Volume 44 Number 2 June 2020 ISSN 0350-5596

Informatica

An International Journal of Computing and Informatics

Special Issue:

SoICT 2019

Guest Editors:

Huynh Thi Thanh Binh

Ichiro Ide

1977

Editorial Boards

Informatica is a journal primarily covering intelligent systems in the European computer science, informatics and cognitive community; scientific and educational as well as technical, commercial and industrial. Its basic aim is to enhance communications between different European structures on the basis of equal rights and international refereeing. It publishes scientific papers accepted by at least two referees outside the author's country. In addition, it contains information about conferences, opinions, critical examinations of existing publications and news. Finally, major practical achievements and innovations in the computer and information industry are presented through commercial publications as well as through independent evaluations.

Editing and refereeing are distributed. Each editor from the Editorial Board can conduct the refereeing process by appointing two new referees or referees from the Board of Referees or Editorial Board. Referees should not be from the author's country. If new referees are appointed, their names will appear in the list of referees. Each paper bears the name of the editor who appointed the referees. Each editor can propose new members for the Editorial Board or referees. Editors and referees inactive for a longer period can be automatically replaced. Changes in the Editorial Board are confirmed by the Executive Editors.

The coordination necessary is made through the Executive Editors who examine the reviews, sort the accepted articles and maintain appropriate international distribution. The Executive Board is appointed by the Society Informatika. Informatica is partially supported by the Slovenian Ministry of Higher Education, Science and Technology.

Each author is guaranteed to receive the reviews of his article. When accepted, publication in Informatica is guaranteed in less than one year after the Executive Editors receive the corrected version of the article.

Special issue on "The Tenth International Symposium on Information and Communication Technology —SoICT 2019"

Since 2010, the Symposium on Information and Communication Technology—SoICT has been organized annually. The symposium series provides an academic forum for researchers to share their latest research findings and to identify future challenges in computer science. The best papers from SoICT 2015, SoICT 2016, and SoICT 2017 have been extended and published in the Special issue "SoICT 2015", "SoICT 2016", and "SoICT 2017" of the Informatica Journal, Vol.40, No.2 (2016), Vol. 41, No. 2 (2017), and Vol. 42, No. 3 (2018), respectively.

In 2019, SoICT was held in the scenic Ha Long bay, Vietnam, during December 4–6, commemorating the tenth event of the symposium series. The symposium covered four major areas of research including Artificial Intelligence and Big Data, Information Networks and Communication Systems, Human-Computer Interaction, and Software Engineering and Applied Computing.

Among 145 submissions from 28 countries, 63 papers were accepted for presentation at SoICT 2019. Among them, the following two papers were carefully selected, after further extension and additional reviews, for inclusion in this special issue.

The first paper, "Privacy Preserving Visual Log Service with Temporal Interval Query using Interval Tree-based Searchable Symmetric Encryption" by Viet-An Pham, Huy-Hoang Huy Chung-Nguyen, Dinh-Hieu Hoang, Mai-Khiem Tran, and Minh-Triet Tran developed a smart secure service for visual logs with a temporal interval query. The proposed scheme achieves efficient search and update time while also maintaining all important security properties such as forward privacy, backward privacy, and it does not leak information outside the desired temporal range.

The second paper, "Cycle Time Enhancement by Simulated Annealing for a Practical Assembly Line Balancing Problem" by Huong Mai Dinh, Dung Viet Nguyen, Long Van Truong, Thuan Phan Do, Thao Thanh Phan, and Nghia Duc Nguyen investigated the assembly line balancing problem. For this problem, they proposed a solution that takes the simulated annealing approach, which was proved to be effective and potentially applicable in practice.

We hope that readers interested in Information and Communication Technology will find this Special Issue a useful collection of papers.

Huynh Thi Thanh Binh
Ichiro Ide

Privacy Preserving Visual Log Service with Temporal Interval Query using Interval Tree-based Searchable Symmetric Encryption

Viet-An Pham, Dinh-Hieu Hoang, Huy-Hoang Chung-Nguyen, Mai-Khiem Tran and Minh-Triet Tran
Faculty of Information Technology - Software Engineering Lab - University of Science, VNU-HCM
E-mail: pvan@apcs.vn, hdhieu@apcs.vn, cnhhoang@apcs.vn, tmkhiem@selab.hcmus.edu.vn,
tmtriet@fit.hcmus.edu.vn

Keywords: searchable symmetric encryption, temporal interval query, visual concept extraction

Received: March 15, 2020

Visual logs become widely available via personal cameras, visual sensors in smart environments, or surveillance systems. Storing such data in public services is a common convenient solution, but it is essential to devise a mechanism to encrypt such data to protect sensitive information while enabling the capability to query visual content even in encrypted format at the services. More precisely, we need smart systems that their security and practicality must be balanced against each other. As far as we know, in spite of their importance in preserving personal privacy, such reliable systems have not gained sufficient attention from researchers. This motivates our proposal to develop a smart secure service for visual logs with a temporal interval query. In our system, visual log data are analyzed to generate high-level contents, including entities, scenes, and activities happening in visual data. Then our system supports data owners to query these high-level contents from their visual logs at the server-side in a temporal interval while the data are still encrypted. Our searchable symmetric encryption scheme TIQSSE utilizes interval tree structure and we prove that our scheme achieves efficient search and update time while also maintaining all important security properties such as forward privacy, backward privacy, and it does not leak information outside the desired temporal range.

Povzetek: Problem uravnoteženja proizvodne poti je predstavljen odprto, brez omejitev npr. števila delavcev, zato je izviren. Avtorji testirajo več algoritmov in predlagajo najboljšega.

1 Introduction

In daily activities, people usually take photos and record video clips to capture moments and events in their lives. Besides, with the booming trend of developing smart interactive environments, such as smart homes, offices, or even cities, visual sensors are densely integrated to our habitats to record then analyse external contexts, such as monitoring users, objects, activities, etc. Consequently, visual lifelogs become increasingly available and are usually uploaded to store in online storage services.

In this paper, we target two challenging problems to better develop an online storage service for private visual data: (i) to search photos or video clips based on their content, and (ii) to protect private data leakage at server-side accidentally or intentionally.

First, we aim to bridge the gap between visual data and their semantics by allowing data owners to search with keywords. Each photo or frame in a video clip is processed to extract high-level concepts, including entities, scene attributes, activities, etc. Different types of high-level concept extractors can be plugged into our framework to meet specific requirements in real applications. Consequently, a photo or video frame can be considered as a document or a set of concepts, which are ready to be retrieved by keywords. We also demonstrate a prototype smart edge camera which can be re-configured remotely to generate visual data with associated extracted concepts.

Second, a typical solution to protect data secrecy is to encrypt before uploading data to an online storage server. However, after encryption, data are no longer suitable to be searched normally. Symmetric Searchable Encryption (SSE), first proposed by Song et al. [23], can be used as a promising solution to privately save data while maintaining the ability to search in a collection of encrypted records. We adopt the approach of SSE in our proposed solution, and carefully design it to ensure the property of a dynamic SSE [13], i.e. to add, update, and delete data efficiently without re-encrypting the whole database.

Besides, we also consider the forward and backward privacy criteria for SSE. Informally, the former means that an update query does not leak information if a newly added document contains keywords that were searched in the past, while the latter is to make sure that it is impossible to retrieve data from deleted files. Forward privacy has been receiving a lot of attention, while backward privacy is only studied in recent years. Most of the existing schemes suffer from key-size overgrowing after deletion queries [2, 4], thus limits the practicality of these schemes.

Moreover, in particular cases, there are new security properties that must be satisfied: search only in a temporal interval, and do not leak any information outside of the requested range. For example, a police wants to check the private security camera of a company from a range of time for a criminal event. The company wants to provide the information exactly from the requested range and not leak any information from other temporal intervals. A similar problem is when we want to search for some disease in a medical database in a temporal interval, it is best to prevent leaking information of patients in other time. This motivates us to define Temporal Interval Query SSE (TIQSSE), a new SSE problem to search by keywords for documents in a particular temporal interval.

This work is the extension paper of previous TIQSSE work [20], with more in-depth explanations and analysis. This paper is also a significantly enhanced version of [7]. Our previous work only guarantees a one-sided access pattern. For more clarity, the one-sided access pattern means that it can only preclude adversaries from extracting information about the documents that were added after the queried interval, while still leaking information of documents that were added before the requested range. In this paper, there is a great improvement on security since our SSE scheme now guarantees two-sided access pattern, which means it also prevents adversaries from gaining information of added documents.

Our newly defined problem is different from the existing range query SSE schemes [1, 14]. In a range query SSE scheme, a server returns every document whose key/identifier is in a queried range. In our temporal interval SSE problem, the server only examines documents whose identifiers are inside the temporal interval to select the documents containing a query keyword w.

Our secure SSE scheme does not suffer from key-size overgrowing after sufficient deleting queries like previous schemes. Our idea is based on $\Sigma o\varphi o\varsigma$ from Raphael Bost et al. [2] in 2016 and modifies it to match our problem. Although there are many improved constructions later [4, 24], these ideas are not suitable for our problems that the use cases we target require efficient deletion operations which (1) have an acceptable time complexity and (2) do not increase server-side usage.

Our main contributions in this paper are as follows.

- We propose a solution for a public visual data storage service to assist data owners to search their photos and video clips with keywords, i.e. concepts extracted from visual content, and preserve data privacy in query and data manipulation (insert, update, delete). We also develop a prototype smart edge camera to handle concept extraction for recorded photos or video clips.

- We also define TIQSSE as a new SSE problem to search with encrypted documents in a temporal interval while preventing data leakage outside the requested range. We then propose an efficient solution to search for a keyword in documents within a determined time range and achieves both forward and backward privacy.

In Section 2, we briefly review approaches and methods related to the two main aspects of our work, visual retrieval with concepts, and searchable symmetric encryption. We propose a smart secure framework for visual data storage service and smart edge camera in Section 3. in Section 4, We review the necessary preliminaries of cryptography, then define the novel TIQSSE problem. Our scheme which tackles this problem is introduced in Section 5. The security analysis of our proposed scheme is presented in Section 6. In Section 7, we draw our conclusion and discuss some fascinating directions for future works.

2 Related work

2.1 Visual retrieval with semantic concepts

Visual log retrieval is one of the important problems to analyse and understand visual content. Different approaches have been proposed to provide users with various modalities to input queries and get retrieved results [17, 18, 26]. Visual semantic concepts from images are usually used as tags or keywords for interactive retrieval systems[26, 25]. The concepts can be detected using available APIs, such as Google Cloud Vision API, or pre-trained object detectors, such as Yolo [21], FasterRCNN [22], etc. Besides, scene attributes and categories [30] can be extracted from images to augment further environmental information of visual data[25]. Some works also utilize captioning [27] or activity recognition to capture the dynamic nature of an image or video clip[16, 15].

In this work, we propose to integrate different concept extractors to create the associated metadata for each photo or video clip stored in the smart visual service. We also develop a prototype smart edge camera that can locally extract concepts in certain tasks before uploading visual data to online storage service (see Section 2.1).

2.2 Searchable symmetric encryption

Song et al. [23] first proposed a solution to Searchable Symmetric Encryption in 2000. Although the first SSE scheme was not efficient, it provided a solid foundation for the problem. Many works were proposed [10, 6] to improve search time and security. However, leakage problems in SSE were not formally defined. Curtmola et al. [8] were the first to explicitly define the general acceptable leakage criteria for SSE problems, including search patterns and access patterns that are frequently used several years later.

Although the previous schemes were optimal in search time, there was no way to update a database without re-encrypting the whole database. To remove this limitation, in 2012, dynamic SSE was proposed by Kamara et al. [13]. Their scheme can efficiently add or remove files with the trade-off by leaking some information when those queries are executed. In particular, forward privacy and backward privacy are not fully satisfied.

SSE problem is continuously studied and improved. Raphael Bost achieved forward privacy in 2016 [2], and also achieved backward privacy one year later [4]. In 2018, Sun et al. [24] proposed Puncturable Symmetric Encryption to construct and improve backward secure. Unfortunately, all schemes mentioned above not only suffer from key-size overgrowing after many deleting queries, but also do not support range query property that we need.

Other than proposing new SSE constructions, many efforts were made to attack the proposed security models. Some notable works are inference attacks on deterministic encryption (DTE) and order-preserving encryption (OPE) [19, 11], leakage-abuse attacks [5, 3, 11, 12] and File-Injection attacks [29, 12].

Before us, there are many works about range queries. However, they all are different from ours. Their solution is used for indexing in relational databases and return entities that have acquired attributes within some range, while in our scheme, we need to return all the files containing the searched keyword in a period.

3 Smart secure framework for visual data storage service

In this section, we present our proposal for a smart secure framework for a visual data storage service. We are inspired by the idea of edge computing to shift the concept extraction task toward the smart camera. There has been an ongoing interest on this shift, particularly from privacy-aware users due to recent breaches in data centers, where sensitive user data is processed and may be used for malicious purposes. If the process is on users' premise, they will have more control over the data that is generated.

3.1 Smart edge camera with concept extraction

Figure 1 illustrates the process for concept extraction from photos/clips in a smart edge camera before uploading visual data with their associated metadata to the secure visual service. Different modules for various concept types can be deployed in the smart edge camera, such as object detection, person recognition, action recognition, scene attribute, category classification, and image captioning.

In our prototype, we utilize NVIDIA Jetson Nano embedded computers with dedicated 128 Maxwell CUDA cores to handle various machine learning tasks. Our smart edge camera prototype can be specialized for various specific tasks with different models to be deployed and updated (see Figure 2). In our model repository, not only there are existing pre-trained models, such as ResNet-50, MobileNet-v2, SSD ResNet-18, SSD Mobilenet-V2, Tiny YOLO V3, but we also prepare our own custom models for other tasks, such as custom object detectors for contexts

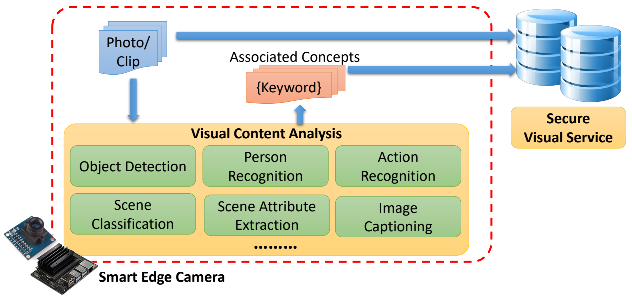


Figure 1: Concept extraction from photos/clips in a smart edge camera.

originated from Vietnam or image captioning with concept augmentation [27].

Future custom models can also be created and further optimized with various techniques such as quantization, fusion, and scheduling available in NVIDIA TensorRT SDK, then deployed to the smart camera. Due to its cloud nature, the devices' software can be remotely updated, and additional machine learning models can be added in a secure manner.

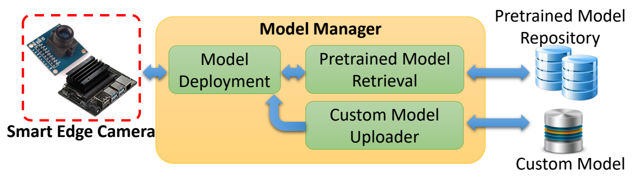


Figure 2: Model update for an edge camera.

3.2 Components in a smart secure visual system

We propose a scenario in which a system collects, processes, and synchronizes the data from various cameras, including the proposed smart edge ones, to a visual data server that utilizes our proposed secure scheme for SSE. Figure 3 illustrates the three key components of the system: a storage and query processing server, camera nodes, and query nodes.

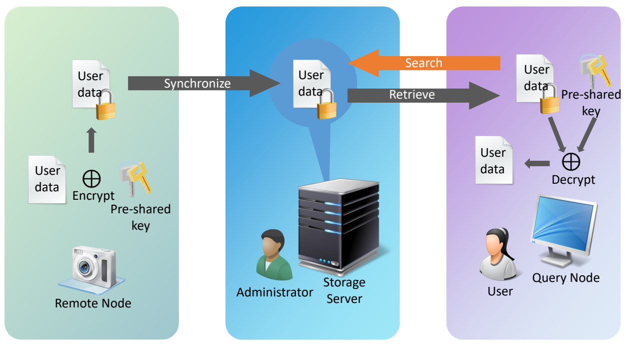


Figure 3: Main components in smart secure visual system.

In our system, the storage and query processing server supports multiple users, and the server owner can be different from the data owners. The owners of the server can fully examine the stored data, but are expected not to understand or to exploit useful information from stored data. Thus, to ensure this crucial property of our visual system, i.e. preserving data privacy for data owners, we define a new problem of Temporal Interval Query SSE (in Section 4) and propose an efficient solution for this problem (in Section 5).

A user, after signing up, is provided a means to submit and retrieve data over commonly utilized protocols, such as HTTP SSL, SMB, or SFTP. Querying is done over an API with a common contract protocol implemented in gRPC, a protocol buffer library that utilizes HTTP2 over an SSL Channel. With gRPC's wide adoption status across numerous languages and libraries, the implementation is relatively easy and open for everyone. Connections to the server are secured with the server's certificate by default. We assume this certificate is self-signed and pre-installed on every query node via personal trusted channels beforehand. A user usually plays both roles as a generator party at upload time from a camera node and a querying party at retrieve time from a query node, which can be his or her mobile device. Thanks to the loosely coupled architecture, our proposed system allows new users to dynamically join in without any interruptions on the server-side using a streamlined user interface.

4 Temporal interval query searchable symmetric encryption

In this section, we first provide background knowledge that includes several cryptographic primitives and the dynamic SSE problem. Then we introduce the definition and security properties for TIQSSE.

4.1 Preliminaries

For consistency in presentation, we denotes:

- $x \overset{\$}{\leftarrow} \{0,1\}^n$ as randomizing n bits then store the result to x.

- n as the number of added files. $\mathbf{F_n}$ as the n-th file. $\mathbf{EF_n}$ as the encrypted file corresponding to $\mathbf{F_n}$.

- $\perp$ as $null$ or $empty$. λ as the security parameter. Security parameter means that unless specified explicitly, the keys used in SSE scheme is λ-bit in length, and the probability for an adversary to break the scheme is $2^{-\lambda}$.

We use several cryptographic primitives from Dan Boneh and Victor Shoup [9] which includes: negligible function, pseudo random generator (PRG), pseudo random function (PRF), simulator, and symmetric encryption. For the symmetric encryption, we denote the encryption of plaintext m with secret key sk as $\mathrm{SE.enc}(sk, m)$, and the decryption of ciphertext c with secret key sk as $\mathrm{SE.dec}(sk, c)$.

We also inherit the idea of trapdoor permutation from Bost et al. [2] and denote the function as π. Formally: One can compute π of p_1 with the secret key K_s: $p_2 \leftarrow \pi(K_s, p_1)$. Given p_2 in π's proper, one can derive the original p_1 with the public key: $p_1 \leftarrow \pi^{-1}(K_p, p_2)$. Finally, for all p we have $p = \pi(K_s, \pi^{-1}(K_p, p)) = \pi^{-1}(K_p, \pi(K_s, p))$.

4.2 Dynamic symmetric searchable encryption

In SSE, we view the database as an array of files $\mathbf{F} = (\mathbf{f_1}, \mathbf{f_2}, ..., \mathbf{f_n})$ where $\mathbf{f_i}$ consists of multiple words $(w_1, w_2, ..., w_{m_i})$. Later when the client request a search on keyword w, the client obfuscate or encrypt w into trapdoor $\mathbf{T}$ and give it to server. The server when receiving $\mathbf{T}$ must return a list of result identifiers $\mathbf{R} = (id_1, id_2, ..., id_r)$ such that when returned to the client, for every i we have $w \in \mathbf{F_{id_i}}$. It is notable that the act of obfuscating w into $\mathbf{T}$ is essential because it hides the original keyword from the server, in this paper we call this as trapdoor generation procedure.

In other words, dynamic SSE consists of one algorithm Setup and two protocols Search and Update.

- In Setup phase, the client creates some keys and keypairs that will be used in the other 2 protocols.

- The Search protocol consists of multiple interactions between client and server when the client request a search. For each client's request, the server should receive the trapdoor $\mathbf{T}$ and return a list of files as we mentioned in the above paragraph.

- The Update protocol is comprised of 2 types of updates which is add a new file and delete an existed file. Depending on which update protocol, the encrypted database on the untrusted server will be modified based on the SSE scheme.

$\mathsf{Correctness}$. An SSE is *correct* if the probability that the search protocol returns the false results to client is negligible.

$\mathsf{Security}$. The SSE scheme Σ is said to be adaptively secure, if for any adversary $\mathcal{A}$ who issues a polynomial number of queries q(λ), there exist a polynomial-time simulator $\mathcal{S}$ such that:

$$|P[\mathrm{SSEReal}^{\Sigma}_{\mathcal{A}}(\lambda, q) = 1] - P[\mathrm{SSEIdeal}_{\mathcal{A},\mathcal{S},\mathcal{L}}(\lambda, q) = 1]| < negl(\lambda)$$

Informally, the simulator $\mathcal{S}$ can be thought of as an efficient probabilistic algorithm such that its output distribution is identical to the real scheme's output distribution. Then, the theorem above can be semantically understood as: if we can prove there exists a simulator $\mathcal{S}$ of SSE scheme Σ, then it is very hard for the adversary to distinguish between the real case with a simulation case. Hence, we achieve adaptive security for SSE.

4.3 Definition of temporal interval query SSE

Temporal Interval Query SSE continues to use the model of the original dynamic Symmetric Searchable Encryption but modifies the **Search** protocol. When the client issues a search request, firstly he chooses a range of interest $[L; R]$, then he chooses a keyword w he wants to search on, then he generates the trapdoor vector $\mathbf{T}$ that represents the keyword w for that range $[L; R]$, finally he gives $(\mathbf{T}, L, R)$ to server. The server when receiving $(\mathbf{T}, L, R)$ must return a list of result identifiers $\mathbf{R_{w,L,R}} = (id_1, id_2, ..., id_r)$ such that when returned to the client, for every i we have $w \in F_{id_i}$ and $L \leq id_i \leq R$ (see Figure 4).

4.4 Security

Forward privacy: Informally, a SSE scheme achieves forward privacy if its Update query does not leak any information about the newly added file even if it contains keywords that are previously searched keywords. For example, the client searched for a keyword w. Later, when the client add a file $\mathbf{F}$ that contains w, the server should not know that w exists inside $\mathbf{F}$. Many researches [5] showed that if a scheme does not attain forward privacy, the client's queries, or even the plaintext, can be revealed even with small leakage. There also exist attacks [29] that can effectively exploit the vulnerability of those schemes to break query privacy. In addition, forward privacy can also improves time and space performances [2].

Backward privacy: To have backward privacy in dynamic SSE, we must prevent the adversary from gaining knowledge of deleted files from new queries. For example, if there exists a deleted file $\mathbf{F}$ that contains a word w and has never been queried, in the future when client search for

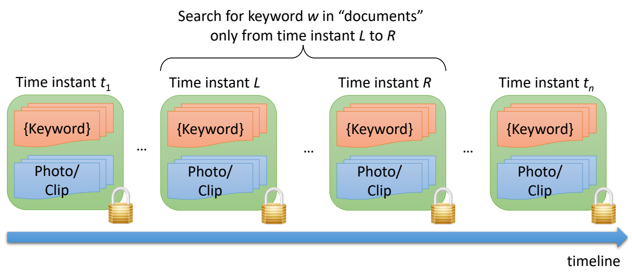


Figure 4: An example of temporal interval query.

word w, it is expected to prevent the server from knowing $w \in \mathbf{F}$.

In order to have searchable property over encrypted data, there must be some leaking information throughout the process. We follow many previous works [2, 4, 24] and call this as leakage function $\mathcal{L} = (\mathcal{L}^{\text{Stp}}, \mathcal{L}^{\text{Srch}}, \mathcal{L}^{\text{Updt}})$. The leakage function $\mathcal{L}$ is used to express the information learned by the untrusted server from 3 protocols **Setup**, **Search**, **Update**.

Setup leakage: In the setup algorithm, the client generates some keys and keypairs for later usage in **Search** and **Update** protocol. Because of that, the leakage of setup phase is the public keys (if there is any) that the client wants to share with the server $\mathcal{L}^{\text{Stp}} = \mathbf{PK}$.

Search leakage: Firstly, let $\mathbf{Q}$ as the search requests of the client where $\mathbf{Q_i} = (\mathbf{T_i}, L_i, R_i)$; $\mathbf{R}$ as the results returned by the untrusted server; $\mathbf{R_i}$ as the result of $\mathbf{Q_i}$ where its content is $(id_{i,1}, id_{i,2}, ..., id_{i,r_i})$; $\mathbf{H}$ as the history of previous searches from the client that $\mathbf{H} = (\mathbf{Q}, \mathbf{R})$. We define the allowed leakage of search protocol is comprised of search pattern $\sigma(\mathbf{H})$ and access pattern $\alpha(\mathbf{H})$.

The access pattern $\alpha(\mathbf{H})$ indicates the leakage of the returned values of the queries. That is for each query we want to only leak the existence of keyword w within the existed files within the interval $[L; R]$ and non elsewhere.

The search pattern $\sigma(\mathbf{H})$ represents the leakage of the query parameters from the client. The search pattern consists of 2 levels of security:

- Perfect security: when analyzing 2 different queries i and j with common keyword w, it is very hard for the server to deduce $\mathbf{T_i}$ and $\mathbf{T_j}$ to be the same keyword w. Because of that in this setup, the client perfectly hide the search pattern and can secure against many inference attack types [19, 11].

- Weak security: when analyzing 2 different queries i and j with common keyword w, the server can easily deduce $\mathbf{Q_i}$ and $\mathbf{Q_j}$ has the same search keyword w if and only if the queried range $[L_i; R_i]$ intersects with $[L_j; R_j]$ at some point.

Update leakage: The update leakage consists of the leakage of add new file protocol and delete file protocol. The add new file protocol leaks n as the number of added files and the size of all the files. The delete file protocol leaks the identifier of deleted files.

5 Proposed scheme for TIQSSE

In this section, the parts are arranged as the followings. Firstly, we outline our scheme at the first part, the remaining parts describe how our scheme works in setup protocol, add new file protocol, search protocol and delete file protocol.

5.1 Scheme outline

There are 7 polynomial-time algorithms in total. The first 6 algorithms are executed by the client, while the last algorithm is done by the server.

- $(sk, \mathbf{K}) \leftarrow \mathrm{KeyGen}(1^\lambda)$: is a probabilistic algorithm that uses the security parameter λ to setup the secret key sk for encryption/decryption and a vector of keypairs $\mathbf{K}$ for trapdoor generation procedure.
- $\mathbf{EF_n} \leftarrow \mathrm{Enc}(sk, \mathbf{F_n})$: is a probabilistic algorithm that encrypt $\mathbf{F_n}$ into $\mathbf{EF_n}$.
- $\mathbf{F_n} \leftarrow \mathrm{Dec}(sk, \mathbf{EF_n})$: is the reverse algorithm of Enc.
- $t_n \leftarrow \mathrm{Trpdr}(sk, \mathbf{K}, n, w)$: is a deterministic algorithm that illustrates the process of generating trapdoor value of keyword w in file $\mathbf{F_n}$ into trapdoor value t_n.
- $\mathbf{I_n} \leftarrow \mathrm{CreateIndex}(sk, n, \mathbf{F_n})$: is a deterministic algorithm that illustrates the process of creating an index file $\mathbf{I_n}$. This index file acts as a look up table for the untrusted server to search on in the search protocol.
- $\mathbf{T} \leftarrow \mathrm{SearchToken}(sk, \mathbf{K}, w, L, R)$: is a deterministic algorithm that illustrates the process when the client prepare the search request. Using the keys and (w, L, R) it outputs trapdoor $\mathbf{T}$ and give it to the server.
- $\mathbf{R_{w,L,R}} \leftarrow \mathrm{Search}(\mathbf{T}, \mathbf{I_{L,\ldots,R}}, L, R)$: be a deterministic algorithm that illustrates how the untrusted server uses the trapdoors $\mathbf{T}$ and the range of interest $[L, R]$ to return the appropriate files back to the client.

5.2 Setup protocol

The client runs KeyGen algorithm: $(sk, \mathbf{K}) \leftarrow \mathrm{KeyGen}(1^\lambda)$ to generate sk and $\mathbf{K}$. For sk, the client can randomize λ bits and store it in sk as: $sk \overset{\$}{\leftarrow} \{0,1\}^\lambda$. For $\mathbf{K}$ which consists of 2 trapdoor permutation keypairs $(\mathbf{K_{ul}}, \mathbf{K_{ur}})$, the client can generate these keypairs by generating trapdoor permutation keypair on λ bits.

A side note here is the client must assure that the id of each files given to the server are incremental. Thus, id starts from -1 and n starts from 0 when no file has been added.

5.3 Add new file protocol

In the following algorithms, let n be the number of added files, $\mathbf{F_n}$ be the new file that client wants to add, $\mathbf{EF_n}$ be the encrypted file of $\mathbf{F_n}$, $\mathbf{I_n}$ be the encrypted index of $\mathbf{F_n}$.

The main idea for the add new file protocol is for each file $\mathbf{F_i}$, the client encrypts $\mathbf{F_i}$ into $\mathbf{EF_i}$ and generates the encrypted index $\mathbf{I_i}$. Finally, the client gives $\mathbf{EF_i}$ and $\mathbf{I_i}$ to the server. The usage of encrypted index $\mathbf{I_i}$ is for the server to indicate whether word w is contained inside the corresponding encrypted file $\mathbf{EF_i}$ or not.

5.3.1 Encrypt file

With the above idea, because the searching step at the server-side only requires the encrypted index, it is trivial to encrypt the file $\mathbf{F_n}$ using popular symmetric encryption algorithm like AES. Plus, the decryption algorithm is the reverse function of encryption. Let SE be the symmetric encryption algorithm:

- Enc : $\mathbf{EF_n} \leftarrow \mathrm{SE.enc}(sk, \mathbf{F_n})$
- Dec : $\mathbf{F_n} \leftarrow \mathrm{SE.dec}(sk, \mathbf{EF_n})$

5.3.2 Create encrypted index

In this section we use a data structure called $\mathbf{map}$ for the encrypted index. Firstly, we initialize $\mathbf{I_n}$ as an empty map. Then for each word w in $\mathbf{F_n}$ we create the trapdoor and store it inside $\mathbf{I_n}$ as following procedure:

1. $t_n \leftarrow \mathrm{Trpdr}(sk, \mathbf{K}, n, w)$.
2. Check whether t_n already exists in $\mathbf{I_n}$ as: $t_n \in \mathbf{I_n}$, if yes, then repeatedly randomizing t_n as: $t_n \overset{\$}{\leftarrow} \{0,1\}^\lambda$ until $t_n \notin \mathbf{I_n}$. After this step, we call t_n *garbage data* if $t_n \neq \mathrm{Trpdr}(sk, \mathbf{K}, n, w)$.
3. Store t_n into $\mathbf{I_n}$.

With the algorithm above, there will not have any word duplication within an index file **I** because whenever a word already exists inside, it will be substituted as a *garbage word* by randomization. Furthermore, when provided a trapdoor t, the server can easily check whether t exists inside an encrypted index **I**, which enables searchability.

5.3.3 Trapdoor generation

The interval tree model is mainly for this step and is quite complicated. Briefly, Interval tree (or Segment tree) is a binary tree where each node contains information about a specific range $[L, R]$. Every none-leaf node also has two child node to manages $[L, M]$ and $[M+1, R]$ where $M = \lfloor \frac{L+R}{2} \rfloor$. The complexity of search and update query is $\mathcal{O}(\log_2 n)$.

Firstly, we visualize how interval tree trapdoor generation works, then we give an example of it in practice. We also provide graphical Figure 5 that is easier for readers to visualize the appearance of interval tree in our model.

- We view the files as nodes at level 0 and is arranged from left to right with incremental id starting from 0 to $n-1$.

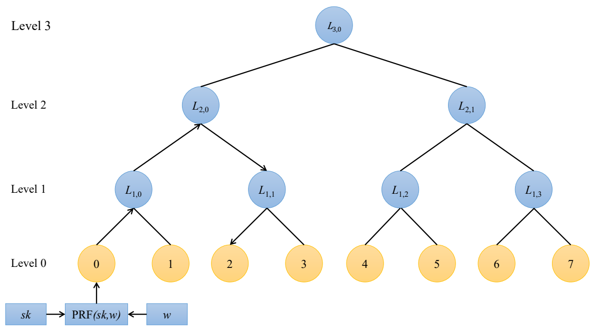


Figure 5: Interval Tree for Trapdoor generation visualization and trapdoor transformation in practice.

- The nodes at higher levels are treated as interval nodes, that is it will cover an interval of continuous files. For example in Figure 5, node $L_{2,1}$ covers files from 4 to 7 and node $L_{1,1}$ covers files 2 and 3.
- From any word w, client can generate trapdoor for file 0 by: $t_0 \leftarrow \text{PRF}(sk, w)$. And from any node, the client can "move" the trapdoor at that node onto the **Up-Right** node by using trapdoor permutation on secret component of $\mathbf{K_{ur}}$ as: $t_{ur} \leftarrow \pi(K_{ur_s}, t)$. And from any node, we can "move" trapdoor to the **Down-Left** node by applying reverse trapdoor permutation using public component of $\mathbf{K_{ur}}$ as: $t_{dl} \leftarrow \pi^{-1}(K_{ur_p}, t)$. In case of moving **Up-Left** and **Down-Right**, we just need to apply π and π^{-1} on $\mathbf{K_{ul}}$ like above steps.

The important point here is the public key of $\mathbf{K_{ur}}$ and $\mathbf{K_{ul}}$ is available for both client and server but only the client holds the secret component of $\mathbf{K_{ur}}$ and $\mathbf{K_{ul}}$. Hence, for any node $L_{i,j}$, the server can only move the trapdoor value to nodes in sub-tree of $L_{i,j}$. However, the client can move anywhere he wants because he holds the secret key.

We denote $t_{i,j}$ to be the trapdoor value at node $L_{i,j}$; t_i and $t_{0,i}$ to be the trapdoor value at F_i. Below we show an example of generating t_2 from keyword w. Figure 5 also demonstrates the process.

1. $t_{0,0} \leftarrow \text{PRF}(sk, w)$
2. $t_{1,0} \leftarrow \pi(K_{ur_s}, t_{0,0})$
3. $t_{2,0} \leftarrow \pi(K_{ur_s}, t_{1,0})$
4. $t_{1,1} \leftarrow \pi^{-1}(K_{ul_p}, t_{2,0})$
5. $t_{0,2} \leftarrow \pi^{-1}(K_{ur_p}, t_{1,1})$

5.4 Search protocol

To search for the existence of keyword w within range $[L; R]$, the client runs SearchToken algorithm to create trapdoor vector $\mathbf{T}$ then give it to the server. The server will use the range $[L; R]$ and $\mathbf{T}$ to run Search algorithm and return the appropriate encrypted files to the client.

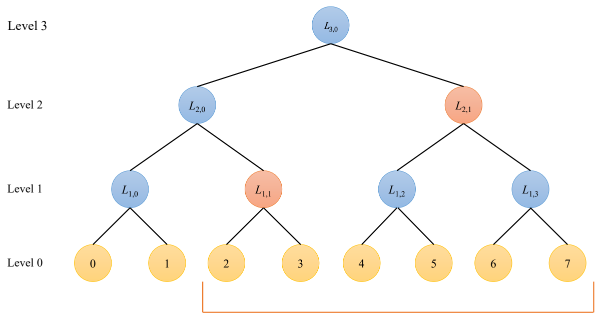


Figure 6: Client issue SearchToken from file 2 to file 7, sending $t_{1,1}$ and $t_{2,1}$ to query the server.

5.4.1 SearchToken algorithm

Previously we mentioned that given a trapdoor $t_{i,j}$ at some node $L_{i,j}$, the server can easily compute all trapdoor values at nodes inside sub-tree of $L_{i,j}$. With this special characteristic, the client only needs to compute trapdoor value at nodes such that it covers only in range $[L; R]$. To optimize the computational complexity, the client needs to find as a minimal number of interval nodes as possible.

The algorithm is simple. Iterates from L to R, let $L_{0,i}$ be our current node, while there exists an up-right parent node and the parent node still covers within the range $[L; R]$, traverse to the parent node and repeat the process. After finding the appropriate parent of $L_{0,i}$, let k be the level of that parent node, we can skip the next 2^k nodes and repeat the process until we cover all nodes from $[L; R]$.

In Figure 6, we demonstrate SearchToken when issuing query from file 2 to file 7, the red nodes $L_{1,1}$ and $L_{2,1}$ is sufficient to cover the range $[2, 7]$. After finding the interval nodes that cover $[L; R]$, the client can use trapdoor generation procedure mentioned earlier to calculate $\mathbf{T} = (t_{1,1}, t_{2,1})$ and send it to server.

5.4.2 Search algorithm

For each value $t_{i,j}$ in the received trapdoor vector $\mathbf{T}$, the server can use π^{-1} with the public key of $\mathbf{K_{ur}}$ and $\mathbf{K_{ul}}$ to traverse down to any nodes in sub-tree of $t_{i,j}$. After traversing down to the level 0 nodes, the server can easily check whether trapdoor $t_{0,i}$ exists inside the encrypted index $\mathbf{I_i}$. Finally, the server returns all the encrypted file $\mathbf{EF_i}$ that satisfies above conditions.

5.5 Delete protocol

To delete a file, the client sends a single variable k to the server to indicates that he wants to delete $\mathbf{EF_k}$ and $\mathbf{I_k}$. After that, the server deletes $\mathbf{EF_k}$ and $\mathbf{I_k}$ in its database/hard drive. Later when Search algorithm occurs, without the data of $\mathbf{I_k}$, the server cannot check trapdoor t exists in $\mathbf{I_k}$ or not because I_k has already been deleted. However, only deleting $\mathbf{EF_k}$ and $\mathbf{I_k}$ is not efficient because the server must iterate through every file at level 0, even the deleted ones, which would result in $\mathcal{O}(R-L)$.

We can optimize the runtime by storing an additional boolean isDeleted in each node $L_{i,j}$. The isDeleted boolean indicates whether the entire sub-tree of $L_{i,j}$ has been deleted or not. Then we fix the deletion algorithm as following: when deleting the k-th file, mark $L_{0,k}$.isDeleted as True. Then iterate to the parent of $L_{0,k}$, if the 2 children of that current parent are also deleted, then mark that parent as deleted and continues to move onto its parent and repeat the process.

With the above optimization, the Search algorithm by the server will be modified a little bit. At server-side while traversing to the nodes of sub-tree of $L_{i,j}$, if server encounters a deleted node, the server can ignore all the nodes in that sub-tree, which can optimize the computation.

6 Scheme analysis

6.1 Correctness

We will prove the *correctness* of our scheme based on the *correctness* that we introduced in section 4.2. Our proof has two parts:

The search result contains all documents having the searched keyword w. Obviously, for each document having the searched keyword w, the tree associated with w must mark it containing this keyword. Thus, on executing the searching protocol, server will see that the document is included in the tree, which means that the document will be listed in the result.

For any document which does not have the searched keyword w, the probability that it is listed in the search result is extremely small. We consider some arbitrary document. Let m be the size of the output of trapdoor function which is used to encode the words, $size$ the size of the document, $amtw$ the amount of valid distinct keywords of that document, and $dictsize$ the amount of valid distinct keywords of the whole dataset.

Obviously, the probability that there does not exist any garbage data that collides with a valid keyword is:

$$p = \binom{2^m - dictsize - amtw}{size - amtw} \Big/ \binom{2^m - amtw}{size - amtw}$$

which is reduced as,

$$p = \prod_{i=1}^{size-amtw} \frac{2^m - dictsize - size + amtw + i}{2^m - size + i}$$

By using a suitable trapdoor permutation whose output size m is big enough, we make the probability that the server falsely determines a searched keyword w exists in a document, which equals $(1-p)$, very small. The bigger m is, the more precise the returned results are.

6.2 Formal adaptive security proof

We formally describe adaptive security proof of our scheme based on section 4.2. We retrieve the following games from the TIQSSE scheme:

Game G_0. The first game G_0 is completely identical to the real world game $\mathrm{SSEReal}^{\Sigma}_{\mathcal{A}}(\lambda, q)$. Thus,

$$P[SSEReal^{\Sigma}_{\mathcal{A}}(\lambda, q) = 1] = P[G_0 = 1]$$

Game G_1. In this game, we replace the function PRF by a truly random key generator. More precisely, G_1 will get a random element in the domain of PRF whenever it comes to a new word w, and stores this element in a table Key containing all key associated with each queried word w. So in order to exist some adversary who can distinguish between G_0 and G_1, he must break the security of PRF. Therefore we have:

$$P(G_0 = 1) - P(G_1 = 1) \leq \mathrm{Avd}^{PRF}_{\mathcal{B}_1}(\lambda)$$

Game G_2. G_2 does not use trapdoor permutation anymore. Instead, it uses random oracles for π and programs π^{-1} such that $\pi_{key}(\pi^{-1}_{key})(i) = i$ for any arbitrary key and i. Obviously, the problem of distinguishing between G_1 and G_2 can be reduced to the problem of cracking the onewayness of π. Since our scheme uses two pair of public-private keys, there exists an efficient adversary $\mathcal{B}_2$ such that:

$$P(G_1 = 1) - P(G_2 = 1) \leq \mathrm{Avd}^{OW}_{\mathcal{B}_2}(\lambda)$$

The simulator $\mathcal{S}$. We construct our simulator $\mathcal{S}$ identical to game G_2 which changes PRF and Trapdoor Permutation as random oracles:

$$P(G_2 = 1) = P(SSEIdeal^{\Sigma}_{\mathcal{A},\mathcal{S},\mathcal{L}_\Sigma} = 1)$$

Combining all above results, we conclude that we can simulate the original scheme and achieve adaptive security mentioned in Section 4.2:

$$\begin{aligned} P(SSEReal^{\Sigma}_{\mathcal{A}} = 1) - P(SSEIdeal^{\Sigma}_{\mathcal{A},\mathcal{S},\mathcal{L}_\Sigma} = 1) \\ \leq \mathrm{Avd}^{PRF}_{\mathcal{B}_1}(\lambda) + \mathrm{Avd}^{OW}_{\mathcal{B}_2}(\lambda) \end{aligned}$$

6.3 Informal adaptive security proof

To make it more understandable, we also provide an informal proof for the adaptive security of SSE. From our scheme we can derive several consequences:

Truly random encrypted file. The encrypted file **EF** is obtained by encrypting the plain file with a secure symmetric encryption algorithm. So to break the randomness of **EF** the adversary must break the symmetric encryption algorithm.

Truly random encrypted index. Let us recall how we create an encrypted index. Firstly from keyword w we apply PRF to generate t_0. So if an adversary able to break the randomness of t_0, he must break PRF. After that, we apply several trapdoor permutation π and π^{-1} to generate other t_i values. Again, to break the randomness of t_i, the adversary must find a way to crack trapdoor permutation. Hence, we claim that we generate truly random encrypted indexes.

Conclusion. From the 2 above proofs, we say that our scheme can use a random oracle to simulate the process of generating encrypted file EF and encrypted index I and therefore we achieve adaptive secure SSE mentioned in section 4.4.

6.4 Access pattern security

Without knowing secret component of $\mathbf{K_{ul}}$ and $\mathbf{K_{ur}}$, when receiving some trapdoor $t_{i,j}$ of node $L_{i,j}$, in order for the adversary to figure out trapdoor value at parent node of $L_{i,j}$, he must find a way to break the trapdoor permutation function. Furthermore in our scheme, the client only gives server values $t_{i,j}$ that cover the interval $[L; R]$. With this, our scheme achieves access pattern security.

6.5 Search pattern security

Assume that the client has issued 2 queries $\mathbf{Q_1} = (\mathbf{T_1}, L_1, R_1)$ and $\mathbf{Q_2} = (\mathbf{T_2}, L_2, R_2)$ where $\mathbf{T_1}$ and $\mathbf{T_2}$ refers to the same keyword w. There are 2 cases:

1. If $[L_1; R_1]$ intersects with $[L_2, R_2]$: let k be a number where $k \in [L_1; R_1] \cap [L_2; R_2]$, the server obviously can check T_1 and T_2 be the same search keyword because the trapdoor t_k of the 2 search queries will be the same.

2. If $[L_1; R_1]$ does not intersect with $[L_2, R_2]$: it is impossible for the untrusted server to check T_1 and T_2 to be the same search keyword unless he can calculate π and achieve trapdoor value at parent nodes of $\mathbf{T}$, which is very hard and the probability is negligible.

By analyzing the 2 above cases, we claim that our scheme achieves weak search pattern security of TIQSSE.

6.6 Forward privacy

We already stated that our scheme achieves access pattern security which prevents the server from gaining knowledge of outside the interval query. Furthermore, it is obvious that newly added files are outside of past search queries. To sum up, we claim that our scheme obtains forward privacy.

6.7 Backward privacy

When receiving a deleting query, the server deletes the encrypted file along with the encrypted index on the database/hardware which prevents the server from gaining more knowledge from it in future searches.

However, if the attacker can clone the encrypted index to elsewhere without the client's knowledge, then our scheme does not achieve backward privacy. Because of that, our scheme can only guarantee backward privacy if we assume that the system is honest-but-curious. The honest-but-curious property implies that the system follows explicitly how the scheme is supposed to do, but still listens to the client's queries and try to exploit for vulnerabilities. This is an important property that has been used widely in many constructions [8, 13, 2, 4, 24].

6.8 Complexity analysis

Search complexity. Let $n_{add} = R - L + 1$ be the number of historically added documents, n be the number of remaining documents, and m be the number of document-deleted segments in the searching range $[L, R]$, respectively. For each value $t_{i,j}$ received from client, server must traverse all nodes of the sub-tree associated with $t_{i,j}$. The size of that sub-tree is equal to $2 \cdot n_{leaf} - 1$ where n_{leaf} is the number of leaves of that sub-tree. However, we don't have to consider nodes which is assigned as deleted. That means we do not traverse any sub-tree whose root is deleted. In conclusion, the search complexity is $\mathcal{O}(n+m)$.

Add document complexity. Obviously, the server's time complexity of Add operation is $\mathcal{O}(1)$. For the client's time complexity of Add operation: The first step (*encrypt file*) implementation time depends on which symmetric encryption is used. Let $size$ be the number of words in file. Because the complexity of Trpdr is $\mathcal{O}(\log(n))$ where n is the identifier of the file, the complexity of creating encrypted index is $\mathcal{O}(\log(n) \cdot size)$.

Delete document complexity. Trivially, the client's time complexity of deleting document is $\mathcal{O}(1)$. On server, the complexity of deleting document is proportional to the number of iterations which is not greater than $log(n)$ where n is the total number of historically added documents.

Storage complexity. The client's storage is $\mathcal{O}(1)$ because the user only needs to store the keys. About the server's storage, we can see that the size of encrypted indexes is the same as that of encrypted data. Plus, each node of the interval tree holds an isDeleted attribute that totally costs the complexity storage as the number of existing nodes in the tree. Therefore, we can conclude that the storage complexity of server is $\mathcal{O}(esize)$ where $esize$ is the size of encrypted data.

7 Discussion and conclusion

In this paper, we first introduce a smart secured multimedia service to provide visual content analysis with smart edge cameras while preserving the privacy of data owners. We also define a new problem for range queries with searchable symmetric encryption to prevent sensitive information leakage to service provider. Finally, we propose a secured scheme for the new problem.

It is impractical that our scheme only supports single keyword search although there has been many previous works toward multi keywords search [28]. To make our scheme works for multi keywords search, we can combine multiple keywords that is near each other into one single word. For example, we can combine k consecutive words into one and mark it as existed in the encrypted index. For further versatile, we can even permute these k words into $k!$. With this approach, we have multi keywords search property.

In order to achieve perfect secrecy in search pattern, we can simply not searching for keyword w of the same range $[L; R]$ that has been issued before. Because the client has already searched for word w within range $[L; R]$, we can store the result in our memory. Later when we want to query the keyword w within other range $[L_2; R_2]$, we first eliminate all the shared intersection on word w that has been searched before, let the interval after elimination be $\mathbf{S}$. Then, we only query on $\mathbf{S}$ and merge the returned results with the solutions in our memory. With this approach, we can achieve perfect secrecy of the search pattern because we never search for keyword w that intersects with the previous search interval. However, the downside of this is the client must have some mechanism to store previously search queries, which makes it impractical.

Currently, we are improving our system to optimize its performance, scalability, and reliability. We also analyse other data structures to enhance the solution for SSE and consider potential leakage via side-channel information.

8 Acknowledgement

This research is partially supported by research funding from Advanced Program in Computer Science and Honors Program, University of Science, Vietnam National University - Ho Chi Minh City. We also would like to express our appreciation to AIOZ Pte Ltd and VINIF for supporting our research team.

References

[1] Boelter, T., Poddar, R., Popa, R.A.: A secure one-roundtrip index for range queries. IACR Cryptology ePrint Archive **2016**, 568 (2016)

[2] Bost, R.: $\sigma o\varphi o\varsigma$: Forward secure searchable encryption. In: Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security. pp. 1143–1154. ACM (2016). https://doi.org/10.1145/2976749.2978303

[3] Bost, R., Fouque, P.A.: Thwarting leakage abuse attacks against searchable encryption-a formal approach and applications to database padding. IACR Cryptology ePrint Archive **2017**, 1060 (2017)

[4] Bost, R., Minaud, B., Ohrimenko, O.: Forward and backward private searchable encryption from constrained cryptographic primitives. In: Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security. pp. 1465–1482. ACM (2017). https://doi.org/10.1145/3133956.3133980

[5] Cash, D., Grubbs, P., Perry, J., Ristenpart, T.: Leakage-abuse attacks against searchable encryption. In: Proceedings of the 22nd ACM SIGSAC conference on computer and communications security. pp. 668–679. ACM (2015). https://doi.org/10.1145/2810103.2813700

[6] Chang, Y.C., Mitzenmacher, M.: Privacy preserving keyword searches on remote encrypted data. In: IACR Cryptology ePrint Archive (2004)

[7] Chung-Nguyen, H.H., Pham, V.A., Hoang, D.H., Tran, M.T.: Keyword-search interval-query dynamic symmetric searchable encryption. In: International Conference on Future Data and Security Engineering. pp. 673–680. Springer (2019). https://doi.org/10.1007/978-3-030-35653-8_46

[8] Curtmola, R., Garay, J., Kamara, S., Ostrovsky, R.: Searchable symmetric encryption: improved definitions and efficient constructions. Journal of Computer Security **19**(5), 895–934 (2011). https://doi.org/10.3233/jcs-2011-0426

[9] Dan Boneh, V.S.: A Graduate Course in Applied Cryptography (2017)

[10] Goh, E.J., et al.: Secure indexes. IACR Cryptology ePrint Archive **2003**, 216 (2003)

[11] Grubbs, P., Sekniqi, K., Bindschaedler, V., Naveed, M., Ristenpart, T.: Leakage-abuse attacks against order-revealing encryption. In: 2017 IEEE Symposium on Security and Privacy (SP). pp. 655–672. IEEE (2017). https://doi.org/10.1109/sp.2017.44

[12] Islam, M.S., Kuzu, M., Kantarcioglu, M.: Access pattern disclosure on searchable encryption: Ramification, attack and mitigation. In: Ndss. vol. 20, p. 12. Citeseer (2012)

[13] Kamara, S., Papamanthou, C., Roeder, T.: Dynamic searchable symmetric encryption. In: Proceedings of the 2012 ACM conference on Computer and communications security. pp. 965–976. ACM (2012). https://doi.org/10.1145/2382196.2382298

[14] Kerschbaum, F., Tueno, A.: An efficiently searchable encrypted data structure for range queries. In: Lecture Notes in Computer Science, pp. 344–364. Springer International Publishing (2019). https://doi.org/10.1007/978-3-030-29962-0_17

[15] Le, N., Nguyen, D., Hoang, T., Nguyen, T., Truong, T., Duy, T.D., Luong, Q., Vo-Ho, V., Nguyen, V., Tran, M.: Smart lifelog retrieval system with habit-based concepts and moment visualization. In: Proceedings of the ACM Workshop on Lifelog Search Challenge, LSC@ICMR 2019, Ottawa, ON, Canada, 10 June 2019. pp. 1–6 (2019). https://doi.org/10.1145/3326460.3329155

[16] Le, N., Nguyen, D., Nguyen, V., Tran, M.: Lifelog moment retrieval with advanced semantic extraction and flexible moment visualization for exploration. In: Working Notes of CLEF 2019 - Conference and Labs of the Evaluation Forum, Lugano, Switzerland, September 9-12, 2019. (2019)

[17] Le, T.K., Ninh, V.T., Dang-Nguyen, D.T., Tran, M.T., Zhou, L., Redondo, P., Smyth, S., Gurrin, C.: Lifeseeker - interactive lifelog search engine at lsc 2019. In: Proceedings of the 2019 ACM Workshop on The Lifelog Search Challenge. ACM (2019). https://doi.org/10.1145/3326460.3329162

[18] Münzer, B., Leibetseder, A., Kletz, S., Primus, M.J., Schoeffmann, K.: lifexplore at the lifelog search challenge 2018. In: Proceedings of the 2018 ACM Workshop on The Lifelog Search Challenge. pp. 3–8 (2018). https://doi.org/10.1145/3210539.3210541

[19] Naveed, M., Kamara, S., Wright, C.V.: Inference attacks on property-preserving encrypted databases. In: Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security. pp. 644–655. ACM (2015). https://doi.org/10.1145/2810103.2813651

[20] Pham, V.A., Hoang, D.H., Chung-Nguyen, H.H., Tran, M.K., Tran, M.T.: Privacy preserving visual log service with temporal interval query using interval tree-based searchable symmetric encryption. In: Proceedings of the Tenth International Symposium on Information and Communication Technology. pp. 425–432 (2019). https://doi.org/10.1145/3368926.3369701

[21] Redmon, J., Farhadi, A.: Yolov3: An incremental improvement (2018), `http://arxiv.org/abs/1804.02767`

[22] Ren, S., He, K., Girshick, R., Sun, J.: Faster r-cnn: Towards real-time object detection with region proposal networks. In: Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1. pp. 91–99. NIPS'15, MIT Press, Cambridge, MA, USA (2015). https://doi.org/10.1109/tpami.2016.2577031

[23] Song, D.X., Wagner, D., Perrig, A.: Practical techniques for searches on encrypted data. In: Proceeding 2000 IEEE Symposium on Security and Privacy. S&P 2000. pp. 44–55. IEEE (2000). https://doi.org/10.1109/secpri.2000.848445

[24] Sun, S.F., Yuan, X., Liu, J.K., Steinfeld, R., Sakzad, A., Vo, V., Nepal, S.: Practical backward-secure searchable encryption from symmetric puncturable encryption. In: Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security. pp. 763–780. ACM (2018). https://doi.org/10.1145/3243734.3243782

[25] Tran, M., Truong, T., Duy, T.D., Vo-Ho, V., Luong, Q., Nguyen, V.: Lifelog moment retrieval with visual concept fusion and text-based query expansion. In: Working Notes of CLEF 2018 - Conference and Labs of the Evaluation Forum, France, September 2018. (2018)

[26] Truong, T.D., Dinh-Duy, T., Nguyen, V.T., Tran, M.T.: Lifelogging retrieval based on semantic concepts fusion. In: Proceedings of the 2018 ACM Workshop on The Lifelog Search Challenge. pp. 24–29. ACM (2018). https://doi.org/10.1145/3210539.3210545

[27] Vo-Ho, V.K., Luong, Q.A., Nguyen, D.T., Tran, M.K., Tran, M.T.: Personal diary generation from wearable cameras with concept augmented image captioning and wide trail strategy. In: Proceedings of the Ninth International Symposium on Information and Communication Technology. pp. 367–374. SoICT 2018, ACM (2018). https://doi.org/10.1145/3287921.3287955

[28] Xia, Z., Wang, X., Sun, X., Wang, Q.: A secure and dynamic multi-keyword ranked search scheme over encrypted cloud data. IEEE transactions on parallel and distributed systems **27**(2), 340–352 (2015). https://doi.org/10.1109/tpds.2015.2401003

[29] Zhang, Y., Katz, J., Papamanthou, C.: All your queries are belong to us: The power of file-injection attacks on searchable encryption. In: 25th {USENIX} Security Symposium ({USENIX} Security 16). pp. 707–720 (2016)

[30] Zhou, B., Lapedriza, A., Khosla, A., Oliva, A., Torralba, A.: Places: A 10 million image database for scene recognition. IEEE Transactions on Pattern Analysis and Machine Intelligence (2017). https://doi.org/10.1109/tpami.2017.2723009

Cycle Time Enhancement by Simulated Annealing for a Practical Assembly Line Balancing Problem

Mai Huong Dinh*
Hanoi University of Science and Technology, Hanoi, Vietnam
Hanoi University of Industry, Hanoi, Vietnam
E-mail: huongdm@haui.edu.vn

Viet Dung Nguyen*, Van Long Truong, Phan Thuan Do*, Thanh Thao Phan and Duc Nghia Nguyen
Hanoi University of Science and Technology, Hanoi, Vietnam
E-mail: dungnv@soict.hust.edu.vn, long.tv166391@sis.hust.edu.vn,
{thuan.dophan, thao.phanthanh}@hust.edu.vn, *in the memory of Professor Duc Nghia Nguyen*

Keywords: assembly line balancing, simulated annealing, meta-heuristic

Received: March 15, 2020

In the garment industry, assembly line balancing is one of the most significant tasks. To make a product, a manufacturing technique called assembly line is utilized, where components are assembled and transferred from workstation to workstation until the final assembly is finished. Assembly line should always be as balanced as possible in order to maximize efficiency. Different types of assembly line balancing problems were introduced along with many proposed solutions. In this paper, we focus on an assembly line balancing problem where the upper bound of the number of workers is given, tasks and workers have to be grouped into workstations so that the cycle time is minimized, the total number of workers is minimized and balance efficiency is maximized. With unfixed number of workstations and many other constraints, our problem is claimed to be novel. We propose three different approaches: exhaustive search, simulated annealing and simulated annealing with greedy. Computational results affirmed that our simulated annealing algorithm performed extremely good in terms of both accuracy and running time. From these positive outcomes, our algorithms clearly show their applicability potential in practice.

Povzetek: Problem uravnoteženja proizvodne poti je predstavljen odprto, brez omejitev npr. števila delavcev, zato je izviren. Avtorji testirajo več algoritmov in predlagajo najboljšega.

1 Introduction

The assembly line consists of a set of workstations arranged along a material transport system. Parts of the clothes are assembled on workstations, which are transported from workstation to workstation in one direction until the final product is completed. Workers perform a number of tasks in each workstation to create the product. The assembly line rhythm, which is called the cycle time, is the average time for each workstation to complete its tasks before transferring the product's subassembly to the next workstation. After that, the workstations receive new subassembles and repeat the assigned work.

In the fashion industry, patterns of clothing are ever-changing, each product has a set of tasks to assemble the details of clothes together. Each task has unique time to be performed on a specific type of machine or done manually. Among tasks there are precedence relationships, where some tasks must be performed before some other tasks to create a certain clothing product. Different products will have different numbers of tasks and the precedence relationships among tasks will be different. The order of execution between tasks is a directed graph with no cycles.

The assembly line balancing problem (ALBP) is to assign tasks to workstations so that one or more objectives are optimized while ensuring that the precedence relationship among tasks are satisfied. Bryton [3] was the first to propose the ALBP in 1954, while Salveson [24] first published it in a mathematical form in 1955.

There are many ways to classify an ALBP. It is classified into simple and generalized problems [2, 26, 19]. The simple assembly line balancing problem (SALBP) is for straight single, dedicated lines, the duration of the task is determined and the only goal is to optimize line efficiency. SALBP according to the research objectives has been classified by Scholl and Becker [26] into 4 categories: SALBP-1, SALBP-2, SALBP-E, SALBP-F. SALBP-1 objective is minimizing the number of workstations given the cycle time [27]. SALBP-2 goal is to minimize the cycle time given the number of workstations [21]. SALBP-E relates to maximizing line efficiency while simultaneously considering the relationship between the number of workstations and the cycle time [11]. SALBP-F is a feasibility study to

* Corresponding authors

determine if there is a balanced assembly line when fixing both the number of workstations and the cycle time. However researchers usually want to challenge with more realistic problems than SALBP. These problems consider relevant factors including equipment selection, parallel workstations, mixed-model production, processing alternatives, etc. All of them form a very large class of problems called generalized ALBP (GALBP) [1].

The ALBP is usually based on a number of assumptions such as: only one type of product is produced on the line, the processing time of each task is determined and given, the processing sequence of tasks is subject to priority constraints (precedence relations), each task is only assigned to one worker, the maximum processing time of a task is less than the cycle time, etc. [12, 1]. When we assume that the maximum processing time of a task is less than the cycle time, the production speed of the assembly line is limited by that maximum processing time. This problem is solved by creating parallel workstations, which consist of two or more copies of the workstation performing the same group of tasks. The purpose of creating parallel workstations is to speed up production and balance time between workstations. However the number of tasks performed by each worker increases then and they require higher skills from workers. In order to control this process, several studies have set conditions when forming parallel workstations. Sarker and Shanthikumari [25] limited the number of parallel workstations. Buxey [4] limits the number of tasks per worker. McMullen et al. [22] allowed to form parallel workstations so that the processing time of workstations is closest to the cycle time. Vilarinho and Simaria [29] allowed the creation of parallel workstations when the maximum processing time of the tasks did not exceed twice of the cycle time and there is no more than two parallel workstations.

Since an ALBP usually falls into the NP-hard class of combinatorial optimization problems [13], heuristic algorithms are constantly being developed to estimate optimal solutions. Methods such as Hoffmann [15], Helgeson-Birnie [14], Kilbridge-Wester [17] have been applied to ALBP and have certain results, but the solution may fall into the local optimization area. To explore the optimal solution area, meta-heuristic methods were applied. Chiang [7], Lapierre et al. [20] applied the Tabu search method to solve type 1 of SALBP. Ponnambalam et al. [23] applied the multi-target genetic algorithm (GA) to consider different criteria such as number of workstations, line efficiency, the maximum difference between the processing time of workstations and CPU.

ALBP studies in the garment industry have been developed with the goals and constraints of various actual production models. Kayar and Akyalçin [16] have applied a number of heuristic methods to balance the line given the cycle time. Chen et al. [6] used GA to solve the ALBP with the goal of minimizing the number of workstations given the cycle time, while taking into consideration the influence of the skill level of each worker. Eryürük utilized heuristic methods to balance the assembly line in the garment industry in articles [9, 10] and compared the line efficiency of the methods applied when the cycle time was constant.

In this paper, we solve a GALBP in economic production conditions of Dong Van Garment Factory of Hanoi Textile and Garment Corporation, Vietnam. This problem is close to the SALBP of type 2, since its main objective is to minimize the cycle time. In our problem, the upper bound of number of workers (operators) is fixed. The factory mainly produces knitting products, each worker is trained to be able to carry out up to 3 different tasks, which requires not too many skills for workers in the garment industry. We have built a model of ALBP where given the upper bound of number of workers, the minimum cycle time has to be determined while ensuring numerous conditions. Having the minimum cycle time, we also need to determine the minimum number of workers and the maximum balance efficiency achievable. We deal with this GALBP by applying the result of our existing paper [8] which solved another GALBP where the minimum number of workers needs to be determined given the cycle time. Our GALBP has many similarities with the ALBP mentioned in [21], where the minimum cycle time has to be determined given the number of workers and parallel workstations. However, our problem has many different points compared to the one in [21]. While they fixed the number of workers on the assembly line, we allow it to be not greater than a certain number. Moreover, we allow each workstation to complete its tasks in a longer period of time than the cycle time, but not longer than *the upper limit of the cycle time* which will be mentioned later. Specifically, our research contributions are consistent with actual production in the following characteristics:

- The number of workers is not fixed, but the upper bound is given. This constraint is a very new factor, which is much more flexible in real conditions when the factory has a fixed budget for hiring labors or when they do not need to use all of their workers.
- Each workstation is allowed to have up to three workers and perform up to three tasks.
- There are up to two devices under specific constraints in each workstation.
- The processing time R (or cycle time) of each workstation should deviate by approximately $\pm\Delta R$ from the cycle time. Depending on the level of organization of the line, one of the following values is assigned to Δ: 5%, 10% or 15%. This is a very different case compared to other studies. The value $R + \Delta R$ is called *the upper limit of the cycle time*, the cycle time of each workstation must not be greater than this value. The interval $[R - \Delta R, R + \Delta R]$ is called the balanced cycle time interval, and workstations having cycle time within the balanced cycle time interval are called balanced work-

stations. These balanced workstations are the key factor to increase balance efficiency.

Tasks are combined into groups and assigned to workstations with the primary objective of minimizing the cycle time, which means maximizing production speed. The secondary goal is to minimize the total number of workers on the assembly line in order to reduce labor costs and save labor for other jobs in the factory. The tertiary goal is to maximize the proportion of balanced workstations out of all workstations created, thus maximize balance efficiency. In our solutions, binary search is used and proven to be correct to find the optimal cycle time, total number of workers and balance efficiency. Within each iteration of the binary search process, a meta-heuristic method is applied for approximate calculations. We propose three different approaches: exhaustive search, simulated annealing (SA) and simulated annealing with greedy. The proposed algorithms have been evaluated on the actual data set of Dong Van Garment Factory, Hanoi Textile and Garment Joint Stock Corporation, Vietnam. Computational results affirmed that our SA algorithm performed extremely good in terms of both correctness and time. From positive outcomes of this paper, our algorithms clearly show their applicability potential in practice.

2 Problem formulation

2.1 Notations

Throughout the paper, the following notations listed in Table 1 are used.

Table 1: Notation list

$Tasks$	Set of all tasks
M	Number of tasks in $Tasks$
$\hat{N}$	Upper bound of number of workers
N	Total number of workers
t_i	Processing time of task i
R	Cycle time
R'	Upper limit of the cycle time
Δ	Deviation coefficient of cycle time
S_i	Set of all tasks in workstation i
T_i	Total processing time of all tasks in workstation i
n_i	Number of workers in workstation i
R_i	Cycle time of workstation i
k	Total number of workstations
k'	Number of balanced workstations
H	Balance efficiency

2.2 Problem statement

Our GALBP has the primary goal of minimizing the cycle time given the upper bound of number of workers. The secondary goal is minimizing the total number of workers on the assembly line. Then the last goal is determining the maximum balance efficiency. Since its main goal is minimizing the cycle time which is also the objective of SALBP type 2, we denote our problem as GALBP-2. Some particular characteristics of it are described below.

- There is a set $Tasks$ of M tasks. The i^{th} task consumes t_i processing time performed by a machine or by manual work. These M tasks need to be assigned to workstations. Each task will be done in only one workstation.
- There are 3 types of tasks: Type 1 includes tasks using common machines, Type 2 includes tasks using special machines which requires a large investment while having short processing time and Type 3 is manual work.
- On the assembly line of a factory, each workstation can have up to three tasks. In a workstation, if two or three tasks use the same kind of machine, it is counted as one machine only. Machines/manual works assigned into a workstation must match with one of the three cases below:
  + All are manual works.
  + There is exactly one machine and other machines/-manual works, if there are any, are all manual works.
  + There are exactly two special machines (from Type 2 tasks).
- The precedence relations are represented as a directed acyclic graph. This precedence graph is used to determine the execution order of tasks during the assembly process. Some tasks must be done before other tasks. If there is an edge from task u to task v, it means that task u must be done before task v. As a consequence, a task u must be done before task v if there is a path from u to v on the precedence graph. Furthermore, between two different workstations X and Y, if there is a task $u \in X$ and a task $v \in Y$ such that u must be done before v, then there must not exist a task $u' \in X$ and a task $v' \in Y$ such that v' must be done before u', otherwise the product cannot be made.
- Workstations run in parallel.
- In each workstations, all workers do the same task at the same time before moving to another task. Therefore, the processing time (or cycle time) of each workstation i to finish all of its tasks, denoted by R_i, is equal to sum of processing time of all of its tasks (T_i) divide by the number of workers in it (n_i). After all tasks in a workstation are done, the process is operated again with the same set of tasks.
- The total number of workers in all workstations, denoted by N, must not exceed $\hat{N}$.

– The cycle time (or rhythm), denoted by R, is the time limit for each workstation to complete the assigned tasks before transferring the product to another workstation. The sum of all tasks' processing time in a workstation must not exceed $3(R + \Delta R)$. The cycle time R_i of each workstation i must not be greater than $R + \Delta R$, where Δ is given and takes one of the following values: 5%, 10% or 15%.

– If R_i lies within the balanced cycle time interval $[R - \Delta R, R + \Delta R]$, then workstation i is called a ***balanced workstation***.

– Balance efficiency, denoted by H, is the percentage of the number of workstations having their cycle time lies within $[R - \Delta R, R + \Delta R]$.

2.3 Optimization formulation

In addition with the problem statement, here are some induced constraints and formulas which completes the definition of our problem:

GALBP-2 objectives:

Primary: minimize R

Secondary: minimize N

Tertiary: maximize H

Input: $Tasks, \hat{N}, \Delta$ ($\Delta \in \{5\%, 10\%, 15\%\}$)

Definitions:

$$M = |Tasks| = \sum_{i=1}^{k} |S_i| \tag{1}$$

$$N = \sum_{i=1}^{k} n_i \tag{2}$$

$$H = \frac{k'}{k}.100(\%) \tag{3}$$

$$R' = R + \Delta R \tag{4}$$

$$\forall i, 1 \le i \le k : T_i = \sum_{j \in S_i} t_j \tag{5}$$

$$\forall i, 1 \le i \le k : n_i = \begin{cases} 1, & T_i \le R' & \text{(6a)} \\ 2, & R' < T_i \le 2R' & \text{(6b)} \\ 3, & 2R' < T_i \le 3R' & \text{(6c)} \end{cases}$$

$$\forall i, 1 \le i \le k : R_i = \frac{T_i}{n_i} \tag{7}$$

Constraints:

$$\forall i, 1 \le i \le k : |S_i| \le 3 \tag{8}$$

$$\forall i, j, 1 \le i < j \le k : S_i \cap S_j = \emptyset \tag{9}$$

$$\forall i, 1 \le i \le k : T_i \le 3R' \tag{10}$$

$$N \le \hat{N} \tag{11}$$

Constraint (8) ensures a workstation always has no more than 3 tasks. Constraint (9) along with definition (1) guarantees that a task can only be assigned to exactly one workstation. Constraint (10) shows that the total processing time of all tasks in a workstation is always less than or equal to 3 times the upper limit of the cycle time. Constraint (11) shows that the total number of workers cannot be greater than the upper bound of the number of workers which is given as input.

2.4 Examples

As an example, in Table 2 we show technological indexes of a Polo-Shirt product at Dong Van Garment Factory, Hanoi Textile & Garment Joint Stock Corporation, Vietnam (Figure 1). The process of manufacturing such a Polo-Shirt includes 25 tasks. In the table, $t_i(s)$ denotes the processing time of Task i.

The precedence graph of the Polo-Shirt product in Table 2 is shown in Figure 2, along with a sample assignment of tasks into workstations. This sample assignment ensures that the constraints about machines in a workstation and precedence relations are satisfied.

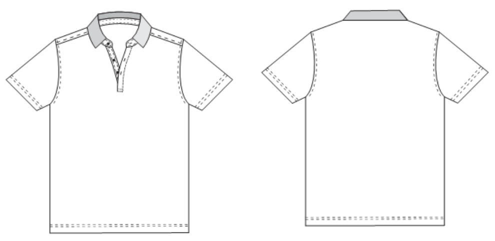


Figure 1: Model of Polo-Shirt.

Table 2: Product technological indexes of Polo-Shirt

No	Task	Machine	Type	t_i(s)
1	Check, mark placket	Check-table	1	32.0
2	Sew placket to front	Lockstitch machine	1	30.0
3	Topstitch placket	Lockstitch machine	1	118.5
4	Trim top of placket	Hand-made	3	12.0
5	Sew collar with collar band	Lockstitch machine	1	59.1

6	Trim bottom edge of collar band	Hand-made	3	12.0
7	Trim bottom edge of collar band	Hand-made	3	10.0
8	Sew shoulder	Overlock machine	1	23.2
9	Topstitch shoulder	1 needle - chainstitch machine	1	11.5
10	Sew collar band with 2 point top of placket	Lockstitch machine	1	27.3
11	Sew collar	Overlock machine	1	32.4
12	Topstitch collar band	Lockstitch machine	1	87.9
13	Attach sleeve set to armhole	Overlock machine	1	43.9
14	Topstitch armhole	1 needle - chainstitch machine	1	24.8
15	Side seam	Overlock machine	1	61.4
16	Hem bottom opening	2 needles - chainstitch machine	1	30.8
17	Hem sleeve opening	2 needles - chainstitch machine	1	41.6
18	Sew bottom of placket	Lockstitch machine	1	15.4
19	Sew bottom opening, sleeve opening	Lockstitch machine	1	23.0
20	Button hole on collar band	Button holing machine	2	9.5
21	Button hole on placket	Button holing machine	2	19.0
22	Button	Button machine	2	38.0
23	Bartack placket	Bartack machine	1	9.5
24	Bartack hem sleeve opening	Bartack machine	1	19.0
25	Trim thread	Hand-made	3	36.0

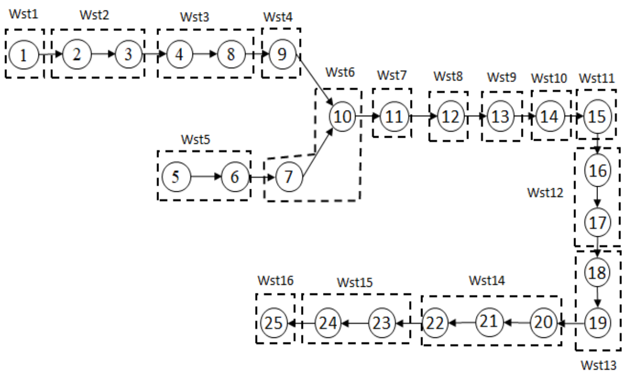


Figure 2: Precedence graph and a sample assignment of tasks into workstations.

3 Solution overview

3.1 Solution outline

To solve the GALBP-2 problem, we simply use binary search method to find the minimum cycle time. Because if there is a solution which consumes no more than $\hat{N}$ workers when $R = x$ then the minimum value of R is certainly not greater than x, and if such a solution does not exist it means that R must be greater than x (the correctness of this argument will be proven in section 3.2). To check for the existence of such a solution, we will have to solve a sub-problem which is also a GALBP: Given the set of all tasks, the values R and Δ, find a way to assign tasks into workstations in order to minimize the total number of workers.

The following **GALBP2** procedure is the framework for our solution. Given the set $Tasks$ of tasks, the upper bound of number of workers $\hat{N}$ and the deviation coefficient of cycle time Δ, **GALBP2** will produce an estimated optimal solution $wstSet$ which is a set of workstations, along with its corresponding values R, N, and H. The return value of **GALBP2** has the form $(R, N, H, wstSet)$. In this procedure, we assume that ϵ is a very small real positive number, α is the maximum processing time of a task in $Tasks$ and β is the sum of processing time of three tasks which have largest processing time in $Tasks$ (if $M \leq 3$ then β is the sum of processing time of all tasks in $Tasks$).

The procedure **GALBP1** inside **GALBP2** solves the sub-problem which is also a GALBP. It takes three parameters: $Tasks$, R and Δ. It generates a solution $wstSet$, which is set of workstations that first minimizes the total number of workers N and then maximizes the balance efficiency H. The return value of **GALBP1** is $(N, H, wstSet)$. This sub-problem is denoted as GALBP-1 because its primary objective is minimizing the total number of workers, close to the SALBP-1 which has the goal of minimizing the number of workstations where each workstation consists of only one worker. Since GALBP-1 is an NP-hard problem, **GALBP1** can only be approximately calculated. Therefore, several meta-heuristic methods are deployed in section 4 to increase the accuracy of

Procedure 1 Solve GALBP-2

Require: $Tasks$: set of all tasks,
$\hat{N}$: upper bound of total no. of workers,
Δ: deviation coefficient of the cycle time.

```
1: procedure GALBP2($Tasks$, $\hat{N}$, $\Delta$)
2:     $lowR \leftarrow \frac{\alpha}{3(1+\Delta)}$                         ▷ min valid $R$
3:     $upR \leftarrow \frac{\beta}{1-\Delta}$                              ▷ max effective $R$
4:     while $upR - lowR > \epsilon$ do
5:         $R \leftarrow \frac{upR+lowR}{2}$
6:         $(N, H, wstSet) \leftarrow$ GALBP1$(Tasks, R, \Delta)$
7:         if $N > \hat{N}$ then
8:             $lowR \leftarrow R$
9:         else
10:            $upR \leftarrow R$
11:    $(N, H, wstSet) \leftarrow$ GALBP1$(Tasks, upR, \Delta)$
12:    if $N > \hat{N}$ then                                               ▷ no solution
13:        return $(\infty, \infty, 0\%, NULL)$
14:    else
15:        return $(upR, N, H, wstSet)$
```

this procedure.

3.2 Binary search correctness

Recall that in section 3.1, we have stated that if there is a solution which consumes no more than $\hat{N}$ workers when $R = x$ then the minimum value of R is certainly not greater than x, and if such a solution does not exist it means that R must be greater than x. The correctness of this argument is proven in Lemma 3.1 below.

Lemma 3.1. *Let $Tasks$ be any set of tasks and $\Delta \in \{5\%, 10\%, 15\%\}$. Let R_1, R_2 such that $\frac{\alpha}{3(1+\Delta)} \leq R_1 < R_2$, where α is the maximum processing time of a task in $Tasks$. Assume that procedure* $\mathbf{GALBP1}$ *can always produce an accurate result, if we set $(N_1, H_1, wstSet_1) = \mathbf{GALBP1}(Tasks, R_1, \Delta)$ and $(N_2, H_2, wstSet_2) = \mathbf{GALBP1}(Tasks, R_2, \Delta)$, then $N_1 \geq N_2$.*

Proof. First we need to show that for all $R \geq \frac{\alpha}{3(1+\Delta)}$, a valid solution for **GALBP1** always exists. Indeed, a solution where each workstation contains exactly one task would fit all the constraints mentioned in the problem statement.

Then, we consider an interesting observation here: $wstSet_1$ is also a solution when $R = R_2$ since all mentioned constraints are still satisfied. Moreover, if $R = R_2$, solution $wstSet_1$ will consumes not as many workers as itself when $R = R_1$, because of the way we calculate the number of workers in each workstations. Assume that when $R = R_2$, $wstSet_1$ consumes N workers, then we have $N_2 \leq N \leq N_1$ which is what we want to prove. □

Actually, when $R < \frac{\alpha}{3(1+\Delta)}$, there will be no solution. Because there exists at least one workstation i which has $T_i > 3(R + \Delta R)$, contradict with problem statement. Therefore setting $lowR = \frac{\alpha}{3(1+\Delta)}$ at the beginning of procedure **GALBP2** is indeed appropriate. Moreover when $R > \frac{\beta}{1-\Delta}$, the minimum number of workers stops to decrease further, so initializing $upR = \frac{\beta}{1-\Delta}$ is suitable too.

4 Methods to estimate the procedure GALBP1

With the application of **GALBP2** procedure, our original GALBP-2 is turned into solving another GALBP-1 in procedure **GALBP1**. GALBP-1 is very similar to the original problem GALBP-2, with all the constraints remain the same except that the number of workers is not bound anymore.

Since the GALBP-1 in procedure **GALBP1** is an NP-hard problem, it cannot be fully solved in polynomial time. Therefore, we tried to apply exhaustive search (brute-force search) along with different meta-heuristic methods such as simulated annealing (SA for short) and SA with greedy to produce answers as close as possible to the optimal ones. For a similar version of this GALBP-1 where H must not be less than 80%, we have already proposed an efficient SA algorithm [8] which performs excellently in terms of accuracy and speed. Therefore, with some reasonable modifications, we could expect our same methods to work well in this GALBP-1.

Throughout section 4, we introduce about our approaches in detail to cope with this GALBP-1. The following section 5 will contain a full evaluation of all methods when being applied to solve our original GALBP-2 based on experimental results on real data of the garment industry.

4.1 Exhaustive search

The exhaustive search finds the optimal result by considering all possible solutions. We design a simple exhaustive search algorithm for this GALBP-1 in the procedure 2.

In this procedure, wst is the current built workstation which consists of tasks, $curSol$ is the current solution which is a set of workstations and $bestSol$ is the current best solution. By initializing $bestSol$ as a random valid solution and calling $\mathbf{exhaustive}(1, 1, \emptyset, \emptyset)$, we will have $bestSol$ as our optimal solution when $\mathbf{exhaustive}$ terminates.

For our Polo-Shirt example, when the number of tasks is not too large, the exhaustive search can still return an optimal solution after a reasonable time. Nonetheless, when input is big enough, it takes hours to run until termination, which is infeasible in industrial environment. Therefore, better approaches should be applied to deal with this problem.

Procedure 2 Exhaustive search for GALBP-1

```
Require: $i$: $1^{st}$ task in current workstation,
         $j$: last added task in current workstation,
         $wst$: current workstation,
         $curSol$: current solution.

 1: procedure exhaustive($i$, $j$, $wst$, $curSol$)
 2:     if $i > M$ then
 3:         if $curSol$ is better than $bestSol$ then
 4:             $bestSol \leftarrow curSol$
 5:     else if $wst = \emptyset$ then
 6:         if $task_i$ is marked then
 7:             exhaustive($i+1, i+1, \emptyset, curSol$)
 8:         else
 9:             Push $task_i$ into $wst$
10:             exhaustive($i, i, wst, curSol$)
11:             Pop $task_i$ out of $wst$
12:     else
13:         if $wst$ is valid then
14:             Mark all tasks in $wst$
15:             Push $wst$ into $curSol$
16:             exhaustive($i+1, i+1, \emptyset, curSol$)
17:             Pop $wst$ out of $curSol$
18:             Unmark all tasks in $wst$
19:         if $|wst| < 3$ then
20:             for $k \leftarrow j+1$ to $M$ do
21:                 if $task_k$ is not marked then
22:                     Push $task_k$ into $wst$
23:                     exhaustive($i, k, wst, curSol$)
24:                     Pop $task_k$ out of $wst$
```

4.2 Simulated annealing

SA algorithm has been widely applied due to its feasibility in NP-hard problem classes through a randomized controlled process with reasonable calculation time. Therefore, the SA algorithm is a good tool for ALBP with a lot of constraints.

4.2.1 Motivation and idea

Simulated annealing (SA for short) was first applied to optimization problems by S. Kirkpatrick et al. [18] and V. Cerny [5]. In the book "Metaheuristics: From design to implementation" of El-Ghazali Talbi [28], the author described almost every aspect of SA in detail. It is a metaheuristic to approximate optimal solution in a large search space for an optimization problem. The idea of SA algorithm is derived from physical metallurgy. The metal is heated to high temperatures and cooled slowly so that it crystallizes in a low energy configuration.

SA is chosen to solve this ABLP because of its simplicity and efficiency. It allows for a more extensive search for the global optimal solution, and can even find a global optimal solution if it runs for enough amount of time.

We represent our SA approach in Procedure 3. This Procedure is a close edition of the general SA algorithm from Talbi's book [28].

Procedure 3 Simulated Annealing

```
Require: $s_0$: initial solution,
         $T_{max}$: starting temperature,
         $L$: neighbor generation loop time limit,
         $T_{dec}$: temperature drops after each step,
         $P$: probability to accept worse solution.

 1: procedure SA($s_0, T_{max}, L, T_{dec}, P$)
 2:     $s \leftarrow s_0$
 3:     $T \leftarrow T_{max}$
 4:     while $T > 0$ do
 5:         for $i \leftarrow 1$ to $L$ do
 6:             Generate a random neighbor $s'$
 7:             if $s'$ is better than $s$ then
 8:                 $s \leftarrow s'$
 9:             else
10:                 Assign $s \leftarrow s'$, probability $P(T)$
11:         $T \leftarrow T - T_{dec}$
12:     return Best solution found
```

There are five parameters that we need to decide for SA: s_0 as the initial solution; starting temperature T_{max}, L and T_{dec} for cooling schedule; and P as the acceptance probability of moving to a worse solution. Also we need to design a procedure to generate a random neighbor s' from a current solution s. All these factors will affect the quality of our algorithm.

4.2.2 Initial solution

In theory, the initial solution s_0 can be any valid solution and it does not affect the quality of SA. However, when the solution searching space is too large, a good initial solution can be a suitable approximation for the global optimum in a short amount of time. In section 4.2 we set a random solution as the initial solution for SA, and in section 4.3 we will assign a solution obtained from a greedy method to s_0. Result comparison between these two approaches shows a remarkable efficiency difference.

4.2.3 Neighbor generation

A neighbor of a solution s is generated simply by moving a task from a workstation to another workstation (including creating a new workstation consist of only that task) or swapping two tasks in two different workstations. There are at most M^2 valid neighbors of a solution.

Among all valid neighbors of s, we just consider its χ best neighbors and randomly choose one of them. The reason why we do not choose among all valid neighbors is to save computation cost without worsening the algorithm efficiency too much.

χ is set high at the beginning and decreases as the temperature decreases, so that when temperature is high a wide range of neighbor is considered and at the end only better solutions are chosen.

4.2.4 Move acceptance

Usually, the probability P that a worse solution is accepted depends on the current temperature T, the current solution s and the new solution s'. One of the most basic forms of P [28] can be written as:

$$P(T, s, s') = e^{-\frac{f(s')-f(s)}{T}} = e^{-\frac{\Delta E}{T}}$$

In which $\Delta E = f(s') - f(s)$ is the different of quality between the new and current solution. However in our SA algorithm, P depends only on T by a simple formula:

$$P(T) = \frac{T}{T_{max}}$$

ΔE is not used in our case since the quality of s and its chosen neighbor s' are not too different, they are even very close. Because s' is generated from s by just moving a task from a workstation to another or swapping two tasks in two workstations, and also s' is chosen among χ best neighbors of s. Therefore, ΔE tends to be very small and negligible. Also, it is very hard to find an ideal formula for calculating the quality of a solution. Any tuned formula for a solution's quality is just overfit to some set of tests and performs badly in other tests.

Computational results show that $P(T)$ works well compared to any tuned version of $P(T, s, s')$ that we design. Moreover, in our case $P(T)$ formula is much simpler and more reasonable.

4.2.5 Cooling schedule

In theory, the higher T_{max} and L are the higher chance for optimal solution to be discoverable. Similarly, the lower T_{dec} is, the better our final solution will be. However, to save computation energy, these three parameters should be carefully tuned.

4.2.6 Multiple execution

Since the solution search space for this GALBP-1 is very large, it is not guaranteed that when SA is applied on a unique input, a unique output will be produced. Therefore, given an input, SA algorithm will be repeated multiple times to provide multiple answers, then the best answer among them will be the solution for **GALBP1** procedure. By experimenting on actual data, we realize that 10 times of repetition is enough to stabilize our SA algorithm without taking too much of time.

4.3 Simulated annealing with greedy

A good initial solution provided by a greedy approach can always be a suitable approximation for the optimal result in a short amount of time. Also, when the solution search space is too large, it could help SA to find better final solution by focusing the process on a critical region only. With our GALBP-1, our initial solution s_0 for SA is constructed by a 5-step algorithm described below:

* **Step 1**: Choose a task u such that there is not any remaining task $v \neq u$ where v must be done before u is processed.

* **Step 2**: Create a workstation X which contains u and some of the remaining tasks so that X is valid and the following Ws_X value is maximize (Ws here stands for "worker saved"):

$$Ws_X = n'_X - n_X$$

Where n'_X is the total number of workers needed to complete all the tasks in workstation X if we divide these tasks into separated one-task-only workstations. If there are many workstations X with the same value Ws_X, choose any workstation which is balanced.

* **Step 3**: Add X to s_0.

* **Step 4**: Remove all tasks belong to X.

* **Step 5**: If there is some task remaining, go back to Step 1.

At step 2 of this algorithm, a greedy strategy is utilized: the best workstation which contains task u is added to the solution. Such a strategy efficiently exploits a signature property of an assembly line: Its precedence graph is almost identical to a tree with only a few number of branches. Therefore, a workstation tends to consist of connected tasks on the precedence graph, and removing them does not affect our future decisions so much. Indeed, experimental results which will be discussed in section 5 show that the SA with greedy solution's efficiency is usually better than that of exhaustive search and traditional SA, in terms of both accuracy and running time.

5 Computational results

If the exhaustive search procedure were allowed to run fully, it would take several hours or even days until termination which is infeasible in industrial environment. Therefore, for each test, we forced it to terminate when it is called more than 6×10^6 times recursively, and its best produced result is collected. Besides that, for all versions of SA, we set $T_{max} = 100, L = 20$ and $T_{dec} = 5$ to guarantee solution quality without consuming too much time. All algorithms are implemented in C++, and run on a computer which has 2.60GHz i7-8850H CPU (12 CPUs), NVIDIA Quadro P1000, 16GB RAM and 512GB SSD.

Our algorithms were tested on real data set related to the production of Polo-Shirt products at Dong Van Garment Factory, Hanoi Textile & Garment Joint Stock Corporation, Vietnam. There are 12 cases, where 6 tests are created from each of these cases by modifying Δ and $\hat{N}$. The values of Δ and $\hat{N}$ for each test are the combinations of three values of Δ $(5\%, 10\%$ and $15\%)$ and two different values of $\hat{N}$ where $\hat{N}_{high}$ the greater is about twice as $\hat{N}_{low}$ the smaller and $\hat{N}_{low} \leq 1.5M$. $\hat{N}_{high}$ and $\hat{N}_{low}$ are different among cases. Hence there are 72 tests in total. The number of tasks M spreads evenly among tests, from 15 to 60. The performance of each algorithm on all tests in terms of the cycle time R, number of workers N, balance efficiency H

Table 3: Results for tests having $\Delta = 5\%$ and $\hat{N} = \hat{N}_{low}$

M	N̂	R-Ex	R-SA	R-SA-gr	N-Ex	N-SA	N-SA-gr	H-Ex (%)	H-SA (%)	H-SA-gr(%)	T-Ex (s)	T-SA (s)	T-SA-gr(s)
15	18	31.429	31.429	31.429	18	18	18	54.545	54.545	54.545	0.055	10	10
20	18	30.952	31.157	30.953	18	18	18	55.556	55.556	50	37	28	28
25	24	41.857	41.857	41.857	24	24	24	29.412	29.412	29.412	11	57	55
30	28	41.81	41.81	41.81	28	28	28	23.81	23.81	19.048	110	89	90
32	26	84.762	72.621	69.02	26	26	26	23.81	38.889	29.412	843	125	128
33	36	93.016	92.245	91.429	36	36	36	33.333	25	37.5	952	125	127
35	32	42.857	37.871	37.66	32	32	32	28	52.381	44.444	911	155	162
47	33	No	35.05	32.273	No	32	33	No	34.783	54.545	1240	439	432
48	40	87.619	61.903	60.272	40	40	40	14.286	32	40.741	1368	343	347
50	40	47.143	34.822	33.225	40	40	40	11.111	33.333	46.154	1353	439	447
52	52	66.952	53.401	48.857	49	51	51	18.421	32	52	1552	531	531
60	40	No	71.011	59.831	No	38	40	No	32.143	58.333	1625	895	897

and running time in seconds is documented to make diagrams on Figure 3.

The top six diagrams on Figure 3 show the cycle time R obtained from exhaustive search, SA and SA with greedy algorithms, divide by a number R_0 which is calculated as:

$$R_0 = \frac{\sum_{i=1}^{M} t_i}{\hat{N}} \qquad (12)$$

R_0 is used as an approximation for the lower bound of R, since if $\Delta = 0\%$ then R_0 is exactly the lower bound of R and actually Δ is quite small ($\Delta \leq 15\%$) which means the real lower bound of R is not so different from R_0. Therefore R_0 is used to normalize R. Among the top six diagrams, the upper three of them consist of tests having $\hat{N} = \hat{N}_{low}$ and the lower three consist of tests having $\hat{N} = \hat{N}_{high}$. Each column contains a pair of diagrams sharing a particular Δ value ($5\%, 10\%$ or 15%). The same order applies to diagrams of the balance efficiency H and running time.

For example, Table 3 shows results of 12 tests having $\Delta = 5\%$ and $\hat{N} = \hat{N}_{low}$. Here "Ex" is exhaustive search, "SA" is simulated annealing and "SA-gr" is simulated annealing with greedy. These results are used to built the top-left diagram in each set of six diagrams in Figure 3.

Since R_0 is an approximation for the lower bound of R, a value of R is a good answer if it is not so far from R_0. When $\hat{N} = \hat{N}_{low}$, based on Figure 3, we can see that both SA and SA with greedy results are as good as results of exhaustive search in small tests but much better than exhaustive search in medium and large tests. Even in some cases, due to early termination, exhaustive search does not provide any valid solution, as opposed to SA algorithms, which still produces quality answers for all tests. In case of $\hat{N} = \hat{N}_{high}$, the results of R may not be close to R_0 since $\hat{N}_{high} \approx 2\hat{N}_{low}$ can be a bit too high which made R_0 too much lower than the real lower bound of R. Nevertheless, SA algorithms still show that they are always not worse than exhaustive search. In addition, SA with greedy is usually slightly better than traditional SA in terms of R, which reveals the effectiveness of greedy initial solution.

For the balance efficiency H, SA algorithms can be slightly worse than brute force when the number of tasks M is small. However as M grows larger, SA algorithms clearly become superior to the exhaustive one. Moreover, H is usually higher than 40% and often fluctuates from 60% to 80% when SA is utilized which are quite satisfying outcomes. A point worth noting is that SA with greedy is remarkably better than exhaustive search and traditional SA in almost all test cases.

In case of running time, SA algorithms completely outperform exhaustive search as expected since they are polynomial time algorithms while exhaustive search theoretically runs in exponential time. Also, results are produced from SA in less than 20 minutes even for the largest test cases. With its fast processing speed, SA is perfectly suitable for real industrial environment.

With all the evaluation above, we can conclude that SA is an efficient meta-heuristic for our GALBP-2. In addition,

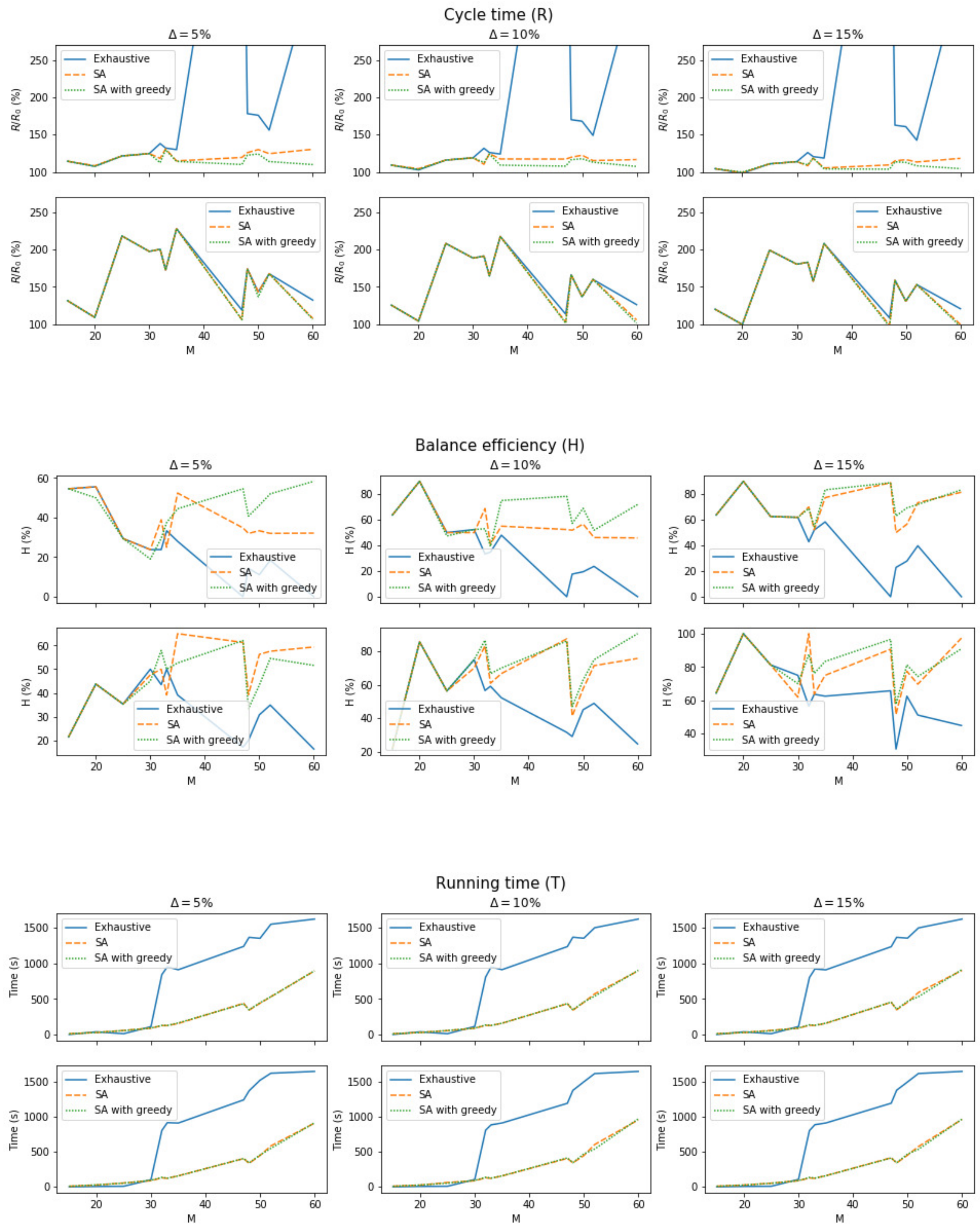


Figure 3: Diagrams of cycle time (R), balance efficiency (H) and running time of exhaustive search, SA and SA with greedy on 72 tests from Dong Van Garment Factory, Hanoi Textile & Garment Joint Stock Corporation, Vietnam.

the SA with greedy version is clearly the most excellent, compared to both exhaustive search and traditional SA.

6 Conclusion

In this paper, we represented a Simulated Annealing algorithm to solve a generalized assembly line balancing problem in the garment industry. Our GALBP-2 has the primary goal of minimizing the cycle time given the upper bound of number of workers. The secondary goal is minimizing the total number of workers on the assembly line. Then the last goal is determining the maximum balance efficiency. We efficiently utilized binary search to turn the original problem into a simpler problem GALBP-1, where the primary objective is minimizing the total number of workers and the secondary goal is maximizing the balance efficiency, given the cycle time. Then we introduced three methods to solve this GALBP-1: exhaustive search, SA and SA with greedy. All of them have their particular advantages in terms of accuracy and running time, depend on different test sizes. These algorithms are good supporting tools for garment factory managers to make plans before decisions. In other real assembly line balancing cases, our mentioned methods should also be considered as promising directions.

Acknowledgments

The authors would like to acknowledge the Dong Van Garment Factory, Hanoi Textile & Garment Joint Stock Corporation, Vietnam for supporting survey, experiment to complete this study.

References

[1] Ilker Baybars. A survey of exact algorithms for the simple assembly line balancing problem. *Management science*, 32(8):909–932, 1986. `https://doi.org/10.1287/mnsc.32.8.909`.

[2] Nils Boysen, Malte Fliedner, and Armin Scholl. A classification of assembly line balancing problems. *European journal of operational research*, 183(2):674–693, 2007. `https://doi.org/10.1016/j.ejor.2006.10.010`.

[3] Benjamin Bryton. *Balancing of a continuous production line*. PhD thesis, Northwestern University, 1954.

[4] GM Buxey. Assembly line balancing with multiple stations. *Management science*, 20(6):1010–1021, 1974. `https://doi.org/10.1287/mnsc.20.6.1010`.

[5] Vladimír Černỳ. Thermodynamical approach to the traveling salesman problem: An efficient simulation algorithm. *Journal of optimization theory and applications*, 45(1):41–51, 1985. `https://doi.org/10.1007/bf00940812`.

[6] James C Chen, Chun-Chieh Chen, Yi-Jhen Lin, CJ Lin, and TY Chen. Assembly line balancing problem of sewing lines in garment industry. In *Proceedings of the 2014 International Conference on Industrial Engineering and Operations Management Bali, Indonesia*, pages 7–9, 2014. `https://doi.org/10.1109/icmlc.2009.5212600`.

[7] Wen-Chyuan Chiang. The application of a tabu search metaheuristic to the assembly line balancing problem. *Annals of Operations Research*, 77:209–227, 1998.

[8] Mai Huong Dinh, Viet Dung Nguyen, Van Long Truong, Phan Thuan Do, Thanh Thao Phan, and Duc Nghia Nguyen. Simulated annealing for the assembly line balancing problem in the garment industry. In *Proceedings of the Tenth International Symposium on Information and Communication Technology*, pages 36–42, 2019. `https://doi.org/10.1145/3368926.3369698`.

[9] Selin Hanife ERYÜRÜK. Clothing assembly line design using simulation and heuristic line balancing techniques. *Journal of Textile & Apparel/Tekstil ve Konfeksiyon*, 22(4), 2012.

[10] SH Eryuruk, F Kalaoglu, and M Baskak. Assembly line balancing in a clothing company. *Fibres & Textiles in Eastern Europe*, 66(1):93–98, 2008.

[11] Rasul Esmaeilbeigi, Bahman Naderi, and Parisa Charkhgard. The type e simple assembly line balancing problem: A mixed integer linear programming formulation. *Computers & Operations Research*, 64:168–177, 2015. `https://doi.org/10.1016/j.cor.2015.05.017`.

[12] Waldemar Grzechca. Assembly line balancing problem with reduced number of workstations. *IFAC Proceedings Volumes*, 47(3):6180–6185, 2014. `https://doi.org/10.3182/20140824-6-za-1003.02530`.

[13] Allan L Gutjahr and George L Nemhauser. An algorithm for the line balancing problem. *Management science*, 11(2):308–315, 1964. `https://doi.org/10.1287/mnsc.11.2.308`.

[14] WB Helgeson and Dunbar P Birnie. Assembly line balancing using the ranked positional weight technique. *Journal of industrial engineering*, 12(6):394–398, 1961.

[15] Thomas R Hoffmann. Assembly line balancing with a precedence matrix. *Management Science*, 9(4):551–562, 1963. `https://doi.org/10.1287/mnsc.9.4.551`.

[16] Mahmut Kayar and Ö C Akyalçin. Applying different heuristic assembly line balancing methods in the apparel industry and their comparison. *Fibres & Textiles in Eastern Europe*, 2014. `https://doi.org/10.5604/12303666.1191438`.

[17] Maurice D Kilbridge and Leon Wester. A heuristic method of assembly line balancing. *Journal of Industrial Engineering*, 12(4):292–298, 1961.

[18] Scott Kirkpatrick, C Daniel Gelatt, and Mario P Vecchi. Optimization by simulated annealing. *science*, 220(4598):671–680, 1983. `https://doi.org/10.1126/science.220.4598.671`.

[19] N Kriengkorakot and N Pianthong. The assembly line balancing problem: Review problem. *J. Ind. Eng*, 6(3):18–25, 1955.

[20] Sophie D Lapierre, Angel Ruiz, and Patrick Soriano. Balancing assembly lines with tabu search. *European journal of operational research*, 168(3):826–837, 2006. `https://doi.org/10.1016/j.ejor.2004.07.031`.

[21] Yuchen Li, Honggang Wang, and Zaoli Yang. Type ii assembly line balancing problem with multi-operators. *Neural Computing and Applications*, 31(1):347–357, 2019. `https://doi.org/10.1007/s00521-018-3834-1`.

[22] Patrick R McMullen and GV Frazier. Using simulated annealing to solve a multiobjective assembly line balancing problem with parallel workstations. *International Journal of Production Research*, 36(10):2717–2741, 1998. `https://doi.org/10.1080/002075498192454`.

[23] SG Ponnambalam, P Aravindan, and G Mogileeswar Naidu. A multi-objective genetic algorithm for solving assembly line balancing problem. *The International Journal of Advanced Manufacturing Technology*, 16(5):341–352, 2000. `https://doi.org/10.1007/s001700050166`.

[24] M. E. Salveson. Induced matchings in intersection graphs. *The Journal of Industrial Engineering*, 6(3):18–25, 1955.

[25] Bhaba R Sarker and JG Shanthikumari. A generalized approach for serial or parallel line balancing. *THE INTERNATIONAL JOURNAL OF PRODUCTION RESEARCH*, 21(1):109–133, 1983. `https://doi.org/10.1080/00207548308942341`.

[26] Armin Scholl and Christian Becker. State-of-the-art exact and heuristic solution procedures for simple assembly line balancing. *European Journal of Operational Research*, 168(3):666–693, 2006. `https://doi.org/10.1016/j.ejor.2004.07.022`.

[27] Yuri N Sotskov, Alexandre Dolgui, Tsung-Chyan Lai, and Aksana Zatsiupa. Enumerations and stability analysis of feasible and optimal line balances for simple assembly lines. *Computers & Industrial Engineering*, 90:241–258, 2015. `https://doi.org/10.1016/j.cie.2015.08.018`.

[28] El-Ghazali Talbi. *Metaheuristics: from design to implementation*, volume 74. John Wiley & Sons, 2009. `https://doi.org/10.1002/9780470496916`.

[29] Pedro M Vilarinho and Ana Sofia Simaria. A two-stage heuristic method for balancing mixed-model assembly lines with parallel workstations. *International Journal of Production Research*, 40(6):1405–1420, 2002. `https://doi.org/10.1080/00207540110116273`.

A Novel Method for Determining Research Groups from Co-Authorship Network and Scientific Fields of Authors

Jan Pisanski
University of Ljubljana, Faculty of Arts
E-mail: Jan.Pisanski@ff.uni-lj.si, http://oddelki.ff.uni-lj.si/biblio/oddelek/osebje/pisanski.html
ORCID 0000-0002-3060-111X

Mark Pisanski

Tomaž Pisanski (corresponding author)
University of Primorska, FAMNIT
E-mail: Tomaz.Pisanski@upr.si, https://en.wikipedia.org/wiki/Tomaz Pisanski,
ORCID 0000-0002-1257-5376

Keywords: co-authorship network, scientific field, maximal spanning tree, induced subnetwork, pruning of networks, pathfinder network, MST, line-cut

Received: March 9, 2020

Large networks not only have a large number of vertices but also have a large number of edges. Although such networks are generally sparse they are usually difficult to visualise, even locally. This paper considers the case where large weights on edges represent similarity between the corresponding end-vertices. We follow two main ideas in this paper. The first one is network pruning, *that is removal of edges that makes the resulting network more manageable while keeping the main characteristic of the original network. The other idea is to partition the network vertex set in such a way that the induced connected components represent groups of network elements that fit together. Furthermore, we assume that the vertices of the network are labeled by* types. *Here we apply our approach to co-authorship network of researchers in Slovenia in order to identify research groups, finding group leaders and the degree of interdisciplinarity of the group. For the network pruning phase we use a MST-pathfinder network and for vertex partition appropriate line-cuts. Each cluster is assigned a distribution of types. In this case, the types correspond to scientific fields, also known as research interests of authors. A measure of interdisciplinarity of research group is derived from such a distribution.*

Povzetek: Velika omrežja nimajo le mnogo vozlišč, ampak imajo tudi mnogo povezav. Čeprav so običajno taka omrežja redka, so nepregledna in jih je težko prikazati na sliki, tudi lokalno. Ta prispevek obravnava omrežja, pri katerih velike vrednosti uteži na povezavah pomenijo podobnost pripadajočih krajišč. V prispevku sledimo dvema glavnima idejama. Prva je kleščenje omrežja, to je odstranitev manj pomembnih povezav, zaradi česar je nastalo omrežje bolj obvladljivo, hkrati pa se ohrani glavna značilnost prvotnega omrežja. Druga ideja je razdeliti vozlišča omrežja tako, da inducirane povezane komponente predstavljajo skupine omrežnih elementov, ki se med seboj prilegajo. Poleg tega predpostavljamo, da so vozlišča omrežja označena s tipi. V tem prispevku uporabljamo naš pristop k omrežju soavtorstev raziskovalcev v Sloveniji z namenom identifikacije raziskovalnih skupin, iskanja voditeljev skupin in stopnje interdisciplinarnosti skupine. Za fazo kleščenja omrežja uporabljamo usmerjevalno omrežje (MST-pathfinder network), za vozliščno razbitje pa ustrezne reze povezav. Vsaki skupini je dodeljena porazdelitev tipov. Mero interdisciplinarnosti raziskovalne skupine izpeljemo iz takšne porazdelitve. V tem primeru tipi predstavljajo znanstvena področja, oz. raziskovalne interese avtorjev.

1 Introduction

In contemporary research community scientists collaborate within formal or informal research groups. Identifying such groups from data available in various bibliometric networks is an interesting challenge. In this note we propose a method that uses the co-authorship network on the one hand and declared scientific field of authors that can be extracted from some bibliographic databases, on the other.

We propose a theoretical model that uses a network, i.e. graph with weights on edges and labels, called types, on its vertices. We may view labels as scientific fields or subfields. Our approach is quite general and can be applied to any weighted network with types. In this paper we apply it to co-authorship networks. Note that scientific fields are sometimes caller research interests.

The method consists of two steps. In the first step the original co-authorship network is pruned in order to reduce the number of edges and increase the number of components, in our case producing research groups. In this step line-cuts are determined. In the second step a collection of induced monotype subnetworks is pruned by applying MST-pathfinder algorithm to further reduce the number of edges while keeping the same connectivity. Our original contribution is combination of both methods and the use of symmetric predicate in the first step; see Algorithm 3. Note that the idea of using MST, pathfinder and MST-pathfinder has been used extensively in the past in variety of contexts of bibliographic and other research[6, 8, 11, 22, 23].

This rough general approach may be refined in several different ways. We present in detail only one such refinement and discuss some others in the conclusion. In general, bibliographic networks are very large and allow for a variety of methods for data mining [15], however in this pilot study we focus our attention on a relatively small data set. The data is restricted to Slovenian researchers and is taken from Slovenian bibliographic system SICRIS. Moreover, only researchers that are co-authors of mathematicians are considered.

2 Pruning of co-authorship network

2.1 Co-authorship network

For basics in graph theory, the reader is referred to [4], for network theory, see for instance [3].

Let V be a list of authors from some bibliographic database. We say that $u, v \in V$ are adjacent: $u \sim v$, if u and v are co-authors of a common work from the corresponding database. Sometimes we restrict our attention to certain types of works or certain types of co-authorships. Usually only scientific works are considered and the co-authorship graph is computed from a two-mode network WA composed of pairs (w, a), works and authors for each co-author a of work w. Since binary relation $\sim$ is irreflexive[1] and symmetric it defines a simple graph $G = (V, \sim)$ that we call the *co-authorship graph*. Let $E = \{uv \in V^2 | u \sim v\}$ denote the set of unordered pairs of adjacent vertices of G. Instead of $G = (V, \sim)$ we may use notation $G = (V, E)$ to denote the same graph. The graph may be weighted where the weights w on the edges represent the number of joint papers between the two authors. In this way a *network* $N = (V, E, w)$ is obtained. Let $w(u, v)$ denote this weight. Sometimes, we may consider the weight of co-authorship differently for different number $n(w)$ of co-authors of work w. Let $W(u, v)$ denote the collection of works co-authored by u and v. For any work w let $n(w)$ denote the number of authors of w. Then

$$w(u, v) = |W(u, v)|.$$

[1]Sometimes one may use also loops at each vertex. The weight associated with a loop may depend on the method that the co-authorship graph is constructed. If it is obtained by multiplication of two-mode networks [4] it represent the number of works for a given author. In the fractional approach it may represent the total contribution of an author. Loops are removed if we follow Newman's approach.

In a fractional approach [2] the weight $f(u, v)$ is defined as:

$$f(u, v) = \sum_{w \in W(u,v)} \frac{2}{n(w)^2}.$$

In case of Newman's normalization the weight is:

$$f(u, v) = \sum_{w \in W(u,v)} \frac{2}{n(w) \cdot (n(w) - 1)}$$

A network N is a weighted graph $N = (V, E, w)$, where $w : E \to \mathbb{R}$ is the *weight function*. In our case it is positive and the value 0 means there is no edge between u and v.

A graph $G = (V, \sim)$ is transformed into the network $N = (V, E, a)$, where $a(u, v) = 1$ for all adjacent pairs of vertices $u \sim v$. The same bibliographic database can produce at least three types of networks for the weight functions a, w, f, defined above:

1. (V, E, a), the *binary case*,
2. (V, E, w) the *standard case*, and
3. (V, E, f), the *fractional case*.

Algorithm 4 Prune the network $N = (V, E, w)$, $n = |V|, m = |E|$

```
 1: Partition the edge set $E$ into subsets $E_i$ with equal weights: $E_i = \{e \in E | w(e) = w_i\}$.
 2: Order the parts in descending order of weights $w_1 > w_2 > \ldots > w_k$
 3: for $u \in V$ do
 4:     $S_u = \{u\}$
 5: $F = \emptyset$
 6: for $i = 1, 2, \ldots$,k do
 7:     $F_i = \emptyset$
 8:     for $e = uv \in E_i$ do
 9:         Let $S_u, S_v$ be the corresponding sets.
10:         if $S_u \neq S_v$ then
11:             Append $e$ to $F_i$.
12:     for $e = uv \in F_i$ do
13:         if $S_u \neq S_v$ then
14:             $S_u = S_v = S_u \cup S_v$
15:     Extend $F$ by $F_i$
16: return subnetwork $Pr(N) = (V, F, w)$.
```

There is another aspect that we have not considered in this paper. Namely, the weight of an edge $e = uv$ between two authors u and v may depend also on the total number of papers authored by each of the two authors. In this case we may modify the network to allow loops and define $w^*(u, v) = w(u, v)$ for $u \neq v$ and let $w^*(u) = w^*(u, u)$ denote the total number of papers having u as an author. Note that in general w^* cannot be computed directly from w since we have no information about the single-authored papers. In this case the best way to compute w^* is to multiply WA^T by WA, where WA represents a two-mode network work-author. The theory of two mode networks and their applications to bibliographic data can be found, for instance in [3].

2.2 Pruning networks

In the analysis of large networks, dense networks present a challenge. Usually one tends to partition the set of vertices and investigate the induced networks on such parts. In [3] one may find a variety of concepts that are useful in such analysis, e.g. cuts, islands, etc. Nevertheless, such subnetworks may be dense again and the role of particular vertices is not clearly visible. For this reason we prune the original network $N = (V, E, w)$ by appropriately selecting a subset of important edges $E' \subset E$. If w' denotes the restriction of w on E', the pruned subnetwork $N' = (V, E', w')$ is obtained.

In case of co-authorship networks large weights indicate close collaboration between authors. When considering research groups one may assume strong collaboration within each group. Hence, in such a case a natural approach to pruning would be to remove all edges of lesser weights, while keeping the same connected components. A possible solution is given by the well-known maximum cost spanning tree. More precisely, in case of a disconnected network the resulting graph is a maximum cost spanning forest.

However, the problem with a maximum cost spanning forest is that, in case when several edges have the same weights, the forest may not be unique. We use a Kruskal-like algorithm that produces a unique pruned network. Algorithm 1 is almost identical to the MST-pathfinder algorithm of [14] and produces the *pathfinder network* $Pn(\infty, n-1)$; for discussion and various aspects see also [19, 5, 20].

It is not hard to see, that the following is true:

Theorem 1. *The network $N'(V, F, w)$ is uniquely determined from the original network. If all weights are positive, the connected components are the same as in the original network.*

Theorem 2. *If all weights in $N(V, E, w)$ are distinct, Algorithm 1 produces the (unique) maximum cost spanning forest. On the contrary, if all weights are the same no edge is discarded.*

Moreover, we easily compute the running time of Algorithm 1.

Theorem 3. *The time complexity of the above algorithm is $\mathcal{O}(m \log m)$.*

In fact, the time complexity is the same as for Kruskal's algorithm[10]. The sorting and partitioning takes $\mathcal{O}(m \log m)$ steps. There are two loops, each with $\mathcal{O}(m)$ steps, and the time complexity for the UNION-FIND is of lesser order.

By applying this pruning method strong ties among the nodes remain visible.

3 Line-cuts

For further refining the network $N(V, E, w)$ one may choose a parameter $t > 0$, the threshold, or cut parameter and prune the edges with weights less than t. In this way the network $N_t(V, E_t, w)$ is obtained, where

$$E_t = \{e \in E | w(e) \geq t\}.$$

The choice of parameter t depends on our aims. There are several obvious goals. For instance:

1. We may choose maximal value of t that guarantees at least a prescribed number of connected components, say κ.

2. An alternative is to insist that all components have at most prescribed number of vertices, say ν.

We present the basic pruning algorithm; see Algorithm 2. It produces essentially a *line-cut*, see for instance [3]. The only difference is that we keep isolated vertices.

Algorithm 5 Prune the network $N = (V, E, w)$, given threshold parameter t. Connected components of the resulting network are called *line-cuts*.

```
1: $F = \emptyset$
2: for $e \in E$ do
3:     if $w(e) \geq t$ then
4:         Append $e$ to $F$.
5: return subnetwork $Pr(N, t) = (V, F, w)$.
```

In Python, Algorithm 2 can be reduced to a single statement:

$$F = [e \text{ for } e \in E \text{ if } w(e) \geq t]$$

4 Pruning networks with vertex types

Let us assume we are given a finite number of types, or colors T, a network $N(V, E, w)$ and a mapping $c : V \to T$. The structure $N(V, E, w, T, c)$ will be called a *weighted network with vertex types*. When pruning network with vertex types, a connected component consisting of vertices of a single type will be called *monotype*. Additionally, we will refer to the number of types used in a connected component as its *type number*. The maximum of type numbers of network components is called the type number of the network, In particular, we are interested in networks of low type number, preferably with monotype networks.

Parameters of pruning may be adjusted in such a way that a monotype network is obtained. For networks with vertex types, in addition to the two goals described in Section 3, a third goal may be considered.

- One may insist that all connected components are monotype, or more general that each component has at most δ types (colors).

The following basic algorithm (Algorithm 3) for a given network with types removes all edges that have endpoints of different types, or more generally, when they satisfy a symmetric predicate P.

Algorithm 6 Prune the network with vertex types $N = (V, E, w, T, c)$ with given threshold parameter t and (symmetric) predicate $P : T^2 :\to \{\top, \bot\}$. Connected components of the resulting network are called *monotype line-cuts*.

```
1: $F = \emptyset$
2: **for** $uv = e \in E$ **do**
3:     **if** $P(c(u), c(v))$ and $w(e) \geq t$ **then**
4:         Append $e$ to $F$.
5: **return** subnetwork $Pr(N, t, P) = (V, F, w, T, c)$.
```

As we mentioned above the predicate P usually is true if both endpoints are of the same type. However, other options are possible. Namely we may have a similarity imposed on the predicates and P signifies that two types are sufficiently similar.

We need an algorithm to analyse the network with vertex types; see Algorithm 4. Using these numbers we may select different parameters and re-run this algorithm to reduce the size of the maximal component or alternatively limit the number of different components. We may also insist that all components be composed of a single type.

5 Interdisciplinarity of research groups and leaders of research groups

For a given network with vertex types one may perform basic statistics on it. Namely, one may compute absolute frequencies of types on the vertex set.

$$f(x) = |\{v \in V | c(v) = x\}|$$

$$f_i(x) = |\{v \in V_i | c(v) = x\}|$$

or relative frequencies

$$\phi(x) = f(x)/n$$

$$\phi_i(x) = f_i(x)/n_i$$

where $n = |V|$ and $n_i = |V_i|$.

We consider two measures:

$$r(N) = \max\{\phi(x) | x \in T\}$$

$$s(N) = |\text{supp } \phi| = |\{x \in T | \phi(x) > 0\}|$$

and for each component:

$$r(V_i) = \max\{\phi_i(x) | x \in T\}$$

$$s(V_i) = |\text{supp } \phi_i| = |\{x \in T | \phi_i(x) > 0\}|$$

Both measure the diversity of research interests in a research group. If $r(V_i) < 0.5$ there is no dominant discipline. If $r(V_i) = 1$, the group is totally homogeneous.

Algorithm 7 Analyse network with vertex types $N = (V, E, w, T, c)$.

```
1: Partition $V$ into connected components $V_1, V_2, \ldots, V_k$
2: Let $d = \max\{|V_j|; j = 1, 2, \ldots, k\}$
3: **for** $j = 1, 2, \ldots, k$ **do**
4:     $b(j)$ = number of different types in $V_j$.
5: Let $\gamma = \max\{b(j); j = 1, 2, \ldots, k\}$
6: **return** number of connected components $k$, order of
   maximal connected component $d$, and maximal number
   of types $\gamma$ in any component.
```

One way to define a leader of a research group is to determine the vertex of maximal degree in the corresponding network, or even better the sum of weights of edges to the neighbouring vertices. There are two parameters that we are interested in. Let m be the number of edges of network N and let d be the maximal degree attained at vertex x. Let d' be the second largest degree. Then x can be defined as a leader of the research group, while *dominance* is the quotient d/m and *absolutism* is defined by expression $1 - d'/d$. Note that it would be also interesting to explore the *diversity index* [21] in this context. However, we will address all of these in a future work.

6 Example

The data used in our experiments was taken from COBIS-S/SICRIS [18] for the works indexed in Scopus [17]. Only papers, where at least one author was a mathematician, were considered. Co-authors that were not registered as researchers in Slovenia were not included. Scientific fields alias research interests, used in Slovenia have three levels. On Level 1 we have:

1	Science
2	Engineering
3	Medicine
4	Biotechnology
5	Social Sciences
6	Humanities
7	Interdisciplinary

The next table shows division of Science on Level 2.

1.01	Mathematics
1.02	Physics
1.03	Biology
1.04	Chemistry
1.05	Biochemistry and Molecular Biology
1.06	Geology
1.07	Comp. Intensive Methods and Appl.
1.08	Environment Protection
1.09	Pharmacy

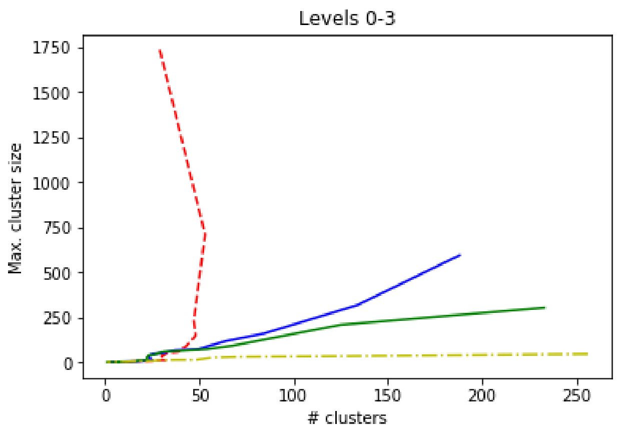


Figure 1: Line-cuts for threshold values $t = 0, 1, 2, \dots$ for four different predicates, each depending on the level $\ell = 0, 1, 2, 3$. Each predicate depends on the interpretation of equality between two research types. Red – $\ell = 0$, blue – $\ell = 1$, green – $\ell = 2$ and yellow – $\ell = 3$.

Finally, the division of Mathematics in the Level 3 is indicated here:

1.01.01	Analysis
1.01.02	Topology
1.01.03	Numerical and Computer Mathematics
1.01.04	Algebra
1.01.05	Graph Theory
1.01.06	Probability and Statistics

Level ℓ may be interpreted as the length of the research interest code that is used to test equality: for $\ell = 0$, the string is not used at all, for $\ell = 1$ only the first characters are compared, for $\ell = 2$, the first four characters are compared, while for $\ell = 3$ all seven characters are compared. Different levels can be associated with the suitable choice of predicate P in Algorithm 3. Let P_ℓ denote the predicate applicable to level ℓ. For instance, for $u = 1.01.01$ and $v = 1.01.04$ $P_2(u, v) = \top$ while $P_3(u, v) = \perp$.

Here we give an example of a pruned research group network. We intend to perform a thorough analysis on more complete data set elsewhere.

Figures 2 and 3 depict the same research group. The network in Figure 3 is tree-like and is obtained from the network of Figure 2 by MST-pathfinder method. Vertex colors denote vertex types: red: 1.01.05, yellow: 1.01.04, blue: 1.01.03 and green 1.01.00. In the database some researchers were assigned research interest at level 2, e.g 1.01 (Mathematics). For consistency, we expanded that to level three as 1.01.00. Note that the research group in Figure 3 is composed of two subgroups, one predominantly interested in graph theory and the other in algebra. There is a central triangle connecting the two subgroups.

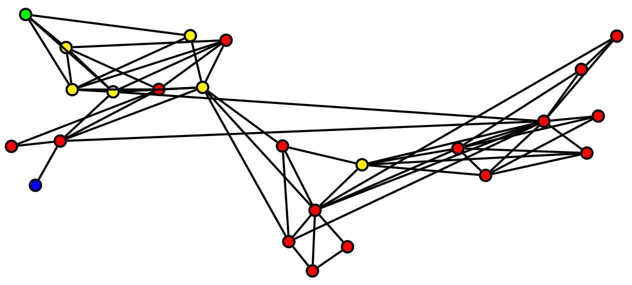


Figure 2: One of several research groups determined by the line-cut for $t = 8$.

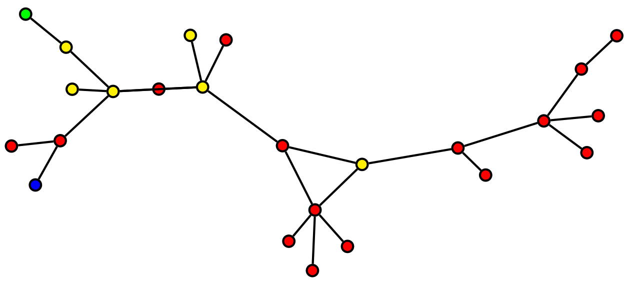


Figure 3: The research group of Figure 2 , pruned by the MST-pathfinder network. Red – Graph Theory, Yellow – Algebra, Blue – Numerical Mathematics, Green – Mathematics

7 Conclusion

Co-authorship graphs and networks are important in the study of research structure and dynamics; see for instance [7, 9, 12, 1]. Their practical value has first been recognised by specialised systems, such as MathSciNet and ZbMath; see [16, 24]. Including them in more general bibliographic systems such as SICRIS [18] would be beneficial for most users. Potential applications are plenty. In this paper we presented only one aspect of such applications. In a recent paper [13] a completely different application is sought, namely, organising talks at a conference in such a way that speakers with similar topics are scheduled at different times.

The data that was available to us has also authors with UNKOWN research interest. In this preliminary study we considered it as a separate research interest. It would be interesting to repeat the study with some flexibility and consider the function: $c : V \rightarrow T \cup \{\text{UNKNOWN}\}$.

Clearly line-cuts refine the vertex partition and apply only within a component. Note that in general one could take different thresholds in different components. In case we intend to have components with given maximal size ν, then indeed different threshold values may be used. In or future more comprehensive work we intend to address some further extensions and applications of the MST-pathfinder method as well as some of the parameters that we have introduced.

Acknowledgement

We would like to thank Vladimir Batagelj for useful advice and fruitful discussion. The work of both referees improved the presentation considerably and is gratefully acknowledged. Work of Jan Pisanski is supported in part by the ARRS grants P5-0361 and J5-8247, while work of Tomaž Pisanski is supported in part by the ARRS grants P1-0294,J1-7051, N1-0032, and J1-9187.

References

[1] T. Bartol, G. Budimir, P. Južnič, K. Stopar. Mapping and classification of agriculture in Web of Science: other subject categories and research fields may benefit. *Scientometrics*, vol. 109 (2016) no. 2, pp. 979-996. https://doi.org/10.1007/s11192-016-2071-6

[2] V. Batagelj. On Fractional Approach to Analysis of Linked Networks, *Scientometrics* (2020). https://doi.org/10.1007/s11192-020-03383-y

[3] V. Batagelj, P. Doreian, A. Ferligoj, N. Kejžar. *Understanding large temporal networks and spatial networks: Exploration, pattern searching, visualization and network evolution*, (2014) John Wiley & Sons.

[4] J.A. Bondy and U.S.R. Murty. *Graph theory*, (2008) Graduate Texts in Mathematics, 244. Springer, New York.

[5] C. Chen. Science mapping: a systematic review of the literature. *Journal of Data and Information Science*, vol. 2 (2017) no.2, pp1-40. https://doi.org/10.1515/jdis-2017-0006

[6] C. Chen, S. Morris. Visualizing evolving networks: Minimum spanning trees versus pathfinder networks. In *IEEE Symposium on Information Visualization 2003* (2003), pp. 67-74. https://doi.org/10.1109/INFVIS.2003.1249010

[7] A. Ferligoj et al. Scientific collaboration dynamics in a national scientific system. *Scientometrics*, vol. 104, (2015), no. 3, pp. 985–1012. https://doi.org/10.1007/s11192-015-1585-7

[8] M. Gallivan, M. Ahuja. . Co-authorship, homophily, and scholarly influence in information systems research. *Journal of the Association for Information Systems*, vol. 16 (2015) no. 12, 2. https://doi.org/10.17705/1jais.00416

[9] L. Kronegger, F. Mali, A. Ferligoj, and P. Doreian. Collaboration structures in Slovenian scientific communities. *Scientometrics*, vol. 90 (2012), no.2, pp. 631–647. https://doi.org/10.1007/s11192-011-0493-8

[10] JB. Kruskal. On the shortest spanning subtree of a graph and the traveling salesman problem. *Proceedings of the American Mathematical Society.* vol.7 (1956) no.1, pp. 48–50. https://doi.org/10.1090/S0002-9939-1956-0078686-7

[11] T. Jacobsen, RL. Punzalan, ML. Hedstrom. Invoking "collective memory": Mapping the emergence of a concept in archival science. *Archival Science*, vol. 13(2013)no. 2-3, pp. 217-251.

[12] S. Pečlin, P. Južnič, R. Blagus, MČ. Sajko, J. Stare. Effects of international collaboration and status of journal on impact of papers. *Scientometrics*, vol. 93 (2012) no. 3, pp. 937-948. https://doi.org/10.1007/s11192-012-0768-8

[13] J. Pisanski, T. Pisanski. The use of collaboration distance in scheduling conference talks. *Informatica : an international journal of computing and informatics,* vol. 43 (2019) no. 4, pp. 461–466, https://doi.org/10.31449/inf.v43i4.2832.

[14] A. Quirin O. Cordón, V. P. Guerrero–Bote, B. Vargas–Quesada, F. Moya–Anegón. A quick MST-based algorithm to obtain Pathfinder networks $(\infty, n-1)$, *Journal of the American Society for Information Science and Technology* vol. 59 (2008) no. 12, pp.1912–1924. https://doi.org/10.1002/asi.20904

[15] J. Leskovec, A. Rajaraman, and J. Ullman. *Mining of Massive Datasets* (2014), Cambridge University Press. https://doi.org/10.1017/CBO9781139924801

[16] MathSciNet:
https://mathscinet.ams.org/mathscinet/index.html

[17] Scopus:
https://www.scopus.com/home.uri

[18] SICRIS:
https://www.sicris.si/public/jqm/cris.aspx?lang=eng

[19] A. Vavpetič, V. Batagelj, V. Podpečan. An implementation of the Pathfinder algorithm for sparse networks and its application on text network, In M. Bohanec (ed.),*12th International Multiconference Information Society,* vol. A (2009) pp. 236–239.

[20] HD. White. Pathfinder networks and author cocitation analysis: A remapping of paradigmatic information scientists. *Journal of the American Society for Information Science and Technology,* vol. 54 (2003) no. 5, pp. 423-434. https://doi.org/10.1002/asi.10228

[21] Diversity index, Wikipedia, https://en.wikipedia.org/wiki/Diversity_index

[22] H. Yang, HJ. Lee. Research trend visualization by MeSH terms from Pubmed. *International journal of environmental research and public health*, vol. 15 (2018) no. 6, 1113. https://doi.org/10.3390/ijerph15061113

[23] SY. Yu. Detecting collaboration patterns among iSchools by linking scholarly communication to social networking at the macro and micro levels. *LIBRES: Library and Information Science Research Electronic Journal* vol. 23 (2013) no. 2, pp. 1–13.

[24] zbMATH:
https://zbmath.org/

Dialogue Act-Based Expressive Speech Synthesis in Limited Domain for the Czech Language

Martin Grůber, Jindřich Matoušek, Zdeněk Hanzlíček and Daniel Tihelka
University of West Bohemia
Faculty of Applied Sciences
NTIS – New Technologies for the Information Society, Department of Cybernetics
Univerzitní 8, Pilsen, Czech Republic
E-mail: gruber@ntis.zcu.cz

Keywords: speech synthesis, unit selection, HMM, expressivity, dialogue act, limited domain

Received: October 30, 2018

This paper deals with expressive speech synthesis in a dialogue. Dialogue acts – discrete expressive categories – are used for expressivity description. The aim of the work is to create a procedure for development of expressive speech synthesis for a dialogue system in a limited domain. The domain is here limited to dialogues between a human and a computer on a given topic of reminiscing about personal photographs. To incorporate expressivity into synthetic speech, modifications of current algorithms used for neutral speech synthesis are made. An expressive speech corpus is recorded, annotated using a predefined set of dialogue acts, and its acoustic analysis is performed. Unit selection and HMM-based methods are used to synthesize expressive speech, and an evaluation using listening tests is presented. The listeners asses two basic aspects of synthetic expressive speech for isolated utterances: speech quality and expressivity perception. The evaluation is also performed for utterances in a dialogue to asses appropriateness of synthetic expressive speech. It can be concluded that synthetic expressive speech is rated positively even though it is of worse quality when comparing with the neutral speech synthesis. However, synthetic expressive speech is able to transmit expressivity to listeners and to improve the naturalness of the synthetic speech.

Povzetek: Razvita je metoda za izrazno govorno sintezo v češčini.

1 Introduction

Nowadays, speech synthesis techniques produce high quality and intelligible speech. However, to use synthetic speech in dialogue systems (ticket booking [1], information on restaurants or hotels [2], flights [3], trains [4] or weather [5]) or in any other human-computer interactive systems (virtual computer companions, computer games), the voice interface should be more friendly to make the user to feel more involved in the interaction or communication. Synthetic speech cannot sound completely natural until it expresses a speaker's attitude. Thus, expressive (or emotional) speech synthesis is a frequently discussed topic and has become a concern of many scientists. Even though some results have already been presented, this task has not been satisfactorily solved yet. Some papers which deal with this problem include, but are not limited to [6, 7, 8, 9, 10, 11, 12].

To reduce the complexity of the general expressive speech synthesis, the task is usually somehow limited (as well as limited domain speech synthesis systems are) and focused on a specific domain, e.g. expressive football announcements [13], sport commentaries [14] or dialogue system in a tourism domain [15]. In this work, we limited the domain to conversations between seniors and a computer. Personal photographs were chosen as the topic for these discussions since the work started as a part of a major project aiming at developing a virtual senior companion with an audiovisual interface [16].

Once the specific limited domain is defined, the task of expressive speech synthesis becomes more easily solvable. However, this work tries to propose a general methodology for designing an expressive speech synthesizer in a limited domain. Thus, it should be possible to create a synthesizer for various limited domains following the procedure described herein.

In the first phase of our research, becoming acquainted with the defined domain was the main goal. Thus, an extensive audiovisual database containing 65 natural dialogues between humans (seniors) and a computer (represented by a 3D virtual avatar) was created using the Wizard-of-Oz method which was proposed in [17] and used e.g. in [18, 19]. Afterwards, the dialogues were manually transcribed so that the text could be used later. The process of the database recording is described in Section 2.

Next, on the basis of these dialogues (the texts and the audio recordings), an expressive speech corpus was designed and recorded. The recording of the expressive corpus was performed in the form of a dialogue between a professional female voice talent and a computer. The dialogues were designed on the basis of the natural dialogues recorded in the previous phase. Thus, the voice talent (acting as the virtual avatar now) was recording predefined sentences as responses to the seniors' speech that the voice talent was listening to. The expressive speech corpus recording process is in more details described in Section 3.

To synthesize expressive speech, an expressivity description has to be defined. Many approaches have been suggested in the past. Continuous descriptions using multidimensional space with several axes to determinate "expressivity position" were described e.g. in [20, 21]. Another option is a discrete division into various groups, for emotions e.g. happiness, sadness, anger, joy, etc. [22]. The discrete description is the most commonly used method and various sets of expressive categories are used, e.g. dialogue acts [23, 15], emotion categories [24, 7, 25] or categories like good news and bad news [8, 26]. Thus, a set of expressive categories was defined and used to annotate the expressive speech corpus. The expressive categories used in our work are presented in Section 4 and annotation of the expressive speech corpus is described in Section 5.

There are various methods to produce synthetic speech, the mostly used are unit selection [27], HMM-based methods [28], DNN-based methods [29] or other methods based on neural networks [30, 31]. These methods can be certainly used also for the expressive speech synthesis. In addition, a method for voice conversion [32] can be taken into consideration. Although this method is primarily used for a conversion of source voice to a target voice in the process of the speech synthesis, it can be also used to convert one speaking style to another [33]. DNN-based approaches then allows e.g. adaptation of an expressive model to a new speaker [34].

To incorporate expressivity into speech using unit selection method, the baseline algorithm used e.g. in [35, 36] was slightly modified. The main modification consists in a different target cost calculation. A prosodic feature representing an expressive category is considered in addition to the current set of features used for the cost calculation. To get specific penalties for speech units labelled with an expressive category different from the requested one, enumerated differences between various expressive categories are used. To compute the penalties, a penalty matrix based on perception and acoustic differences is used. The complex acoustic analysis of the expressive speech corpus along with the unit selection method modifications is described in Section 6.

Even though this work is mainly focused on using the unit selection method for expressive speech synthesis, a brief description of preliminary experiments with HMM-based method is also presented. The HMM-based TTS system settings is described in Section 7.

The results and evaluation are presented in Section 8. The expressivity perception ratio is investigated for natural speech and for synthetic speech generated by both the unit selection based TTS system and the HMM-based TTS system. The synthetic speech quality is also discussed in that section. As the results of this work are to be used in a dialogue system, the suitability of produced expressive synthetic speech is evaluated also directly in dialogues.

2 Natural dialogues

To become acquainted with the limited domain, an extensive audiovisual database[1] of natural dialogues was created using the Wizard-of-Oz method. This means that each dialogue was recorded as a dialogue between a human (senior) and a computer (avatar) which was allegedly controlled only by the human voice. However, the computer was covertly controlled by human operators from another room. Thus, the operators were controlling the course of the dialogue whereas the recorded human subjects thought they are interacting with an independent system based on artificial intelligence. The avatar was using neutral TTS system ARTIC [35] to speak to the human subjects. The recording procedure is described in [37] in more details.

2.1 Recording setup

A soundproof recording room has been established for the recording purposes (the scheme is shown in Figure 1). In the recording room, the human subject faces an LCD screen and two speakers. The speech is recorded by two wireless high-quality head microphones (one for the human subject and one for the computer avatar), and the video is captured by three miniDV cameras. A surveillance web-camera was placed in the room to monitor the situation, especially the senior's state. The only contact between a user and the computer was through speech, there was no keyboard nor mouse on the table.

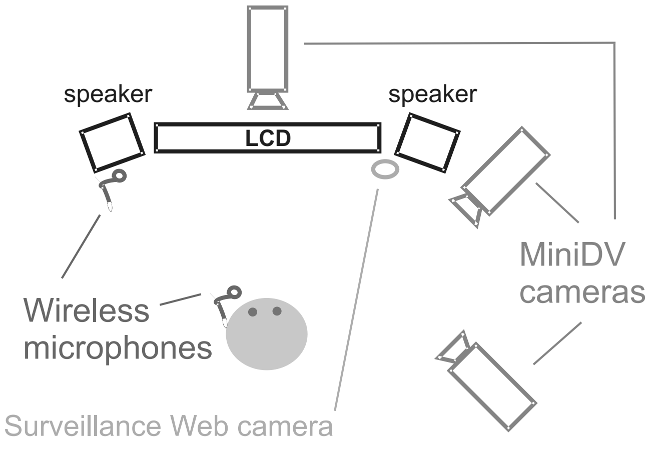


Figure 1: Recording room setup.

A snapshot captured by the miniDV cameras during a recording session is presented in Figure 2. The cameras

[1]The video recordings are not used for the purposes of the expressive TTS system design. They were just archived and are intended for future use in audiovisual speech recognition, emotion detection, gesture recognition, etc.

were positioned to be able to capture the subject from three different views to provide data usable in various ways.

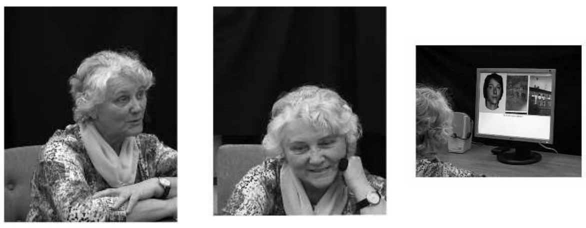


Figure 2: Screenshot captured by the miniDV cameras during a recording session.

2.2 Recording application description

A snapshot of the screen presented to human subjects is shown in Figure 3 (“Presenter” interface). On the left upper part of the LCD screen, there is visualized 3D model of a talking head. This model is used as the avatar, the impersonate companion that should play a role of the partner in the dialogue. Additionally, on the right upper part, there is shown a photograph which is currently being discussed. On the lower half of the screen, there is a place used for displaying subtitles (just in case the synthesized speech is not intelligible sufficiently). The subtitles were displayed only during the first few sessions and then they were switched off as the generated speech turned out to be understandable enough.

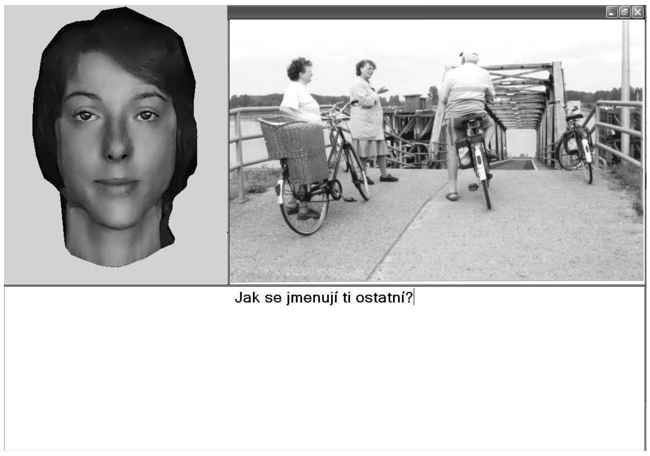
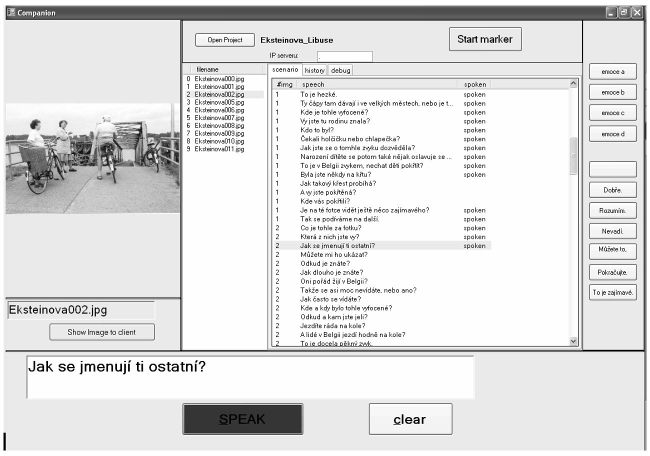

Jak se jmenují ti ostatní?

Figure 3: Snapshot of the WoZ system interface - the user’s side.

In Figure 4, a screen of the operator’s part of the recording application is shown (“Wizard” interface). The interface provides the human operators with possibilities of dialogue flow controlling. The middle part of the screen serves to display the pre-prepared scenario for a dialogue. Note that the wizards could select the sentences from the scenario, the assumption on how the dialogue could develop, by clicking on them. Each sentence of the scenario was given a number related to the picture displayed on the left. This enabled the orientation in large pre-prepared scenarios. Under the picture there is a button for displaying the picture on the “Presenter” screen. Once a sentence is selected by clicking on the list, it appears in the bottom edit box just above the buttons “SPEAK” and “clear”. The displayed sentence can be modified before pressing “SPEAK” button and also an arbitrary text can be typed into the edit box. The right part of the screen is intended for displaying buttons bearing non-speech acts (smile, laughter, assentation, hesitation) and quick phrases (Yes. No. It’s nice. Alright. Doesn’t matter. Go on; etc.).



	filename
0	Eksteinova000.jpg
1	Eksteinova001.jpg
2	Eksteinova002.jpg
3	Eksteinova005.jpg
4	Eksteinova006.jpg
5	Eksteinova007.jpg
6	Eksteinova008.jpg
7	Eksteinova009.jpg
8	Eksteinova010.jpg
9	Eksteinova011.jpg

#img	speech	spoken
1	To je hezké.	spoken
1	Ty čápy tam dávají i ve velkých městech, nebo je t...	spoken
1	Kde je tohle vyfocené?	spoken
1	Vy jste tu rodinu znala?	spoken
1	Kdo to byl?	spoken
1	Čekali holčičku nebo chlapečka?	spoken
1	Jak jste se o tomhle zvyku dozvěděla?	spoken
1	Narození dítěte se potom také nějak oslavuje se ...	spoken
1	To je v Belgii zvykem, nechat děti pokřtít?	spoken
1	Byla jste někdy na křtu?	spoken
1	Jak taková křest probíhá?	
1	A vy jste pokřtěná?	
1	Kde vás pokřtili?	
1	Je na té fotce vidět ještě něco zajímavého?	spoken
1	Tak se podíváme na další.	spoken
2	Co je tohle za fotku?	spoken
2	Která z nich jste vy?	spoken
2	Jak se jmenují ti ostatní?	spoken
2	Můžete mi ho ukázat?	
2	Odkud je znáte?	
2	Jak dlouho je znáte?	
2	Oni pořád žijí v Belgii?	
2	Takže se asi moc nevídáte, nebo ano?	
2	Jak často se vídáte?	
2	Kde a kdy bylo tohle vyfocené?	
2	Odkud a kam jste jeli?	
2	Jezdíte ráda na kole?	
2	A lidé v Belgii jezdí hodně na kole?	
2	To je docela pěkný zvyk.	

Jak se jmenují ti ostatní?

Figure 4: Snapshot of the WoZ system interface - the operator’s side.

2.3 Audiovisual database statistics

Almost all audio recordings are stored using 22kHz sample rate and 16-bit resolution. The first six dialogues were recorded using 48kHz sample rate, later it was reduced to the current level according to requirements of the cooperating team dealing with ASR (automatic speech recognition). The total number of recorded dialogues is 65. Based on gender, the set of speakers can be divided into 37 females and 28 males. Mean age of the speakers is 69.3 years; this number is almost the same for both male and female speakers. The oldest person was a female, 86 years old. The youngest one was also a female, 54 years old. All the recorded subjects were native Czech speakers; two of them (1 male and 1 female) spoke a regional Moravian dialect. This dialect differs from regular Czech language in pronunciation and also a little in vocabulary. Approximately one half of the subjects stated in the after recording form that they have a computer at home. Nevertheless, most of them do not use it very often. Almost all the dialogues were rated as friendly and smooth. And even more, the users were really enjoying reminiscing on their photos, no matter that the partner in the dialogue was an avatar. Duration of each dialogue was limited to 1 hour, as this was the capacity of tapes used in miniDV cameras, resulting in average duration 56 minutes per dialogue. During the conversation, 8 photographs were discussed in average (maximum was 12, minimum 3).

3 Expressive corpus recording

3.1 Texts preparation

For developing a high-quality expressive speech synthesis system, an expressive speech corpus has to be created. Such a corpus can be then merged or just enhanced by a neutral one to create a robust corpus containing neutral speech as well as expressivity while keeping a maximum speech units coverage (phonetic balance). The process of designing texts for the expressive corpus recording is very important. The real natural dialogues and their transcriptions were taken as a basis for such a design. Thus, almost all the texts (more than 7000 sentences) uttered by the computer avatar during the natural dialogues were used. Texts containing unfinished phrases due to e.g. speakers overlapping were omitted. These texts form a set of sentences to be recorded.

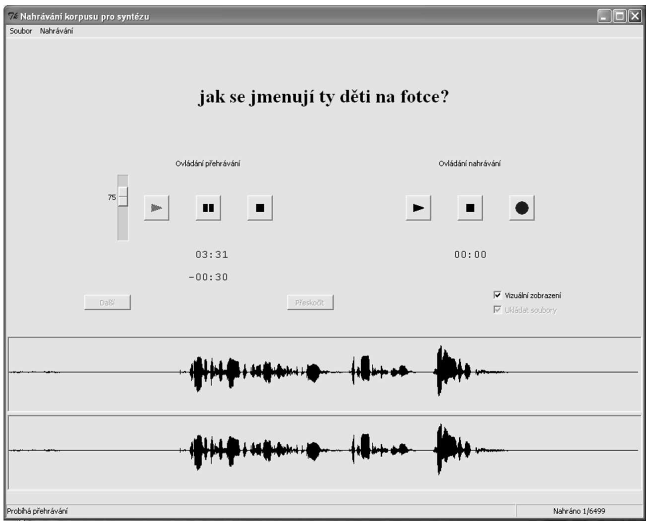
3.2 Recording process

For the expressive corpus recording, a method using so-called scenarios was applied. A scenario in our case can be viewed as a natural dialogue whose course is prepared in advance, just with missing audio of one of the participants (the avatar). This means that the parts of the dialogues to be uttered by a voice talent represent the computer avatar responses and order of these parts is fixed. The parts also follow the natural dialogues and are accompanied with the other participant's original speech to provide the voice talent with information about the context. Actually, the recording was a simulation of the natural dialogues where the voice talent was standing for the computer avatar and was pronouncing its sentences. This should stimulate the voice talent to became naturally expressive while recording.

As the voice talent, a female professional stage-player experienced in speech corpora recording was chosen. The voice talent had already recorded the neutral speech corpus for our neutral TTS system. This corresponds with the intention suggested in Section 3.1 that the expressive corpus should be enhanced by the neutral one to keep the speech units coverage. To improve the performance of tools processing the recorded speech corpora, glottal signal was captured along with the speech signal during the recording.

3.3 Recording application description

To record the expressive corpus using the above described method, a special recording application was developed. The application interface is depicted in Figure 5.


Nahrávání korpusu pro syntézu

Soubor Nahrávání

jak se jmenují ty děti na fotce?

Ovládání přehrávání

75

03:31

-00:30

Ovládání nahrávání

00:00

Další

Přeskočit

[✓] Vizuální zobrazení

[✓] Ukládat soubory

Probíhá přehrávání

Nahráno 1/6499

Figure 5: Interface of the application for expressive corpus recording.

On the upper part of the application window, the text to be recorded is displayed. However, the voice talent was allowed to change the exact sentence wording if unclear[2] while keeping the same meaning. On the middle part, there are, among other things, control buttons for recording and listening. On the bottom, the waveform of the just recorded sentence is shown. The application can be also controlled via keyboard short-cuts to make it more comfortable for the voice talent.

4 Expressivity description

To incorporate expressivity in synthetic speech, some kind of its description is necessary. A general description of expressivity is a very complex task that has not been satisfactorily solved yet even though there are some studies (e.g. [38]) dealing with this topic. For various research fields and their tasks, there are various possibilities of expressivity description. In our work, a description making use of so-called dialogue acts was used. It is a categorical description based on a classification of expressivity into pre-defined classes (used also in [39, 23, 15]).

Although there are several schemas describing expressivity using dialogue acts (including DAMSL [40, 41], SWBD-DAMSL [42], VERBMOBIL [43, 44] or AT&T schema [39]), a new schema was employed to describe expressivity in our limited domain in question. The set of proposed dialogue acts is shown in Table 1 along with a few examples.

The definition of the dialogue acts was based on the audiovisual database of the natural dialogues (described in Section 2) and on the expressive speech corpus (described

[2]Since the texts for the recording were prepared automatically and were not manually checked due to their high number, they could contain some typos, unintelligibilities or unclarities.

dialogue act	*example*
directive	Tell me that. Talk.
request	Let's get back to that later.
wait	Wait a minute. Just a moment.
apology	I'm sorry. Excuse me.
greeting	Hello. Good morning.
goodbye	Goodbye. See you later.
thanks	Thank you. Thanks.
surprise	Do you really have 10 siblings?
sad empathy	I'm sorry to hear that.
	It's really terrible.
happy empathy	It's nice. Great.
	It had to be wonderful.
showing interest	Can you tell me more about it?
confirmation	Yes. Yeah. I see. Well. Hmm.
disconfirmation	No. I don't understand.
encouragement	Well. For example?
	And what about you?
not specified	Do you hear me well?
	My name is Paul.

Table 1: Set of dialogue acts.

in Section 3). These dialogue acts are than used for expressive corpus annotation (Section 5) and also in the process of the expressive speech synthesis (Section 6).

The need for a new dialogue act schema was driven by a definition of our specific limited domain. Most of the dialogue acts are intended to encourage the (human) partner in a dialogue to talk more about a topic while keeping the computer dialogue system to behave more like a patient listener.

Even though the dialogue acts schemas are generally supposed to describe various phases of dialogues, we assume that in various dialogues' phases a speaker can present his state of mind, mood or personal attitude in a specific way. We believe that the proposed set of dialogue acts can be used not only for description of various dialogue phases but that it also represents the speaker's attitude and affective state expressed by expressive speech. Using these dialogue acts in this limited domain, the synthetic speech is supposed to become more natural for the listeners (seniors in this case).

5 Expressive corpus annotation

The expressive speech corpus was annotated by dialogue acts using a listening test. The test was aimed to determine objective annotation on the basis of several subjective annotations as the perception of expressivity is always subjective and may vary depending on a particular listener. Preparation works, listening test framework, evaluation of listening test result and a measure of inter-rater agreement analysis is presented in the following paragraphs.

5.1 Listening test background

The listening test was organized on the client-server basis using a specially developed web application. This way, listeners were able to work on the test from their homes without any contact with the test organizers. The listeners were required to have only an internet connection, any browser installed on their computers and some device for audio playback. Various measures were undertaken to detect possible cheating, carelessness or misunderstandings.

Potential test participants were addressed mostly among university students from all faculties and the finished listening test was financially rewarded (to increase motivation for the listeners). The participants were instructed to listen to the recordings very carefully and subsequently mark dialogue acts that are expressed within the sentence. The number of possibly marked dialogue acts for one utterance was just upon the listeners, they were not limited anyhow. Few sample sentences labelled with dialogue acts were provided and available to the listeners on view at every turn. If any listener marked one utterance with more than one dialogue act, he was also required to specify whether the functions occur in that sentence consecutively or concurrently. If the dialogue acts are marked as consecutive in a particular utterance, this utterance is omitted from further research for now. These sentences should be manually reviewed later and either divided into more shorter sentences or omitted completely.

Finally, 12 listeners successfully finished the listening test. However, this way we obtained subjective annotations that vary across the listeners. To objectively annotate the expressive recordings, a proper combination of the subjective annotations was needed. Therefore an evaluation of the listening test was made.

5.2 Objective annotation

We utilized two ways to deduce the objective annotation.

The first way is a simple majority method. Using this easy and intuitive approach, each sentence is assigned a dialogue act that was marked by the majority of the listeners. In case of less then 50% of all listeners marked any dialogue act, the classification of this sentence is considered as untrustworthy.

The second approach is based on maximum likelihood method. Maximum likelihood estimation is a statistical method used for fitting a statistical model to data and providing estimates for the model's parameters. Under certain conditions, the maximum likelihood estimator is consistent. The consistency means that having a sufficiently large number of observations (annotations in our case), it is possible to find the value of statistical model parameters with arbitrary precision. The parameter calculation is implemented using the EM algorithm [45]. Knowing the model parameters, it is possible to deduce true observation which is called objective annotation. Precision of the estimate is one of the outputs of this model. Using the precision, any untrustworthy assignment of a sentence with a dialogue act can be eliminated.

Comparing these two approaches, 35 out of 7287 classifications were marked as untrustworthy using maximum likelihood method and 571 using simple majority method. The average ratio of listeners who marked the same dialogue act for particular sentence using simple majority approach was 81%, when untrustworthy classifications were excluded. Similar measure for maximum likelihood approach cannot be easily computed as the model parameters and the estimate precision depend on number of iteration in the EM algorithm.

We decided to use the objective annotation obtained by maximum likelihood method. It is an asymptotically consistent, asymptotically normal and asymptotically efficient estimate. This approach was also successfully used in other works regarding speech synthesis research, see [46].

Further, we need to confirm that the listeners marked the sentences with dialogue acts consistently and achieved some measure of agreement. Otherwise the subjective annotations could be considered as accidental or the dialogue acts inappropriately defined and thus the acquired objective annotation would be false. For this purpose, we make use of two statistical measures for assessing the reliability of agreement among listeners.

One of the measures used for such evaluation is Fleiss' kappa [47, 48]. It is a statistical measure for assessing the reliability of agreement between a fixed number of raters when assigning categorical ratings to a number of items or classifying items. We calculated this measure among all listeners separately for each dialogue act. Computation of overall Fleiss' kappa is impossible because the listeners were allowed to mark more than one dialogue act for each sentence. However, the overall value can be evaluated as the mean of Fleiss' kappas of all dialogue acts.

Another measure used here is Cohen's kappa [49, 48]. It is a statistical measure of inter-rater agreement for categorical items and takes into account the agreement occurring by chance as well as Fleiss' kappa. However, Cohen's kappa measures the agreement only between two listeners. We decided to measure the agreement between each listener and the objective annotation obtained by maximum likelihood method. Again, calculation of Cohen's kappa was made for each dialogue act separately. Thus, we can find out whether particular listener was in agreement with the objective annotation for certain dialogue act. Finally, the mean of Cohen's kappas of all dialogue acts can be calculated.

Results of agreement measures are presented in Table 2. Values of Fleiss' and Cohen's kappas vary between 0 and 1, the higher value the better agreement. More detailed interpretation of measure of agreement is e.g in [50].

The Fleiss' kappa mean value of 0.5434 means that the measure of inter-listeners agreement is moderate. As it is obvious from Table 2, dialogue acts *OTHER* and *NOT-SPECIFIED* should be considered as poorly recognizable. It is understandable when taking into consideration their definitions. After eliminating values of these dialogue acts the mean value of 0.6191 is achieved, which means substantial agreement among the listeners.

The Cohen's kappa mean value of 0.6632 means that the measure of agreement between listeners and objective annotation is substantial. Moreover, we can again eliminate dialogue acts *OTHER* and *NOT-SPECIFIED* as they were poorly recognizable also according to Cohen's kappa. Thus, mean value of 0.7316 is achieved. However, it is still classified as a substantial agreement.

As it is shown in Table 2, agreement among listeners regarding classification of consecutive dialogue act was measured too. The listeners agreed on this label moderately among each other and substantially with the objective annotation. There are also shown ratios of the particular dialogue acts occurrence when maximum likelihood method was used for the objective annotation obtaining. It is obvious that dialogue acts *SHOW-INTEREST* and *ENCOURAGE* are the most frequent.

6 Unit selection

6.1 General unit selection approach

In general, a unit selection algorithm (for our system described e.g. in [51]) is used to form resulting synthetic speech from speech units that are selected from a list of corresponding candidate units. These candidates are stored in a unit inventory which is built up on the basis of a speech corpus. The unit selection process usually respects two various groups of candidates' features.

6.1.1 Concatenation cost

Features in one group are used for a concatenation cost computation. This cost reflects continuity distortion, i.e. how smoothly each candidate for unit u_{i-1} will join with each candidate for unit u_i in the sequence. The lower the cost is, the less the unit boundaries are noticeable. In this group of features, there are usually included mostly ordinal values (acoustic and spectral parameters of the speech signal), e.g. some acoustic coefficients, energy values, F0 values, their differences, etc. The concatenation cost for candidate u_i is then calculated as follows:

$$C_i = \frac{\sum_{j=1}^{n} w_j d_j}{\sum_{j=1}^{n} w_j}, \tag{1}$$

where C_i is the concatenation cost of a candidate for unit u_i, n is a number of features under consideration, w_j is a weight of *j-th* feature and d_j is an enumerated difference between corresponding features of two potentially adjacent candidates for units u_{i-1} and u_i — for unit u_i the features from the end of the originally preceding (adjacent in the original corpus) unit are compared with the same features from the end of unit u_{i-1}.

dialogue act	Fleiss's kappa	Measure of agreement	Cohen's kappa	Cohen's kappa SD	Measure of agreement	Occurr. probab.
DIRECTIVE	0.7282	Substantial	0.8457	0.1308	Almost perfect	0.0236
REQUEST	0.5719	Moderate	0.7280	0.1638	Substantial	0.0436
WAIT	0.5304	Moderate	0.7015	0.4190	Substantial	0.0073
APOLOGY	0.6047	Substantial	0.7128	0.2321	Substantial	0.0059
GREETING	0.7835	Substantial	0.8675	0.1287	Almost perfect	0.0137
GOODBYE	0.7408	Substantial	0.7254	0.1365	Substantial	0.0164
THANKS	0.8285	Almost perfect	0.8941	0.1352	Almost perfect	0.0073
SURPRISE	0.2477	Fair	0.4064	0.1518	Moderate	0.0419
SAD-EMPATHY	0.6746	Substantial	0.7663	0.0590	Substantial	0.0344
HAPPY-EMPATHY	0.6525	Substantial	0.7416	0.1637	Substantial	0.0862
SHOW-INTEREST	0.4485	Moderate	0.6315	0.3656	Substantial	0.3488
CONFIRM	0.8444	Almost perfect	0.9148	0.0969	Almost perfect	0.1319
DISCONFIRM	0.4928	Moderate	0.7153	0.1660	Substantial	0.0023
ENCOURAGE	0.3739	Fair	0.5914	0.3670	Moderate	0.2936
NOT-SPECIFIED	0.1495	Slight	0.3295	0.2292	Fair	0.0736
OTHER	0.0220	Slight	0.0391	0.0595	Slight	0.0001
mean	*0.5434*	*Moderate*	*0.6632*		*Substantial*	
consecutive DA	0.5138	Moderate	0.6570	0.2443	Substantial	0.0374

Table 2: Fleiss' and Cohen's kappa and occurrence ratio for various dialogue acts and for the "consecutive DAs" label. For Cohen's kappa, mean value and standard deviation is presented, since Cohen kappa is measured between annotation of each listener and the reference annotation.

6.1.2 Target cost

Features in the other group are used for a target cost computation. This cost reflects the level of an approximation of a target unit by any of the candidates; in other words, how a candidate from the unit inventory fits a corresponding target unit — a theoretical unit whose features are specified on the basis of the sentence to be synthesized. In this group, there are usually included mostly nominal features, e.g. phonetic context, prosodic context, position in word, position in sentence, position in syllable, etc. The target cost for candidate u_i is then calculated as follows:

$$T_i = \frac{\sum_{j=1}^{n} w_j d_j}{\sum_{i=j}^{n} w_j}, \tag{2}$$

where T_i is the target cost of a candidate for unit u_i, n is a number of features under consideration, w_j is a weight of *j-th* feature and d_j is an enumerated difference between *j-th* feature of a candidate for unit u_i and target unit t_i. The differences of particular features (d_j) can be also referred to as penalties.

For our ARTIC TTS system, the features that are considered when calculating the target cost are shown in Table 3.

feature	weight
position in a prosodic word	7.0
left phoneme context	3.0
right phoneme context	3.0
prosodeme type	14.0
voicing – at the beginning	8.5
voicing – at the end	8.5

Table 3: Prosodic features along with their weights used for target cost calculation in the ARTIC TTS system.

6.2 Basic target cost for expressive speech synthesis

When using the expressive speech corpus, the set of the features used for the target cost computation is extended with one more feature. Regarding the aforementioned expressivity description, it is called *dialogue act*. The penalty d_{da} between a candidate u_i of a target unit t_i can be in the easiest way calculated as follows:

$$d_{da} = \begin{cases} 0 & \text{if } da_t = da_c \\ 1 & \text{otherwise} \end{cases}, \tag{3}$$

where d_{da} is a difference (penalty), da_t is a dialogue act of the target unit t_i and da_c is a dialogue act of the candidate u_i.

Finally, a weight for this penalty needs to be set since the target cost is calculated as a weighted sum of particular penalties.

6.3 Advanced target cost for expressive speech synthesis

The target cost calculation presented in equation 3 is very simple and it assumes that penalties for different expressive categories (represented by the dialogue acts) are the same. However, this is not true in most cases. For instance, the difference between *SAD-EMPATHY* and *HAPPY-EMPATHY* should be probably greater than a difference between *SAD-EMPATHY* and *NEUTRAL* — this means that when synthesizing a sentence in the *SAD-EMPATHY* manner and there is no available or suitable candidate labelled with this dialogue act, it is probably better to consider a candidate labelled with *NEUTRAL* dialogue act than considering a candidate labelled as *HAPPY-EMPATHY*. Therefore, it is necessary to enumerate differences between various dialogue acts and use them for the target cost calculation. The basics of the procedure are described in [52], a bit enhanced version is presented here.

6.3.1 General penalty matrix

The differences are assumed to be coded in a penalty matrix **M**, where coefficients m_{ij} represents a difference (a penalty) between a dialogue act *i* and a dialogue act *j*.

To determine coefficients of the matrix, i.e. the differences in dialogue acts, two aspects should be considered: human perception of the speech and acoustic measures calculated from the signal. Thus, two separate matrices are created and then combined. Coefficients of the first matrix **P** are calculated on the basis of a listening test that was performed to annotate the dialogue acts in the expressive speech corpus [37] (see Section 6.3.2). The second matrix **A** is then based on results of an acoustic analysis of expressive speech [53] (see Section 6.3.3). The combined final penalty matrix **M** represents the overall differences (penalties) between various dialogue acts.

6.3.2 Listening test based differences

Given the annotations of the expressive recordings presented in Section 5, a penalty matrix **P** was created. Its coefficients p_{ij} were calculated according to the following equation:

$$p_{ij} = \frac{\text{abs}(\log(\frac{num_{ij}}{max_i}))}{K}, \tag{4}$$

where num_{ij} represents how many times recordings with dialogue act *i* (according to the objective annotation as presented in Section 5.2) were labelled with dialogue act *j* (calculated over all listeners and all recordings), max_i represents the maximum value of num_{ij} for fixed *i* and *K* is a constant defined as $K \geq K_{min}$ where:

$$K_{min} = \max_{\forall i,j}(\text{abs}(\log(\frac{num_{ij}}{max_i})), \tag{5}$$

where $\max_{\forall i,j}$ is the maximum value for all i, j for which the *log* is defined. For situations where the *log* is not defined, the p_{ij} was set as $p_{ij} = K$. In our experiments, the $K = 5 \approx 2 \times K_{min}$. The *log* was used to emphasize differences between calculated ratios and we also assumed that the human perception is logarithmic-based (as suggested e.g. by The Weber-Fechner Law).

6.3.3 Acoustic analysis based differences

An extensive acoustic analysis of the expressive corpus was performed in [53]. On the basis of this analysis, a penalty matrix **A** was created. Its coefficients a_{ij} were calculated as the Euclidean distance between numeric vectors representing the dialogue acts *i* and *j* in a 12-dimensional space. The components of the vector consist of normalized values of 4 statistical characteristics (mean value, standard deviation, skewness, kurtosis) for 3 acoustic parameters (F0 value, RMS energy and unit duration). The acoustic analysis proved that these features can be used as acoustic distance measures for this purpose. It is likely that there other features not considered in this work which may affect the measure in any way and whose influence should be explored in the future.

6.3.4 Final penalty matrix

The final penalty matrix containing numeric differences between various dialogue acts is an appropriate combination of two separate penalty matrices (matrix **P** based on the annotations and matrix **A** based on the acoustic analysis). The coefficients m_{ij} of matrix **M** can be calculated as follows:

$$m_{ij} = \frac{w_p \cdot p_{ij} + w_a \cdot a_{ij}}{w_p + w_a}, \qquad (6)$$

where p_{ij} and a_{ij} represent coefficients from matrices **P** and **A**, w_p and w_a are corresponding weights.

After several experiments, values $w_p = 3$ and $w_a = 1$ were used as the weights. Using this setting, the best results were achieved when subjectively comparing resulting synthetic speech. We also believe that the perceptual part should be emphasized. The final penalty matrix is depicted in Table 4.

6.3.5 Weight tuning for dialogue act feature

Proper setting of a weight for any of the features is not an easy task. Some techniques for automatic settings have also been developed [54, 55]. However, in our system the settings shown in Table 3 is used as it was proved to be appropriate in applications of our TTS.

To set the weight for the dialogue act feature, sets of synthetic utterances were generated for various settings. Using a subjective evaluation (a brief listening test) and considering weights for other features, the final weight was defined as $w_{DA} = 12.0$. When compared with Table 3, this weight is one of the highest among others.

7 HMM algorithm modification/training

Along with the concatenative unit selection method, statistical parametric speech synthesis based on using hidden Markov models (abbreviated as HMM-based speech synthesis) is one of the most researched synthesis methods [28]. Several experiments on using this synthesis method for generating expressive speech are described in [14]. In the HMM approach, statistical models (an extended type of HMMs) are trained from natural speech database. Spectral parameters, fundamental frequency and eventually some excitation parameters are modelled simultaneously by the corresponding multi-stream HMMs.

The variability of speech is modelled by using models with large context description, i.e. individual models are defined for various phonetic, prosodic and linguistic contexts, that are described by so-called contextual factors. The contextual factors employed in our experiments are listed in Table 5. For more details, see e.g. [56].

To increase the robustness of the estimated model parameters, models of acoustically similar units are clustered by a decision-tree-based context-clustering algorithm. As a result, similar units share one common model.

Within the HMM-based speech synthesis, various methods for modelling the expressivity or speaking styles have been introduced. The simplest one uses so-called style dependent models [59], i.e. an independent set of HMMs is trained for each expression. An obvious drawback of this approach is a large amount of training data required for particular expressions.

A better solution are so-called style mixed models [59], where one set of HMMs is trained for all expressions together and particular expressions are distinguished by introducing an additional contextual factor. Then, models of units that are acoustically similar for more expressions are clustered. Independent models are trained only when there is a significant difference between particular expressions.

Another option of modelling expressions are methods based on model adaptation [60, 61]; they are usually preferred because they allow to control the speech style or expression more precisely and require less training data. However, the style mixed model utilizing an additional contextual factor for dialogue act was used in this work.

8 Evaluation & results

This section deals with an evaluation of the procedure described in this paper to verify that it fulfils the goals which were specified at the beginning. Especially, it should be verified that listeners perceive the synthetic speech produced by the developed system as expressive (Section 8.2.1) and also how the quality of synthetic speech changed in comparison with the baseline system (Section 8.2.2). Since the proposed TTS system is focused on a usage in a specified dialogue system, the suitability of the

	APOLOGY	CONFIRM	DIRECTIVE	DISCONFIRM	ENCOURAGE	GOODBYE	GREETING	HAPPY-EMPATHY	NOT-SPECIFIED	OTHER	REQUEST	SAD-EMPATHY	SHOW-INTEREST	SURPRISE	THANKS	WAIT	NEUTRAL
APOLOGY	0.00	0.58	0.39	0.40	0.83	0.25	0.90	0.48	0.36	0.50	0.84	0.17	0.45	0.48	0.83	0.47	0.78
CONFIRM	0.96	0.00	0.71	0.58	0.71	0.95	0.92	0.40	0.48	0.72	0.93	0.41	0.50	0.53	0.74	0.70	0.72
DIRECTIVE	0.90	0.56	0.00	0.58	0.26	0.86	0.46	0.50	0.36	0.65	0.33	0.49	0.38	0.81	0.92	0.59	0.41
DISCONFIRM	0.34	0.33	0.84	0.00	0.43	0.87	0.86	0.28	0.23	0.40	0.44	0.28	0.42	0.42	0.90	0.45	0.54
ENCOURAGE	0.83	0.50	0.48	0.64	0.00	0.83	0.60	0.35	0.31	0.43	0.27	0.39	0.14	0.27	0.75	0.52	0.55
GOODBYE	0.30	0.53	0.34	0.87	0.47	0.00	0.59	0.19	0.17	0.50	0.82	0.25	0.42	0.82	0.25	0.55	0.61
GREETING	0.90	0.63	0.86	0.86	0.58	0.90	0.00	0.52	0.31	0.68	0.89	0.90	0.53	0.87	0.93	0.54	0.42
HAPPY-EMPATHY	0.44	0.25	0.58	0.41	0.24	0.39	0.89	0.00	0.21	0.45	0.54	0.29	0.27	0.28	0.46	0.54	0.58
NOT-SPECIFIED	0.39	0.26	0.25	0.35	0.12	0.19	0.26	0.15	0.00	0.34	0.20	0.22	0.13	0.18	0.32	0.38	0.46
OTHER	0.95	1.00	0.29	0.97	0.94	0.36	0.97	0.96	0.30	0.00	0.94	0.99	0.94	0.95	0.92	0.89	0.84
REQUEST	0.84	0.93	0.30	0.88	0.17	0.82	0.35	0.45	0.31	0.40	0.00	0.51	0.28	0.80	0.87	0.58	0.59
SAD-EMPATHY	0.28	0.28	0.37	0.41	0.26	0.32	0.90	0.33	0.28	0.50	0.49	0.00	0.25	0.33	0.89	0.59	0.67
SHOW-INTEREST	0.88	0.53	0.39	0.62	0.15	0.85	0.87	0.43	0.27	0.58	0.32	0.41	0.00	0.34	0.90	0.57	0.45
SURPRISE	0.86	0.29	0.47	0.40	0.05	0.82	0.87	0.15	0.14	0.37	0.35	0.24	0.06	0.00	0.59	0.51	0.51
THANKS	0.83	0.54	0.92	0.90	0.53	0.46	0.93	0.88	0.91	0.92	0.87	0.89	0.90	0.89	0.00	0.88	0.86
WAIT	0.47	0.58	0.24	0.89	0.32	0.93	0.88	0.55	0.56	0.89	0.29	0.57	0.39	0.90	0.88	0.00	0.63
NEUTRAL	0.78	0.72	0.41	0.54	0.55	0.61	0.42	0.58	0.46	0.84	0.59	0.67	0.45	0.51	0.86	0.63	0.00

Table 4: Final penalty matrix **M**.

Contextual factor	Possible values
Left and right phonetic context	Czech phonetic alphabet [57]
Phone position in prosodic word (forward and backward)	1, 2, 3, 4, 5 ...
Prosodic word position in clause (forward and backward)	
Prosodeme	terminating satisfactorily, terminating unsatisfactorily, non-terminating, null
Dialogue act	see Section 4

Table 5: A list of contextual factors and their values. Prosodic words, clauses and prosodemes are thoroughly described in [58].

expressive speech synthesis in such a dialogue system is also evaluated (Section 8.4).

During the design of the expressive TTS system, it turned out that some of the dialogue acts (further referred to as DAs) appear much more frequently than others, some of them are very rare. Thus, only the most frequent DAs were used to evaluate the system and they were divided into two separate groups:

Expressive dialogue acts:

- *SHOW-INTEREST* – relative frequency 34.9 %;
- *ENCOURAGE* – relative frequency 29.4 %;
- *CONFIRM* – relative frequency 13.2 %;
- *HAPPY-EMPATHY* – relative frequency 8.6 %;
- *SAD-EMPATHY* – it was added because it is considered to be an opposite to *HAPPY-EMPATHY* dialogue act; relative frequency 3.4%;

Neutral dialogue acts:

- *NOT-SPECIFIED* – besides it is one of the most frequently occurring DAs, it should also represents the neutral synthetic speech; relative frequency 7.4 %;
- *NEUTRAL* – this is not a DA per se, it is defined here to represent the neutral speech produced by the current baseline TTS system for the purposes of the evaluation.

All the listening tests described further were performed using the same system as it was used for the expressive corpus annotation (described in Section 5.1). Of course, the questions and options were different within this evaluation but the core of the system is the same. The majority of listening tests participants were experts in speech or language processing, some of them were university students. Texts of synthesized utterances were not a part of the corpora, new texts were created for this purpose. The content of the texts corresponds to the dialogue act to be synthesized (for expressive synthesis), or it is neutral (for neutral synthesis).

8.1 Expressivity perception in natural speech

Before assessing the synthetic expressive speech, a listening test focused on expressivity perception in natural speech was performed. This gives us a brief overview of how the listeners are able to perceive the expressivity and later a comparison between expressivity perception in natural and synthetic speech can be presented.

All the listeners were assessing randomly selected utterances form the natural corpora (neutral and expressive) and their task was to mark if they perceive any kind of expressivity or not or if they are not able to make a decision. 14 listeners participated in this test, each listener was presented with 34 utterances – 4 for each expressive dialogue act being evaluated and 7 for each dialogue act considered as neutral (i.e. *NOT-SPECIFIED* and *NEUTRAL*). The results are depicted in Figure 6 and also shown in Table 6.

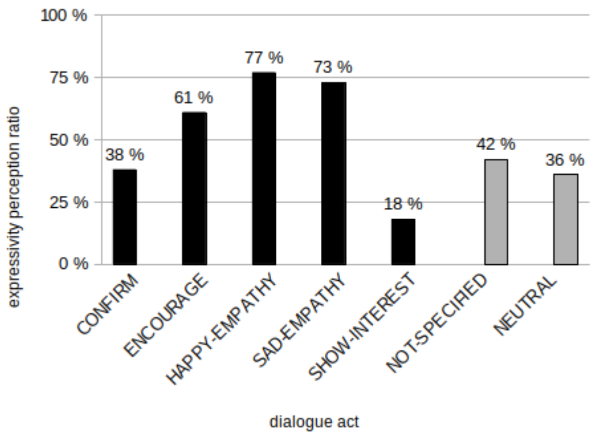


Figure 6: Expressivity perception in natural speech.

dialogue act	expressivity perception ratio	cannot decide
CONFIRM	38 %	3 %
ENCOURAGE	61 %	7 %
HAPPY-EMPATHY	77 %	4 %
SAD-EMPATHY	73 %	6 %
SHOW-INTEREST	18 %	11 %
mean	**53** %	**6** %
NOT-SPECIFIED	42 %	13 %
NEUTRAL	36 %	3 %
mean	**39** %	**7** %

Table 6: Expressivity perception in natural speech.

The results are quite surprising, especially for neutral speech. In 39 % of neutral natural utterances (in average, including *NOT-SPECIFIED*), the listeners perceived an expressivity. It seems that some kind of expressivity is included even in the neutral corpus and the listeners are very sensitive to that, and they are able to perceive it. This fact can be related to the content of speech since as it was described in [62], the content as such might also influence the listeners' expressivity perception.

The results for the expressive DAs depends on a particular DA. For instance, utterances marked as *HAPPY-EMPATHY* and *SAD-EMPATHY* are mostly recognized as expressive whereas utterances marked as *SHOW-INTEREST* are not.

These results give us a baseline for the evaluation of expressive synthetic speech. Since for some DAs the listeners don't perceive expressivity even in natural speech, it's unlikely that they will perceive it in synthetic speech.

8.2 Evaluation of the unit selection based expressive speech synthesis

During the evaluation of expressive synthetic speech, two main factors were investigated – expressivity perception and speech quality. It's supposed that the quality of synthetic speech will be affected by the expressivity integration as the expressive speech is much more dynamic and thus more artificial artifacts may occur. This section deals with the evaluation of expressive synthetic speech produced by the unit selection TTS system. The evaluation of HMM-based TTS system is presented in section 8.3.

In the listening tests regarding expressive synthetic speech evaluation, 13 listeners assessed 30 utterances – 4 for each DA in question and 2 for natural neutral speech (so that a comparison of speech quality can be performed).

8.2.1 Expressivity perception in synthetic speech

The results for expressivity perception in synthetic expressive speech are depicted in Figure 7 and presented in Table 7.

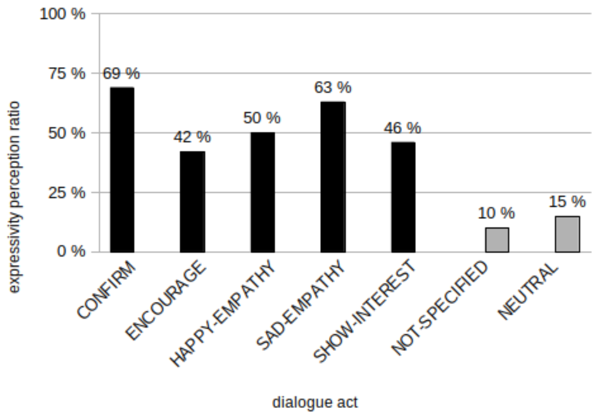


Figure 7: Expressivity perception in synthetic speech (unit selection).

Again, a surprising result can be observed for natural neutral speech as an expressivity was perceived at a quite high ratio (42 %). However, it is consistent with the previous results presented in Table 6 (39 %).

dialogue act	expressivity perception ratio	cannot decide
CONFIRM	69 %	4 %
ENCOURAGE	42 %	8 %
HAPPY-EMPATHY	50 %	10 %
SAD-EMPATHY	63 %	4 %
SHOW-INTEREST	46 %	4 %
mean	**54** %	**6** %
NOT-SPECIFIED	10 %	0 %
NEUTRAL	15 %	0 %
mean	**13** %	**0** %
natural speech (neutral)	42 %	4 %

Table 7: Expressivity perception in synthetic speech (unit selection).

For synthetic speech generated as *NOT-SPECIFIED* and for baseline neutral synthetic speech (marked as *NEUTRAL*), almost no expressivity was perceived. On the other hand, for expressive DAs, the expressivity perception ratio was quite high (mean value 54 %) and it was even slightly higher than for expressive natural speech (mean value 53 %, see Table 6).

To verify that the achieved results are not random, a statistical measure for listeners agreement (the Fleiss' kappa was used here) was calculated. Its value varies in the range $< -1, 1 >$ and a positive value indicates an agreement above the chance level. In our experiment, the Fleiss' kappa was calculated as $\kappa_F = 0.37$ which means a moderate agreement.

In addition, other measures might be used to verify the results; for instance *precision*, *recall*, *F1* and *accuracy* measures which are mostly used for evaluation of classifiers in classification tasks. However, the presented listening test can be also viewed as a classification task where the listeners as classifiers classify into two distinct classes: *perceive* and *do not perceive* expressivity (the *cannot decide* answers were not considered in this verification). The measure are determined as follows:

$$P = \frac{t_p}{p_p}, \quad R = \frac{t_p}{a_p}$$

$$F1 = \frac{2 * P * R}{P + R}, \quad A = \frac{t_p + t_n}{a_p + a_n}$$

where P is *precision*, the ability of a listener not to perceive a neutral sentence as expressive; R is *recall* (also *sensitivity*), the ability of a listener not to perceive expressive sentences as neutral; A is *accuracy*, the ability of a listener to perceive expressivity in expressive sentences and not to perceive it in neutral sentences; $F1$ is the harmonic mean of precision and recall; t_p means “true positives” (i.e. the number of expressive sentences correctly perceived as expressive); t_n means “true negatives” (i.e. the number of neutral sentences correctly perceived as neutral); p_p stands for “predicted positives” (i.e. the number of all sentences perceived as expressive); a_p stands for “actual positives” (i.e. the number of all actual expressive sentences); a_n means “actual negatives” (i.e. the number of all actual neutral sentences).

The calculated values of these measures are presented in Table 8 altogether with values that would be achieved in case the expressivity perception is assessed completely at random.

measure	real listeners	random assessment
precision	0.92	0.72
recall	0.58	0.50
F1 measure	0.71	0.59
accuracy	0.66	0.50

Table 8: Statistical measures for expressivity perception listening test and comparison with completely random assessment.

As the verification indicates, the expressivity perception ratio in synthetic speech is not a result of a random process. It’s necessary to note that there are two main facts which affect the expressivity perception. The first one is the TTS system and the synthetic speech whose evaluation is the main goal. The second fact is the listeners – each of them might perceive (assess) various intensity of various expressivity categories differently. However, the main task here is not to evaluate the listeners and if they are or they are not able to perceive an expressivity (which is basically impossible). The listeners are just believed to and the only thing that can be done is to perform some kind of agreement measure calculation.

In synthetic expressive speech generated with a particular DA in mind, the relative ratio between units originally coming from utterances labelled with this DA and units coming from other utterances can be measured. The ratio might vary depending on setting of the weight for the *dialogue act* feature. The calculated ratios for the current weight settings (as designed in Section 6.3.5) are shown in Figure 8.

It’s worth noting that the measure is very low for *NOT-SPECIFIED* DA. However, after further investigation, it turned out that when synthesizing utterances for this DA, units coming from the neutral corpus (*NEUTRAL*) were mostly selected. It supports the assumption that the *NOT-SPECIFIED* DA represents neutral speech (although in the final penalty matrix **M** the distance between *NOT-SPECIFIED* and *NEUTRAL* was calculated as 0.46 which is quite high). It also seems that there is no strong relation between this measure and the expressivity perception results presented in Table 7.

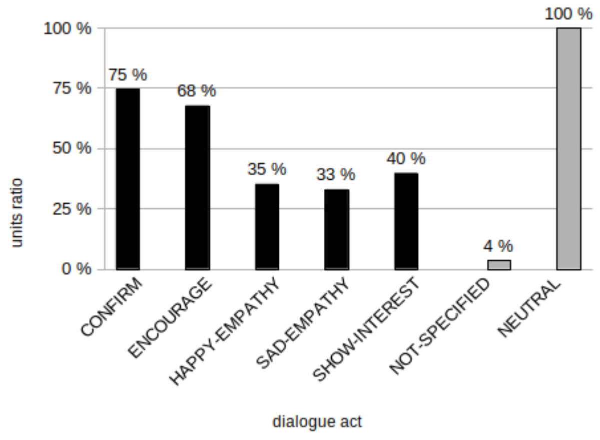


Figure 8: Relative ratio of units coming from utterances labelled with the DA which was intended to be synthesized.

8.2.2 Quality evaluation

To investigate whether the synthetic speech quality deteriorated by adding the expressivity, a MOS test evaluation was performed. In the MOS test, the listeners assess the speech quality using a 5-point scale where, in theory, the natural speech should be evaluated as 5 ($100\ \%$) and a very unnatural speech as 1 ($0\ \%$). The test was running along with the expressivity perception test, i.e. the test conditions, test utterances and the listeners were the same as for the evaluation that is presented in Section 8.2.1. The results of this MOS test are shown in Figure 9 and also in Table 9 altogether with a relative comparison with the natural speech (whose result is evaluated as $100\ \%$).

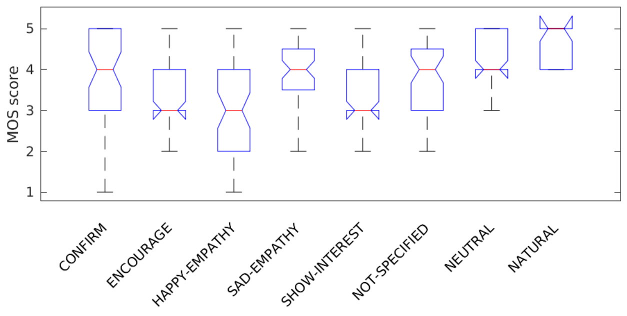


Figure 9: Evaluation of speech quality using a MOS test (unit selection).

The results suggest that the quality of expressive synthetic speech is worse than the quality of neutral synthetic speech by 0.49 of the MOS score ($13\ \%$) in average. It is almost the same difference as between natural speech and neutral synthetic speech (0.65 of the MOS score). This deterioration is probably caused by greater variability of the acoustic signal of expressive speech. Thus, the artifacts might occur more often than in neutral synthetic speech.

An auxiliary measure called *smooth joints* can be also calculated. A smooth joint is a concatenation point of two

dialogue act	MOS score		comparison with
	mean	std	natural speech
CONFIRM	3.87	1.11	79 %
ENCOURAGE	3.48	0.97	68 %
HAPPY-EMPATHY	3.10	1.00	58 %
SAD-EMPATHY	3.87	0.94	79 %
SHOW-INTEREST	3.25	0.92	62 %
mean	**3.51**	**0.99**	**69** %
NOT-SPECIFIED	3.92	0.78	81 %
NEUTRAL	4.08	0.78	83 %
mean	**4.00**	**0.78**	**82** %
natural speech	4.65	0.48	100 %

Table 9: Evaluation of speech quality using a MOS test (unit selection).

speech units that were originally adjacent in the speech corpus and thus their concatenation is natural. The smooth joints measure indicates the relative ratio of such joints with respect to the number of all concatenation points. The calculated values are presented in Figure 10. It is assumed that the less smooth joints in synthetic speech, the more artifacts can occur, causing the synthetic speech quality to be worse.

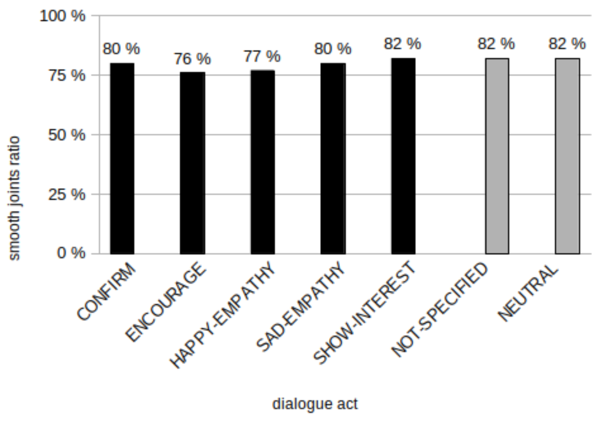


Figure 10: Relative ratio of smooth joints.

It is obvious that the relative ratio of smooth joints is almost the same regardless of the DA (mean 79 %) and also in comparison with neutral synthetic speech (mean 82 %). Also, this measure seems to be unrelated to the expressivity perception measure or the MOS score.

8.3 Evaluation of the HMM-based expressive speech synthesis

Even though this work deals mostly with the unit selection speech synthesis, the results of an experiment with the HMM-based expressive speech synthesis are to be briefly discussed in this section. The used method is based on the HTS system [63] and adapted to the Czech language [56]. The experiment is described in more details in [62] and the HMM approach is also briefly presented in Section 7. The aim is to evaluate the capability of the HMM-based TTS system to produce expressive speech (shown in Table 10) and to evaluate its quality (Table 11). The presented results are summarized and various DAs are not differentiated. There were 12 listeners participating in these listening tests.

dialogue act	expressivity perception ratio	cannot decide
expressive	15 %	5 %
NOT-SPECIFIED	8 %	3 %

Table 10: Expressivity perception in synthetic speech (HMM).

dialogue act	MOS score	comparison with
	mean	natural speech
expressive DAs + NOT-SPECIFIED	2.71	50 %
natural speech	4.44	100 %

Table 11: Evaluation of speech quality using a MOS test (HMM).

The expressivity perception ratio in synthetic speech produced by the HMM-based expressive TTS system is at a very low level (15 %) in comparison with the unit selection TTS system (54 %). Also the quality of synthetic speech is much worse, 2.7 of the MOS score (50 % of natural speech) for the HMM-based system and 3.5 (69 %) for the unit selection system. Generally, the HMM-based speech synthesis for the Czech language is not yet at such a high level as the unit selection approach is. Moreover, by adding expressive speech into this process, the trained HMM models may in fact mix natural and expressive acoustic signal depending on how the decision trees were created. Thus, in such synthetic speech of a lower quality, it is probably hard to identify any kind of expressivity.

8.4 Evaluation of the expressivity in dialogues

Since the unit selection expressive speech synthesis is going to be used in a specific dialogue system (conversations between seniors and a computer; see Section 1 and 2), it is necessary to evaluate it also with respect to this purpose. A preference listening test was used to perform this kind of evaluation. The test stimuli were prepared as follows:

– 6 appropriate parts of the natural dialogues (see Section 2), each approximately 1 minute in length, were randomly selected. The appropriateness were determined on the basis of sufficiency of the avatar's interactions within the dialogues. These parts will be further referred to as *minidialogues*.

- The acoustic signal of each minidialogue was splitted into parts where the person is speaking and parts where the avatar responses are expressed by the neutral speech synthesis.

- The text contents of the avatar's responses were slightly modified so that the newly generated responses are really to be synthesized and not only played back. The sense of the utterances was of course kept the same so that the dialogue flow is not disrupted.

- The new texts (avatar's responses) were synthesized using both the baseline neutral TTS system and the newly developed expressive TTS system – before the expressive speech synthesis, the texts were labelled by presumably appropriate DAs.

- In some parts of the minidialogues where the person is originally speaking, little modifications were done so that the length of the person's speech was shortened – for instance, the parts where the person was speaking for a long time or where a long silence was detected were removed. Again, the natural dialogue flow was not disrupted.

- the parts of the minidialogues were joint together so that two versions of each minidialogue were created – the first one with the avatar's responses with neutral synthetic speech and the second one with the avatar's responses with expressive synthetic speech.

Each of the 6 minidialoges contains 4 avatar's responses in average expressing various DAs, mostly *SHOW-INTEREST* or *ENCOURAGE*. However, each evaluated DA was included at least once in the responses. The minidialogues were then presented to the listeners within a listening test, both minidialogue's variants in a single test query. The task for the listeners was to decide which variant is more natural, more pleasant and which one would they prefer when being in place of the human minidialogue participant. The results of this evaluation are presented in Table 12; there were 11 listeners participating in this listening test.

synthesis variant	preference
neutral	8 %
expressive	83 %
cannot decide	9 %

Table 12: Evaluation of neutral vs. expressive speech synthesis in dialogues.

It's obvious that the listeners preferred the expressive speech synthesis to the neutral one (83 % preference ratio). This is one of the most important results indicating that the developed system increases the user experience with the TTS system for this limited domain task.

To verify that the avatar's responses were indeed synthesized and not only played back, the measure of smooth joints can be used. The mean value of this measure for the expressive avatar's responses is 86 % which is slightly higher than it was measured in Figure 10 of Section 8.2.1 (mean 82 % for neutral speech and 79 % for expressive speech). However, it still means that the responses were really synthesized.

9 Conclusion

It is necessary to incorporate some kind of expressivity into synthetic speech as it improves the user experience with systems using speech synthesis technology. Expressive speech sounds more naturally in dialogues between humans and computers. There are several ways to make the synthetic speech sound expressively. In this work, expressivity described by dialogue acts was employed and the algorithms of the TTS system were modified to use that information when producing synthetic speech.

The results presented in Section 8 suggest that in speech produced by the expressive TTS system the listeners perceived some kind of expressivity. More importantly, it was also confirmed that in the dialogues within the discussed limited domain, expressive speech is more suitable and preferred than the pure neutral speech produced by the baseline TTS system even though its quality is little bit worse.

Although the development of the expressive TTS system was done within a limited domain of conversations about personal photos between humans and a computer, the whole procedure – data collecting, data annotation, expressive corpus preparation and recording, expressivity description and TTS system modification – can be used within any other limited domain if appropriate expressivity definition is used. Thus, an expressivity can be incorporated to any other dialogue system with a similar structure.

Acknowledgement

This research was supported by the Czech Science Foundation (GA CR), project No. GA19-19324S and by the Ministry of Education, Youth and Sports of the Czech Republic project No. LO1506.

References

[1] J. D. Williams, S. Young, Partially observable Markov decision processes for spoken dialog systems, Computer Speech and Language 21 (2) (2007) 393–422.
`https://doi.org/10.1016/j.csl.2006.06.008`

[2] O. Lemon, K. Georgila, J. Henderson, M. Stuttle, An ISU dialogue system exhibiting reinforcement learning of dialogue policies: generic slot-filling

in the TALK in-car system, in: Proceedings of the Eleventh Conference of the European Chapter of the Association for Computational Linguistics: Posters & Demonstrations, EACL '06, Association for Computational Linguistics, Stroudsburg, PA, USA, 2006, pp. 119–122.
`https://doi.org/10.3115/1608974.1608986`

[3] X. Wu, M. Xu, W. Wu, Preparing for evaluation of a flight spoken dialogue system, in: Proceedings of ISCSLP, 2002, paper 50.

[4] J. Švec, L. Šmídl, Prototype of Czech spoken dialog system with mixed initiative for railway information service, in: P. Sojka, A. Horák, I. Kopecek, K. Pala (Eds.), Text, Speech and Dialogue, Vol. 6231 of Lecture Notes in Computer Science, Springer, Berlin-Heidelberg, Germany, 2010, pp. 568–575.
`https://doi.org/10.1007/978-3-642-15760-8\_72`

[5] A. Meštrović, L. Bernić, M. Pobar, S. Martinčič-Ipšić, I. Ipšić, Overview of a croatian weather domain spoken dialog system prototype, in: 32nd International Conference on Information Technology Interfaces (ITI), Cavtat, Dubrovnik, 2010, pp. 103–108.

[6] A. W. Black, Unit selection and emotional speech, in: Proceedings of Eurospeech, Geneva, Switzerland, 2003, pp. 1649–1652.

[7] M. Bulut, S. S. Narayanan, A. K. Syrdal, Expressive speech synthesis using a concatenative synthesiser, in: Proceedings of the 7th International Conference on Spoken Language Processing – ICSLP, Denver, CO, USA, 2002, pp. 1265–1268.

[8] W. Hamza, R. Bakis, E. M. Eide, M. A. Picheny, J. F. Pitrelli, The IBM expressive speech synthesis system, in: Proceedings of the 8th International Conference on Spoken Language Processing – ISCLP, Jeju, Korea, 2004, pp. 2577–2580.
`https://doi.org/10.1109/tasl.2006.876123`

[9] I. Steiner, M. Schröder, M. Charfuelan, A. Klepp, Symbolic vs. acoustics-based style control for expressive unit selection, in: Seventh ISCA Tutorial and Research Workshop on Speech Synthesis, Kyoto, Japan, 2010, pp. 114–119.

[10] J. Lorenzo-Trueba, G. E. Henter, S. Takaki, J. Yamagishi, Y. Morino, Y. Ochiai, Investigating different representations for modeling and controlling multiple emotions in DNN-based speech synthesis, Speech Communication 99 (2018) 135–143.
`https://doi.org/10.1016/j.specom.2018.03.002`

[11] S. An, Z. Ling, L. Dai, Emotional statistical parametric speech synthesis using LSTM-RNNs, in: 2017 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC), IEEE, 2017, pp. 1613–1616.
`https://doi.org/10.1109/apsipa.2017.8282282`

[12] H. Li, Y. Kang, Z. Wang, EMPHASIS: An emotional phoneme-based acoustic model for speech synthesis system, in: Proceedings of Interspeech, 2018.
`https://doi.org/10.21437/interspeech.2018-1511`

[13] S. Krstulovic, A. Hunecke, M. Schroder, An HMM-based speech synthesis system applied to German and its adaptation to a limited set of expressive football announcements, in: Proceedings of Interspeech, Antwerp, Belgium, 2007, pp. 1897–1900.

[14] B. Picart, R. Brognaux, , T. Drugman, HMM-based speech synthesis of live sports commentaries: Integration of a two-layer prosody annotation, in: 8th ISCA Speech Synthesis Workshop, Barcelona, Spain, 2013.

[15] H. Yang, H. Meng, L. Cai, Modeling the acoustic correlates of dialog act for expressive Chinese TTS synthesis, IET Conference Publications 2008 (CP544) (2008) 49–53.
`https://doi.org/10.1049/cp:20080758`

[16] P. Ircing, J. Romportl, Z. Loose, Audiovisual interface for Czech spoken dialogue system, in: IEEE 10th International Conference on Signal Processing Proceedings, Institute of Electrical and Electronics Engineers, Inc., Beijing, China, 2010, pp. 526–529.
`https://doi.org/10.1109/icosp.2010.5656088`

[17] J. F. Kelley, An iterative design methodology for user-friendly natural language office information applications, ACM Transactions on Information Systems 2 (1) (1984) 26–41.
`https://doi.org/10.1145/357417.357420`

[18] S. Whittaker, M. Walker, J. Moore, Fish or fowl: A Wizard of Oz evaluation of dialogue strategies in the restaurant domain., in: Language Resources and Evaluation Conference, Gran Canaria, Spain, 2002.

[19] M. Hajdinjak, F. Mihelič, The Wizard of Oz system for weather information retrieval, in: V. Matoušek, P. Mautner (Eds.), Text, Speech and Dialogue, proceedings of the 6th International Conference TSD, Vol. 2807 of Lecture Notes in Computer Science, Springer, Berlin-Heidelberg, Germany, 2003, pp. 400–405.
`https://doi.org/10.1007/978-3-540-39398-6\_57`

[20] J. A. Russell, A circumplex model of affect, Journal of Personality and Social Psychology 39 (1980) 1161–1178.

[21] A. Mehrabian, Pleasure-arousal-dominance: A general framework for describing and measuring individual differences in temperament, Current Psychology 14 (1996) 261–292.
https://doi.org/10.1007/BF02686918

[22] R. R. Cornelius, The science of emotion: Research and tradition in the psychology of emotions, Prentice-Hall, Englewood Cliffs, NJ, USA, 1996.

[23] A. K. Syrdal, A. Conkie, Y.-J. Kim, M. Beutnagel, Speech acts and dialog TTS, in: Proceedings of the 7th ISCA Speech Synthesis Workshop – SSW7, Kyoto, Japan, 2010, pp. 179–183.

[24] E. Zovato, A. Pacchiotti, S. Quazza, S. Sandri, Towards emotional speech synthesis: A rule based approach, in: Proceedings of the 5th ISCA Speech Synthesis Workshop – SSW5, Pittsburgh, PA, USA, 2004, pp. 219–220.

[25] J. M. Montero, J. Gutiérrez-Ariola, S. Palazuelos, E. Enríquez, S. Aguilera, J. M. Pardo, Emotional speech synthesis: From speech database to TTS, in: Proceedings of the 5th International Conference on Spoken Language Processing – ICSLP, Vol. 3, Sydney, Australia, 1998, pp. 923–926.

[26] J. F. Pitrelli, R. Bakis, E. M. Eide, R. Fernandez, W. Hamza, M. A. Picheny, The IBM expressive text-to-speech synthesis system for American English, IEEE Transactions on Audio, Speech, and Language Processing 14 (4) (2006) 1099–1108.
https://doi.org/10.1109/tasl.2006.876123

[27] A. J. Hunt, A. W. Black, Unit selection in a concatenative speech synthesis system using a large speech database, in: IEEE International Conference on Acoustics, Speech, and Signal Processing, Vol. 1, 1996, pp. 373–376.
https://doi.org/10.1109/ICASSP.1996.541110

[28] H. Zen, K. Tokuda, A. W. Black, Statistical parametric speech synthesis, Speech Communication 51 (2009) 1039–1064.
https://doi.org/10.1016/j.specom.2009.04.004

[29] H. Zen, A. Senior, M. Schuster, Statistical parametric speech synthesis using deep neural networks, in: 2013 IEEE International Conference on Acoustics, Speech and Signal Processing, IEEE, 2013, pp. 7962–7966.
https://doi.org/10.1109/ICASSP.2013.6639215

[30] A. van den Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, K. Kavukcuoglu, Wavenet: A generative model for raw audio, in: Arxiv, 2016. arXiv:1609.03499v2.

[31] Y. Wang, R. Skerry-Ryan, D. Stanton, Y. Wu, R. J. Weiss, N. Jaitly, Z. Yang, Y. Xiao, Z. Chen, S. Bengio, et al., Tacotron: Towards end-to-end speech synthesis, arXiv preprint arXiv:1703.10135
https://doi.org/10.21437/interspeech.2017-1452

[32] A. Kain, M. W. Macon, Spectral voice conversion for text-to-speech synthesis, in: Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing, Vol. 1, 1998, pp. 285–288.
https://doi.org/10.1109/icassp.1998.674423

[33] H. Kawanami, Y. Iwami, T. Toda, H. Saruwatari, K. Shikano, GMM-based voice conversion applied to emotional speech synthesis, IEEE Tranactions on Speech and Audio Processing 7 (1999) 2401–2404.

[34] J. Parker, Y. Stylianou, R. Cipolla, Adaptation of an expressive single speaker deep neural network speech synthesis system, in: 2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2018, pp. 5309–5313.
https://doi.org/10.1109/ICASSP.2018.8461888

[35] J. Matoušek, D. Tihelka, J. Romportl, Current state of Czech text-to-speech system ARTIC, in: Text, Speech and Dialogue, proceedings of the 9th International Conference TSD, Vol. 4188 of Lecture Notes in Computer Science, Springer, Berlin-Heidelberg, Germany, 2006, pp. 439–446.
https://doi.org/10.1007/11846406_55

[36] D. Tihelka, J. Kala, J. Matoušek, Enhancements of Viterbi search for fast unit selection synthesis, in: Proceedings of Interspeech, Makuhari, Japan, 2010, pp. 174–177.

[37] M. Grůber, M. Legát, P. Ircing, J. Romportl, J. Psutka, Czech Senior COMPANION: Wizard of Oz data collection and expressive speech corpus recording and annotation, in: Z. Vetulani (Ed.), Human Language Technology. Challenges for Computer Science and Linguistics, Vol. 6562 of Lecture Notes in Computer Science, Springer, Berlin-Heidelberg, Germany, 2011, pp. 280–290.
https://doi.org/10.1007/978-3-642-20095-3_26

[38] R. Cowie, Describing the emotional states expressed in speech, in: ISCA Workshop on Speech and Emotion, Newcastle, uk, 2000, pp. 11–18.

[39] A. K. Syrdal, Y.-J. Kim, Dialog speech acts and prosody: Considerations for TTS, in: Proceedings of Speech Prosody, Campinas, Brazil, 2008, pp. 661–665.

[40] M. G. Core, J. F. Allen, Coding dialogs with the DAMSL annotation scheme, in: Working Notes of the AAAI Fall Symposium on Communicative Action in Humans and Machines, Cambridge, MA, USA, 1997, pp. 28–35.

[41] J. Allen, M. Core, Draft of DAMSL: Dialog act markup in several layers, WWW page, [online] (1997).

[42] D. Jurafsky, L. Shrilberg, D. Biasca, Switchboard-DAMSL labeling project coder's manual, Tech. Rep. 97–02, University of Colorado, Institute of Cognitive Science, Boulder, Colorado, USA (1997).

[43] S. Jekat, A. Klein, E. Maier, I. Maleck, M. Mast, J. J. Quantz, Dialogue acts in VERBMOBIL, Tech. rep., German Research Center for Artificial Intelligence (DFKI), Saarbrücken, Germany (1995).

[44] J. Alexandersson, B. Buschbeck-Wolf, T. Fujinami, M. Kipp, S. Koch, E. Maier, N. Reithinger, B. Schmitz, M. Siegel, Dialogue acts in VERBMOBIL-2 - second edition, Tech. rep., German Research Center for Artificial Intelligence (DFKI), Saarbrücken, Germany (1998).

[45] A. P. Dempster, N. M. Laird, D. B. Rubin, Maximum likelihood from incomplete data via the EM algorithm, J. Roy. Statist. Soc. Ser. B 39 (1) (1977) 1–38, with discussion.

[46] J. Romportl, Prosodic phrases and semantic accents in speech corpus for Czech TTS synthesis, in: Text, Speech and Dialogue, proceedings of the 11th International Conference TSD, Vol. 5246 of Lecture Notes in Artificial Intelligence, Springer, Berlin–Heidelberg, Germany, 2008, pp. 493–500. `https://doi.org/10.1007/978-3-540-87391-4_63`

[47] J. L. Fleiss, Measuring nominal scale agreement among many raters, Psychological Bulletin 76 (5) (1971) 378–382. `https://doi.org/10.1037/h0031619`

[48] J. L. Fleiss, J. Cohen, The equivalence of weighted kappa and the intraclass correlation coefficient as measures of reliability, Educational and Psychological Measurement 33 (3) (1973) 613–619. `https://doi.org/10.1177/001316447303300309`

[49] J. A. Cohen, A coefficient of agreement for nominal scales, Educational and Psychological Measurement 20 (1) (1960) 37–46. `https://doi.org/10.1177/001316446002000104`

[50] J. R. Landis, G. G. Koch, The measurement of observer agreement for categorical data., Biometrics 33 (1) (1977) 159–174. `https://doi.org/10.2307/2529310`

[51] D. Tihelka, J. Matoušek, Unit selection and its relation to symbolic prosody: a new approach, INTERSPEECH 2006 – ICSLP, proceedings of 9th International Conference on Spoken Language Procesing 1 (2006) 2042–2045.

[52] M. Grůber, Enumerating differences between various communicative functions for purposes of Czech expressive speech synthesis in limited domain, in: Proceedings of Interspeech, Portland, Oregon, USA, 2012, pp. 650–653.

[53] M. Grůber, Acoustic analysis of Czech expressive recordings from a single speaker in terms of various communicative functions, in: Proceedings of the 11th IEEE International Symposium on Signal Processing and Information Technology, IEEE, 345 E 47TH ST, NEW YORK, NY 10017, USA, 2011, pp. 267–272. `https://doi.org/10.1109/isspit.2011.6151576`

[54] L. Latacz, W. Mattheyses, W. Verhelst, Joint target and join cost weight training for unit selection synthesis, in: Proceedings of Interspeech, ISCA, Florence, Italy, 2011, pp. 321–324.

[55] X. L. F. Alias, Evolutionary weight tuning for unit selection based on diphone pairs, in: Proceedings of Eurospeech, Vol. 2, Geneve, Switzerland, 2003, pp. 1333–1336.

[56] Z. Hanzlíček, Czech HMM-based speech synthesis, in: Text, Speech and Dialogue, proceedings of the 13th International Conference TSD, Vol. 6231 of Lecture Notes in Computer Science, Springer, Berlin-Heidelberg, Germany, 2010, pp. 291–298. `https://doi.org/10.1007/978-3-642-15760-8_37`

[57] J. Nouza, J. Psutka, J. Uhlíř, Phonetic alphabet for speech recognition of czech, Radioengineering 6 (4) (1997) 16–20.

[58] J. Romportl, J. Matoušek, D. Tihelka, Advanced prosody modelling, in: Text, Speech and Dialogue, proceedings of the 7th International Conference TSD, Vol. 3206 of Lecture Notes in Artificial Intelligence, Springer, Berlin-Heidelberg, Germany, 2004, pp. 441–447. `https://doi.org/10.1007/978-3-540-30120-2_56`

[59] J. Yamagishi, K. Onishi, T. Masuko, T. Kobayashi, Modeling of various speaking styles and emotions for HMM-based speech synthesis, in: Proceedings of Eurospeech, Geneva, Switzerland, 2003, pp. 2461–2464.

[60] K. Miyanaga, T. Masuko, T. Kobayashi, A style control technique for HMM-based speech synthesis, in: Proceedings of Interspeech, 2004, pp. 1437–1440.

[61] T. Nose, Y. Kato, T. Kobayashi, A speaker adaptation technique for MRHSMM-based style control of synthetic speech, in: Proceedings of ICASSP, 2007, pp. 833–836. `https://doi.org/10.1109/icassp.2007.367042`

[62] M. Grůber, Z. Hanzlíček, Czech expressive speech synthesis in limited domain: Comparison of unit selection and HMM-based approaches, in: Text, Speech and Dialogue, Vol. 7499 of Lecture Notes in Computer Science, Springer, Berlin-Heidelberg, Germany, 2012, pp. 656–664. `https://doi.org/10.1007/978-3-642-32790-2_80`

[63] K. Tokuda, H. Zen, J. Yamagishi, T. Masuko, S. Sako, A. W. Black, The HMM-based speech synthesis system (HTS), [online].

Knowledge Redundancy Approach to Reduce Size in Association Rules

Julio César Díaz Vera
University of Informatics Sciences, Havana, Cuba
E-mail: jcdiaz@uci.cu

Guillermo Manuel Negrín Ortiz
University of Informatics Sciences, Havana, Cuba
E-mail: gmnegrin@uci.cu

Carlos Molina
University of Jaen, Jaen, Spain
E-mail: carlosmo@ujaen.es

Maria Amparo Vila
University of Granada, Granada, Spain
E-mail:vila@decsai.ugr.es

Keywords: association rule mining, redundant rules, knowledge guided post-processing

Received: June 19, 2019

Association Rules Mining is one of the most studied and widely applied fields in Data Mining. However, the discovered models usually result in a very large set of rules; so the analysis capability, from the user point of view, is diminishing. Hence, it is difficult to use the found model in order to assist in the decision-making process. The previous handicap is hightened in the presence of redundant rules in the final set. In this work, a new definition of redundancy in association rules is proposed, based on user prior knowledge. A post-processing method is developed to eliminate this kind of redundancy, using association rules known by the user. Our proposal allows finding more compact models of association rules to ease its use in the decision-making process. The developed experiments have shown reduction levels that exceed 90 percent of all generated rules, using prior knowledge always below ten percent. So, our method improves the efficiency of association rules mining and the exploitation of discovered association rules.

Povzetek: Opisan je sistem za zmanjševanje števila in dolžine pravil s pomočjo analize redundantnosti za metode asociativnega učenja.

1 Introduction

Mining for association rules has been one of the most studied fields in data mining. Its main goal is to find unknown relations among items in a database.

Given a set of items I which contains all the items in the domain and a transactional database D where every transaction is composed by a transaction id (tid) and a set of items, subset of I (itemset).

An association rule is presented as an implication $X \rightarrow Y$ where X is the antecedent and Y is the consequent of the rule. Both X and Y are itemsets and usually, but not necessarily, they check $X \cap Y = \emptyset$ property. Association rules reflect how much the presence of the rule antecedent influences the presence of the rule consequent in the database records.

What generally makes a rule meaningful are two statistical factors: support and confidence. The support of a rule $supp(X \rightarrow Y)$ refers to the portion of the database transaction for which $X \cap Y$ is true while confidence $conf(X \rightarrow Y)$ is a measure of certainty to evaluate the validity of the rule, it is a measure for the portion of record which contains Y from those that contain X. The problem with association rule mining deals with finding all the rules that satisfy a user-given threshold for support and confidence. Most algorithms face the challenge in a two steps procedure

1. Find all the itemsets which support value is equal or greater than the support threshold.

2. Generate all association rules $X \rightarrow (Y - X)$, considering: Y is a frequent itemset, $X \subset Y$, and $conf(X \rightarrow Y)$ is equal or greater than the confidence threshold value.

The discovering of meaningful association rules can help in the decision-making process but the quite large number of rules usually makes it difficult for decision-makers in order to process, interpret and apply them. A significant part of the rules presented to the user are irrelevant because they are obvious, too general, too specific or because they are not relevant for the decision topic. Several methods were proposed in the literature to overcome this handicap such as interest measures development, concise representations of frequent itemsets and redundancy reduction. Section 2 discusses some of the most important works in the field.

This paper proposes a new approach to deal with redundancy, taking into account user previous knowledge about the studied domain. Previous knowledge is used to detect and prune redundant rules. We adapt the concept of redundancy and we propose a procedure to develop the redundancy reduction process in the post-processing stage.

The paper is organized as follows. Section 2 discusses related work. In section 3 we propose an algorithm to find and prune redundant rules. In section 4 the proposed algorithm is used over three datasets one with data about financial investment [1], other with data about the USA census [2] and the other with data about Mushrooms [2]. Section 5 closes the paper with conclusions.

2 Related work

Interestingness is difficult to define quantitatively [3] but most interestingness measures are classified in objective measures and subjective measures. Objective measures are domain-independent, one of them is the interestingness which is expressed in terms of statistic or information theory applied over the database. Several surveys [4, 5, 6] summarize and compare objective measures. The explosion of objective measures has raised a new problem: What are the best metrics to use in a specific situation and a particular application field? Several papers attempt to solve it [8, 9] but it is far from being solved. The correlation between 11 objective rule interestingness measures and real human interest over eight different datasets were computed in [10] and there was not a clear "winner", the correlation values associated with each measure varied considerably across the eight datasets.

Subjective measures were proposed in order to involve explicitly user knowledge in the selection of interesting rules so that the user can make a better selection. According to [11] subjective measures are classified in:

- Unexpectedness: a pattern is interesting if it is surprising to the user.
- Actionability: a pattern is interesting if it can help the user to take some actions.

Actionability started as an abstract notion, with an unclear definition, but nowadays, several researchers are interested in it. The actionability problem is discussed in [12].

Unexpectedness or novelty [13] was proposed in order to solve the pattern triviality problem, assessing the surprise level of the discovered rules. Several techniques have been used to accomplish this aim:

- Templates: Templates are syntactic constraints that allow the user to define a group of rules that are interesting or not to him/her [14, 15]. A template is defined as $A_1 ... A_n \rightarrow A_{n+1}$ where A_i is a class name in a hierarchy or an expression E over a class name. Templates may be inclusive or restrictive. A rule is considered interesting if it matches an inclusive template and uninteresting if it matches a restrictive template. The use of templates is quite restrictive because the matching method requires each rule element to be an instance of the elements in templates, and all template elements must have at least one instance in the rule. Moreover, the template definition makes hard to use it for declaring restrictive templates because it should be composed of elements subsuming all attributes of the rule, being in a subsuming relation with the inclusive template elements.

  The best known form of templates is meta-rules [16, 40] a meta-rule is the relationship between two association rules. The main drawback of this approach is that meta-rules are restricted to having a single rule in their antecedent and consequent, because of this some important information may be lost.

- Belief: Silbershatz and Tuzilin [11] defined user knowledge as a set of convictions, denominated belief. They are used in order to measure the unexpectedness of a pattern. Each belief is defined as a predicate formula expressed in first-order logic with a degree of confidence associated, measuring how much the user trusts in the belief. Two types of belief were defined:

  - Soft belief is that knowledge user accepts to change if new evidence contradicts the previous one. The interestingness of the new pattern is computed by how the new pattern changes the degree of beliefs.
  - Hard belief is that knowledge user will not change whatever new patterns are extracted. They are constraints that cannot be changed with new evidence.

  This approach is still in a development stage, no further advances were published, so it is not functional.

- General Impressions: were presented in [17] and later developed in [18] and [19]. They developed a specification language to express expectations and goals. Three levels of specification were established: General Impressions, Reasonably Precise Concept and Precise Knowledge. Item taxonomies concept was integrated in the specification languages in order to generalize rule selection. The matching process involved a syntactic comparison between antecedent/-consequent elements. Thus, each element in the general impression should find a correspondent in the association rule.

- Logical Contradiction: was developed in [20]. It consists in extracting only those patterns which logically contradict the consequent of the corresponding belief.

An association rule $X \rightarrow Y$ is unexpected with respect to some belief $A \rightarrow B$ if:

- $Y \wedge B \models FALSE$ B and Y are in logical contradiction;
- $X \wedge B$ has an important support in the database. This condition eliminates those rules which could be considered unexpected, but not those concerning the same transaction in the database;
- $A, X \rightarrow B$ exists.

- Preference Model: was proposed in [21]. It is a specific type of user knowledge representing how the basic knowledge of the user, called knowledge rules (K), will be applied over a given scenario or tuples of the database. The user proposes a covering knowledge (Ct) for each tuple (t) - a subset of the knowledge rule set K that the user prefers to apply to the tuple t. The approach validates the transactions which satisfy the extracted rule.

All the previously presented works use some kind of knowledge to reduce the number of useless association rules in the final set. In this way, our approach is similar to them but there are some remarkable differences.

Like in templates our approach uses the syntactical notation of association rules to represent knowledge. Templates use this knowledge to constraint the structure of selected rules, pruning out those rules which do not satisfy the template but produce a lot of association rules with similar information. On the other hand, we use the knowledge to remove those rules with similar information, presenting to the user a set of unexpected rules that can help him to better understand the underlying domain.

The approach followed by Belief tries to find just unknown rules, this is our main goal too but, they use a complex and fixed formal knowledge representation based on first order logic and degrees of belief with no clear way of building and maintaining the belief system. Instead, we use a simpler and natural rule-based form of knowledge, focused on the enhanced capability to increase interactively the knowledge system.

2.1 Rule redundancy reduction

Research community accepts the semantical definition of association rule redundancy given in [22] "an association rule is redundant if it conveys the same information - or less general information - than the information conveyed by another rule of the same usefulness and the same relevance". But several formal definitions have been proposed over time. In table 1, a sample transactional database is presented. Defining a support threshold of 0.15 and a confidence threshold of 0.75, an association rule model with 92 rules is obtained. It is used to show redundancy definitions.

Income	Balance	Sex	Unemployed	Loan
High	High	F	No	Yes
High	High	M	No	Yes
Low	Low	M	No	No
Low	High	F	Yes	Yes
Low	High	M	Yes	Yes
Low	Low	F	Yes	No
High	Low	M	No	Yes
High	Low	F	Yes	Yes
Low	Medium	M	Yes	No
High	Medium	M	No	Yes
Low	Medium	F	Yes	No
Low	Medium	M	No	No

Table 1: Sample transactions

Definition 1. ***Minimal non-redundant association rules[22]:*** *An association rule $R : X \rightarrow Y$ is a minimal non-redundant association rule if there is not an association rule $R_1 : X_1 \rightarrow Y_1$ with:*

- $support(R) = support(R_1)$
- $confidence(R) = confidence(R_1)$
- $X_1 \subseteq X$ *and* $Y \subseteq Y_1$

From data on table 1 we can obtain the rules: $R : \{[balance].[medium]\} \rightarrow \{[income].[low], [loan].[no]\}$ $supp = 0.25$, $conf = 0.75$ and $R_1 : \{[balance].[medium]\} \rightarrow \{[loan].[no]\}$ $supp = 0.25$, $conf = 0.75$. According to definition 1 R is a redundant rule. No new information is provided by its inclusion into the association rules model.

Several works have been developed to prune that kind of redundancy. Mining Closed Associations, uses frequent closed itemsets [23] tries to produce the set of minimal generators for each itemset. The number of closed association rules is linear to the number of closed frequent itemsets. It can be large for sparse and large datasets.

The Generic Basis (GB) and the Informative Basis (IB) [22] used the Galois connections to propose two condensed basics that represent non-redundant rules. The Gen-GB and Gen-RI algorithms were presented to obtain a generic basis and a transitive reduction of the IB. The reduction ratio of IB was improved by [24] maximal closed itemsets. The Informative Generic Basis [25] also uses the Galois connection semantics but taking the support of all frequent itemsets as an entry, so it can calculate the support and confidence of derived rules. The augmented Iceberg Galois lattice was used to construct the Minimal Generic Basis (MGB) [26]. The concept of generator was incorporated into high utility itemsets mining in [27].

The redundancy definition presented in definition 1 requires that a redundant rule and its corresponding non-redundant rule must have identical confidence and identical support. From data on table 1 we can obtain the

rules:
$R : \{[income].[high], [unemployed].[no]\} \rightarrow \{[loan].[yes]\}$ $supp = 0.33, conf = 1.0$, and $R_1 : \{[income].[high]\} \rightarrow \{[loan].[yes]\}$ $supp = 0.41, conf = 1.0$ those rules are non-redundant ones, but the consequent of R can be obtained from R_1 a rule with the same confidence and fewer conditions. So without R the same results are achieved, rule R must be a redundant rule. Xu [28] formalizes this kind of redundancy in definition 2.

Definition 2. ***Redundant rules[28]:*** *Let $X \rightarrow Y$ and $X_1 \rightarrow Y_1$ be two association rules with confidence cf and cf_1, respectively. $X \rightarrow Y$ is said to be a redundant rule to $X_1 \rightarrow Y_1$ if*

– $X_1 \subseteq X$ *and* $Y \subseteq Y_1$

– $cf \leq cf_1$

Based on definition 2 the Reliable basis was proposed. It consists of two bases the ReliableApprox used in partial rules, and ReliableExact used in exact rules. Frequent closed itemsets are used to perform the reliable redundancy reduction process. It generates rules with minimal antecedent and maximal consequent. The reliable basis removes a great amount of redundancy without reducing the inference capacity of the remaining rules. Phan [29] uses a more radical approach to define redundancy see definition 3.

Definition 3. ***Representative association rules[29]:*** *Let $X \rightarrow Y$ an association rule. $X \rightarrow Y$ is said to be a representative association rule if there is not other interesting rule $X_1 \rightarrow Y_1$ such that $X_1 \subseteq X$ and $Y \subseteq Y_1$.*

The redundancy definitions presented above do not guarantee the exclusion of all non-interesting patterns of the final model. Example 1 shows a group of rules with no new information to the user, and they are not classified as redundant by the previous definitions.

Example 1. ***A set of redundant rules from data in table 1*** *Let's see a subset of association rules obtained from table 1:*

$R_1 : \{[income].[high]\} \rightarrow \{[loan].[yes]\}$
$R_2 : \{[sex].[female], [unemployed].[no]\} \rightarrow \{[income].[high]\}$
$R_3 : \{[sex].[female], [unemployed].[no]\} \rightarrow \{[income].[high], [loan].[yes]\}$
$R_4 : \{[sex].[female], [unemployed].[no]\} \rightarrow \{[loan].[yes]\}$
$R_5 : \{[income].[high], [loan].[yes]\} \rightarrow \{[unemployed].[no]\}$
$R_6 : \{[income].[high], [loan].[yes], [sex].[male]\} \rightarrow \{[unemployed].[no]\}$
$R_7 : \{[balance].[high], [income].[high], [loan].[yes]\} \rightarrow \{[unemployed].[no]\}$

If we analyze the rules R_1 and R_3 we see that item [loan].[yes] in R_3 consequent provides no new information, because this is known by R_1. So rule R_3 is redundant but this kind of redundancy is not detected by the previous definitions. Analyzing rules R_1, R_2 and R_4 we can check that combining, transitively, of R_1 and R_2 it will produce R_4 so, R_4 is redundant. One more time this kind of redundancy is not detected by previous definitions. In R_5,R_6 and R_7 antecedent the item $[loan].[yes]$ provides no new information because this is known by R_1. It is redundant and must be pruned, but it can not be detected by redundancy definitions.

2.2 Post-processing

Since the year 2000, the interest in post-processing methods in association rules has been increasing. Perhaps the most accurate definition of post-processing tasks were done by Baesens et al. [30] Post-processing consists of different techniques that can be used independently or together: pruning, summarizing, grouping and visualization. We have a special interest in pruning techniques that prune those rules that do not match to the user knowledge. Those techniques are associated with interestingness measures that may not satisfy the downward closure property, so it is impossible to integrate them in Apriori like extraction algorithms.

An element to consider is the nature of Knowledge Discovery in Databases (KDD) as an interactive and iterative user-centered process. Enforcing constraints during the mining runs neglects the character of KDD [31], [32]. A single and possibly expensive mining run is accepted but all subsequent mining questions are supposed to be satisfied with the initial result set.

In this work, a method is developed to obtain non-redundant association rules about user knowledge. It is important to ensure the user capability to refine his/her knowledge in an interactive and iterative way, accepting any of the discovered associations or discarding some previous associations and updating prior knowledge. This approach also makes possible to fulfill the mining question of different users, with different domain knowledge, in a single mining run.

3 A knowledge guided approach

3.1 Knowledge based redundancy

In example 1, a group of redundant rules, which are currently not covered by the definitions of redundancy, are showed. Our interest is to eliminate these forms of redundancy in association rule models. Based on a core set of rules that represent the user belief; a result of his experience working in the subject area. This knowledge is more general than rules obtained in the mining process which only represent a particular dataset with partial information so the quality metric value for this kind of rule is considered maximal. This set of rules will be named prior knowledge. A rule that does not contradict prior knowledge of the user will be considered redundant. We formalize the notion of prior knowledge redundancy in definition 4. User can represent previous knowledge in different ways like semantic networks, ontologies, among others.

Considering that, the expert is interested in association rules discovering, prior knowledge is incorporated to the model using association rules format. For example an expert working with the dataset presented in table 1 knows that customers with high income ($[income].[high]$) pay their loans on time and therefore these must be approved. This knowledge can be represented as the association rule $\{[income].[high]\} \to \{[loan].[yes]\}$.

Definition 4. ***Knowledge Based Redundancy:*** *Let $\mathcal{S}$ be a set of association rules and $\mathcal{S}_c$ a set of prior known rules, defined over the same domain of $\mathcal{S}$. An association rule $R : X \to Y \in \mathcal{S}$ is redundant with respect to $\mathcal{S}_c$ if there is a rule $R^{'} : X^{'} \to Y^{'} \in \mathcal{S}_c$ and fulfills some of the following conditions.*

1. $X^{'} \subseteq X \wedge Y^{'} \cap Y \neq \{\emptyset\}$
   A rule is redundant if there is another rule presented in $\mathcal{S}_c$ that contains more general information.

2. $X^{'} \subseteq X \wedge \exists R^{''} : X^{''} \to Y^{''} \in \mathcal{S}_c : X^{''} \subseteq Y^{'} \wedge Y \subseteq Y^{''}$
   A rule R is redundant if there is a rule $R^{'}$ in $\mathcal{S}_c$ that contains part or the whole antecedent and there is a third rule $R^{''}$ in $\mathcal{S}_c$ that shares information with $R^{'}$ and its consequent contains R consequent.

3. $X^{'} \subseteq X \wedge Y^{'} \cap X \neq \{\emptyset\}$
   A rule is redundant if its antecedent contains a part or the whole information of a previously known rule.

4. $X^{'} \subseteq Y \wedge Y^{'} \cap Y \neq \{\emptyset\}$
   A rule is redundant if its consequent contains a part or the whole information of a previously known rule.

Reviewing rules in example 1 with definition 4 we have: $\mathcal{S}_c = \{\{[income].[high]\} \to \{[loan].[yes]\}, \{[sex].[female], [unemployed,].[no]\} \to \{[income].[high]\}\}$

Rule R_3 : $\{[sex].[female], [unemployed].[no]\} \to \{[income].[high], [loan].[yes]\}$ fulfills condition 1 in definition 4 because:

1. $[sex].[female], [unemployed].[no] \subseteq [sex].[female], [unemployed].[no]$
2. $[income].[high] \subseteq [income].[high], [loan].[yes]$

Rule R_3 : $\{[sex].[female], [unemployed].[no]\} \to \{[income].[high], [loan].[yes]\}$ fulfills condition 4 in definition 4 because:

1. $[income].[high] \subseteq [income].[high], [loan].[yes]$
2. $[loan].[yes] \subseteq [income].[high], [loan].[yes]$

Rule R_4 : $\{[sex].[female], [unemployed].[no]\} \to \{[loan].[yes]\}$ fulfills condition 2 in definition 4 because:

1. $[sex].[female], [unemployed].[no] \subseteq [sex].[female], [unemployed].[no]$
2. $[income].[high] \subseteq [income].[high]$
3. $[loan].[yes] \subseteq [loan].[yes]$

Rule R_5 : $\{[income].[high], [loan].[yes]\} \to \{[unemployed].[no]\}$ fulfills condition 3 in definition 4 because:

1. $[income].[high] \subseteq [income].[high], [loan].[yes]$
2. $[loan].[yes] \subseteq [income].[high], [loan].[yes]$

Rule R_6 : $\{[income].[high], [loan].[yes], [sex].[male]\} \to \{[unemployed].[no]\}$ fulfills condition 3 in definition 4 because:

1. $[income].[high] \subseteq [income].[high], [loan].[yes], [sex].[male]$
2. $[loan].[yes] \subseteq [income].[high], [loan].[yes], [sex].[male]$

Rule R_7 : $\{[balance].[high], [income].[high], [loan].[yes]\} \to \{[unemployed].[no]\}$ fulfills condition 3 in definition 4 because:

1. $[income].[high] \subseteq [balance].[high], [income].[high], [loan].[yes]$
2. $[loan].[yes] \subseteq [balance].[high], [income].[high], [loan].[yes]$

Armstrong's axioms [33] are a set of inference rules. They allow to obtain the minimum set of functional dependencies that are maintained in a database. The rest of functional dependencies can be derived from this set. They are part of clear mechanisms designed to find smaller subsets of a larger set of functional dependencies called "covers" that are equivalent to the "bases" in Closure Spaces and Data Mining.

Armstrong's axioms can not be used as an inference mechanism in association rules [34] because it is impossible to obtain the values of support and confidence in the derived rules:

– Reflexivity (if $B \subset A$ then $A \to B$) holds because $conf(A \to B) = \frac{supp(A \cap B)}{supp(A)} = \frac{supp(A)}{supp(A)} = 1$

– Transitivity if $A \to B$ and $B \to C$ both hold with confidence $\geq threshold$ we can not know the value for $conf(AD \to C)$ so the Transitivity does not hold.

– Augmentation (if $A \to B$ then $AC \to B$) does not hold. Enlarging the antecedent of a rule may give a rule with much smaller confidence, even zero: think of a case where most of the times X appears it comes with Z, but it only comes with Y when Z is not present; then the confidence of $X \rightarrow Z$ may be high whereas the confidence of $XY \rightarrow Z$ may be null.

Our intention is to use Armstrong's axioms in order to assess if a rule has Prior Knowledge Redundancy over a set of rules S_c from previous knowledge. So they must verify the condition presented in definition 4.

Condition $X^{'} \subseteq X \wedge Y^{'} \cap Y \neq \{\emptyset\}$ represents the classical definition of redundancy like in definition 1, definition 2 and definition 3. This condition is fulfilled if a single attribute in Y is redundant. Armstrong's axioms can be used to perform this operation. Let $R_1 : X \rightarrow Y$ and $R_2 : X^{'} \rightarrow Y^{'}$ be association rules. Suppose $Y^{'} \cap Y = Y_1$. Then by the reflexivity axiom on R_2 consequent $R_3 : Y \rightarrow Y_1$ and by reflexivity on R_1 consequent $R_4 : Y^{'} \rightarrow Y_1$. By transitivity between R_1 and R_3 we have $R_5 : X \rightarrow Y_1$, applying transitivity between R_2 and R_4 we have $R_6 : X^{'} \rightarrow Y_1$. $X^{'} \subseteq X$ by statement condition, applying augmentation in R_6 until $X^{'} = X$, $R_7 : X \rightarrow Y_1$. Therefore Armstrong's axioms check the condition. For example, the rule $R : \{[income].[high], [sex].[male]\} \rightarrow \{[loan].[yes], [unemployed].[no]\}$ is part of the association model generated from the dataset in table 1. This rule can be classified as redundant by condition 1 of definition 4 with respect to prior knowledge. $S_c = \{R_{s1} : [income].[high] \rightarrow [loan].[yes]$, $R_{s2} : [sex].[female], [unemployed].[no] \rightarrow [income].[high]\}$. By the application of Reflexivity, we have that $R_1 : [loan].[yes] \rightarrow [loan].[yes]$ by Augmentation of $[unemployed].[no]$ on R_1 we have $R_2 : [loan].[yes], [unemployed].[no] \rightarrow [loan].[yes]$ and by Transitivity between R and R_2 we have $R_3 : [income].[high], [sex].[male] \rightarrow [loan].[yes]$, the same procedure must be followed to $[unemployed].[no]$. Now by Augmentation of $[sex].[male]$ in rule $[income].[high] \rightarrow [loan].[yes] \in S_c$ we have $R_4 : [income].[high], [sex].[male] \rightarrow [loan].[yes]$ $R_4 = R_3$ so item $[loan].[yes]$ is redundant in R and therefore R is also redundant.

Condition $X^{'} \subseteq X \wedge \exists R^{''} : X^{''} \rightarrow Y^{''} \in \mathcal{S}_c : X^{''} \subseteq Y^{'} \wedge Y \subseteq Y^{''}$ represents the notion of transitivity a common term in human thinking. This condition is fulfilled if a single attribute in Y is redundant. Let $R_1 : X \rightarrow Y$, $R_2 : X^{'} \rightarrow Y^{'}$ and $R_3 : X^{''} \rightarrow Y^{''}$ be rules. Suppose $Y^{''} \cap Y = Y_1$. Then by the reflexivity axiom on R_1 consequent $R_4 : Y \rightarrow Y_1$ by transitivity between R_1 and R_4 we have $R_5 : X \rightarrow Y_1$. By statement condition $X^{''} \subseteq Y^{'}$ so by reflexivity on R_2 consequent we have $R_6 : Y^{'} \rightarrow X^{''}$. By transitivity between R_2 and R_6 we have $R_7 : X^{'} \rightarrow X^{''}$ now by transitivity between R_2 and R_7 we have $R_8 : X^{'} \rightarrow Y^{''}$. Applying augmentation in R_8 until we have $R_9 : X \rightarrow Y^{''}$. By reflexivity in R_9 consequent $R_{10} : Y \rightarrow Y_1$ and by transitivity between R_9 and R_{10} we have $R_{11} : X \rightarrow Y_1$. Therefore Armstrong's axioms check the condition. For example, taking into account rule $R : \{[sex].[female], [unemployed].[no]\} \rightarrow \{[loan].[yes]\}$ and prior knowledge $S_c = \{R_{s1} : [income].[high] \rightarrow [loan].[yes]$, $R_{s2} : [sex].[female], [unemployed].[no] \rightarrow [income].[high]\}$. R is classified as redundant according to condition 2 in definition 4. R is a single consequent rule so no separation is needed. By the application of Transitivity between $[income].[high] \rightarrow [loan].[yes]$ and $[sex].[female], [unemployed].[no] \rightarrow [loan].[yes]$ both in S_c the rule $R_1 : [sex].[female], [unemployed].[no] \rightarrow [loan].[yes]$ is obtained $R = R_1$ so R is a redundant rule.

Condition $X^{'} \subseteq X \wedge Y^{'} \cap X \neq \{\emptyset\}$ represents the case when any item in the antecedent of a rule is a redundant one. Let $R_1 : X \rightarrow Y$ and $R_2 : X^{'} \rightarrow Y^{'}$ be rules. Suppose $Y' \cap X = X_1$. Then by augmentation of X_1 in R_2 we have $R_3 : X^{'}X_1 \rightarrow X_1Y^{'}$ and by transitivity between R_3 and R_1 $R_4 : X \rightarrow Y$. Therefore Armstrong's axioms fulfill the condition. For example, with $R : \{[income].[high], [loan].[yes]\} \rightarrow \{[unemployed].[no]\}$ and $S_c = \{R_{s1} : [income].[high] \rightarrow [loan].[yes]$, $R_{s2} : [sex].[female], [unemployed].[no] \rightarrow [income].[high]\}$ R is classified as redundant by condition 3 in definition 4. Applying Reflexivity of $[income].[high]$ in $[income].[high] \rightarrow [loan].[yes]$ rule $R_1 : [income].[high] \rightarrow [income].[high], [loan].[yes]$ is obtained by Transitivity between R_1 and R we have $R_2 : [income].[high] \rightarrow [income].[high]$ R_2 is simpler than R with the same information so R is a redundant rule. However, by Augmentation of $[loan].[yes]$ in R_2 we have $R_3 : [income].[high], [loan].[yes] \rightarrow [unemployed].[no]$ $R = R_3$.

Condition $X^{'} \subseteq Y \wedge Y^{'} \cap Y \neq \{\emptyset\}$ represents the case when any item in the consequent of R is redundant with respect to other item in consequent. This condition is fulfilled if a single attribute in Y is redundant. Let $R_1 : X \rightarrow Y$ and $R_2 : X^{'} \rightarrow Y^{'}$ be rules. Suppose $Y \cap Y^{'} = Y_1$. Then by the reflexivity axiom on R_2 consequent $R_3 : Y^{'} \rightarrow Y_1$ by transitivity between R_2 and R_3 we have $R_4 : X^{'} \rightarrow Y_1$. By statement condition we have $X \subseteq Y$ so by transitivity between R_1 and R_4 we have $R_5 : X \rightarrow Y_1$. Therefore Armstrong's axioms fulfill the condition. For example, $R : \{[balance].[high], [unemployed].[no]\} \rightarrow \{[income].[high], [loan].[yes]\}$ and $S_c = \{R_{s1} : [income].[high] \rightarrow [loan].[yes]$, $R_{s2}[sex].[female], [unemployed].[no] \rightarrow [income].[high]\}$. R is redundant according to condition 4 in definition 4. Applying Reflexivity, Augmentation and Transitivity we obtain $R_1 : [balance].[high], [unemployed].[no] \rightarrow [income].[high]$ and $R_2 : [balance].[high], [unemployed].[no] \rightarrow [loan].[yes]$ now by Transitivity between R_1 and $[income].[high] \rightarrow [loan].[yes] \in S_c$ we have $R_3 : [balance].[high], [unemployed].[no] \rightarrow [loan].[yes]$. $R_2 = R_3$ so R is a redundant rule.

We do not use Armstrong's Axioms as an inference mechanism so, we do not worry if it is not able to ensure the support and confidence threshold in the inferred rules.

3.2 Algorithm to eliminate prior knowledge redundancy in association rules

In this section we present an algorithm to determine if a rule contains redundant items, see Fig. 1. The closure algorithm presented in [35] is used to compute X^+.

```
Require: Set of previous knowledge rules $S_c$
    A rule $R_i$ in form $X \to Y$
Ensure: Boolean value to indicate if the rule is redundant
 1: $i = 0$
 2: $n = |Y|$
 3: while $i < n$ do
 4:     if $Y[i] \in X^+_{S_c \cup X \to (Y - \{Y[i]\})}$ then
 5:         return true
 6:     end if
 7:     $i = i + 1$
 8: end while
 9: $i = 0$
10: $n = |X|$
11: while $i < n$ do
12:     if $X[i] \in (X - X[i])^+_{S_c \cup (X - X[i]) \to Y}$ then
13:         return true
14:     end if
15:     $i = i + 1$
16: end while
17: return false
```

Algorithm 1: Prior Knowledge Redundancy detection

To determine the redundancy of a rule $X \to Y$ we have to prove if any item A in the rule's antecedent is redundant or if an item W in the consequent is redundant. The item A is redundant if the consequent can be derived from the prior knowledge without A. The first part of algorithm 1 performs this task for all items $A \in X$ by calculating the closure of the new antecedent $X - \{A\}$ over the previous knowledge rules joined to the studied rule focus, and comparing results with the closure of the same antecedent over the set of previous rules joined to a new rule, where the item A is not a part of the antecedent. If both results are equal, then the item A is redundant and the entire rule is also redundant. To test if item W is redundant we have to apply a similar procedure, the second part of algorithm 1 performs this task.

Example 2. ***Prior Knowledge Redundancy detection:*** *We use the following Prior Knowledge* $S_c = \{R_{s1} : [income].[high] \to [loan].[yes]$, $R_{s2} : [sex].[female], [unemployed].[no] \to [income].[high]\}$ *and the rules* $R_1 : \{[balance].[high], [unemployed].[no]\} \to \{[income].[high], [loan].[yes]\}$ *and* $R_2 : \{[income].[high], [loan].[yes]\} \to \{[unemployed].[no]\}$ *to show the performance of algorithm 1. For* R_1 *we have:*

The first step is to compute $F = S_c \cup R_i$ *for* R_1 $F = \{R_{f1} : [income].[high] \to [loan].[yes]$, $R_{f2} : [sex].[female], [unemployed].[no] \to [income].[high]$, $R_{f3} : [balance].[high], [unemployed].[no] \to [income].[high], [loan].[yes]\}$.

Second, checks the redundancy in the antecedent, computing closure of $[balance].[high]$ *over* F. *This is* $[balance].[high]^+_F = [balance].[high]$ *and comparing with closure of* $[balance].[high]$ *over* G *where* $G = ((F - \{R_1\}) \cup ([balance].[high]) \to [income].[high], [loan].[yes])$, $[balance].[high]^+_G = [balance].[high], [income].[high], [loan].[yes]$. *They are different so* $[unemployed].[no]$ *is not redundant. The item* $[balance].[high]$ *is also non-redundant.*

And last, checks the redundancy in the consequent.
$F^{'} = \{(F - R_1 \cup ([balance].[high], [unemployed].[no] \to [income].[high])\}$
$[balance].[high], [unemployed].[no]^+_F = [balance].[high], [unemployed].[no], [income].[high], [loan].[yes]$,
$[balance].[high], [unemployed].[no]^+_{F'} = [balance].[high], [unemployed].[no], [income].[high], [loan].[yes]$. *They are the same so the item* $[loan].[yes]$ *and the rule* R_1 *are redundant.*

For R_2 *we have:*

- $F^{'} = (F - R_1) \cup [income].[high] \to [unemployed].[no]$.
  $F = \{(F - R_1) : [income].[high] \to [loan].[yes]$, $R_{f2}[sex].[female], [unemployed].[no] \to [income].[high]$, $R_{f3}[income].[high], [loan].[yes] \to [unemployed].[no]\}$.
- $[income].[high]^+_F = [income].[high], [loan].[yes], [unemployed].[no]$, $[income].[high]^+_{F'} = [income].[high], [loan].[yes], [unemployed].[no]$. *They are the same so the rule is redundant.*

3.2.1 Correctness

We first prove that closure algorithm [35] can be used to detect redundancy according to definition 4. Closure algorithm applies Armstrong's axioms to find all items implied by a given itemset.

Theorem 1. *Let* $\mathcal{S}_c$ *be a set of prior known rules and* $R : X \to Y$ *an association rule. If there is a rule* $R^{'} : X^{'} \to Y^{'} \in \mathcal{S}_c$ *and* $X^{'} \subseteq X \wedge Y^{'} \cap Y \neq \{\emptyset\}$ *then* $Y^{'} \cap Y \in X^+_{\mathcal{S}_c}$

Proof. Assume $X^{'} \subseteq X \wedge Y^{'} \cap Y \neq \{\emptyset\}$. Then $X^{'} \in X^+_{\mathcal{S}_c}$ by assumption $X^{'} \subseteq X$ and reflexivity axiom. So

$Y^{'} \in X^{+}_{\mathcal{S}_c}$ by transitivity between $X \rightarrow X^{'}$ and $X^{'} \rightarrow Y^{'}$. Therefore $Y^{'} \cap Y \in X^{+}_{\mathcal{S}_c}$ by definition of set intersection. □

Theorem 2. *Let $\mathcal{S}_c$ be a set of prior known rules and $R : X \rightarrow Y$ one association rule. If there is a rule $R^{'} : X^{'} \rightarrow Y^{'} \in \mathcal{S}_c$ and $X^{'} \subseteq X \wedge \exists R^{''} : X^{''} \rightarrow Y^{''} \in \mathcal{S}_c : X^{''} \subseteq Y^{'} \wedge Y \subseteq Y^{''}$ then $Y \in X^{+}_{\mathcal{S}_c}$.*

Proof. Assume $X^{'} \subseteq X \wedge \exists R^{''} : X^{''} \rightarrow Y^{''} \in \mathcal{S}_c : X^{''} \subseteq Y^{'} \wedge Y \subseteq Y^{''}$. Then $X^{'} \in X^{+}_{\mathcal{S}_c}$ by assumption $X^{'} \subseteq X$ and reflexivity axiom. $Y^{'} \in X^{+}_{\mathcal{S}_c}$ by transitivity between $X \rightarrow X^{'}$ and $X^{'} \rightarrow Y^{'}$. $X^{''} \in X^{+}_{\mathcal{S}_c}$ by assumption $X^{''} \subseteq Y^{'}$ and subset definition. So $Y^{''} \in X^{+}_{\mathcal{S}_c}$ by transitivity between $X \rightarrow X^{''}$ and $X^{''} \rightarrow Y^{''}$. Therefore $Y \in X^{+}_{\mathcal{S}_c}$ by assumption $Y \subseteq Y^{''}$ and subset definition. □

Theorem 3. *Let $\mathcal{S}_c$ be a set of prior known rules and $R : X \rightarrow Y$ one association rule. If there is a rule $R^{'} : X^{'} \rightarrow Y^{'} \in \mathcal{S}_c$ and $X^{'} \subseteq X \wedge Y^{'} \cap X \neq \{\emptyset\}$ then $Y^{'} \cap X \in (X - (Y^{'} \cap X))^{+}_{\mathcal{S}_c}$.*

Proof. Assume $X^{'} \subseteq X \wedge Y^{'} \cap X \neq \{\emptyset\}$. Then $X^{'} \in (X - (Y^{'} \cap X))^{+}_{\mathcal{S}_c}$ by assumption $X^{'} \subseteq X$ and reflexivity axiom. $Y^{'} \in (X - (Y^{'} \cap X))^{+}_{\mathcal{S}_c}$ by transitivity between $X \rightarrow X^{'}$ and $X^{'} \rightarrow Y^{'}$. Therefore $Y^{'} \cap X \in (X - (Y^{'} \cap X))^{+}_{\mathcal{S}_c}$ by definition of set intersection. □

Theorem 4. *Let $\mathcal{S}_c$ be a set of prior known rules and $R : X \rightarrow Y$ one association rule. If there is a rule $R^{'} : X^{'} \rightarrow Y^{'} \in \mathcal{S}_c$ and $X^{'} \subseteq Y \wedge Y^{'} \cap Y \neq \{\emptyset\}$ then $Y^{'} \cap Y \in X^{+}_{\mathcal{S}_c \cup X \rightarrow (Y - (Y^{'} \cap Y))}$.*

Proof. Assume $X^{'} \subseteq Y \wedge Y^{'} \cap Y \neq \{\emptyset\}$. Then $X^{'} \in X^{+}_{\mathcal{S}_c \cup X \rightarrow (Y - (Y^{'} \cap Y))}$ by assumption $X^{'} \subseteq Y$ and association rule property $X \cap Y = \emptyset$. $Y^{'} \in X^{+}_{\mathcal{S}_c \cup X \rightarrow (Y - (Y^{'} \cap Y))}$ by transitivity between $X \rightarrow X^{'}$ and $X^{'} \rightarrow Y^{'}$. Therefore $Y^{'} \cap Y \in X^{+}_{\mathcal{S}_c \cup X \rightarrow (Y - (Y^{'} \cap Y))}$ by definition of set intersection. □

Theorem 5. *If $(\exists A_i \in X \wedge A_i \in (X - A_1)^{+}_{\mathcal{S}_c \cup (X - A_i) \rightarrow Y}) \vee (\exists W_i \in Y \wedge W_i \in X^{+}_{\mathcal{S}_c \cup X \rightarrow Y - W_i})$ then rule $X \rightarrow Y$ has prior knowledge redundancy over $\mathcal{S}_c$.*

Proof. Direct from theorem 1, theorem 2, theorem 3 and theorem 4. □

Hoare triple was introduced by C. A. R. Hoare [38] as $\{P\}C\{Q\}$, for specifying what a program does. In such a Hoare triple:

- C is a program.
- P and Q are assertions, conditions on the program variables used in C. They will be written using standard mathematical notation together with logical operators. We can use functions and predicates to express high-level properties based on a domain theory [39] covering specifics of the application area.

We say $\{P\}C\{Q\}$ is true, if whenever C is executed in a state satisfying $\{P\}$ and if the execution of C finishes, then the state in which C execution finishes satisfies Q. If there is a loop in C, loop invariants must be used to prove correctness. If loop invariants are proved to be true after each loop iteration then the postcondiction must be proven true.

In algorithm 1 lines one through eight and lines nine through sixteen perform basically the same operation, one over the rule antecedent and the other over the rule consequent. So we analize them only one time. Line four checks if $Y[i]$ is subset of the closure. So closure algorithm must be computed, this algorithm has been proved as correct[35]. The search of $Y[i]$ within closure can be done by a well known linear search algorithm, we assume it is correct.

Preconditions:

- $\mathcal{S}_c$ is a set of previous knowledge rules.
- $X \rightarrow Y$ is an association rule with $X = X_1, .., X_n$ and $Y = Y_1, .., Y_m$

Postcondition: If $(\exists A_i \in X \wedge A_i \in (X - A_1)^{+}_{\mathcal{S}_c \cup (X - A_i) \rightarrow Y}) \vee (\exists W_i \in Y \wedge W_i \in X^{+}_{\mathcal{S}_c \cup X \rightarrow Y - W_i})$ the return value is $true$.

Loop invariants: If the loop is executed j or more times, then after j executions

- $i = j$
- $0 \leq i \leq n$
- $Y[h] \notin X^{+}_{\mathcal{S}_c \cup X \rightarrow (Y - \{Y[i]\})}$ for $0 \leq h < i$

Proving the loop invariant: (by induction on j) **Base Case**: $j = 0$

- before first execution of loop $i = 0$
- loop invariant holds, $i = 0 \Rightarrow (0 \leq h < 0)$. No such h value.

Inductive hypothesis: assume that, if the loop iterates j times then the loop invariant holds $i_{old} = j$. Proving that if the loop iterates $j + 1$ times, then the loop invariant holds for $i_{new} = j + 1$. If true for iteration $i_{old} = j$ then $Y[h] \notin X^{+}_{S_c \cup X \rightarrow (Y - \{Y[i]\})}$ for $0 \leq h < i_{old}$.

- if loop iterates then $Y[i_{old}] \notin X^{+}_{\mathcal{S}_c \cup X \rightarrow (Y - \{Y[i_{old}]\})}$ and $i_{new} = i_{old} + 1$.
- thus $Y[h] \notin X^{+}_{\mathcal{S}_c \cup X \rightarrow (Y - Y[h])}$ for $0 \leq h < i_{new}$.
- because loop iterated for $i_{old} = j$ we have $i_{old} < n$ and $i_{new} \leq n$

Thus, the loop invariant holds for $j + 1$.

When the loop test fails, the loop invariant holds and either $i \geq n$ or $Y[i] \in X^+_{S_c \cup X \to (Y - Y[i])}$

- **Case 1** ($j \geq n$): loop invariant implies that $Y[h] \notin X^+_{S_c \cup X \to (Y - Y[h])}$ for $0 \leq h < n$, so no element in cosequent is a redundant one.

- **Case 2** ($j < n$): loop invariant implies that $Y[i] \in X^+_{S_c \cup X \to (Y - Y[i])}$ and $true$ is returned

Conclusions: Poscondition is satisfied in either case, so the algorithm is correct.

3.2.2 Complexity analysis

Time complexity of an algorithm is a function $T(n)$ limiting the maximum number of steps in the algorithm for an input size n. $T(n)$ depends on what is counted as one computation step, the random access machine (RAM) model is the most extended one. RAM is a model for a simple digital computer with random access memory. For the sake of simplicity $T(n)$ is approximated by a simplest function, it is written $T(n) = O(f(n))$ if there are constants $c \geq 0$ and $n_1 \geq 0$ such that: $T(n) \leq cf(n)$ for all $n \geq n_1$.

For algorithm in Fig 1 we considered a as the number of different attribute symbols in S_c and p the number of previous knowledge rules presented in S_c. The complexity order to compute the closure is $O(n)$ see [35]. The execution time of the first **while** loop (the consequent of the rule) takes $a * p$ since the number of rules in F is p, and we compute the closure with a cost of $O(p)$. The execution time of the second **while** loop (the antecedent of the rule) takes the same value of $a * p$ because it performs the same operation and in the same way the complexity of the steps is $O(ap)$. To compute the complexity of the entire algorithm, the complexity of the first and second **while** loops must be added so it is $O(ap) + O(ap) = 2O(ap)$ but the constant 2 can be ignored and the final value for complexity of the algorithm is $O(ap)$.

Association rules extraction algorithms have much higher complexity [36] than the reduction approach presented here. This difference led us to propose a reduction mechanism in which rule extraction algorithm is executed once and then, in the post processing stage, the reduction algorithm is fired to prune the redundant rules, rather than applying prior knowledge as restriction within the extraction algorithms, which would force to execute it for each different user and even for each change on a user's prior knowledge. The computational cost for the constraint approach is very high. However, our approach, in post processing stage, allows us to run a simpler routine when the user changes or the user prior knowledge is updated. The temporal cost of this approach did not exceed 5 seconds in any of the applied tests.

4 Experimental results

4.1 Methodology

In order to verify the effectiveness of our approach we performed experiments with four datasets. The first one with data about USA census[2], the second one with data about stock market investments [1], the third one with data about hypothetical samples of mushroom[2] and the last one with data about breast cancer[2]. Prior knowledge consists of 6 rules for each dataset. We use Pruning Ratio metric $PR = (PrunedRules/TotalRules) \times 100$ to evaluate our results.

Table 2 shows the result of the experiments. Each row corresponds to an experiment following the next steps:

1. Find the complete set of rules using as support threshold the value in column 2 and confidence threshold the value in column 3. The number of rules is showed in column 4.

2. Apply the steps presented in algorithm 1. The number of pruned rules are presented in column 5 of Table 2.

3. After applying the algorithm to the dataset, the final number of rules is presented in column 6 of Table 2 while column 7 contains the pruning ratio. The execution time is presented in column 8.

4.2 Results and discussion

Pruning Ratio changes according to support in Census and Stocks datasets, first increasing while the support increases, but when the support is greater than 0.07 for the Census dataset and greater than 0.5 for Stocks dataset, the Pruning Ratio decreases while the support increases. The behavior in Mushroom dataset is the opposite, the Pruning Ratio decreases while support increases until the support reaches the 0.5 value then the Pruning Ratio increases while the support value increases.

This behavior shows a relation between support and previous knowledge patterns. If the support is increased, then a number of rules do not meet the support threshold and they are discarded. Hence the discarded rules have no major impact on the rules derived from previous knowledge, Pruning Ratio will be increased, but as the support increases it starts to reduce the rules derived from previous knowledge, so the Pruning Ratio will be decreased.

In Fig 1, Fig 2 and Fig 3 the mean value of Pruning Ratio is shown for several support values in Census, Stocks and Mush datasets respectively using combination of all six rules in S_c.

Dataset	Support	Confidence	Rules	Pruned Rules	Final Rules	Pruning Ratio	Time
Census	0.01	0.4	3408	942	2466	27	0.589
Census	0.03	0.4	835	242	593	28	0.079
Census	0.05	0.4	458	158	300	32	0.043
Census	0.07	0.4	229	79	150	34	0.021
Census	0.09	0.4	163	51	112	31	0.015
Census	0.11	0.4	114	23	91	20	0.010
Stocks	0.2	0.4	11010	5592	5418	50	2.170
Stocks	0.3	0.4	3314	2225	1089	67	0.536
Stocks	0.4	0.4	1230	904	326	73	0.116
Stocks	0.5	0.4	349	294	55	84	0.039
Stocks	0.6	0.4	212	64	148	30	0.020
Mushroom	0.3	0.5	78998	29154	49844	36	11.245
Mushroom	0.4	0.5	5767	1225	4542	21	0.852
Mushroom	0.5	0.5	1148	200	948	17	0.098
Mushroom	0.6	0.5	266	88	178	33	0.025
Mushroom	0.7	0.5	180	83	97	46	0.017
Breast	0.01	0.4	210500	98582	111918	47	27.732
Breast	0.1	0.4	28808	13695	15113	47	4.190
Breast	0.2	0.4	6092	2982	3110	49	0.859
Breast	0.3	0.4	5284	2398	2886	45	0.798
Breast	0.4	0.4	1246	449	797	36	0.118

Table 2: Experiment's result

4.3 Traditional vs. knowledge based reduction

The approach developed in this paper differs from those published until now. Previous woks are concerned with the structural relationship between association rules and mechanisms to reduce redundancy using inference rules and maximal itemsets. We use the user experience to prune rules that do not bring new knowledge to the user, simplifying decision making. Both approaches are not comparable in essence, but we carried out experiments to compare KBR's pruning ratio with previous works.

Fig 4 shows the pruning ratio of some relevant works in redundancy reduction, over a Mushroom dataset with a support value of 0.3. We used Mushroom dataset because we can access to author experiments and it is sufficient to test our case. The values for pruning ratio are taken from the author's papers: MinMax, Reliable, GB, CHARM, CRS and MetaRules.[40]

Reliable has the best Pruning Ratio, see Fig 4, so we compare it with our approach at different support values, see Table 3.

Reliable Pruning Ratio is the best of KBR_{6rules}, KBR_{9rules} and $KBR_{12rules}$. Nevertheless, $KBR_{15rules}$ reaches better Pruning Ratio than Reliable for all supports except 0.4, see Fig. 6. A previous knowledge of 15 rules is equivalent to 0.018% of the whole rule set, for a support value of 0.3, and 7.9% for a support value of 0.7.

With very few rules in KBR is possible to exceed the Pruning Ratio of previous works. Of course there is a narrow relationship between the Pruning Ratio and the repercussion of the previous knowledge rules over the whole set of rules. The Pruning Ratio of knowledge rules increases in the same way that they are able to describe the domain under study. The better KBR results are, the better the user will know the domain under study. Our approach has the possibility to determine when a model can not be improved like in the case of $KBR_{15rules}$ for a support value of 0.7 where the Pruning Ratio is 100%.

4.4 Knowledge vs knowledge based reduction

In section 2 we surveyed some works that used knowledge to reduce the number of association rules presented to the final user. The main goal of those papers is to obtain a set of association rules that satisfies some constraint provided by users, using different forms of knowledge representation. They are able to reduce the association rules set cardinality but generate a lot of rules that represent the same knowledge. Strictly speaking we can not compare our proposal with those ones because of the difference between goals, but we want to test the association rules model cardinality reduction capability of our approach with template, the best known form of knowledge approach.

We compare the pruning ratio of our approach with the template implementation proposed in [41] that up-perform the implementation proposed in [16] across five dataset from [2].

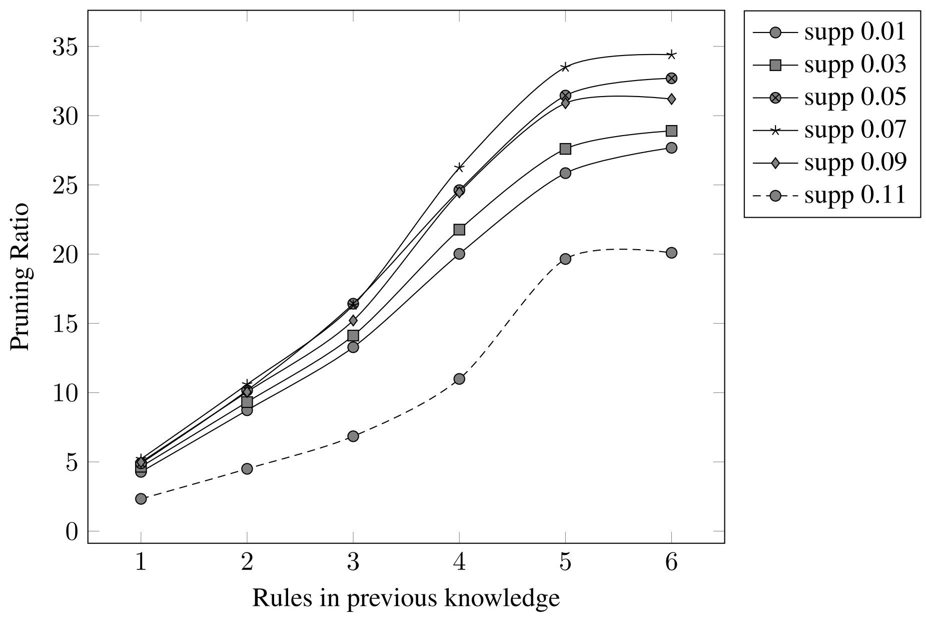


Figure 1: Rules pruned in census dataset

Support	Reliable	KBR_{6rules}	KBR_{9rules}	$KBR_{12rules}$	$KBR_{15rules}$
0.3	95	36	76	80	96
0.4	90	21	37	47	84
0.5	89	17	30	44	93
0.6	74	33	40	62	97
0.7	78	46	46	75	100
Average	85	32,5	45,8	61,5	94

Table 3: Pruning Ratio

– Mushroom data (mush)

– Johns Hopkins University Ionosphere data (ion)

– Statlog Project Heart Disease data (hea)

– Thyroid Disease data (thy)

– Attitudes Toward Workplace Smoking Restrictions data (smo)

The continuous attributes in the data sets used were discretized using a 4-bin equal-frequency discretization. Support and Confidence were set to the same values used in [16]. In table 4 we present the result of our pruning approach (KBR) and compare it with the previous work (MetaRules) [41].

Each row in table 4 represents an experiment where column Dataset contains the dataset id, column TotalRules shows the total number of rules produced by extraction algorithms, MetaRules presents the remaining rules after the application of the aplgorithm proposed in [41] while column KBR contains the average of remaining rules of ten runs of knowledge based redundancy elimination algorithm using a random knowledge of ten rules for each execution. The remaining rules in our approach are lower than the number of rules in metarules approach for all datasets.

Dataset	TotalRules	MetaRules	KBR
mush	1374	138	120.2
ion	1215	452	402.6
hea	371	246	176.7
thy	1442	502	431.6
smo	797	300	283.3

Table 4: Remaining rules

5 Conclusion

The fundamental idea in this work is linked to the main definition of data mining: analysis of large amount of data to extract interesting patterns, previously unknown and the consideration that an association rule that correspond to prior knowledge is a redundant one[37]. Our approach prunes those rules, presenting a simpler model to the final user.

The main contribution in this work is the definition of redundancy of association rules with respect to prior knowledge, and the definition of a mechanism to eliminate this kind of redundancy from the final model of association

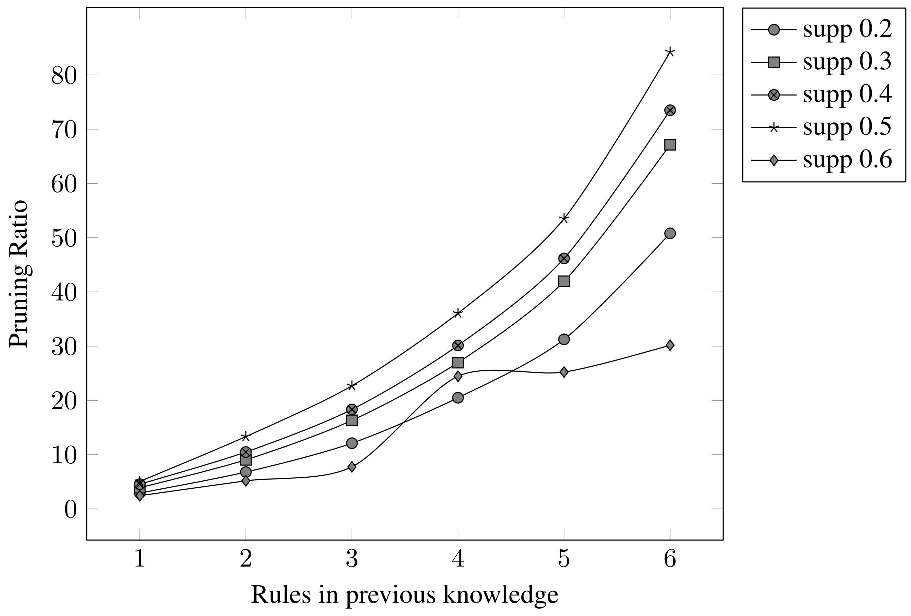


Figure 2: Rules pruned in stocks dataset

rules presented to the end user. The redundancy elimination is performed in two procedures, the first one to detect and prune redundant element in rules antecedent and consequent, and the second one to detect if all information provided by a rule is redundant with respect to prior knowledge and then to prune it.

The results of this study confirm it is possible to use prior knowledge of experts to reduce the volume of association rules. Models of association rules with fewer rules can be interpreted more clearly by specialists so they can generate advantages in decision making process. The experimental results show that prior knowledge of less than 10% can reach a reduction ratio above 90%.

Acknowledgement

This research is partially funded by the research project TIC1582 by "Consejeria de Economia, Innovacion, Ciencia y Empleo from Junta de Andalucia" (Spain).

References

[1] J. Núñez, (2007), "Empleo de Fuzzy OLAP para Obtener Reglas que Caractericen Estrategias de Inversión".

[2] D. J. Newman, (2007), "UCI Repository of Machine Learning Databases",University of California, School of Information and Computer Science, Irvine, CA.

[3] Sisodia, Dilip Singh and Singhal, Riya and Khandal, Vijay, (2018), "Comparative performance of interestingness measures to identify redundant and non-informative rules from web usage data", International Journal of Technology. https://doi.org/10.14716/ijtech.v9i1.1510

[4] Ali Yousif Hasan, (2019), "Evaluation and Validation of the Interest of the Rules Association in Data-Mining", International Journal of Computer Science and Mobile Computing, Vol.8 Issue.3, pp. 230-239.

[5] N. Bhargava, M. Shukla, (2016), "Survey of Interestingness Measures for Association Rules Mining: Data Mining, Data Science for Business Perspective", International Journal of Computer Science and Information Technology (IJCSITS), Vol.6, No.2, Mar-April 2016, pp. 74-80.

[6] Sudarsanam, Nandan and Kumar, Nishanth and Sharma, Abhishek and Ravindran, Balaraman, (2019), "Rate of change analysis for interestingness measures", Knowledge and Information Systems. https://doi.org/10.1007/s10115-019-01352-3

[7] J. Blanchard, F. Guillet, P. Kuntz, (2009), "Semantics-based classification of rule interestingness measures in Post-mining of association rules: techniques for effective knowledge extraction", IGI Global, pp. 56-79. https://doi.org/10.4018/978-1-60566-404-0.ch004

[8] V. de Carvalho, V. Oliveira, R. de Padua, S. Oliveira, (2016), "Solving the Problem of Selecting Suitable Objective Measures by Clustering Association Rules Through the Measures Themselves", SOFSEM 2016: Theory and Practice of Computer Science. Springer Berlin Heidelberg. pp. 505-517. https://doi.org/10.1007/978-3-662-49192-8_41

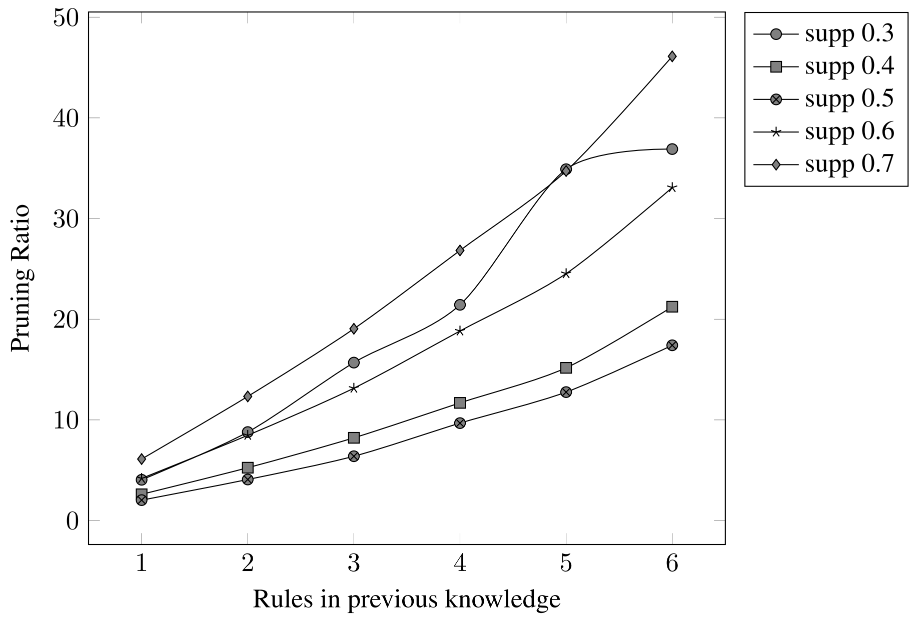


Figure 3: Rules pruned in Mushroom dataset

[9] V. Oliveira, D. Duarte, M. Violante, W. dos Santos, R. de Padua, S. Oliveira, (2017), "Ranking Association Rules by Clustering Through Interestingness", in Mexican International Conference on Artificial Intelligence, pp 336-351. Annals of Data Science 1.1 (2014): pp. 25-39.

[10] D. R. Carvalho, A. A. Freitas, N. Ebecken, (2005), "Evaluating the correlation between objective rule interestingness measures and real human interest", Knowledge Discovery in Databases: PKDD 2005, Springer, pp. 453-461. https://doi.org/10.1007/11564126_45

[11] A. Silberschatz, A. Tuzhilin, (1996), "What makes patterns interesting in knowledge discovery systems", IEEE Trans. Knowledge Data Eng, vol. 8, no. 6, pp. 970-974. https://doi.org/10.1109/69.553165

[12] R. Batra, M. A. Rehman, (2019), "Actionable Knowledge Dsicovery for Increasing Enterprise Profit, Using Domain Driven Data Mining.", IEEE Acces vol.7, pp. 182924-182936. https://doi.org/10.1109/access.2019.2959841

[13] R. Sehti, B. Shekar, (2019), "Subjective interestingness in Association Rule Mining: A Theoretical Analysis", Digital Business, Springer Charm, pp. 375-389. https://doi.org/10.1007/978-3-319-93940-7_15

[14] L. Greeshma, G. Pradeepini, (2016), "Unique Constraint Frequent Item Set Mining", Advanced Computing (IACC), 2016 IEEE 6th International Conference on pp. 68-72. IEEE. https://doi.org/10.1109/iacc.2016.23

[15] A. Kaur, V. Aggarwal, S. K. Shankar, (2016), "An efficient algorithm for generating association rules by using constrained itemsets mining", Recent Trends in Electronics, Information Communication Technology (RTEICT), IEEE International Conference on (pp. 99-102). IEEE. 2016. https://doi.org/10.1109/rteict.2016.7807791

[16] Berrado, G. C. Runger, (2007), "Using metarules to organize and group discovered association rules", Data Mining and Knowledge Discovery, vol. 14, no. 3, pp. 409-431. https://doi.org/10.1007/s10618-006-0062-6

[17] W. Liu, W. Hsu, S. Chen, (1997), "Using General Impressions to Analyze Discovered Classification Rules", KDD, pp. 31-36

[18] W. Liu, W. Hsu, K. Wang, S. Chen, (1999), "Visually aided exploration of interesting association rules", Methodologies for Knowledge Discovery and Data Mining, Springer, pp. 380-389. https://doi.org/10.1007/3-540-48912-6_52

[19] B. Liu, W. Hsu, S. Chen, Y. Ma, (2000), "Analyzing the subjective interestingness of association rules", Intell. Syst. Their Appl. IEEE, vol. 15, no. 5, pp. 47-55. https://doi.org/10.1109/5254.889106

[20] B. Padmanabhan, A. Tuzhilin, (2000), "Small is beautiful: discovering the minimal set of unexpected patterns", Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and

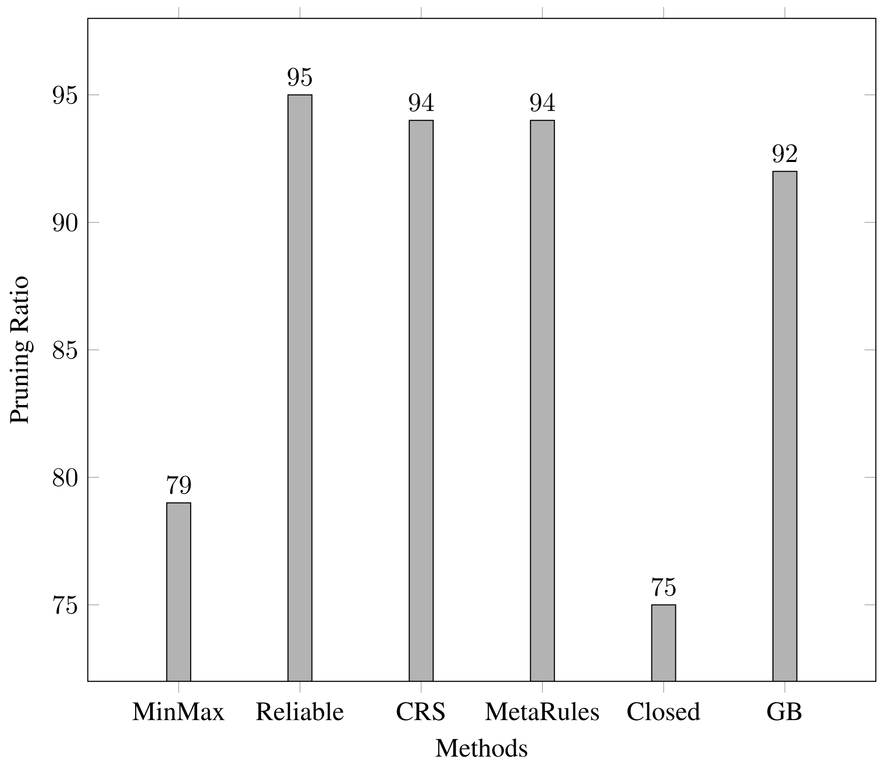


Figure 4: Pruning Ratio for different approaches

data mining, pp. 54-63. https://doi.org/10.1145/347090.347103

[21] K. Wang, Y. Jiang, L. V. Lakshmanan, (2003), "Mining unexpected rules by pushing user dynamics", Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 246-255. https://doi.org/10.1145/956750.956780

[22] Y. Bastide, N. Pasquier, R. Taouil, G. Stumme, and L. Lakhal, (2000), "Mining minimal nonredundant association rules using frequent closed itemsets", Proc. International Conference on Computational Logic (CL 2000), pp. 972-986. https://doi.org/10.1007/3-540-44957-4_65

[23] M. Quadrana, A. Bifet, R. Gavalda, (2015), "An efficient closed frequent itemset miner for the MOA stream mining system", AI Communications 28.1: pp. 143-158. https://doi.org/10.3233/aic-140615

[24] L. Greeshma, G. Pradeepini, (2016), "Mining Maximal Efficient Closed Itemsets Without Any Redundancy", Information Systems Design and Intelligent Applications. Springer India. pp. 339-347. https://doi.org/10.1007/978-81-322-2755-7_36

[25] G. Gasmi, S. B. Yahia, E. M. Nguifo, Y. Slimani, (2005), "A New Informative Generic Base of Association Rules", Advances in Knowledge Discovery and Data Mining, pp. 81-90, Springer Berlin Heidelberg. https://doi.org/10.1007/11430919_11

[26] C. L. Cherif, W. Bellegua, S. Ben Yahia, G. Guesmi, (2005), "VIE-MGB: A Visual Interactive Exploration of Minimal Generic Basis of Association Rules", Proc. International Conferences on Concept Lattices and Applications (CLA 2005), pp.179-196.

[27] P. Fournier-Viger, Wu C.-W., V. S. Tseng, (2014), "Novel Concise Representations of High Utility Itemsets using Generator Patterns", Proc. 10th International Conference on Advanced Data Mining and Applications, Springer LNAI. https://doi.org/10.1007/978-3-319-14717-8_3

[28] Y. Xu, Y. Li, G. Shaw, (2011), "Reliable representations for association rules", Data and Knowledge Engineering, vol. 70, no. 6, pp. 555-575. https://doi.org/10.1016/j.datak.2011.02.003

[29] Phan-Luong, (2001), "The representative basis for association rules", Proc. IEEE. International Conference on Data Mining (ICDM 2001), pp. 639-640. https://doi.org/10.1109/icdm.2001.989588

[30] B. Baesens, S. Viaene, and J. Vanthienen, (2000), "Post-processing of association rules", DTEW Res. Rep. 0020, pp. 118.

[31] J. Hipp, U. Gntzer, (2002), "Is pushing constraints deeply into the mining algorithms really

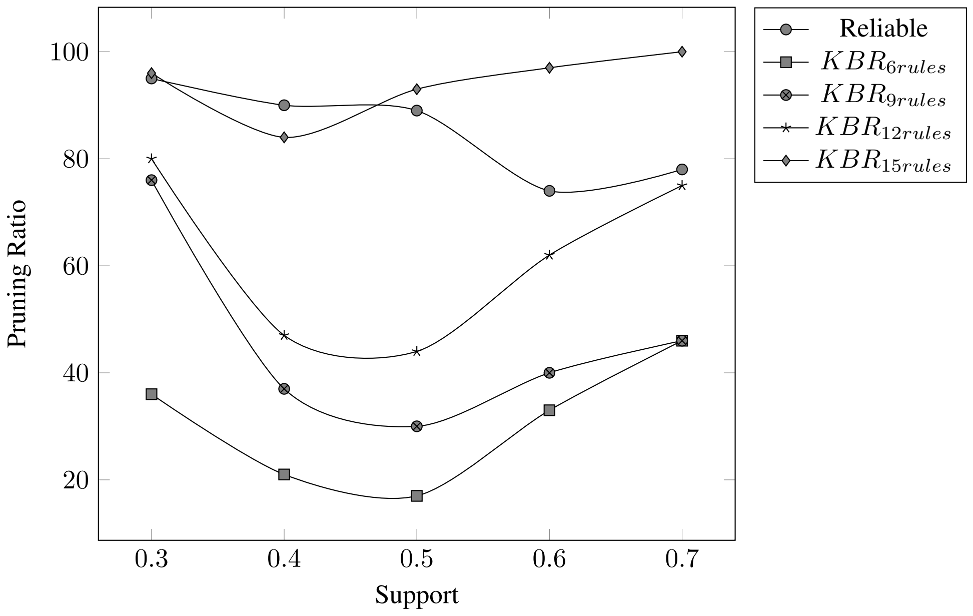


Figure 5: Reliable vs KBR pruning ratio

what we want?: an alternative approach for association rule mining", ACM SIGKDD Explorations 4(1), pp.50-55. https://doi.org/10.1145/568574.568582

[32] R. J. Bayardo, (2005), "The Hows, Whys, and Whens of Constraints and Itemset and Rule Discovery", Constraint-Based Mining and Inductive Databases LNCS3848, Springer: pp.1-13. https://doi.org/10.1007/11615576_1

[33] W. Armstrong, (1974), "Dependency structures of database relationships", IFIP Congress, pp. 580-583.

[34] Tirnuc, Cristina and Balcázar, José L. and Gómez-Pérez, Domingo, (2020), "Closed-SetBased Discovery of Representative Association Rules", International Journal of Foundations of Computer Science, vol. 31, no.1, pp. 143-156. https://doi.org/10.1142/s0129054120400109

[35] D. Maier, (1983), "Theory of Relational Database".

[36] W. A. Kosters, W. Pijls, V. Popova, (2003), "Complexity analysis of depth first and fp-growth implementations of apriori", Machine Learning and Data Mining in Pattern Recognition, Springer, pp. 284-292. https://doi.org/10.1007/3-540-45065-3_25

[37] H. Toivonen, M. Klemettinen, P. Ronkainen, K. Htnen, H. Mannila, (1995), "Pruning and grouping discovered association rules", MLnet Wkshp. on Statistics, Machine Learning, and Discovery in Databases.

[38] C. A. R. Hoare, (1972), "An axiomatic basis for computer programming", Communications of the ACM, 12, pp. 334-341.

[39] C. A. Furia, B. Meyer, S. Velder, (2014), "Loop invariants: Analysis, classification, and examples", ACM Computing Surveys (CSUR), vol. 46, no 3, p. 34. https://doi.org/10.1145/2506375

[40] Y. Xu, Y. Li, G. Shaw, (2011), "Reliable representations for association rules". Data & Knowledge Engineering, 70(6), 555-575. Elsevier. https://doi.org/10.1016/j.datak.2011.02.003

[41] Djenouri, Y., Belhadi, A., Fournier-Viger, P., Lin, J. C. W. (2018). Discovering strong meta association rules using bees swarm optimization. In Pacific-Asia Conference on Knowledge Discovery and Data Mining (2018, June) (pp. 195-206). Springer, Cham. https://doi.org/10.1007/978-3-030-04503-6_21

Hybrid Bees Approach Based on Improved Search Sites Selection by Firefly Algorithm for Solving Complex Continuous Functions

Mohamed Amine Nemmich and Fatima Debbat
Department of Computer Science, University Mustapha Stambouli of Mascara, Mascara, Algeria
E-mail: amine.nemmich@gmail.com, debbat_fati@yahoo.fr

Mohamed Slimane
Université de Tours, Laboratoire d'Informatique Fondamentale et Appliquée de Tours (LIFAT), Tours, France
E-mail: mohamed.slimane@univ-tours.fr

Keywords: optimization, bees algorithm, firefly algorithm, memory scheme, adaptive neighborhood search, swarm intelligence, bio-Inspired computation

Received: July 25, 2018

The Bees Algorithm (BA) is a recent population-based optimization algorithm, which tries to imitate the natural behavior of honey bees in food foraging. This meta-heuristic is widely used in various engineering fields. However, it suffers from certain limitations. This paper focuses on improvements to the BA in order to improve its overall performance. The proposed enhancements were applied alone or in pair to develop enhanced versions of the BA. Three improved variants of BA were presented: BAMS-AN, HBAFA and HFBA. The new BAMS-AN includes memory scheme in order to avoid revisiting previously visited sites and an adaptive neighborhood search procedure to escape from local optima during the local search process. HBAFA introduces the Firefly Algorithm (FA) in local search of BA to update the positions of recruited bees, thus increasing exploitation in each selected site. The third improved BA, i.e. HFBA, employs FA to initialize the population of bees in the BA for a best exploration and to start the search from more promising regions of the search space. The proposed enhancements to the BA have been tested using several continuous benchmark functions and the results have been compared to those achieved by the standard BA and other optimization techniques. The experimental results indicate that the improved variants of BA outperform the standard BA and other algorithms on most of the benchmark functions. The enhanced BAMS-AN performs particularly better than others improved BAs in terms of solution quality and convergence speed.

Povzetek: Za reševanje kompleksnih zveznih funkcij so razvili nov pristop na osnovi hibridnega čebeljega algoritma (BA) in algoritma Firefly.

1 Introduction

Metaheuristic algorithms generally mimic the more successful behaviors in nature. These methods present best tools in reaching the optimal or near optimal solutions for complex engineering optimization problems [1].

As a branch of metaheuristic methods, swarm intelligence (SI) is the collective behavior of populated, self-organized and decentralized systems. It concentrates specifically on insects or animals behavior in order to develop different metaheuristics that can imitate the capabilities of these agents in solving their problems like nest building, mating and foraging. These interactions have been effectively appropriated to solve large and complex optimization problems [2]. For instance, the behaviors of social insects, such as ants and honey bees, can be patterned by the Ant Colony Optimization (ACO) [3] and Artificial Bee Colony (ABC) [4] algorithms. These methods are generally utilized to describe effective food search behavior through self-organization of the swarm.

In SI, honey bees are one of the most well studied social insects. Furthermore, it is in a growing tendency in the literature for the past few years and it will continue. Many intelligent popular search algorithms are developed such as Honey Bee Optimization (HBO), Beehive (BH), Honeybees Mating Optimization (HBMO), Bee Colony Optimization (BCO), Artificial Bee Colony (ABC) and the Bees Algorithm (BA) [5].

The Bees Algorithm (BA) is one of optimization technique that imitates the foraging behavior of honeybee in nature to solve optimization problems [6]. The main advantage of Bees Algorithm is has power and equilibrate in local search (exploitation) and global random search, (exploration), where both are completely decoupled, and can be clearly varied through the learning parameters [7]. It is very efficient in finding optimal solutions and overcoming the problem of local optima, easy to apply, and available for hybridization combination with other methods [5].

The BA has been successfully applied in many different engineering problems, such as supply chain optimization [8], production scheduling [9], numerical functions optimization [5, 6, 7, 10], solving timetabling problems [11], control system tuning [12], protein conformation search [13], test form construction [14], Placements of FACTS devices [15], pattern recognition [16], robotic swarm coordination [17], data mining [18, 19, 20], chemical process [21], mechanical design [22], wood defect classification [23], Printed Circuit Board (PCB) assembly optimization [24], image analysis [25], and many other applications [26].

The BA has attracted attention of many researchers in different fields since it has been proved to be efficient and robust optimization tool. It can be split up into four components: parameter tuning, initialization of population, exploitative neighborhood or local search (intensification) and exploratory global search (diversification) [26]. However, despite different successful applications of the BA, the algorithm has some limitations and weaknesses. The algorithm efficiency is much influenced by initialization of the different parameters that need to be tuned. Additionally, the BA suffers from slow convergence caused by many repeated iterations in local search [5].

Many different investigations have been made to improve BA performance. Certain of these studies focus on the parameter tuning and setting component [27, 28]. Others developed different concepts and techniques for the local search neighborhood stage [5, 7], or global search stage [9] or for both the local and global stage [29]. Nonetheless, limited interest has been given to the improvement of the initialization stage.

In order to deal with some weaknesses of BA, this paper considers different improvements based on other strategies and procedures. Hybridization of different techniques may improve the solutions quality and enhance the efficiency of the algorithms.

The present work is an extension of the methods presented in [30]. The Firefly Algorithm (FA), which is swarm intelligence metaheuristic for optimization problems that based upon behavior and motion of fireflies [31], is applied to initialize the bee population of Basic BA for a best exploration of research space and start the search from more promising locations. FA is also introduced in the local search part of Basic BA in order to increase exploitation in each research zone. As a result, two BA variants called the Hybrid Firefly Bee Algorithm (HFBA) and Hybrid Bee Firefly Algorithm (HBAFA), respectively.

We also investigate the behavior of local search and global search of the BA by introducing two strategies: memory scheme (local and global memories) to overcome repetition and unnecessarily spinning inside the same neighborhood and avoid revisiting previously visited sites, and adaptive neighborhood search procedure to jump from the sites of similar fitness values, thus improving the convergence speed of the BA to the optimal solution. This improved variant of BA is called, Bees Algorithm with Memory Scheme and Adaptive Neighborhood (BAMS-AN).

The rest of this article is organized as follows. Section 2 provides a description of Basic BA, FA and three improved variants of BA with different strategies. Section 3 presents and discusses the computational simulations and results obtained on benchmark instances, while Section 4 presents our conclusions and highlights some suggestions and future research directions.

2 Materials and methods

2.1 The standard bees algorithm

The Bees Algorithm (BA) is a bee swarm-based metaheuristic algorithm, which is derived from the food foraging behavior of honey bees in nature. It was originally proposed by Pham et al. [6].

The BA starts out by initializing the population of scouts randomly on the space of search. Then the fitness of the points (i.e. solutions) inspected by the scouts is evaluated. The scouts with the highest fitness are selected for neighborhood search (i.e. local search) as “selected bees” [6]. To avoid duplication, a neighborhood called a “flower patch” is created around every best solution; furthermore the forager bees are recruited and affected randomly within the neighborhoods. If one of the recruited bees lands on a patch of higher fitness than the scout, that recruited bee turns into the new scout and participates in the waggle dance in the next generation. This step is called local search or exploitation [7]. Finally, the remaining of the colony bees (i.e. population) is assigned around the space of search scouting in a random manner for new possible solutions. This activity is called global search (i.e. exploration). In order to establish the exploitation areas, these solutions with those of the new scouts are calculated with a number of better solutions being reserved for the succeeding learning cycle of the BA. This procedure is repeated in cycles until it is necessary to converge to the optimal global solution [5].

The Bees Algorithm detects the most promising solutions, and explores in selective manner their neighborhoods to find the global minimum of the objective function. When the best solutions are selected, the BA in its basic version makes a good balance between a local (or exploitative) search and a global (or exploratory) search. The both develop random search [7]. The pseudo-code of the Standard Bees Algorithm is given below in Figure 1.

A certain number of parameters are required for the BA, called: number of the scout bees or sites (n), (m) sites selected for local search among n sites, (e<m) elite sites chosen from the m selected site for a more intense neighborhood search, (nre) recruited bees for the elite sites, (nrb<nre) bees recruited for neighborhood search around other m-e best sites, initial size of all selected patch (ngh) around each scout for neighborhood search (i.e. flower patch), stopping criterion for the algorithm to terminate and number of iterations [5].

From the control settings of algorithm, the bees' population size p can be determined as below:

$$p = ns + e * nre + (m - e) * nrb \tag{1}$$

Bees Algorithm (BA)

```
Initialize population with random solutions by sending ($n$) scout bees.
Iterate until a stopping criterion is met
1.  Evaluate the fitness of the population.
2.  Sort the solutions according to their fitness.
// Waggle dance
3.  Select the highest-ranking $m$ solutions (i.e. sites or patches) for neighborhood search.
4.  Determine the patch of each selected site with an initial size of ($ngh_{init}$).
5.  Recruit $nr = nre$ foragers to each of the $e \leq m$ top-ranking elite solutions.
6.  Recruit $nr = nrb \leq nre$ foragers to each of the remaining $m - e$ selected solutions.
// Local search
7.  For each of the $m$ selected solutions
a.  Create $nr$ new solutions by perturbing the selected solution randomly or otherwise and evaluate their fitness.
b.  Retain the best solution amongst the selected and new solutions.
// Neighborhood Shrinking Strategy
c.  Adjust the neighborhood of the retained solution (i.e. Shrink patches whose neighborhood has achieved no improvement by a shrinking factor ($sf$)).
// Site Abandonment Strategy
d.  Abandon sites where no progress has been made for a number ($stlim$) of consecutive iterations and re-initialize them (Save the location of the abandoned site if it represents the current best solution).
// Global search
8.  Assign the remaining bees ($n - m$) to search randomly and evaluate their fitness.
9.  Form new population ($m$ solutions from local search, $n - m$ from global search)
```

Figure 1: Pseudo code for the Standard Bees Algorithm (BA).

where; ns is (n – m) remaining bees.

For random scouting in the initialization phase as well as for the unselected bees, the following equation is adopted:

$$x_{rand} = x_{\min} + rand * (x_{\max} - x_{\min}) \quad (1)$$

where; rand is a random vector element between 0 to 1, $x_{\max}$, $x_{\min}$ are the upper and lower bound to the solution vector, respectively.

Two additional steps are called when neighborhood search does not improve any fitness enhancement in a neighborhood: the neighborhood shrinking and the site abandonment. The primary objective is to enhance the computation performance and the search accuracy. This implementation is called the Standard Bees Algorithm (Standard-BA) [7].

2.1.1 Neighborhood shrinking strategy

A large value is defined for the initial size of the neighborhoods. For each ai of a = {a1, …,an} is set as follows:

$$a_i(t) = ngh(t) * (\max{}_i - \min{}_i)$$

$$ngh(0) = 1.0 \quad (2)$$

where t represents the t-th iteration of the BA main loop.

The initial size of flower patches is on a wide area to promote the exploration of the search space. Whilst the local search procedure gives better fitness, the size of patches is maintained unchanged, and is then gradually reduced to yield the search more exploited round the local optimal [7]. The ultimate objective of the neighborhood shrinking approach is to make the search progressively concentrated in the proximity of the local peak of fitness. The updating formula is given as follows:

$$ngh(t+1) = 0.8 * ngh(t) \quad (3)$$

A new variant of BA can be obtained by applying the shrinking procedure over Basic BA. This implementation is named the Shrinking-based BA [26]. It can be deducted from the first paper of Pham et al. [6].

2.1.2 Site abandonment strategy

To enhance the efficiency of the local search, the site abandonment procedure is introduced, in which there is no more enhancement of the fitness value of the fittest bee after a predetermined number (stlim) of successive stagnation cycles. If the abandoned site corresponds to the fitness best value, the location of the peak is registered. If no other flower patch will generate a better value of fitness in the rest of the search, the better registered fitness position is taken as the final solution [7].

To sum up, in the literature review of Bees Algorithm, three important implementations could be discovered, which are Basic-BA, Shrinking-based-BA, and Standard-BA.

2.2 The standard firefly algorithm

Firefly Algorithm (FA) is a recent biologically-inspired algorithm originally introduced by [31]. FA imitates the communication behavior of tropical fireflies and the idealized flashing features of fireflies [31].

The primary strengths of FA are namely exploitation and exploration. In addition, the proposal time of this algorithm is relatively short, the parameters needing to be adjusted are quite few due to its simple structure and less complexity without requiring complex evolutionary operations; and the optimizing search ability is also relatively good [32].

FA has been successfully applied to solve different optimization problems such as software testing [33], dynamic multidimensional knapsack problems [34], combinatorial problem [35], mobile robotics [36], network route selection [37], linear antenna array failure correction [38], optimized gray-scale image watermarking [39], automatic control system optimization [40, 41], price budget [42], multi-objective economic emission dispatch solution [43], and so on. Generally, it can get the optimal solution with its exploitation [44].

According to [32], two important things need to be defined in FA: the change of the attractiveness (β) and the formulation of the light intensity or brightness (I).

The strength of attraction is proportional to the intensity of the brighter firefly and the distance between the two fireflies, so the brightness reduces with the distance from its source and the media will absorb the light. The light intensity of a firefly is given according to the value of the objective function. For simplicity, different features of fireflies are idealized into diverse rules detailed in [32]. The descriptive pseudo code of the FA is given in Figure 2.

The attractiveness β between two fireflies can be determined by the following equation:

$$\beta = \beta_0 \exp\left(-\gamma * r^2\right) \quad (4)$$

where; β0 is the attractiveness at r = 0, r is the distance of two fireflies and γ is the light absorption coefficient. γ is the most crucial parameter in FA method, it performs a very important role in determining the convergence speed and how FA behaves. Hence, nearly all the applications it differs from 0.1 to 10 [45].

The distance among any two fireflies i and j at xi and xj, respectively, calculated by the Euclidean or Cartesian distance formulation as follows:

$$r_{ij} = \left\| x_i - x_j \right\| = \sqrt{\sum_{k=1}^{d} \left(x_{i,k} - x_{j,k} \right)^2} \quad (5)$$

Firefly i is attracted to another more attractive (brighter) firefly j and its movement is calculated by equation (7) [31]. In this equation the second part is the attraction, whilst the third is randomization with the control parameter α ∈ [0, 1], and εi is a vector of random numbers being generated from different distributions such as uniform distribution, Gaussian distribution and Lévy flight. The most basic form is εi could be changed by (rand − ½) where rand is a random real number between interval [0 1] [32]. It is interesting that (7) is a random-walk partial to the brighter fireflies. If β0 = 0, it turns into a simple random walk. Furthermore, the randomization term can easily be prolonged to different distributions such as Lévy flights [31].

Firefly Algorithm (FA)
```
Define the objective function of $f(x)$, where $x=(x_1,........,x_d)^T$
Generate initial population of fireflies $x_i$ $(i = 1, 2,..., n)$ randomly.
Determine the light intensity of Ii at $x_i$ via $f(x_i)$
Define $\alpha$, $\beta$ and $\gamma$
**While** (t <MaxIterFA)
    **For** i = 1 :n (all n fireflies)
        **For** j = 1 : n (all n fireflies)
            **If** (Ii <Ij ), Move firefly i towards j by using equation (7); **end if**
            Vary attractiveness with distance r via $exp[-\gamma r]$
            Evaluate new solutions and update light intensity
        **Endfor** j
    **Endfor** i
    Rank the fireflies according to light intensity and find the current global best
**End while**
Post-process results and visualization
```

Figure 2: Pseudo code for the Firefly Algorithm (FA).

$$x_i^{t+1} = x_i^t + \beta_0 e^{-\gamma r_{ij}^2} \left(x_i^t - x_j^t \right) + \alpha \varepsilon_i^t \quad (6)$$

2.3 The improved bees algorithms

In this subsection, three improved variants of BA are presented: Bees Algorithm with Memory Scheme and Adaptive Neighborhood (BAMS-AN), Hybrid Firefly-Bee Algorithm (HFBA) and Hybrid Bee Firefly Algorithm (HBAFA).

2.3.1 Improved bees algorithm with memory scheme and adaptive neighborhood

In this improved variant, two modifications of BA are introduced. The first one is to improve the BA in more efficient and natural manner with memory integration (local and global memories) to avoid revisiting previously visited sites and overcome repetition and unnecessarily spinning inside the same neighborhood. Thus, the search in the BA can prevent searching within infertile areas and can jump from potential local optimums. The second modification is to deploy an adaptive neighborhood search procedure to escape from local optima during the local search process.

The memory is part of the honey bees' nature, but was not included in basic BA. The strategy is to force the bees to stay away from the neighborhoods studied and experienced with no beneficial results. This will enforce the bees to visit different solutions for investigation. This needs the integration of a memory mechanism into the bees to allow them to respect the experiences and not waste their energy and time. This type of memory will be integrated to the scout and the follower bees where these bees could employ information and details about previously visited sites (i.e. positions and finesses). Different structures of arrays are used to save and handle the memory of each bee (scout and follower). The memory contains the information obtained during direct communication with the environment that used to decide the way of patch visiting based on a waggle dance or depend on it. This information is very important and affects the way forager bees (both scout and follower bees) choose the patch to visit and whether it will be based on a waggle dance or not (i.e., choosing a familiar food source). Foragers depend on their own memories to discover particular locations during the visit of food patches repeatedly [16, 19].

The memorized experiences are continuously updated and utilized to lead the further foraging of the bees, contributing to a more effective search procedure than basic BA.

The pseudo-code below shows the procedures added. It is included in the local neighborhood search for follower bees and in the global search for scout bees. This procedure decides if the new position (in different dimensions) of any bee is close to its previous one recorded in its memory.

```
If the bee is a scout then
    r = x1 * $x^d_{max}$ - $x^d_{min}$;
Else if the bee is a follower bee then
    r = x2 * ngh;
End If
While (number_iterations<itmax) do
    If the new position is close to its previous position recorded in the private memory (by using the Euclidian distance) then
        Find a new position randomly;
    End If
End While
```

Figure 3: The pseudo code of the procedures added.

where r is the radius, x^d_{max} and x^d_{min} are the upper and the lower bound to the solution vector respectively, itmax is the maximum number of iterations and ngh is the neighborhood size.

The radius is the Euclidean distance from the field center to the border. It defines the size of field. It is applied so that different patches do not congregate in the same area. As can be seen from Figure 3, the value of radius r is based on the size of the solution space for scouts and the size of ngh for followers, which involves that r, will be greater than ngh for the scouts and less than ngh for followers.

In addition, this enhanced variant of BA introduces an improvement to the Basic-BA by using two approaches simultaneously; adaptive change to the size of neighborhood of a selected patch and abandonment of site procedure, in the local search part. The neighborhood shrinking method bears similarities with Simulated Annealing procedure. The flower patches are initially set to a large region to foster the exploration and favor large “jumps” across the fitness landscape. Then, the neighborhood shrinking method is applied to a patch after predefined number of consecutive stagnation cycles. If shrinking the patch size does not result in any improvement in the local search after a predefined number of iterations, an enhancement procedure is applied. The enhancement procedure is applied for a number of consecutive iterations of no progress. Finally, after this number of iterations, the patch is assumed to be trapped within a local minimum (or peak) and it is abandoned, and the scout bee is randomly re-initialized. This process is known as site abandonment. If the abandoned location corresponds to the best-so-far solution, it is stored temporarily. If no better solution is subsequently found, the saved best solution is taken as the final one. The objective is to improve the local intensification, evade from local optima and speed the convergence to the optimal solution.

2.3.2 Hybrid firefly-bee algorithm

In initialization step of BA, the scout bees fly at random to initial resources. This stage is performed by distributing the scouts uniformly in random manner on search space. The initial position of scout bees relative to the optimal source may influence the level of optimality of other algorithm steps. Therefore, the population initialization of BA is critical and important factor because it can significantly influence the convergence speed to the optimal target, the quality of the resource found and the stability [26].

In the second variant of BA, an enhancement is introduced based on the Firefly Algorithm (FA) to improve the initialization stage of the BA. The proposed approach is called Hybrid Firefly Bee Algorithm (HFBA).

Although the BA has a combination of both local and global search abilities, it nonetheless has some drawbacks such as a high level of randomness and processing time. The results generated by the standard BA are subject to the random initialization step. This may not contain a sufficient amount of information and details about the space surrounding. Nevertheless, due to its randomness, it is infeasible for random initialization to obtain a high-quality of initial solution regularly. It affects convergence speed, accuracy of final solution and stability. Practically, we should be searching in all directions simultaneously; put differently, initial solutions must be distributed uniformly through the overall search space. Hence, to obtain a better view of the surroundings of all random positions visited by each scout bee and to begin a local search from more promising sites, a FA search step is introduced during the first scouting procedure. The aim of the FA is to enhance the performance of the initialization stage of the BA, which will offer a better exploration of the search-space and empower the global performance of the method. High exploration would provide a high chance to obtain a solution which is close to global optima. FA based search process encourages the fireflies to look for new areas including some lesser explored areas and improves the fireflies’ ability to explore the large search space. In other words, the FA not only integrates the self-improving approach with the current search space, but it also integrates the enhancement among its own space from the prior phases [30, 44]. This contributes to the discovery of better fitness valued patches from where the neighborhood search will be performed since if the method begins its search from a promising location, it is evident that it will have better chance to converge to the global optima. In addition, FA has fast convergence capability.

In the proposed algorithm, FA explores the search place to either isolate the more advantageous region of the search space, and then to escape from trapping into local optima and enhance global search, it is introduced BA to explore search space (starting with the best solutions found by FA) and discover new better solutions. The pseudo-code and flow chart of the proposed hybrid HFBA are given in Figures 4 and 5.

The main steps of the novel hybrid algorithm (HFBA) can be summarized as follows: First, The initial population of n fireflies should be generated, and then, each firefly ought to be assigned the random position and the fitness (light intensity) for that position should also be calculated. Then, a new function is introduced to increase the convergence by calculating the step α as mentioned in Equation (8). This additional function is designated to modify the basic value of α parameter and design a dynamic adjusting scheme. The idea to minimize randomness is to maximize the algorithm's search capability [30, 46]. The step α can be determined as following:

$$\alpha(t) = 0.4/\left(1 + \exp\left(0.015 * \left(t - MaxIterFA\right)/3\right)\right)$$
(8)

where; t and MaxIterFA represent current and maximum number of iterations, respectively. The next step is the process of moving each firefly in research space towards other brighter and attractive firefly. The position of each firefly is updated by Equation (7). The research process of firefly is controlled by the maximum number of iterations MaxIterFA. When the main loop of FA ends, the n best results generated from FA are communicated to the BA as an initial population of n scout bees. Thus, the BA will start with a population, and the rest of the method behaves like the standard (normal) BA algorithm in Figure 1.

To sum up, the main difference between the novel proposed HFBA and the Standard BA lies in the first step of initialization, with the introduction of a new mechanism (FA) for generating the initial population of scout bees. Initialization phase of the improved BA is controlled by firefly fitness and optimized by α and MaxIterFA, which are important HFBA control parameters.

2.3.3 Hybrid bee firefly algorithm

In the third improved variant of BA, the FA algorithm is called to improve the local search of Bees Algorithm. We name this proposed algorithm by Hybrid Bee Firefly Algorithm (HBAFA).

Disadvantage of BA is that it can be relatively weak in neighborhood search activities, and it suffers from slow convergence phenomena caused by the repetition of unneeded similar process in the local search. The repetitive iteration of the algorithm triggers a supplementary computational time in producing the solution. This makes the whole process slow. To overcome these problems, the intelligent algorithm FA is introduced, so as to increase the speed of convergence and avoid running the method with additional iteration without any improvements.

The basic idea behind the proposed HBAFA is to introduce a FA search strategy into local search of BA. The main difference between the Standard BA and the hybrid HBA-FA is the manner of how the different selected patches types to be exploited. The Standard BA employs random mutations as the only means of neighborhood search, whereas in HBAFA, FA has been applied to improve the local search ability of BA and reconfigures the neighborhood dance search as a FA search. FA enhances the capability to exploit the whole local search space in order to either isolate the most promising solution of each selected site. Figure 6 shows the flow chart of the Hybrid HBAFA. The pseudo code of the improved local search part of BA is presented in Figure 7.

The general steps of HBAFA algorithm are as follows. The proposed algorithm uses the Standard BA as a core, where FA works as a local search to refine the best sites found by the scout bees. Firstly, the best sites of BA are given and the control parameters of FA are set. Then, the FA algorithm starts its search process in each best site and it is run (as a normal FA algorithm in Figure. 2) until its stopping criterion is met. The initial population of nr fireflies (i.e. nre fireflies per site for elite sites and nrb fireflies per site for best sites) should be generated and then, each firefly ought to be assigned the random position in the neighborhood of each selected site, and the fitness (light intensity) for that position should also be calculated. The next step is the process of moving each firefly in research space of each selected site towards other brighter and attractive fireflies. The position of each firefly is updated by equation (7). The value of the parameter alpha (α) impacts the algorithm convergence. Therefore, the initial value of step size α is modified and reduced according to equation (8) to maximize the algorithm's search [30].

The FA search process is essential because there might be better solutions than the original solution in the neighborhood region. The research procedure of FA is controlled by the maximum number of iterations MaxIterFA. When the main loop of FA ends, it returns a single global best position. If it lands in a position of higher fitness than the scout bee of the selected site, that forager is maintained as the new scout bee. When local search does not improve any fitness enhancement in a neighborhood, the neighborhood shrinking and the site abandonment procedures are called.

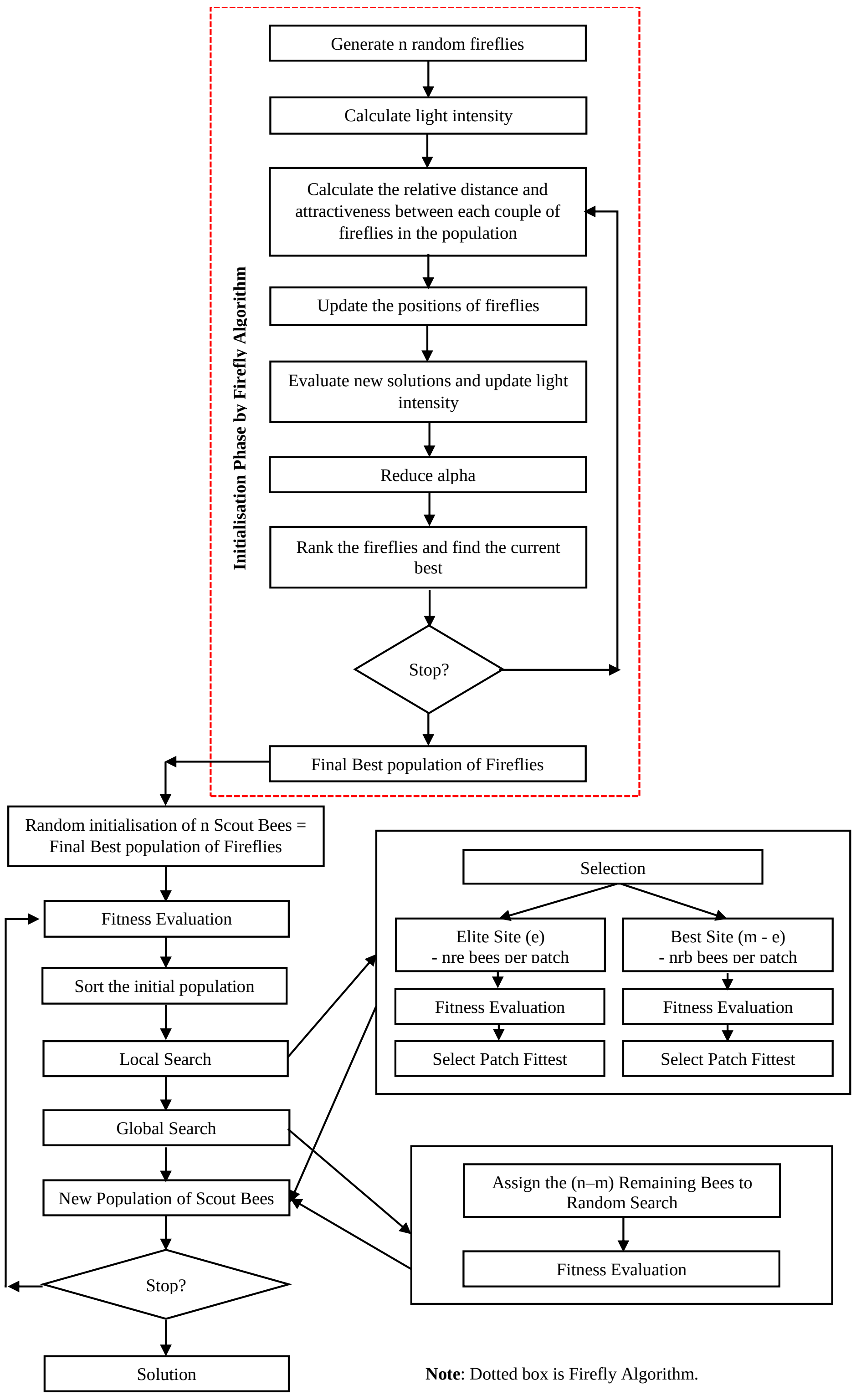
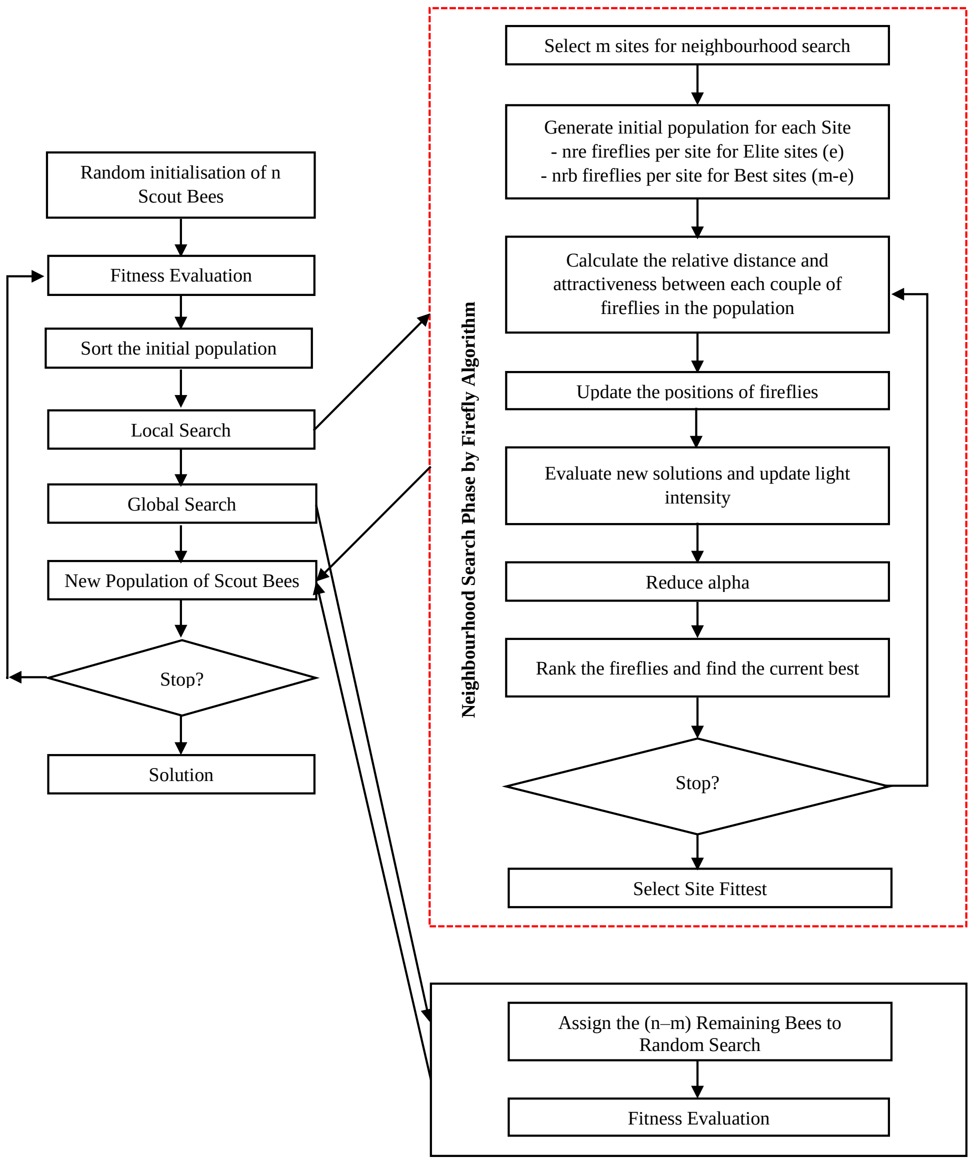

Initialisation Phase by Firefly Algorithm
Generate n random fireflies
Calculate light intensity
Calculate the relative distance and attractiveness between each couple of fireflies in the population
Update the positions of fireflies
Evaluate new solutions and update light intensity
Reduce alpha
Rank the fireflies and find the current best
Stop?
Final Best population of Fireflies
Random initialisation of n Scout Bees = Final Best population of Fireflies
Fitness Evaluation
Sort the initial population
Local Search
Global Search
New Population of Scout Bees
Stop?
Solution
Selection
Elite Site (e) - nre bees per patch
Best Site (m - e) - nrb bees per patch
Fitness Evaluation
Fitness Evaluation
Select Patch Fittest
Select Patch Fittest
Assign the (n–m) Remaining Bees to Random Search
Fitness Evaluation
Note: Dotted box is Firefly Algorithm.

Figure 4: Flow chart of Hybrid Firefly-Bee Algorithm (HFBA).


Random initialisation of n Scout Bees
Fitness Evaluation
Sort the initial population
Local Search
Global Search
New Population of Scout Bees
Stop?
Solution
Neighbourhood Search Phase by Firefly Algorithm
Select m sites for neighbourhood search
Generate initial population for each Site
- nre fireflies per site for Elite sites (e)
- nrb fireflies per site for Best sites (m-e)
Calculate the relative distance and attractiveness between each couple of fireflies in the population
Update the positions of fireflies
Evaluate new solutions and update light intensity
Reduce alpha
Rank the fireflies and find the current best
Stop?
Select Site Fittest
Assign the (n–m) Remaining Bees to Random Search
Fitness Evaluation

Note: Dotted box is improved local search of BA by FA.

Figure 6: Flow chart of Hybrid Bee Firefly Algorithm (HBAFA).

```
// Improved Local Search phase of BA
7.  For each of the $m$ selected solutions
a.  // Perform neighborhood search using Firefly Algorithm
Generate initial population of fireflies $x_i(i = 1, 2,\dots, nr)$ randomly (i.e. nre fireflies per site for elite sites and nrb fireflies per site
for best sites).
Determine the light intensity of Ii at xi via the objective function $f(x_i)$
Define $\alpha$, $\beta$ and $\gamma$
While (t <MaxGeneration)
    For i = 1 :nr (all nr fireflies)
        For j = 1 : nr (all nr fireflies)
            If (Ii <Ij), Move firefly i towards j by using equation (7); end if
            Vary attractiveness with distance r via exp[− γr]
            Evaluate new solutions and update light intensity
        Endfor j
    Endfor i
    Reduce alpha (α) according to Eq. (8)
    Rank the fireflies according to light intensity and find the current global best
End while
Output optimum solution value and its locate
b.  Retain the best solution amongst the selected and new optimum solution.
c.  Adjust the neighborhood of the retained solution (i.e. Shrink patches whose neighborhood has achieved no improvement
by a shrinking factor (sf)).
d.  Abandon sites where no progress has been made for a number (stlim) of consecutive iterations and re-initialize them
(Save the location of the abandoned site if it represents the current best solution).
```

Figure 7: Pseudo code of the Improved Local Search part of BA.

3 Experimental setup

3.1 Benchmark functions

To measure the performance of the improved BAs, some well known continuous type benchmark problems were selected. Each of these functions has its own properties whether it unimodal, multimodal, separable, non-separable, and multidimensional, so obtained results illustrate strengths and weaknesses of the algorithm in different situations.

Some functions are unimodal which mean there is only one global optimum. It can easily locate the point; thus they are useful for evaluating the exploitation ability of an optimization algorithm. If a function has two or more local optima, this function is called multimodal. Multimodal functions are suitable for benchmarking the global exploration capability of an optimization algorithm. If the exploration operation of a method is mediocre that it cannot perform an efficient search in whole space, this method gets trapped in local minima. Multimodal benchmarks are difficult for algorithms as well as benchmarks having flat surfaces because the flatness of the benchmark does not offer the algorithm any kind of information to lead the search process towards the minima. A function is regarded as separable if it can be expressed as a sum of function just from one variable. Non-separable benchmarks cannot be expressed in this form. Consequently, non-separable benchmarks are considerably more difficult compared to the separable benchmarks. Meanwhile, dimensionality of the search area reflects the number of parameters to be optimized. It is an essential characteristic that determines the difficulty of the problem [4].

The Martin and Gaddy and Hypersphere are unimodal benchmarks which are fairly simple for optimization tasks. The Easom function is a unimodal test function. It is characterized by a single narrow mode and the global minima have a small region relative to the search space. The Goldstein & Price is a multimodal function (i.e., including multiple local optima, but just one global optimum) and it is easy optimization task with only two variables. Schaffer benchmark is symmetric complex multimodal function that has a global optimum very close to a local optimum. The Rosenbrock function is complex and unimodal and the minimum is at the lowest part of a long, narrow and parabolic shaped valley. To find the valley is trivial and it can be easily discovered with a few repetitions of an optimization methods, however, to locate the exact global minima is quite difficult. Rastrigin, Griewank and Ackley have an overall unimodal behavior and local multi-modal pockets created by a cosinusoidal “noise” component. The Schwefel function is well-known to be a complex and tough problem. It is a multimodal function and the global minimum is geometrically distant from the second best local minima. Consequently, the optimization algorithms are potentially susceptible to convergence in the incorrect direction. Thus, the algorithm never gets the same value on the same position. Algorithms that fail to tackle this function will do poorly in real world problems with noisy data. The combination of these characteristics determines the complexity of the continuous functions.

No	Benchmark	Function	Min	Interval	P
1	Easom 2D	$\min F = -\cos(x_1)\cos(x_2)e^{\left[(x_1-\pi)^2-(x_2-\pi)^2\right]}$	0	[-100, 100]	UN
2	Goldstein & Price 2D	$\min F = \left[1+(x_1+x_2+1)^2(19-14x_1+3x_1^2-14x_2+6x_1x_2+3x_2^2)^2\right] \times \left[30+(2x_1-3x_2)^2(18-32x_1+12x_1^2+48x_2-36x_1x_2+27x_2^2)\right]$	3	[-2, 2]	MN
3	Martin & Gaddy 2D	$\min\ F = (x_1-x_2)^2+((x_1+x_2-10)/3)^2$	0	[-20, 20]	
4	Schaffer 2D	$\min F = 0.5+\frac{\left(\sin\sqrt{x_1^2+x_2^2}\right)^2-0.5}{\left[1.0+0.001(x_1^2+x_2^2)\right]^2}$	0	[-100, 100]	MN
5	Schwefel 2D	$\min F = \sum_{i=1}^{2}\left(-x_1\sin\left(\sqrt{\lvert x_i\rvert}\right)\right)$	0	[-500, 500]	MS
6	Ackley 10D	$\min F = -20e^{-0.2\sqrt{\frac{\sum_{i=1}^{10}X_i^2}{10}}} - e^{\frac{\sum_{i=1}^{10}\cos(2\pi X_i)}{10}} + 20 + e$	0	[-32, 32]	MN
7	Griewank 10D	$\min F = \frac{1}{4000}\sum_{i=0}^{10}(x_i-100)^2-\prod_{i=0}^{10}\cos\left(\frac{x_i-100}{\sqrt{i+1}}\right)$	0	[-600, 600]	MN
8	Hypersphere 10D	$\min F = \sum_{i=1}^{10}x_i^2$	0	[-100, 100]	US
9	Rastrigin 10D	$\min F = 100+\sum_{i=1}^{10}\left(x_i^2-10\cos(2\pi x_i)\right)$	0	[-5.12, 5.12]	MS
10	Rosenbrock 10D	$\min F = \sum_{i=1}^{10}\left[100(x_i^2-x_{i+1})^2+(1-x_i)^2\right]$	0	[-50, 50]	UN

Table 1: Summary characteristics of test functions used. D: Dimension, Min: Global Minimum, P: Properties, U: Unimodal, M: Multimodal, S: Separable, N: Non-Separable.

Interval, full equation, dimensions, theory global optimal solutions and properties of these functions are shown in Table 1 as reported in [4]. Readers are referred to [47] for more details on the benchmark functions used in present investigation and its characteristics.

3.2 Performance measures

Performance assessment of the different algorithms is based on two metrics: namely, the accuracy and the average evaluation numbers and results were compared to Standard Bees Algorithm (BA), Firefly Algorithm (FA), and other well known optimization techniques such as Particle Swarm Optimization (PSO), Evolutionary Algorithm (EA). These are given in Tables 3, 4 and 5.

The accuracy of algorithms (E) was defined as the difference of the fittest solution obtained by algorithms and the value of the global optimum. It was chosen as a performance indicator to ascertain the quality of the solutions obtained. According to this approach, the more accurate results are closer to zero. However, the number of function evaluations (NFE), which is the number of times any benchmark problem had to be assessed to grasp the optimal solution, was used to evaluate the convergence speed. If the algorithm could not find E less than 0.001, the final fitness value is recorded with maximum function evaluations. Lower value of NFE for a method on a problem means the method is faster in solving that problem.

The process will run until stopping criteria are met. Each time, the optimization algorithm is run until the accuracy (error) is less than 0.001, or the maximum number of iterations (5000) is elapsed. For each configuration, 50 independent minimization trials are performed on each benchmark function.

3.3 Parameters settings

In order to obtain a reliable and fair comparison amongst standard and novel improved BA, the same parameters and values were utilized for all benchmarks to achieve acceptable results within the required tolerance without careful tuning, except for the parameter α, β and γ which were only set for the two variants of BA based on FA and FA. The Firefly Algorithm (FA) was implemented in this study according to the method described in [31]. The parameter configuration recommended by [32] is used. The parameters of the implemented BA used in this paper have been empirically tuned and the optimal parameter settings are used to find the optimal solution. The Standard BA implemented in this work is called BA1.

The simulation results of our experiment are compared with ones reported in [7]. This comparison was carried out between the BA1, FA and the proposed variants of BA with the standard BA introduced in [7] (noted by BA2) and other well-known optimization techniques such as EA and PSO. Pham and Castellani (2009) were analyzed in [7] the learning results of EA, PSO and BA2 algorithms, and the parameter setting that gives the most consistent performance across all the benchmarks was found for each algorithm. The parameter settings of the algorithms are given in Table 2. Given an optimization algorithm, the No Free Lunch Theorem of optimization (NFLT) entails that there is no universal tuning that guarantees top performance on all possible optimization problems [48].

Algorithm	Parameters	Value
BA1, BAMS-AN, HBAFA, HFBA - Common Parameters	Scout bees (n)	12
	Elite sites (e)	2
	Best sites (m)	6
	Recruited elite (nre)	29
	Recruited best (nrb)	9
	Stagnation limit (stlim)	10
	Neighborhood size (ngh_{init})	(Search range)/2
	Shrinking factor (sf)	0.8
	Evolution cycles (ec)	5000
FA, HFBA, HBAFA - Common Parameters	Population size	40
	Initial attractiveness (β0)	1
	Minimum value of beta (βmin)	0.2
	Light absorption coefficient (γ)	1
	Control parameter (α)	0.2
	FA cycles (MaxIterFA)	12500
HFBA & HBAFA	FA cycles in inner loop of HFBA	[3 10]
	FA cycles in inner loop of HBA-FA	5
BA2 [7]	Scout bees (n)	11
	Elite sites (e)	2
	Best sites (m)	6
	Recruited elite (nre)	30
	Recruited best (nrb)	10
	Stagnation limit (stlim)	10
	Neighborhood size (ngh_{init})	Not presented
	Shrinking factor (sf)	0.8
	Evolution cycles (ec)	5000
EA [7]	Population size	100
	Evolution cycles (max number)	5000
	Children per generation	99
	Mutation rate (variables)	0.8
	Mutation rate (mutation width)	0.8
	Initial mutation interval width a (variables)	0.1
	Initial mutation interval width p (mutation width)	0.1
PSO [7]	Population size	100
	Connectivity (no. of neighbors)	2
	Maximum particle velocity (u)	0.05
	c1	2.0
	c2	2.0
	wmax	0.9
	wmin	0.4
	PSO cycles	5000

Table 2: Parameters setting of the algorithms.

The algorithms developed in this study were implemented using Octave programming language and all experiments were performed on Intel Core i3-370M 2.4 GHz and 4 GB RAM running a 64-bit operating system.

4 Results and discussion

In this work, initialization stage, local search and global search parts of BA were investigated. The attention was on improving the performance of the BA by increasing the accuracy and the speed of the search.

The comparisons were carried out between improved variant of BAs (BAMS-AN, HFBA and HBAFA), the standard BA (BA1) and FA. Then, the performance of these algorithms is compared against other well-known optimization techniques presented in [7] such as standard BA (BA2), EA and PSO. The same stopping criterion is used for all the algorithms (see Section. 3.2).

The Average accuracy (μE) and their standard deviation (σE), and the mean numbers of function evaluations (Mean) and standard deviation (Std) for 50 runs are compared in Tables 3 and 4, respectively. Bold values represent the best results. The second best results are in italics. Additionally, Table 5 displays the percentage of improvements in term of reducing the number of evaluations (NFE) when comparing the variants BAs with FA and the Standard BA.

First, we compare the performance of our implemented BA (BA1) to the Standard BA exposed by Pham and Castellani (BA2) in [7]. The two algorithms use different combinations of parameters (Table 2). After finishing the simulation, it is found from Table 3 that the two parameter combinations are capable to find the global optimum on 7 cases out of 10 benchmark functions. For the rest, the average accuracy (μE) found for the Rastrigin function with 10 dimensions was better for the BA1 compared with the BA2 with 7.8601 against 8.8201 respectively. However, BA2 performed slightly better than BA1 on Rosenbrock 10D with 0.0293 against 0.1093, respectively. For Griewank 10D, the result is almost slightly the same for both.

In order to examine and compare the convergence speed of these algorithms, the average number of function evaluations (Mean) is considered. On the other hand, evaluate which parameter combination is better than another. Parameter combination with the minimum number of function evaluations achieving global optimum over all benchmarks and all dimensions is distinguished as the best. It is immediately obvious from Table 4, that BA1 outperformed BA2, except for Schaffer 2D and Griewank 10D functions. Hence, the combination of parameters used for our implementation of BA (BA1) algorithm gave the best results in terms of number of function evaluations. For this reason, it has been selected for the proposed variants of BA. This suggests that if a user faces a problem, this parameter combination ought to be used as default setting.

The second comparison of performances is achieved between the improved algorithms (BAMS-AN, HBAFA, HFBA), BA1 and FA and others well-known optimization algorithms EA, PSO and BA2.

The algorithms are tested first for accuracy, and compared on the benchmark functions. According to Table 3, BA and its variants found the minimum results for the most of the functions with good standard deviations. BAMS-AN is the best performing approach which have been successful in achieving the minimum error in 8 functions out of 10 benchmarks followed by standard HBAFA, HFBA and BA in 7 out of 10 functions. Each of PSO, EA and FA has been capable of obtaining the theoretical value of 0 in, respectively, 6, 5 and 4 functions out of 10 benchmarks.

BAMS-AN did not reach the minimum average accuracy only in Schwefel 2D, Rastrigin 10D functions while the other variants of BA (Standard BA, HBAFA and HFBA) are not capable of finding function minimum on Griewank, Rastrigin and Rosenbrock benchmarks with 10 dimensions. PSO cannot find the global minimum for Schwefel 2D, Rastrigin 10D, Griewank 10D and Rosenbrock 10D functions. In addition to these benchmarks, EA is not capable of finding function minimum on Schaffer 2D benchmark and FA on Easom 2D, Schwefel 2D and Ackley 10D benchmark functions.

All BA algorithms managed to achieve the theoretical optimal value with the Schwefel 2D function except for BAMS-AN where the mean value obtained is 0.0003 which is very close to the theoretical optimum value. However, BAMS-AN can find the optimal values for Griewank 10D and Rosenbrock10D functions or near optimal value for Rastrigin 10D function with 0.0002, while the other variants of BA and other algorithms fail.

HBAFA comes in second place (after BAMS-AN) and outperformed other compared methods (EA, PSO, FA, BA1, BA2 and HFBA) when solving the Rosenbrock 10D function with 0.0234 which is closest to the theoretical optimum value of 0, followed by HFBA with 0.1062. BA2 and BA1 become the third and the fourth best algorithm with 0.0241 and 0.1093, respectively. However, the results found using the FA and EA were not good and far from the theoretical optimal value (0.00) with 10.1469 and 61.5213, respectively. The Rosenbrock 10D benchmark was hard for the EA and the FA algorithms.

The Rastrigin 10D function proved to be difficult tasks, particularly for the EA and none of the algorithms achieved the best minimization performance. HBAFA has become the second best algorithm after BAMS-AN algorithm with 2.9848 and PSO has become the third best with 4.8162. The best fourth one is FA where it managed to achieve 6.0693.

The second best results for the Griewank benchmark function in 10 dimensions are when applying BA2 with 0.0089. HFBA, HBAFA and BA1 perform almost the same on Griewank 10D function with 0.0120 but slightly better than FA with 0.0178.

The results in Goldstein & Price 2D, Martin & Gaddy 2D and Hypersphere 10D functions indicated that all algorithms have achieved the optimal values. Therefore, it is noticeable that BA, HBAFA and HFBA share the only performing approaches in Schwefel 2D function where all of them managed to find the theoretical optimal solution.

For Easom 2D function, all algorithms managed to achieve the theoretical optimal value, except for FA where the mean value obtained is -0.7996.

Besides the average error, the assessment of the performance involves also the average number of evaluations. To compare the convergence speed of the algorithms (FA, BA1, BAMS-AN, HBAFA and HFBA), we calculated and compared the means and standard deviation of number of evaluations (Mean and Std.) generated by each algorithm after 50 runs.

It can be clearly observed in Table 4 that the implementation of BA that utilizes the memory scheme and adaptive neighborhood search (BAMS-AN) achieved the smallest expected numbers of function evaluations (the smallest values of Mean) on 6 out of the 10 tested problems, followed by the modified BA by improving the local search part with FA (HBAFA) on 4 cases out of 10. The proposed BAMS-AN performed significantly better than the other methods on most of the test functions such as Easom 2D, Goldstein & Price 2D, Griewank 10D, Hypersphere 10D, Rastrigin 10D and Rosenbrock 10D (see Table 4). However, the results found from Martin & Gaddy 2D, Schaffer 2D, Schwefel 2D and Ackley 10D using the HBAFA algorithm were better than the proposed BAMS-AN, basic BA and other approaches. Thus, it can be concluded that BAMS-AN and HBAFA algorithms were able to converge to the optimal solution much faster than HFBA, BA, FA and other approaches.

The third best result is for HFBA algorithm which was capable to reduce NFE and performed better in 8 and 7 out of 10 benchmark functions compared to BA1 and BA2, respectively. BA2 produces better results than HFBA on Schwefel 2D, Griewank 10D and Rosenbrock 10D benchmarks. In the meantime, BA1 is selected as the best performing method in only Schwefel 2D and Griewank 10D functions compared to HFBA algorithm.

Comparing HFBA, BA1 and with FA, PSO and EA algorithms, it is apparent that on the most benchmarks, the majority of the results for HFBA and BA were found to be better than the results of the other algorithms, except for Rastrigin 10D function where PSO followed by FA excelled better.

On the whole, BAMS-AN and HBAFA can be classified as the first and the second best performers in terms of reduced number of evaluations, respectively, followed by HFBA and BA. From the observed Table, it is also obvious that PSO, EA and FA have a fairly low convergence rate during the entire process compared with BA variants but PSO performs slightly better than EA and FA.

Table 5 presents the percentage of improvements in term of reducing NFE when comparing the enhanced variants of BA with the Standard BA.

Based on these results, BAMS-AN is the most efficient approach. If NFE of all the functions were totaled, BAMS-AN is capable to minimize the total NFE to 86.39% and 52.62% compared to FA and Standard BA, respectively. The improvement percentage shows the superiority of the BAMS-AN algorithm and how the integration of memory scheme in local and global

No	**Benchmark**	**EA**		**PSO**		**FA**		**BA2**		**BA1**		**BAMS-AN**		**HFBA**		**HBAFA**	
		µE	σE	µE	σE	µE	σE	µE	σE	µE	σE	µE	σE	µE	σE	µE	σE
1	Easom 2D	0.0000	0.0000	0.0000	0.0000	-0.7996	0.4039	0.0000	0.0000	0.0000	0.0000	0.0000	0.0002	0.0000	0.0000	0.0000	0.0000
2	Goldstein & Price 2D	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0001	0.0000	0.0000	0.0000	0.0000
3	Martin &Gaddy 2D	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
4	Schaffer 2D	0.0009	0.0025	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
5	Schwefel 2D	9.4751	32.4579	4.7376	23.4448	59.2194	64.4283	0.0000	0.0000	0.0000	0.0000	0.0003	0.0007	0.0000	0.0000	0.0000	0.0000
6	Ackley 10D	0.0000	0.0000	0.0000	0.0000	0.0010	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0003	0.0000	0.0000	0.0000	0.0000
7	Griewank 10D	0.0210	0.0130	0.0199	0.0097	0.0178	0.0223	***0.0089***	***0.0059***	0.0120	0.0081	**0.0000**	**0.0001**	0.0120	0.0075	0.0127	0.0080
8	Hypersphere 10D	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
9	Rastrigin 10D	17.4913	7.3365	4.8162	1.4686	6.0692	2.7579	8.8201	2.2118	7.8601	2.1010	**0.0002**	**0.0004**	7.2632	2.1010	***2.9848***	***0.8287***
10	Rosenbrock 10D	61.5213	132.6307	1.7879	1.5473	10.1469	39.2052	0.0293	0.0068	0.1093	0.5627	**0.0000**	**0.0003**	0.1062	0.5637	***0.0234***	***0.0022***

Table 3: Accuracy of improved BA algorithms compared with FA and BA and other well-known optimization techniques.

No	**Benchmark**	**EA**		**PSO**		**FA**		**BA2**		**BA1**		**BAMS-AN**		**HFBA**		**HBAFA**	
		Mean	Std.	Mean	Std.	Mean	Std.	Mean	Std.	Mean	Std.	Mean	Std.	Mean	Std.	Mean	Std.
1	Easom 2D	36440	28121	97136	45642	189158.40	159102.21	3866	819	3326.9	415.2	**2126**	**2260**	3316.90	469.69	***2658***	***858***
2	Goldstein & Price 2D	5816	2259	4836	2361	57360	26870.34	2714	454	2658.4	383.4	**1 522**	**660**	2568.4	372.1	***2332***	***479***
3	Martin &Gaddy 2D	3248	1602	2512	781	12729.60	9644.23	2248	329	2180.3	363.7	2130	3034	***1992.8***	***585.4***	**880**	**532**
4	Schaffer 2D	219376	183373	35474	27151	121040.80	100516.67	27890	27335	41472.7	38237.9	***12618***	***4330***	37077.2	34805.7	**10298**	**11777**
5	Schwefel 2D	51468	133632	84572	90373	301726.40	193824.47	5006	2110	***3963.2***	***590.1***	124108	0	4027.1	397.9	**2390**	**1080**
6	Ackley 10D	50344	3949	261608	9165	497459.20	4413.18	12186	3553	11836.2	3532.3	18398	7302	***10402.2***	***516.4***	**9938**	**422**
7	Griewank 10D	490792	65110	497714	16164	474366.40	43760.27	***447064***	***126512***	460894.8	102457.2	**245 426,34**	**198326,6**	470787.2	118.6821	455307	127061
8	Hypersphere 10D	36376	2736	223082	10872	356958.40	6627.86	8288	403	8212	425	**5944**	**3926**	7670.02	384.4	***6962***	***332***
9	Rastrigin 10D	500000	0	500000	0	500000	0.00	500000	0.00	500180	9.9	**15 118**	**14186**	500300	7.92	500000	0.00
10	Rosenbrock 10D	500000	0	500000	0	500000.00	0.00	500000	0	500000	0	**19 642**	**6666**	491924.1	57954.4	500000	0.00

Table 4: Mean number of function evaluations of improved BA algorithms compared with FA and BA and other well-known optimization techniques.

searches with adaptive neighborhood search procedure in basic BA affects the results. The second best algorithm is HBAFA with the average percentage improvement of 67.84% and 23.94% compared to FA and BA, respectively. This variant of BA uses FA to improve the local search stage of Basic BA. On average, the capability to reduce the total NFE when applying Firefly Algorithm in initialization step of BA is 64.42% and 3.95% compared to FA and Standard BA respectively, which ranks HFBA in the third place.

Therefore, the enhanced variants of BA delivered a highly significant improvement in terms of overall performance. An interesting finding is that the introduced strategies and procedures helped BA to converge to good solutions quickly and robustly.

No	Benchmark	Improvement % BAMS-AN		Improvement % HFBA		Improvement % HBAFA	
		FA	BA	FA	BA	FA	BA
1	Easom 2D	**98.88**	**36.08**	**98.25**	**0.30**	**98.59**	**20.11**
2	Goldstein & Price 2D	**97.35**	**42.74**	**95.52**	**3.39**	**95.93**	**12.28**
3	Martin &Gaddy 2D	**83.27**	**2.31**	**84.34**	**8.60**	**93.09**	**59.64**
4	Schaffer 2D	**89.58**	**69.58**	**69.37**	**10.60**	**91.49**	**75.17**
5	Schwefel 2D	**58.867**	**-96.81**	**98.67**	-1.61	**99.21**	**39.70**
6	Ackley 10D	**96.30**	**55.44**	**97.91**	**12.12**	**98.00**	**16.04**
7	Griewank 10D	**48.26**	**46.75**	**0.75**	-2.15	**4.02**	**1.21**
8	Hypersphere 10D	**98.33**	**27.62**	**97.85**	**6.60**	**98.05**	**15.22**
9	Rastrigin 10D	**96.98**	**96.98**	-0.06	-0.02	**0.00**	**0.04**
10	Rosenbrock 10D	**96.07**	**96.07**	**1.62**	**1.62**	**0.00**	**0.00**
	Average improvement (%)	**86,39**	**52,62**	**64,42**	**3,95**	**67,84**	**23,94**

Table 5: Percentage of improvements of BA variants in term of Mean numbers of function evaluations in comparison to the BA and FA.

5 Conclusion

In this paper, enhancements to the Bees Algorithm (BA) have been presented for solving continuous optimization problems. The basic BA was modified first to find the most promising patches, by using memory scheme in order to avoid revisiting previously visited sites, thus increasing the accuracy and the speed of the search, followed by adaptive neighborhood search procedure to escape from local optima during the local search process. This implementation is called BAMS-AN. The second improved version of BA, called HBAFA, uses Firefly Algorithm (FA) to update the positions of recruited bees in the local search part of BA, thus improving the convergence speed of the BA to good solutions. The third variant of BA, i.e. HFBA, developed a new strategy based on Firefly Algorithm (FA) to initialize the population of bees in the Basic BA, enhance the population diversity and start the search from more promising locations.

We evaluated the improved algorithms on several widely used benchmark functions and compared the results with those from the basic BA, FA and other state-of-the-art algorithms found in the literature. These benchmarks cover a range of characteristics including unimodal, multimodal, separable, and inseparable. The results have shown that BAMS-AN followed by HBAFA could track the optimal solution and give reasonable solutions most of the time. By including the improvements, both the search speed was improved and more accurate results were obtained. The comparisons among BA-based algorithms showed that BAMS-AN outperformed HBAFA, HFBA and the conventional BA. The experiments have also indicated that the improved variants of BA performed much better than the standard BA, PSO, FA and EA algorithms.

Testing the improved algorithms further on real world optimization problems and looking for algorithmic enhancements remains as future work.

6 References

[1] Mehmet Polat Saka, O. Hasançebi, and Zong Woo Geem. Metaheuristics in structural optimization and discussions on harmony search algorithm. *Swarm and Evolutionary Computation*, 28: 88-97, 2016. https://doi:10.1016/j.swevo.2016.01.005

[2] Lale Özbakir, and Pinar Tapkan. Bee Colony Intelligence in Zone Constrained Two-Sided Assembly Line Balancing Problem. *Expert Systems with Applications*, 38(9): 11947-11957, 2011. https://doi:10.1016/j.eswa.2011.03.089

[3] Marco Dorigo, and Gianni Di Caro. Ant Colony Optimization: A New Meta-heuristic. *Proceedings of the 1999 Congress on Evolutionary Computation-CEC99 (Cat. No. 99TH8406)*, Washington, DC, USA, https://doi:10.1109/cec.1999.782657

[4] Dervis Karaboga, and Bahriye Akay. A Comparative Study of Artificial Bee Colony Algorithm. *Applied Mathematics and Computation*, 214(1): 108-132, 2009. https://doi:10.1016/j.amc.2009.03.090

[5] Baris Yuce Michael S. Packianather, Ernesto Mastrocinque, Duc Truong Pham, and Alfredo Lambiase. Honey Bees In-spired Optimization Method: The Bees Algorithm. *Insects*, 4(4): 646-662, 2013. https://doi:10.3390/insects4040646

[6] Duc Truong Pham, Afshin Ghanbarzadeh, Ebubekir Koç, Sameh Otri, Shafqat Rahim, and Muhamad Zaidi. The Bees Algorithm, A Novel Tool for Complex Optimization Problems. *Proceedings of the Second International Virtual Conference on Intelligent production machines and systems (IPROMS 2006), Elsevier*, Oxford. 454-459, 2006 https://doi.org/10.1016/j.engappai.2018.04.012

[7] Duc Truong Pham, and Marco Castellani. The Bees Algorithm: Modeling Foraging Behavior to Solve Continuous Optimization Problems. *Proceedings of*

the Institution of Mechanical Engineers Part C, Journal of Mechanical Engineering Science, 223(12): 2919–2938, 2009. https://doi:10.1243/09544062JMES1494

[8] Nuntana Mayteekrieangkrai, and Wuthichai Wongthatsanekorn. Optimized Ready Mixed Concrete Truck Scheduling for Uncertain Factors Using Bee Algorithm. S*ongklanakar in Journal of Science & Technology*, 37(2), 2015. https://doi.org/10.4271/j2967_201308

[9] Michael S. Packianather, Baris Yuce, Ernesto Mastrocinque, Fabio Fruggiero, Duc Truong Pham, and Alfredo Lambiase. Novel Genetic Bees Algorithm Applied to Single Machine Scheduling Problem. *World Automation Congress (WAC), IEEE*, 906–911, 2014. https://doi:10.1109/wac.2014.6936194

[10] Mohamed Amine Nemmich, and Fatima Debbat. Bees Algorithm and its Variants for Complex Optimization Problems. *Proceedings of the second International Conference on Applied Automation and Industrial Diagnostics (ICAAID17)*, Djelfa, Algeria.

[11] Khang Nguyen, Phuc Danh Nguyen, and Nuong Tran. A hybrid algorithm of Harmony Search and Bees Algorithm for a University Course Timetabling Problem. *International Journal of Computer Science Issues*, 9(1): 12–17, 2012.

[12] Duc Truong Pham, Ahmed Haj Darwish, and Eldaw Elzaki Eldukhri. Optimization of A Fuzzy Logic Controller Using the Bees Algorithm. *International Journal of Computer Aided Engineering and Technology*, 1(2): 250-264, 2009. https://doi:10.1504/ijcaet.2009.022790

[13] Nanda Dulal Jana, Jaya Sil, and Swagatam Das. Improved Bees Algorithm for Protein Structure Prediction Using AB Off-Lattice Model. *Advances in Intelligent Systems and Computing Mendel*, 39-52, 2015. https://doi:10.1007/978-3-319-19824-8_4

[14] Pokpong Songmuang, and Maomi Ueno. Bees Algorithm for Construction of Multiple Test Forms in E-Testing. *IEEE Transactions on Learning Technologies*, 4(3): 209-221, 2011. https://doi:10.1109/tlt.2010.29

[15] Razali bin Idris, Azhar Khairuddin, and Mohd Wazir Mustafa. Optimal Choice of FACTS Devices for ATC Enhancement Using Bees Algorithm. *International Journal of Electrical and Computer Engineering*, 3(6): 1-9, 2009. https://doi.org/10.2316/p.2012.785-026

[16] Salima Nebti, and Abdellah Boukerram. Handwritten Characters Recognition Based On Nature-Inspired Computing AndNeuro-evolution. *Applied Intelligence*, 38(2): 146-159, 2013. https://doi:10.1007/s10489-012-0362-z

[17] Aleksandar Jevtic, Álvaro Gutiérrez, Diego Andina, and Mo M. Jamshidi. Distributed Bees Algorithm for Task Allocation in Swarm of Robots. *IEEE Systems Journal*, 6(2): 296-304, 2012. https://doi:10.1109/jsyst.2011.2167820

[18] Er. Poonam, and Rajeev Dhaiya. Artificial Intelligence Based Cluster Optimization for Text Data Mining. *International Journal of Computer Science and Mobile Computing*, 4(9): 8-15, 2015.

[19] Mohamed Amine Nemmich, Fatima Debbat, and Mohamed Slimane. A Data Clustering Approach Using Bees Algorithm with a Memory Scheme. *Lecture Notes in Networks and Systems, 261–270, 2018.* https://doi.org/10.1007/978-3-319-98352-3_28

[20] Hadj Ahmed Bouarara, Reda Mohamed Hamou, and Abdelmalek Amine. Text Clustering using Distances Combination by Social Bees. *International Journal of Information Retrieval Research*, 4(3): 34-53, 2014. https://doi:10.4018/ijirr.2014070103

[21] Marco Castellani, Q. Tuan Pham, Duc Truong Pham. Dynamic Optimization by A Modified Bees Algorithm. Proceedings of the Institution of Mechanical Engineers, Part I: *Journal of Systems and Control Engineering*, 226(7): 956-971, 2012. https://doi:10.1177/0959651812443462

[22] Abbas Moradi, Ali Mirzakhani Nafchi, and A. Ghanbarzadeh. Multi-Objective Optimization of Truss Structures Using The Bee Algorithm. *ScientiaIranica. Transaction B, Mechanical Engineering*, 22(5): 1789-1800, 2015.

[23] Michael S. Packianather, and Bharat Kapoor. A Wrapper-Based Feature Selection Approach Using Bees Algorithm for A Wood Defect Classification System. *Proceedings of Conference the 10th System of Systems Engineering Conference (SoSE)*, 498–503, 2015. https://doi:10.1109/sysose.2015.7151902

[24] Duc Truong Pham, Sameh Otri, and Ahmed Haj Darwish. Application of the Bees Algorithm to PCB Assembly Optimization. *Proceedings of the 3rd virtual international conference on intelligent production machines and systems (IPROMS 2007)*, 511–516, 2007. Whittles, Dunbeath, Scotland

[25] Milad Azarbad, Attaollah Ebrahimzade, and Vahid Izadian. Segmentation of infrared Images and Objectives Detection Using Maximum Entropy Method Based on the Bee Algorithm. *International Journal of Computer information Systems and industrial Management Applications (IJCISIM)*, 3: 026-033, 2011.

[26] Wasim Abdulqawi Hussein, Shahnorbanun Sahran, and Siti Norul Huda Sheikh Abdullah. The Variants of the Bees Algorithm (BA): a survey. *Artificial Intelligence Review*, 1-55, 2017. https://doi.org/10.1007/s10462-016-9476-8

[27] Azar Imanguliyev. Enhancements for the Bees Algorithm [dissertation]. Cardiff University at Cardiff; UK, 2017.

[28] Duc Truong Pham, and Ahmed Haj Darwish. Fuzzy Selection of Local Search Sites in the Bees Algorithm. *Proceedings of the 4th International Virtual Conference on Intelligent Production Machines and Systems (IPROMS 2008)*, 1–14, 2008.

[29] Wasim Abdulqawi Hussein, Shahnorbanun Sahran, and Siti Norul Huda Sheikh Abdullah. An Improved Bees Algorithm for Real Parameter Optimization. *International Journal of Advanced Computer Science and Applications*, 6(10), 2015. https://doi.org/10.14569/ijacsa.2015.061004

[30] Mohamed Amine Nemmich, Fatima Debbat, and Mohamed Slimane. Hybridizing Bees Algorithm with Firefly Algorithm for Solving Complex Continuous Functions. *International Journal of Applied Metaheuristic Computing (IJAMC), 11(2): 27-55, 2020.* https://doi:10.4018/IJAMC.2020040102

[31] Xin-She Yang. Nature-Inspired Metaheuristic Algorithms, 2008

[32] Xin-She Yang. Firefly Algorithms for Multimodal Optimization. Stochastic Algorithms: *Foundations and Applications Lecture Notes in Computer Science*, 169-178, 2009. https://doi:10.1007/978-3-642-04944-6_14

[33] Praveen Ranjan Srivatsava, B. Mallikarjun, and Xin-She Yang. Optimal Test Sequence Generation Using Firefly Algorithm, *Swarm and Evolutionary Computation*, 8: 44–53, 2013. https://doi.org/10.1016/j.swevo.2012.08.003

[34] Adil Baykasoğlu, and Fehmi Burcin Ozsoydan. An Improved Firefly Algorithm for Solving Dynamic Multidimensional Knapsack Problems. *Expert Systems with Applications*, 41(8), 3712–3725, 2014. https://doi.org/10.1016/j.eswa.2013.11.040

[35] Xin-She Yang, and Suash Deb. Eagle Strategy Using Lévy Walk and Firefly Algorithms for Stochastic Optimization. *Nature Inspired Cooperative Strategies for Optimization (NICSO 2010) Studies in Computational Intelligence*, 101–111, 2010. https://doi.org/10.1007/978-3-642-12538-6_9

[36] Krishnanand N. Kaipa, Debasish Ghose. Glowworm Swarm Based Optimization Algorithm for Multimodal Functions with Collective Robotics Applications, *Multiagent and Grid Systems*, 2(3): 209–222, 2006. https://doi.org/10.3233/mgs-2006-2301

[37] K. Chandrasekaran, and Sishaj P. Simon. Network and Reliability Constrained Unit Commitment Problem Using Binary Real Coded Firefly Algorithm. *International Journal of Electrical Power & Energy Systems*, 43(1): 921–932, 2012. https://doi.org/10.1016/j.ijepes.2012.06.004

[38] Narwant Singh Grewal, Munish Rattan, and Manjeet Singh Patterh. A Linear Antenna Array Failure Correction with Null Steering using Firefly Algorithm. *Defence Science Journal*, 64(2): 136–142, 2014. https://doi.org/10.14429/dsj.64.4250

[39] Anurag Mishra, Charu Agarwal, Arpita Sharma, and Punam Bedi. Optimized Gray-scale Image Watermarking Using DWT–SVD and Firefly Algorithm. *Expert Systems with Applications*, 41(17): 7858-7867, 2012. https://doi:10.1016/j.eswa.2014.06.011

[40] Leandro dos Santos Coelho, and Viviana Cocco Mariani. Firefly Algorithm Approach Based on Chaotic Tinkerbell Map Applied to Multivariable PID Controller Tuning. *Computers & Mathematics with Applications*, 64(8): 2371–2382, 2012. https://doi.org/10.1016/j.camwa.2012.05.007

[41] Mohammad Kazem Sayadi, Ashkan Hafezalkotob, and Seyed Gholamreza Jalali Naini. Firefly-Inspired Algorithm for Discrete Optimization Problems: an Application to Manufacturing Cell Formation. *Journal of Manufacturing Systems*, 32(1): 78–84, 2013. https://doi.org/10.1016/j.jmsy.2012.06.004

[42] Ahmad Kazem, Ebrahim Sharifi, Farookh Khadeer Hussain, Morteza Saberi, and Omar Khadeer Hussain. Support Vector Regression with Chaos-Based Firefly Algorithm for Stock Market Price Forecasting. *Applied Soft Computing*, 13(2): 947-958, 2013. https://doi:10.1016/j.asoc.2012.09.024

[43] Mimoun Younes, Fouad Khodja, and Riad Lakhdar Kherfane. Multi-Objective Economic Emission Dispatch Solution Using Hybrid FFA (Firefly Algorithm) and Considering Wind Power Penetration. *Energy*, 67: 595–606, 2014. https://doi.org/10.1016/j.energy.2013.12.043

[44] Qiang Fu, Zheng Liu, Nan Tong, Mingbo Wang, and Yiming Zhao. A Novel Firefly Algorithm based on Im-proved Learning Mechanism. *Proceedings of the International Conference on Logistics, Engineering, Management and Computer Science*, 2015. https://doi:10.2991/lemcs-15.2015.268

[45] Sankalap Arora, and Satvir Singh. The Firefly Optimization Algorithm: Convergence Analysis and Parameter Selection. *International Journal of Computer Applications*, 69(3): 48-52, 2015. https://doi:10.5120/11826-7528

[46] Shuhao Yu, Shenglong Zhu, Yan Ma, and Demei Mao. A Variable Step Size Firefly Algorithm for Numerical Optimization. *Applied Mathematics and Computation*, 263: 214-220, 2015. https://doi:10.1016/j.amc.2015.04.065

[47] Momin Jamil, and Xin She Yang. A Literature Survey Of Benchmark Functions for Global Optimization Problems. *International Journal of Mathematical Modelling and Numerical Optimization*, 4(2): 150, 2013. https://doi:10.1504/ijmmno.2013.055204

[48] David Wolpert, and William Macready. (1997) No Free Lunch Theorems for Optimization. *IEEE Trans-actions on Evolutionary Computation*, 1(1): 67-82, 1997. https://doi:10.1109/4235.585893

Computing Dynamic Slices of Feature-Oriented Programs with Aspect-Oriented Extensions

Madhusmita Sahu and Durga Prasad Mohapatra
Department of Computer Science and Engineering
National Institute of Technology, Rourkela-769008
Rourkela, Odisha, India
E-mail: madhu_sahu@yahoo.com and durga@nitrkl.ac.in

Keywords: feature-oriented programming (FOP), aspect-oriented programming (AOP), composite feature-aspect dependence graph (CFADG), mixin layer, refinement chain

Received: September 10, 2018

This paper proposes a technique to compute dynamic slices of feature-oriented programs with aspect-oriented extensions. The technique uses a dependence based intermediate program representation called composite feature-aspect dependence graph (CFADG) to represent feature-oriented software that contain aspects. The CFADG of a feature-oriented program is based on the selected features that are composed to form a software product and the selected aspects to be weaved. The proposed dynamic slicing technique has been named feature-aspect node-marking dynamic slicing (FANMDS) algorithm. The proposed feature-aspect node marking dynamic slicing algorithm is based on marking and unmarking the executed nodes in the CFADG suitably during run-time. The advantage of the proposed approach is that no trace file is used to store the execution history. Also, the approach does not create any additional nodes during run-time.

Povzetek: Prispevek predstavlja izvirni pristop pri programiranju na osnovi sprejemljivk z aspektno orientiranimi podaljški. Gre za računanje dinamičnih odsekov omenjenih programov.

1 Introduction

Weiser [33] first introduced the concept of a program slice. Program slicing decomposes a program into different parts related to a particular computation. A *slicing criterion* is used to construct a program slice. A slicing criterion is a tuple, $< s, v >$, consisting of a statement s, in a program and a variable v, used or defined at that statement s. Program slicing technique is employed in many areas of software engineering including debugging, program understanding, testing, reverse engineering, etc.

Feature-oriented programming (FOP) is concerned with the separate definition of individual features and the composition of required features to build varieties of a particular software product. The functionalities of a software product are identified as features in FOP paradigm. FOP is used to implement software product lines and incremental designs. A family of software systems constitutes a software product line [20].

Motivation: Today, the variability of software products is crucial for successful software development. One mechanism to provide the required variability is through Software Product Lines, which is inspired by product lines used in the industry, like product lines used in the production of a car or a meal at some fast-food restaurant. Feature-Oriented Programming (FOP) approach is used to implement software product lines. Despite the advantages, feature-oriented programming (FOP) yields some problems in expressing features such as lack of expressing crosscutting modularity. During software evolution, a software should adapt the unanticipated requirements and circumstances. This leads to modifications and extensions that crosscut many existing implementation units. The problem of crosscutting modularity is solved by using aspect-oriented programming (AOP). Kiczales et al. [8] proposed AOP paradigm to separate and modularize the crosscutting concerns like exception handling, synchronization, logging, security, resource sharing, etc. The modularity of crosscutting concerns in FOP can be improved by integrating AOP paradigm into FOP paradigm. In dynamic slicing techniques, first an intermediate representation of the program is statically created in the form of a dependence graph. Then, the dynamic slice is computed by traversing the graph, starting from the point specified in the slicing criterion, using an algorithm. For the programs in general languages like C/C++, Java etc., a single dependence graph is created. There is no composition of features in these languages. FOP is used to develop a family of software products. In FOP (with AOP extensions), multiple dependence graphs are created depending upon the composition of features and aspects. For example, if there are four features and two aspects in a product line out of which two features and one aspect are mandatory, then there are eight possible combinations of features and aspects. Each possible combination of features and aspects creates a different product. Thus, there are eight software products in the product line. Accordingly, there are eight dependence graphs, one graph for each product. Dynamic slice for each possible combination of features and aspects is computed using the corresponding dependence graph. The dynamic slice consists of statements from the composed program that is generated after composition of features and aspects. These statements are again mapped back to the program used for composition. This mapping is not required in general languages like C/C++, Java etc. Again, feature-oriented programs have some special characteristics such as mixins, mixin layers, refinements etc. which are not present in case of general languages like C/C++, Java etc. These characteristics of feature-oriented programs require inclusion of some new nodes/edges in the dependence graph. Similarly, these characteristics require introduction of some new steps/phases in the slicing algorithm (e.g., the handling mixins, the handling of mixin layers, etc.), which are not required in the case of general languages like C/C++, Java, etc. The existing dynamic slicing algorithms for aspect-oriented programs cannot be directly applied for slicing of feature-oriented programs with aspect-oriented extensions due to the specific features of feature-oriented programs such as the presence of mixin layers, refinements of classes, refinements of constructors etc. These characteristics of feature-oriented programs requires inclusion of some new nodes/edges in the dependence graph. Similarly, these characteristics require the introduction of some new steps/phases in the slicing algorithm. Although FOP is an extension of OOP, the existing dynamic slicing algorithms for C/C++, Java cannot be directly applied for slicing of feature-oriented programs due to the presence of aforementioned specific features. Since, program slicing has many applications including testing, software maintenance etc., there is an increasing demand for slicing of feature-oriented programs.

Objective: The main objectives of this work are to develop a suitable intermediate representation of feature-oriented programs with aspect-oriented extension and to propose an efficient slicing algorithm to compute dynamic slices for the above types of programs using the developed intermediate representation. A dependence graph is used to signify the intermediate representation. For a single feature-oriented program, more than one dependence graph can be obtained depending on the number of features to be composed and the number of aspects to be captured. We also aim at calculating the slice computation time for different compositions of features and different aspects captured.

Organization: The organization of rest of the paper is as follows. Section 2 provides a brief introduction to feature-oriented programming (FOP) and program slicing. Section 3 discusses the construction of *composite feature-aspect dependence graph* (CFADG), which is a dependence based intermediate representation of feature-oriented programs containing aspects. In Section 4, the details of our proposed algorithm named *feature-aspect node marking dynamic slicing* (FANMDS) algorithm, is discussed. This section also presents the space and time complexity of FANMDS algorithm. Section 5 furnishes a brief overview of the implementation of FANMDS algorithm along with experimental results. A brief comparison of the proposed work with some other related work is furnished in Section 6. Section 7 concludes the paper along with some possible future work.

2 Basic concepts

In this Section, we provide some basic concepts of feature-oriented programming and outline the features of Jak language, which is a feature-oriented programming language. We also discuss the problems of feature-oriented programming and solutions to these problems through aspect-oriented programming extensions.

2.1 Feature-oriented programming (FOP)

Prehofer [1] was the pioneer to coin the term feature-oriented programming (FOP). The key idea behind FOP is to build the software by composing *features*. Features are the basic building blocks that are used to satisfy user requirements on a software system. The *step-wise refinement* where features incrementally refine other features leads to a stack of features that are arranged in layers. One suitable technique to implement the features is through the use of *Mixin Layers*. A Mixin Layer is a static component that encapsulates fragments of several different classes (Mixins) to compose all fragments consistently. Several languages like Jak [2],[1], Fuji[2], FeatureHouse[3], FeatureRuby [40, 41], FeatureC++ [5, 3, 4] support the concept of features explicitly. FeatureIDE [6][4] is a tool that supports Feature-Oriented Software Development (FOSD) in many languages. We have taken Jak program as input in our proposed approach as it is supported by *Algebraic Hierarchical Equations for Application Design* (AHEAD) tool suite, which is a composer. AHEAD tool suite is a group of tools to work with Jak language. Other languages have their own composers, and those composers are not a group of tools. Jak (short for Jakarta) is a language that extends Java by incorporating feature-oriented mechanisms [2]. Jak-specific tools like *jampack* and *mixin* are invoked to compose Jak files. A Jak file is translated to its Java counterpart using another tool called *jak2java*. The different features supported by Jak language are *Super()* references, an extension of constructors, declaration of local identifiers, etc. The details of Jak language and its features can be found in [2].

[1]`https://juliank.files.wordpress.com/2012/04/jlang1.pdf`

[2]`http://www.infosun.fim.uni-passsau.de/spl/apel/fuji`

[3]`http://www.fosd.de/fh`

[4]`http://wwwiti.cs.uni-magdeburg.de/iti\_db/research/featureide/deploy`

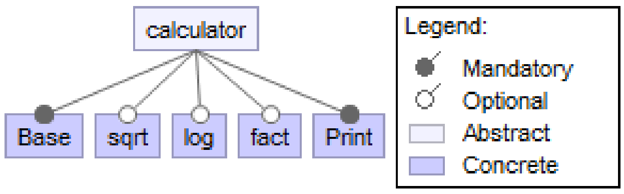


Figure 1: Features supported by Calculator Product Line (CPL)

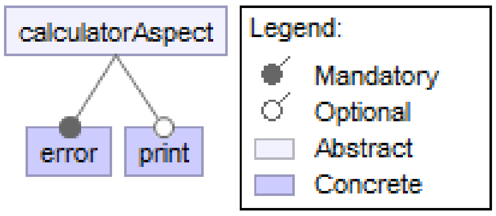


Figure 2: Aspects captured in Calculator Product Line (CPL)

Example 2.1. ***Calculator Product Line (CPL) [45]:*** *This program calculates the factorial, square root and logarithmic value with base 10 of a number. This program is referred to as Calculator Product Line (CPL).*

A *feature-tree* depicts various features supported by a product line in a hierarchical manner. The feature-tree for Example 2.1 is given in Figure 1. The various aspects that are captured for Example 2.1 are given in Figure 2. Figure 3 depicts the source code for each feature given in Figure 1 and each aspect given in Figure 2. Figure 4(a) – Figure 4(b) show the resultant files generated after the composition of all features.

2.2 Problems in feature-oriented programming (FOP)

Feature-oriented programming (FOP) suffers from many problems in modularization of crosscutting concerns [3, 4, 5]. The presence of these problems leads to degradation in modularity of program family members and also decrease in maintainability, customizability, and evolvability. Some of the problems of FOP are discussed below.

1. FOP is unable to express *dynamic crosscutting concerns* that affect the control flow of a program. It can only express *static crosscutting concerns* that affect the structure of the program. AOP languages can handle dynamic crosscutting concerns in an efficient manner through the use of pointcuts, advices etc.

2. FOP languages support only *heterogeneous crosscutting concerns* where different join points are provided with different pieces of codes. In contrast, AOP languages support *homogeneous crosscutting concerns* where different join points are provided with the same piece of code.

3. FOP suffers from excessive method extension problem when a feature crosscuts a large fraction of existing classes because of refinements. A lot of methods are to be overridden for each method on which a crosscut depends. This is because FOP is unable to modularize homogeneous crosscutting concerns. AOP uses wildcards in pointcuts to deal with this problem.

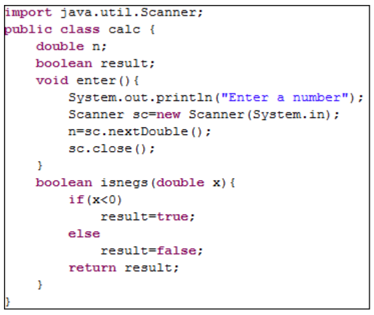
2.3 AOP extensions to FOP

AOP can be used to solve the above problems of FOP by integrating AOP language features like wildcards, pointcuts, and advices into FOP languages. The different approaches used for integrating AOP language features into FOP languages are *Multi Mixins*, *Aspectual Mixin Layers*, *Aspectual Mixins*. More details about these approaches can be found in [5, 3, 4]. The *Aspectual Mixin Layers* approach is a popular one amongst all the approaches since this approach overcomes all the aforementioned problems. Other approaches overcome some of the problems. We have used the approach of aspectual mixin layers in our work. We have separated the aspects from mixin layers for easy understanding of our approach. Our mixin layers contain only a set of classes. Aspects are designed as different layers.

2.4 Program slicing

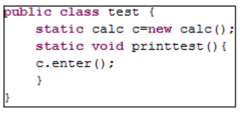
Program slicing is a technique which is employed to analyze the behavior of a program on the basis of dependencies that exist between various statements. It takes out statements from a program related to a specific computation. The extracted statements constitute a slice. Thus, a slicing criterion is employed to compute a slice. A slicing criterion consists of a statement s (or location in the program) and a variable v (or set of variables), and it is represented as a tuple $< s, v >$. Program slicing technique can be either *static* or *dynamic* based on the input to the program. A program slicing technique is said to be *static* when it extracts all the statements from a program with respect to a slicing criterion *irrespective* of the input to the program. On the other hand, a program slicing technique is said to be *dynamic* when all the statements from a program are extracted with respect to a slicing criterion for a *specific* input to the program.

The difference between *static slicing* and *dynamic slicing* can be understood by taking an example. Let us consider the example C program given in Figure 5. The static slice with respect to slicing criterion $< 11, y >$ is depicted as the bold italic statements in Figure 6. It includes statements 1, 2, 3, 4, 6, 7, 9, and 11. The dynamic slice with respect to slicing criterion $< \{x = 10\}, 11, y >$ is depicted as the bold italic statements in Figure 7. It includes statements 1, 2, 3, 6, 9, and 11. For finding the slices of a program, first an intermediate representation of the program is constructed. Then, the slices are found out by using some algorithm on the intermediate representation. There are many slicing algorithms found in the literature



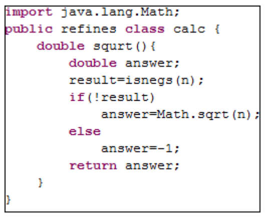
```
import java.util.Scanner;
public class calc {
    double n;
    boolean result;
    void enter(){
        System.out.println("Enter a number");
        Scanner sc=new Scanner(System.in);
        n=sc.nextDouble();
        sc.close();
    }
    boolean isnegs(double x){
        if(x<0)
            result=true;
        else
            result=false;
        return result;
    }
}
```

(a) Base/calc.jak



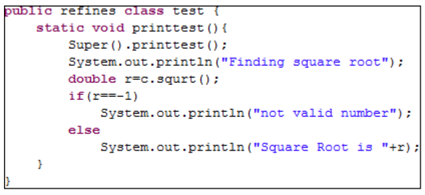
```
public class test {
    static calc c=new calc();
    static void printtest(){
    c.enter();
    }
}
```

(b) Base/test.jak



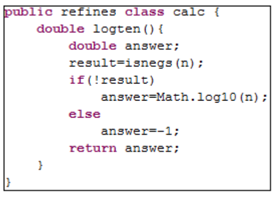
```
import java.lang.Math;
public refines class calc {
    double squrt(){
        double answer;
        result=isnegs(n);
        if(!result)
            answer=Math.sqrt(n);
        else
            answer=-1;
        return answer;
    }
}
```

(c) sqrt/calc.jak



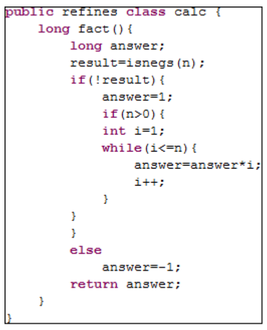
```
public refines class test {
    static void printtest(){
        Super().printtest();
        System.out.println("Finding square root");
        double r=c.squrt();
        if(r==-1)
            System.out.println("not valid number");
        else
            System.out.println("Square Root is "+r);
    }
}
```

(d) sqrt/test.jak



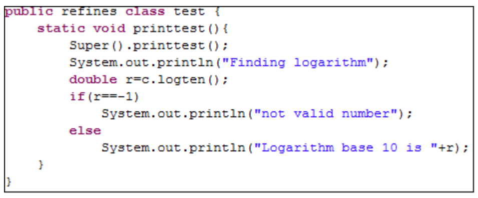
```
public refines class calc {
    double logten(){
        double answer;
        result=isnegs(n);
        if(!result)
            answer=Math.log10(n);
        else
            answer=-1;
        return answer;
    }
}
```

(e) log/calc.jak



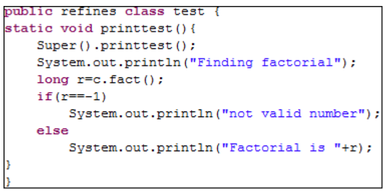
```
public refines class calc {
    long fact(){
        long answer;
        result=isnegs(n);
        if(!result){
            answer=1;
            if(n>0){
            int i=1;
            while(i<=n){
                answer=answer*i;
                i++;
            }
        }
        }
        else
            answer=-1;
        return answer;
    }
}
```

(f) fact/calc.jak



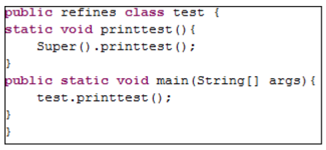
```
public refines class test {
    static void printtest(){
        Super().printtest();
        System.out.println("Finding logarithm");
        double r=c.logten();
        if(r==-1)
            System.out.println("not valid number");
        else
            System.out.println("Logarithm base 10 is "+r);
    }
}
```

(g) log/test.jak



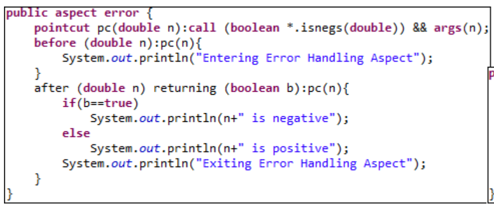
```
public refines class test {
static void printtest(){
    Super().printtest();
    System.out.println("Finding factorial");
    long r=c.fact();
    if(r==-1)
        System.out.println("not valid number");
    else
        System.out.println("Factorial is "+r);
}
}
```

(h) fact/test.jak



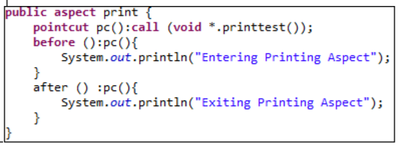
```
public refines class test {
static void printtest(){
    Super().printtest();
}
public static void main(String[] args){
    test.printtest();
}
}
```

(i) Print/test.jak



```
public aspect error {
    pointcut pc(double n):call (boolean *.isnegs(double)) && args(n);
    before (double n):pc(n){
        System.out.println("Entering Error Handling Aspect");
    }
    after (double n) returning (boolean b):pc(n){
        if(b==true)
            System.out.println(n+" is negative");
        else
            System.out.println(n+" is positive");
        System.out.println("Exiting Error Handling Aspect");
    }
}
```

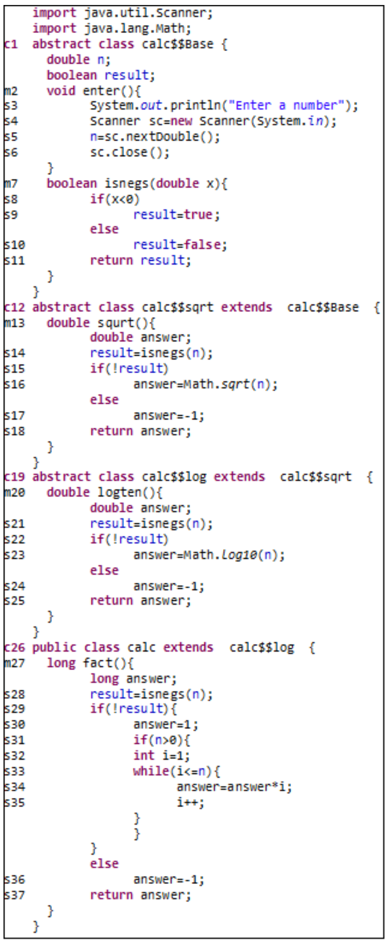
(j) error.aj



```
public aspect print {
    pointcut pc():call (void *.printtest());
    before ():pc(){
        System.out.println("Entering Printing Aspect");
    }
    after () :pc(){
        System.out.println("Exiting Printing Aspect");
    }
}
```

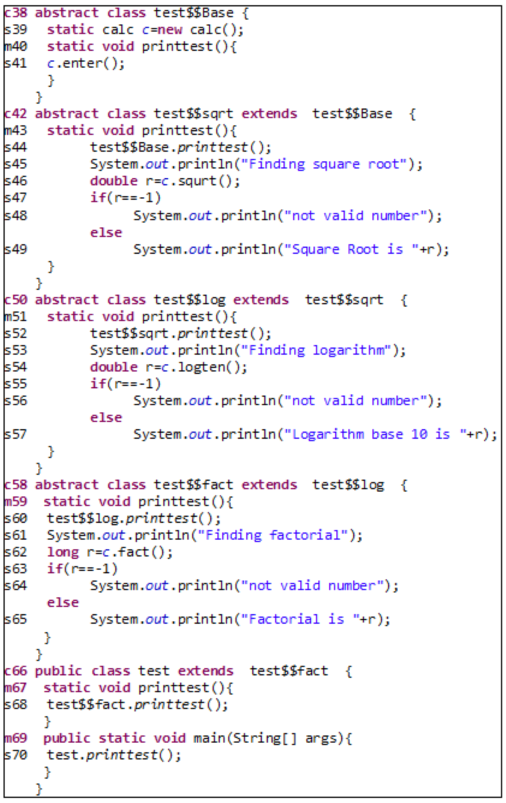
(k) print.aj

Figure 3: Jak program for Calculator Product Line (CPL) along with aspect code. Figure 3(a)–Figure 3(i) represent base code or non-aspect code written in Jak language and Figure 3(j)–Figure 3(k) represent aspect code written in AspectJ



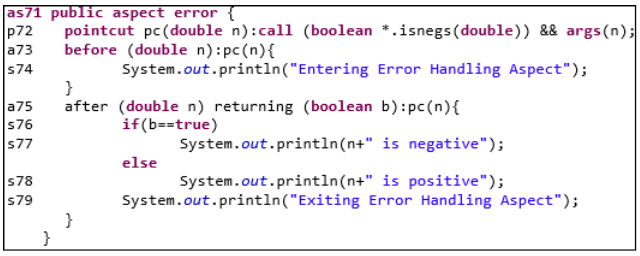
```
    import java.util.Scanner;
    import java.lang.Math;
c1  abstract class calc$$Base {
      double n;
      boolean result;
m2    void enter(){
s3          System.out.println("Enter a number");
s4          Scanner sc=new Scanner(System.in);
s5          n=sc.nextDouble();
s6          sc.close();
      }
m7    boolean isnegs(double x){
s8          if(x<0)
s9                result=true;
            else
s10               result=false;
s11         return result;
      }
    }
c12 abstract class calc$$sqrt extends  calc$$Base  {
m13   double squrt(){
            double answer;
s14         result=isnegs(n);
s15         if(!result)
s16               answer=Math.sqrt(n);
            else
s17               answer=-1;
s18         return answer;
      }
    }
c19 abstract class calc$$log extends  calc$$sqrt  {
m20   double logten(){
            double answer;
s21         result=isnegs(n);
s22         if(!result)
s23               answer=Math.Log10(n);
            else
s24               answer=-1;
s25         return answer;
      }
    }
c26 public class calc extends  calc$$log  {
m27   long fact(){
            long answer;
s28         result=isnegs(n);
s29         if(!result){
s30               answer=1;
s31               if(n>0){
s32               int i=1;
s33               while(i<=n){
s34                     answer=answer*i;
s35                     i++;
                  }
                  }
            }
            else
s36               answer=-1;
s37         return answer;
      }
    }
```

(a) calc.java



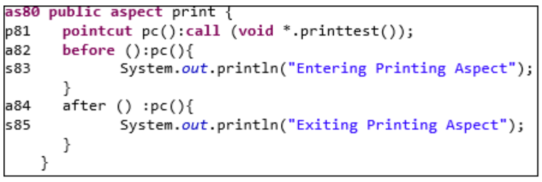
```
c38 abstract class test$$Base {
s39   static calc c=new calc();
m40   static void printtest(){
s41   c.enter();
      }
    }
c42 abstract class test$$sqrt extends  test$$Base  {
m43   static void printtest(){
s44         test$$Base.printtest();
s45         System.out.println("Finding square root");
s46         double r=c.squrt();
s47         if(r==-1)
s48               System.out.println("not valid number");
            else
s49               System.out.println("Square Root is "+r);
      }
    }
c50 abstract class test$$log extends  test$$sqrt  {
m51   static void printtest(){
s52         test$$sqrt.printtest();
s53         System.out.println("Finding logarithm");
s54         double r=c.logten();
s55         if(r==-1)
s56               System.out.println("not valid number");
            else
s57               System.out.println("Logarithm base 10 is "+r);
      }
    }
c58 abstract class test$$fact extends  test$$log  {
m59  static void printtest(){
s60   test$$log.printtest();
s61   System.out.println("Finding factorial");
s62   long r=c.fact();
s63   if(r==-1)
s64         System.out.println("not valid number");
      else
s65         System.out.println("Factorial is "+r);
     }
    }
c66 public class test extends  test$$fact  {
m67  static void printtest(){
s68   test$$fact.printtest();
     }
m69  public static void main(String[] args){
s70   test.printtest();
     }
    }
```

(b) test.java



```
as71 public aspect error {
p72     pointcut pc(double n):call (boolean *.isnegs(double)) && args(n);
a73     before (double n):pc(n){
s74             System.out.println("Entering Error Handling Aspect");
        }
a75     after (double n) returning (boolean b):pc(n){
s76             if(b==true)
s77                     System.out.println(n+" is negative");
                else
s78                     System.out.println(n+" is positive");
s79             System.out.println("Exiting Error Handling Aspect");
        }
    }
```

(c) error.aj



```
as80 public aspect print {
p81     pointcut pc():call (void *.printtest());
a82     before ():pc(){
s83             System.out.println("Entering Printing Aspect");
        }
a84     after () :pc(){
s85             System.out.println("Exiting Printing Aspect");
        }
    }
```

(d) print.aj

Figure 4: Java codes generated (Figure 4(a)–Figure 4(b)) after the composition of all features depicted in Figure 1. Figure 4(c)–Figure 4(d) are the AspectJ codes for the aspects captured.

```
   #include<stdio.h>
1  main(){
   int x,y,z;
2  scanf("%d",&x);
3  if(x<0){
4  y=x+5;
5  z=x*2;
   }
6  else if(x==0){
7  y=x+20;
8  z=x*8;
   }
   else{
9  y=x+4;
10 z=x*3;
   }
11 printf("y= %d",y);
12 printf("z= %d",z);
 }
```

Figure 5: An example C program

```
   #include<stdio.h>
1  main(){
   int x,y,z;
2  scanf("%d",&x);
3  if(x<0){
4  y=x+5;
5  z=x*2;
   }
6  else if(x==0){
7  y=x+20;
8  z=x*8;
   }
   else{
9  y=x+4;
10 z=x*3;
   }
11 printf("y= %d",y);
12 printf("z= %d",z);
 }
```

Figure 6: Static slice with respect to slicing criterion $< 11, y >$

```
   #include<stdio.h>
1  main(){
   int x,y,z;
2  scanf("%d",&x);
3  if(x<0){
4  y=x+5;
5  z=x*2;
   }
6  else if(x==0){
7  y=x+20;
8  z=x*8;
   }
   else{
9  y=x+4;
10 z=x*3;
   }
11 printf("y= %d",y);
12 printf("z= %d",z);
 }
```

Figure 7: Dynamic slice with respect to slicing criterion $< \{x = 10\}, 11, y >$

[10, 11, 12, 14, 15, 16, 17, 19, 21, 25, 26, 37, 38, 42]. For the details of the intermediate program representations and different slicing algorithms, the readers may refer to [10, 11, 12, 14, 15, 16, 17, 19, 21, 25, 26, 37, 38, 42]. In the next section, we propose an intermediate program representation for feature-oriented programs, on which our slicing algorithm can be applied.

3 Composite feature-aspect dependence graph (CFADG): an intermediate representation for feature-oriented programs

We have proposed an intermediate representation for feature-oriented programs, called Composite Feature-Aspect Dependence Graph (CFADG). CFADG is an arc-classified digraph, $G = (N, E)$, where N is the set of vertices depicting the statements and E is the set of edges symbolizing the dependence relationships between the statements. The set E captures various dependencies that exist between the statements in various mixin layers and aspects in a feature-oriented program. CFADG is constructed based on the composition of different features and aspects captured. Thus, there will be different types of CFADGs according to the features composed and aspects captured. Figure 8 shows the CFADG for the composition given in

Figure 3. The square box with a1_in: n_in=n etc. specifies the actual and formal parameters. For example: a1_in: n_in=n specifies that n is an actual-in parameter. Similarly, a2_in: b_in=b specifies that b is an actual-in parameter, f1_in: x=n_in specifies that x is a formal-in parameter, f2_in: b=b_in specifies that b is a formal-in parameter. These notations are adopted from Horwitz et al. [47]. The construction of CFADG consists of the following steps:

- Constructing Procedure Dependence Graph (PDG) for each method in a mixin.
- Constructing Mixin Dependence Graph (MxDG) for each mixin.
- Constructing System Dependence Graph (SDG) for each mixin layer.
- Constructing Advice Dependence Graph (ADG) for each advice.
- Constructing Introduction Dependence Graph (IDG) for each introduction.
- Constructing Pointcut Dependence Graph (PtDG) for each pointcut.
- Constructing Aspect Dependence Graph (AsDG) for each aspect.
- Constructing Composite Feature Aspect Dependence Graph (CFADG) by combining all the SDGs and AsDGs.

Below, we briefly explain the steps for constructing the CFADG and the pseudocode.

(1) Construction of Procedure Dependence Graph (PDG)

A *procedure dependence graph* (PDG) depicts the control and data dependence relationships that exist between the statements in a program with only one function/method/procedure. The nodes in the graph correspond to the program statements and edges correspond to the dependence relationships between the statements.

(2) Construction of Mixin Dependence Graph (MxDG)

A *mixin dependence graph* (MxDG) is used to capture all dependencies within a mixin. A MxDG has a *mixin entry vertex* that connects the method entry vertex of each method in the mixin by a *mixin membership edge*. Each method entry in the MxDG is associated with *formal-in* and *formal-out* parameter nodes. The interactions among methods in a mixin occur by calling each other. This effect of method calls is symbolized by a *call* node in a MxDG. *Actual-in* and *actual-out* parameter nodes are created at each call node corresponding to formal-in and formal-out parameter nodes. The effect of *return* statements in a MxDG is represented by joining each *return* node to its corresponding *call* node through a *return dependence edge*.

(3) Construction of System Dependence Graph (SDG) for each Mixin Layer

A single mixin layer may contain more than one mixin. A mixin may derive another mixin through inheritance. The MxDG for the derived class is constructed. The mixin membership edges connect the mixin entry vertex of derived class to the method entry vertices of all those methods that are defined and inherited in the derived class. The SDG for a mixin layer is constructed by joining all the mixin dependence graphs for that mixin layer through parameter edges, call edges and summary edges.

(4) Construction of Advice Dependence Graph (ADG)

An *advice dependence graph* (ADG) represents an advice in an aspect. The statements or predicates in the advice are represented as vertices and dependencies amongst statements are represented as edges in an ADG. Each ADG is associated with a unique vertex called *advice start vertex* to signify entry into the advice.

(5) Construction of Introduction Dependence Graph (IDG)

An *introduction dependence graph* (IDG) represents an introduction in an aspect. If an introduction is a method or constructor, then its IDG is similar to PDG of a method. A unique vertex, called *introduction start vertex*, is used in IDG to signify the entry into the introduction.

(6) Construction of Pointcut Dependence Graph (PtDG)

Pointcuts in an aspect contain no body. Therefore, to represent pointcuts, only a *pointcut start vertex* is created to denote the entry into the pointcut.

(7) Construction of Aspect Dependence Graph (AsDG)

An *aspect dependence graph* (AsDG) is used to represent a single aspect. It consists of a collection of ADGs, IDGs, PtDGs that are connected by some special kinds of edges. Each AsDG is associated with a unique vertex called *aspect start vertex*, to represent entry into the aspect. An *aspect membership edge* is used to represent the membership relationships between an aspect and its members. This edge connects the aspect start vertex to each start vertex of an ADG, IDG or PtDG. Each pointcut start vertex is connected to its corresponding advice start vertex by call edges.

(8) Construction of Composite Feature-Aspect Dependence Graph (CFADG)

The CFADG is constructed by combining the SDGs for all mixin layers present in the composition and the AsDGs through special kinds of edges. The SDGs for all mixin layers in a composition are connected using refinement edges, mixin call edges, mixin data dependence edges, and mixin return dependence edges. The AsDGs are connected to all the SDGs through weaving edges and aspect data dependence edges. The mixin membership edges and aspect membership edges along with mixin start vertices and aspect start vertices are removed during construction of CFADG. The CFADG for the program given in Figure 3 is shown in Figure 8. A CFADG contains the following types of edges:

(a) **Control dependence edge:** A *control dependence edge* in a CFADG from a node n_1 to a node n_2 indicates that either node n_2 is under the scope of node n_1 or node n_1 controls the execution of node n_2 where node n_1 is a predicate node.
In Figure 8, edge $(m20, s21)$ is a control dependence edge.

(b) **Data dependence edge:** A *data dependence edge* in a CFADG from a node n_1 to a node n_2 indicates that node n_2 uses a variable that is assigned a value at node n_1 or n_1 creates an object o and o is used at n_2.
In Figure 8, edges $(s21, s22)$, $(s39, s41)$, $(p72, a73)$ are data dependence edges.

(c) **Mixin data dependence edge:** A *mixin data dependence edge* in a CFADG from a node n_1 to a node n_2 indicates that node n_2 in a mixin layer defines a variable which is used at node n_1 in another mixin layer.
In Figure 8, edges $(s5, s16)$ and $(s39, s54)$ are mixin data dependence edges.

(d) **Aspect data dependence edge:** An *aspect data dependence edge* in a CFADG from a node n_1 to a node n_2 indicates that node n_2 in an aspect uses the value of a variable and that variable is defined at node n_1 in a mixin.
In Figure 8, edge $(s5, a1\_in)$ is an aspect data dependence edge.

(e) **Call edge:** A *call edge* in CFADG from a node n_1 to a node n_2 indicates that node n_1 calls a method defined at node n_2. Both the nodes n_1 and n_2 are in same mixin layer.
In Figure 8, edge $(s41, m2)$ is a call edge.

(f) **Mixin call edge:** A *mixin call edge* in CFADG from a node n_1 to a node n_2 indicates that node n_1 in a mixin layer calls a method that is defined in a different mixin layer at node n_2.
In Figure 8, edge $(s28, m7)$ is a mixin call edge.

(g) **Return dependence edge:** A *return dependence edge* in a CFADG from node n_1 to node n_2 indicates that node n_1 in a mixin layer returns a value to node n_2 in the same mixin layer and node n_2 calls a method where node n_1 is present.
In Figure 8, edge $(s18, s46)$ is a return dependence edge.

(h) **Mixin return dependence edge:** A *mixin return dependence edge* in a CFADG from node n_1 to node n_2 indicates that node n_1 in one mixin layer returns a value to node n_2 in another mixin layer and node n_2 calls a method where node n_1 is present.
In Figure 8, edge $(s11, s21)$ is a mixin return dependence edge.

(i) **Parameter-in edge:** *Parameter-in edge* in CFADG is added from actual-in parameters to corresponding formal-in parameters to indicate the receipt of values from the calling method to the called method.
In Figure 8, edges $(s14 \rightarrow a1\_in, m7 \rightarrow f1\_in)$, $(s21 \rightarrow a1\_in, m7 \rightarrow f1\_in)$, and $(s28 \rightarrow a1\_in, m7 \rightarrow f1\_in)$ are parameter-in edges.

(j) **Parameter-out edge:** *Parameter-out edge* is added from formal-out parameters to corresponding actual-out parameters to indicate the return of values from the called method to the calling method. If an actual parameter is modified inside a method, then the modified value becomes an actual-out parameter and the original value becomes an actual-in parameter. The parameter used to hold the value of actual-in parameter in method definition becomes a formal-in parameter and the parameter used to hold the modified value becomes a formal-out parameter.
In Figure 8, there are no parameter-out edges, since, in our example, no parameter is modified inside a method.

(k) **Summary edge:** The *summary edge* is used to represent the transitive flow of dependence between an actual-in parameter node and an actual-out parameter node if the value of the actual-in parameter node affects the value of the corresponding actual-out vertex.
In Figure 8, edges $(s14 \rightarrow a1\_in, s14)$, $(s21 \rightarrow a1\_in, s21)$, and $(s28 \rightarrow a1\_in, s28)$ are summary edges.

(l) **Message dependence edge:** A message dependence edge from a node n_1 to another node n_2 in a dependency graph signifies that node n_1 represents a statement outputting some message without using any variable and node n_2 represents an input statement, a computation statement, a method call statement, or a predicate statement.
In Figure 8, there exists a message dependence edge $(s3, s4)$. Similarly, edges $(s45, s46)$, $(s53, s54)$, and $(s61, s62)$ are message dependence edges.

(m) **Refinement edge:** A *refinement edge* in a CFADG from a node n_1 to a node n_2 indicates that node n_1 in child mixin layer calls a method $k()$ by prefacing *Super()* call and $k()$ is defined at node n_2 in parent mixin layer.
In Figure 8, the edge $(s44, m40)$ is a refinement edge. Similarly, edges $(s68, m59)$, $(s60, m51)$, and $(s52, m43)$ are refinement edges.

(n) **Weaving edge:** A weaving edge from a node n_1 to node n_2 indicates that

- node n_1 is a method call node and node n_2 is a *before* advice node capturing the method called at n_1 and node n_2 executes before the method called by n_1 executes. OR
- node n_1 is the last statement in *before* advice and node n_2 is the method entry node of the method captured by the advice and node n_2 executes after node n_1 executes. OR
- node n_2 is an *after* advice node and node n_1 is the last statement in the method captured by node n_2 and node n_2 executes after node n_1 executes. OR
- node n_1 is the last statement in an *after* advice and node n_2 is the statement followed by method call node and the method is captured by the advice and node n_1 executes before node n_2 executes.

In Figure 8, edge $(s14, a73)$ is a weaving edge.

The brief pseudocode for constructing the CFADG for a feature-oriented program is given below, and the complete algorithm is given in Algorithm 8 in Appendix A.
CFADG construction Algorithm

```
(1) For each mixin layer
    (a) For each mixin
        i. Create mixin entry vertex
        ii. For each method
            A. Compute control and data dependences.
            B. Construct PDG using control & data dependence edges.
        iii. For each method call
            A. Create actual parameter vertices.
        iv. For each method definition
            A. Create method entry vertex.
            B. Create formal parameter vertices.
        v. Construct MxDG by connecting all PDGs through method call edges, parameter edges and summary edges and connecting each method vertex to mixin start vertex through mixin membership edges.
    (b) Construct SDG by connecting all MxDGs through method call edges, parameter edges.
(2) For each aspect
    (a) Create aspect entry vertex.
    (b) For each advice
        i. Create advice start vertex.
        ii. Compute control and data dependences.
        iii. Construct ADG using control & data dependence edges.
    (c) For each introduction
        i. Create introduction start vertex.
        ii. If introduction is a field then
            Do not create any dependence graph.
            Else if introduction is a method then
            Construct IDG using control and data dependence edges.
    (d) For each pointcut
        i. Create pointcut start vertex.
        ii. Construct PtDG.
    (e) Construct AsDG by connecting advice start vertices, introduction start vertices, pointcut start vertices to aspect start vertex through aspect membership edges.
(3) Remove mixin membership edges, aspect membership edges, mixin start vertices, and aspect start vertices.
(4) Connect all SDGs through refinement edges, mixin call edges, mixin data dependence edges, and mixin return dependence edges.
(5) Connect all AsDGs to all SDGs through weaving and aspect data dependence edges.
```

4 Feature-aspect node-marking dynamic slicing (FANMDS) algorithm

In this section, we present our proposed algorithm for computing dynamic slices of feature-oriented programs using CFADG. We have named our algorithm *Feature-Aspect Node-Marking Dynamic Slicing* (FANMDS) algorithm as it is based on marking and unmarking the nodes of CFADG. Before presenting our FANMDS algorithm, we first introduce some definitions which will be used in our algorithm.

4.1 Definitions

Definition 1: Defn(v): Let v be a variable or an object in program P. A node u in the CFADG is said to be $Defn(v)$ node if u corresponds to a definition statement that defines a value to variable v or u represents a statement that creates object v.
In the CFADG given in Figure 8, nodes $s23$, and $s24$ represent *Defn(answer)* nodes in the method *logten()* in mixin *calc* in *log* mixin layer.
Definition 2: DefnSet(v): The set of all $Defn(v)$ nodes is referred to as $DefnSet(v)$.
In the CFADG given in Figure 8, $DefnSet(answer) = \{s23, s24\}$ in the method *logten()* in mixin *calc* in *log* mixin layer.
Definition 3: RecDefn(v): For each variable v, $RecDefn(v)$ represents the node corresponding to the most recent definition of v with respect to some point s in an execution.

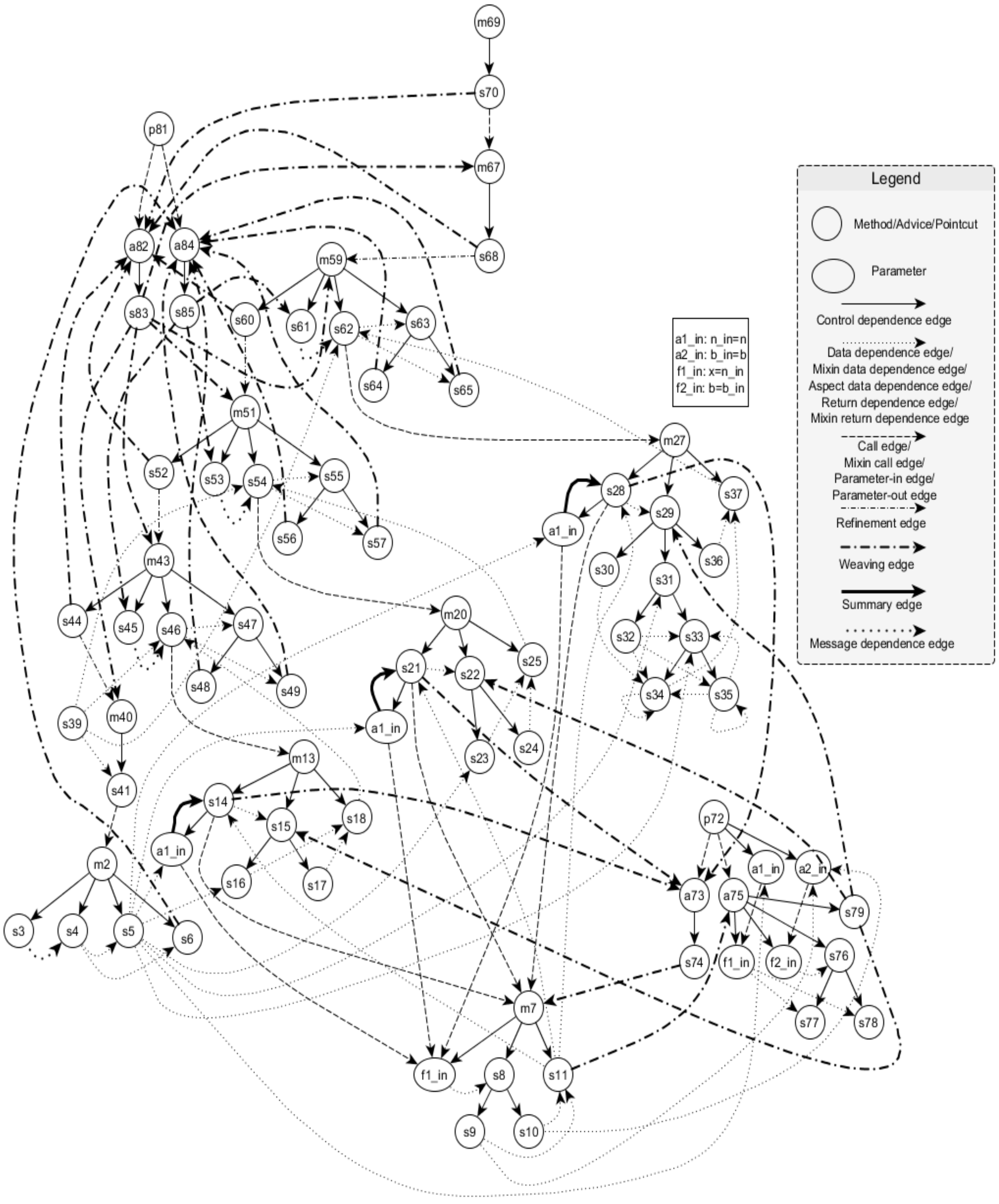


Figure 8: Composite Feature-Aspect Dependence Graph (CFADG) for the program given in Figure 4

In the CFADG of Figure 8, $RecDefn(i)$ is at statement $s32$ before while loop and it is at statement $s35$ during execution of while loop.

Definition 4: Usage(v): Let v be a variable or an object in program P. A node u in the CFADG is said to be $Usage(v)$ node if u represents a statement that uses the variable v or u represents a statement that uses the object v to call a method on that object or to assign the object v with another object.

In the CFADG given in Figure 8, nodes $s47$, and $s49$ represent $Usage(r)$ nodes. Similarly, node $s77$, and $s78$ are $Usage(n)$ nodes.

Definition 5: UsageSet(v): The set of all $Use(v)$ nodes is referred to as $UsageSet(v)$.

In the CFADG given in Figure 8, and $UsageSet(r) = \{s47, s49\}$, $UsageSet(n) = \{s77, s78\}$.

4.2 Overview of FANMDS algorithm

Before execution of a feature-oriented program FP, the features required for composition and the aspects to be captured are selected. Then, the selected features are composed and selected aspects are weaved. The CFADG is constructed statically only once based on the composition of selected features and weaving of selected aspects. The program is executed for a specified input. The executed nodes in CFADG are marked and unmarked during program execution depending upon the arise and cease of dependences respectively. When a statement executes a *Super()* node, it is marked by the algorithm. Also the corresponding method entry node, the associated actual and formal parameter nodes are marked. When there is an invocation of a method, the corresponding call node, the corresponding method entry node, the associated actual and formal parameter nodes are also marked. Whenever a pointcut is executed, the corresponding advice nodes are marked. When an advice is executed, the corresponding formal parameter nodes are marked. During execution, the dynamic slice of each executed statement is computed. After execution of each node and computation of dynamic slice at that node, the algorithm unmarks it.

Let $dyn\_slice(u)$ denote the dynamic slice with respect to the most recent execution of node u. Let (e_1, u), $(e_2, u), \ldots, (e_k, u)$ be all the marked predecessor nodes of u in the CFADG after execution of node u. The dynamic slice with respect to the present execution of node u is computed as
$dyn\_slice(u) = \{u, e_1, e_2, \ldots, e_k\} \cup dyn\_slice(e_1) \cup dyn\_slice(e_2) \cup \ldots \cup dyn\_slice(e_k)$.

Our FANMDS algorithm computes the dynamic slice with respect to the specified slicing criterion by simply looking up the corresponding dyn_slice computed during run-time. Below, we present the pseudocode of our FANMDS algorithm in brief. Algorithm 9 in Appendix B presents our FANMDS algorithm in detail.

Feature-Aspect Node-Marking Dynamic Slicing (FANMDS) Algorithm

(1) CFADG Construction: Construct the CFADG for the given feature-oriented program with aspect-oriented extensions, statically only once.

(2) Initialization: Do the followings before each execution of FP.

(a) Unmark all nodes of CFADG.

(b) Set $dyn\_slice(u) = \phi$ for every node u.

(c) Set $RecDefn(v) = NULL$ for every variable v of the program FP.

(3) Run time updations: Execute the program for the given set of input values and carry out the followings after each statement s of the program FP is executed. Let node u in CFADG corresponds to the statement s in the program FP.

(a) For every variable v used at node u,
Update $dyn\_slice(u) = \{u, e_1, e_2, \ldots, e_k\} \cup dyn\_slice(e_1) \cup dyn\_slice(e_2) \cup \ldots \cup dyn\_slice(x_k)$ where $e_1, e_2, \ldots, e_k$ are the marked predecessor nodes of u in CFADG.

(b) If u is $defn(v)$ node, then

  i. Unmark the node $RecDefn(v)$.

  ii. Update $RecDefn(v) = u$.

(c) Mark node u.

(d) If u is a *method call* node or `new` operator node or *polymorphic* node or *mixin call* node, then

  i. Mark node u.

  ii. Mark the associated *actual-in* and *actual-out* nodes corresponding to the present execution of u.

  iii. Mark the corresponding *method entry* node for the present execution of u.

  iv. Mark the associated *formal-in* and *formal-out* parameter nodes.

(e) If u is a *Super()* method node

  i. Mark node u.

  ii. Mark the associated *actual-in* and *actual-out* nodes corresponding to the present execution of u.

  iii. Mark the corresponding *method entry* node present in the parent mixin layer for the present execution of u.

  iv. Mark the *formal-in* and *formal-out* parameter nodes associated with the *method entry* node.

(f) If u is a *pointcut* node

  i. Mark node u.

  ii. Mark the corresponding *advice* nodes for present execution of u.

(g) If u is an *advice* node

  i. Mark node u.

  ii. Mark the *formal-in* and *formal-out* parameter nodes associated with the *advice* node.

(h) If u is an *introduction* node such that u is a method

  i. Mark node u.

  ii. Mark the *formal-in* and *formal-out* parameter nodes.

(i) If u is an *introduction* node such that u is a field

  i. Mark node u.

  ii. Mark the node that defines a value to u for the current execution of u.

  iii. Mark the node that uses the value of u for the current execution of u.

(4) Slice Look Up

(a) For a given slicing command
$< u, v >$, do

i. Look up $dyn\_slice(u)$ for variable v for the content of the slice.
ii. Map the Java statements included in the computed dynamic slice to the corresponding composed Jak statements to get the final dynamic slice
iii. Display the resulting slice.

(b) If the program has not terminated, go to Step 3.

Working of the Algorithm

The working of FANMDS algorithm is illustrated through an example. Consider the feature-oriented program given in Figure 3 and the selected features for composition and aspects given in Figure 1 and Figure 2 respectively. After the composition of the selected features, the files that are generated are depicted in Figure 4. The corresponding CFADG is shown in Figure 8. During the initialization step, our algorithm first unmarks all the nodes of the CFADG and sets $dyn\_slice(u) = \phi$ for every node u of the CFADG. Now, for the input data $n = 5$, the program will execute the statements $m69$, $s70$, $p81$, $a82$, $s83$, $m67$, $s68$, $a82$, $s83$, $m59$, $s60$, $a82$, $s83$, $m51$, $s52$, $a82$, $s83$, $m43$, $s44$, $a82$, $s83$, $s39$, $m40$, $s41$, $m2$, $s3$, $s4$, $s5$, $s6$, $a84$, $s85$, $s45$, $s46$, $m13$, $s14$, $p72$, $a73$, $s74$, $m7$, $s8$, $s10$, $s11$, $a75$, $s76$, $a78$, $s79$, $s15$, $s16$, $s18$, $s47$, $s49$, $a84$, $s85$, $s53$, $s54$, $m20$, $s21$, $a73$, $s74$, $m7$, $s8$, $s10$, $s11$, $a75$, $s76$, $s78$, $s79$, $s22$, $s23$, $s25$, $s55$, $s57$, $a84$, $s85$, $s61$, $s62$, $m27$, $s28$, $a73$, $s74$, $m7$, $s8$, $s10$, $s11$, $a75$, $s76$, $s78$, $s79$, $s29$, $s30$, $s31$, $s32$, $s33$, $s34$, $s35$, $s37$, $s63$, $s65$, $a84$, $s85$, $a84$, $s85$ in order. So, our FANMDS algorithm marks these nodes. Our algorithm also marks the associated actual parameter vertices at the calling method and the formal parameter vertices at the called method.

Now, the dynamic slice is to be computed with respect to variable n at statement $s78$, i.e., with respect to slicing criterion $< \{n = 5\}, s78, n >$ by traversing the CFADG in backward manner. According to the FANMDS algorithm, the dynamic slice with respect to variable n at statement $s78$ is given by the expression
$dyn\_slice(s78) = \{s78, s76, a75 \rightarrow f1\_in\} \cup dyn\_slice(s76)$
$\cup\, dyn\_slice(a75 \rightarrow f1\_in)$.
By evaluating the above expression in a recursive manner, we get the final dynamic slice consisting of the statements corresponding to the nodes $m2$, $s3$, $s4$, $s5$, $m7$, $s8$, $s10$, $s11$, $m13$, $s14$, $m20$, $s21$, $m27$, $s28$, $s39$, $m40$, $s41$, $m43$, $s44$, $s45$, $s46$, $s47$, $s49$, $m51$, $s52$, $s53$, $s54$, $s55$, $s57$, $m59$, $s60$, $s61$, $s62$, $m67$, $s68$, $m69$, $s70$, $p72$, $a73$, $s74$, $a75$, $s76$, $s78$, $p81$, $a82$, $s83$, $a84$, $s85$. These are indicated as bold vertices in Figure 9 and the corresponding statements are indicated in rectangular boxes in Figure 10. Similarly, dynamic slice with respect to any slicing criterion can be computed using FANMDS algorithm.

5 Implementation

This section briefly describes the implementation of FANMDS algorithm. A dynamic slicing tool has been developed to implement the algorithm which has been named *feature-aspect dynamic slicing tool* (FADST). Figure 11 depicts the architectural design of the slicing tool FADST. The working of our slicing tool is depicted in Figure 12, through a flow chart.

In Figure 11, the executable components are depicted in rectangular boxes and the passive components are depicted in ellipses. First, the features required to compose and aspects to be captured are selected. The selected features, the selected aspects, and the slicing criterion consisting of input, line number, and variable are provided to FADST through the *Graphical User Interface* (GUI) component. The *Dynamic Slicer* component interacts with the GUI component and produces the required result as output back to GUI. The *AHEAD composer* [2] composes the selected features to generate a set of Java programs. These Java programs and the selected aspects are fed to *AspectJ composer*. AspectJ composer weaves the aspects at the appropriate join points, and the result is a composed AspectJ program. The *lexical analyzer* component reads the composed AspectJ program and generates tokens from these programs. Upon encountering a useful token, the lexical analyzer component returns the token along with its type to the *parser and semantic analyzer* component. The *parser and semantic analyzer* component takes the token and analyzes it using the grammatical rules designed for the input programs.

The *code instrumentor* component instruments the composed AspectJ programs. The classes are instrumented with line numbers prefixed with c, the aspects are instrumented with line numbers prefixed with as, the methods are instrumented with line numbers prefixed with m, the pointcuts are instrumented with line numbers prefixed with p, the advices are instrumented with line numbers prefixed with a, and the statements containing assignments, computations, predicates are instrumented with line numbers prefixed with s.

The *CFADG constructor* component constructs the CFADG using the required program analysis information such as type of statement, sets of variables defined or used at a statement etc. The *dynamic slicer* component implements the FANMDS algorithm. We have used Java language for our implementation. A compiler writing tool, *ANTLR* (Another Tool for Language Recognition)[5], has been used for *lexical analyzer*, *parser and semantic analyzer* components of FADST.

An adjacency matrix `adj[][]` has been used for storing the CFADG with respect to a selected composition of features of the given feature-oriented program.

Arrays are used to store the sets *Defn(v)*, *Usage(v)*, *RecDefn(v)*, and *dyn_slice(u)*.

[5] `www.antlr.org`

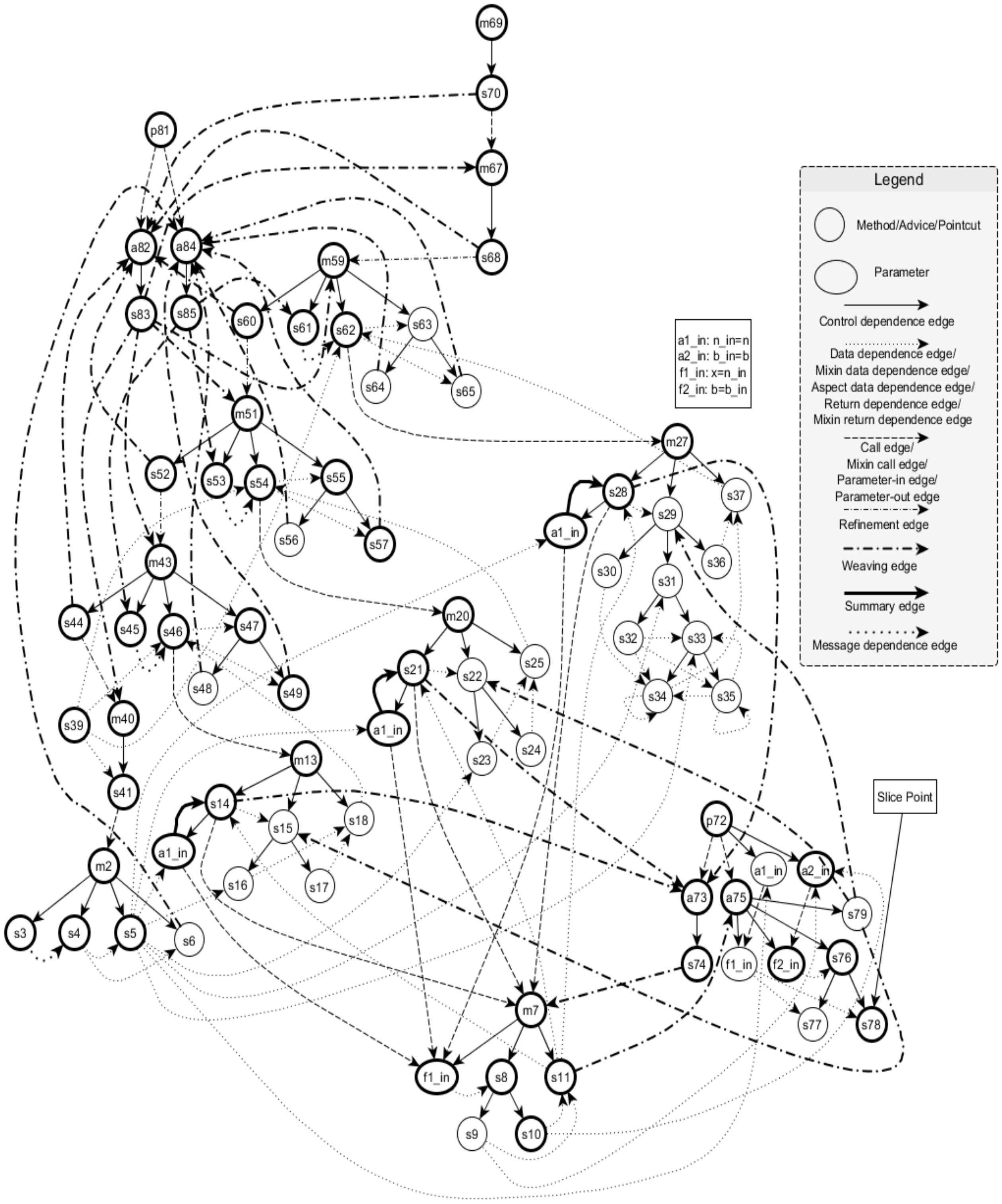
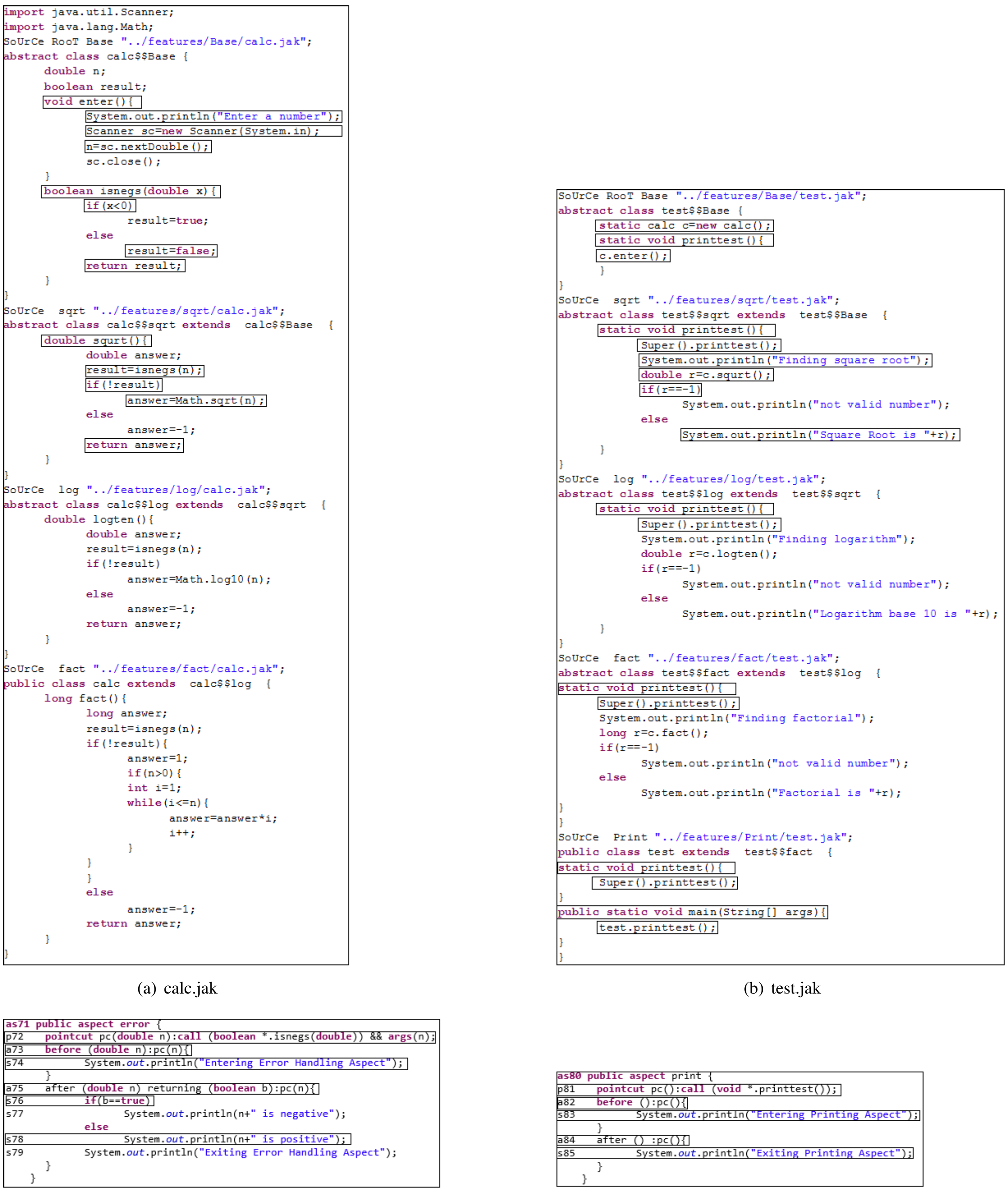


Figure 9: CFADG showing statements included in dynamic slice as bold nodes



```
import java.util.Scanner;
import java.lang.Math;
SoUrCe RooT Base "../features/Base/calc.jak";
abstract class calc$$Base {
	double n;
	boolean result;
	void enter(){
		System.out.println("Enter a number");
		Scanner sc=new Scanner(System.in);
		n=sc.nextDouble();
		sc.close();
	}
	boolean isnegs(double x){
		if(x<0)
			result=true;
		else
			result=false;
		return result;
	}
}
SoUrCe  sqrt "../features/sqrt/calc.jak";
abstract class calc$$sqrt extends  calc$$Base  {
	double squrt(){
		double answer;
		result=isnegs(n);
		if(!result)
			answer=Math.sqrt(n);
		else
			answer=-1;
		return answer;
	}
}
SoUrCe  log "../features/log/calc.jak";
abstract class calc$$log extends  calc$$sqrt  {
	double logten(){
		double answer;
		result=isnegs(n);
		if(!result)
			answer=Math.log10(n);
		else
			answer=-1;
		return answer;
	}
}
SoUrCe  fact "../features/fact/calc.jak";
public class calc extends  calc$$log  {
	long fact(){
		long answer;
		result=isnegs(n);
		if(!result){
			answer=1;
			if(n>0){
			int i=1;
			while(i<=n){
				answer=answer*i;
				i++;
			}
		}
		}
		else
			answer=-1;
		return answer;
	}
}
```

(a) calc.jak

```
SoUrCe RooT Base "../features/Base/test.jak";
abstract class test$$Base {
	static calc c=new calc();
	static void printtest(){
	c.enter();
	}
}
SoUrCe  sqrt "../features/sqrt/test.jak";
abstract class test$$sqrt extends  test$$Base  {
	static void printtest(){
		Super().printtest();
		System.out.println("Finding square root");
		double r=c.squrt();
		if(r==-1)
			System.out.println("not valid number");
		else
			System.out.println("Square Root is "+r);
	}
}
SoUrCe  log "../features/log/test.jak";
abstract class test$$log extends  test$$sqrt  {
	static void printtest(){
		Super().printtest();
		System.out.println("Finding logarithm");
		double r=c.logten();
		if(r==-1)
			System.out.println("not valid number");
		else
			System.out.println("Logarithm base 10 is "+r);
	}
}
SoUrCe  fact "../features/fact/test.jak";
abstract class test$$fact extends  test$$log  {
static void printtest(){
	Super().printtest();
	System.out.println("Finding factorial");
	long r=c.fact();
	if(r==-1)
		System.out.println("not valid number");
	else
		System.out.println("Factorial is "+r);
}
}
SoUrCe  Print "../features/Print/test.jak";
public class test extends  test$$fact  {
static void printtest(){
	Super().printtest();
}
public static void main(String[] args){
	test.printtest();
}
}
```

(b) test.jak

```
as71 public aspect error {
p72    pointcut pc(double n):call (boolean *.isnegs(double)) && args(n);
a73    before (double n):pc(n){
s74            System.out.println("Entering Error Handling Aspect");
       }
a75    after (double n) returning (boolean b):pc(n){
s76            if(b==true)
s77                    System.out.println(n+" is negative");
               else
s78                    System.out.println(n+" is positive");
s79            System.out.println("Exiting Error Handling Aspect");
       }
   }
```

(c) error.aj

```
as80 public aspect print {
p81    pointcut pc():call (void *.printtest());
a82    before ():pc(){
s83            System.out.println("Entering Printing Aspect");
       }
a84    after () :pc(){
s85            System.out.println("Exiting Printing Aspect");
       }
   }
```

(d) print.aj

Figure 10: Dynamic slice with respect to slicing criterion $< \{n = 5\}, s78, n >$ depicted as statements in rectangular boxes

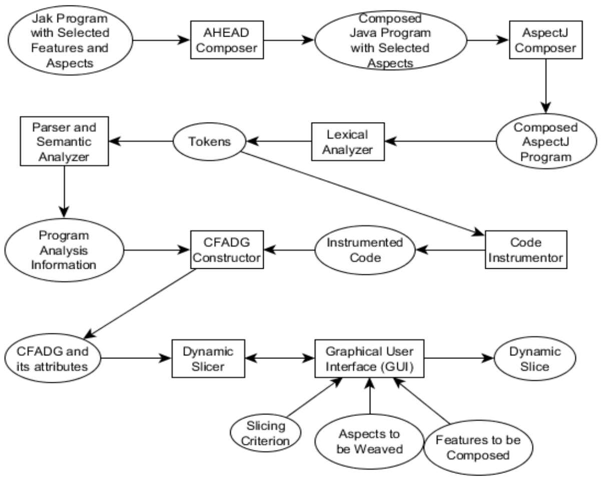


Figure 11: Architecture of the slicing tool

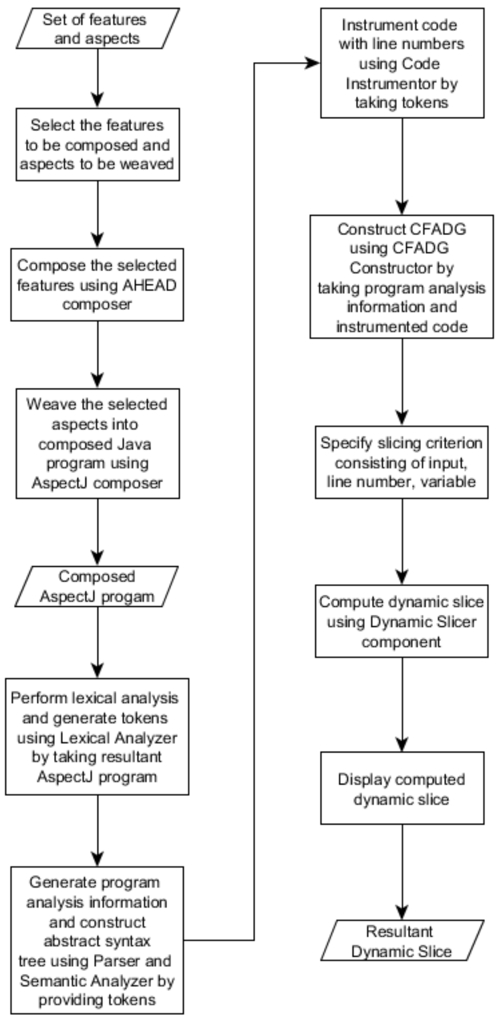

Set of features and aspects
Select the features to be composed and aspects to be weaved
Compose the selected features using AHEAD composer
Weave the selected aspects into composed Java program using AspectJ composer
Composed AspectJ program
Perform lexical analysis and generate tokens using Lexical Analyzer by taking resultant AspectJ program
Generate program analysis information and construct abstract syntax tree using Parser and Semantic Analyzer by providing tokens
Instrument code with line numbers using Code Instrumentor by taking tokens
Construct CFADG using CFADG Constructor by taking program analysis information and instrumented code
Specify slicing criterion consisting of input, line number, variable
Compute dynamic slice using Dynamic Slicer component
Display computed dynamic slice
Resultant Dynamic Slice

Figure 12: Flowchart for working of the slicing tool given in Figure 11

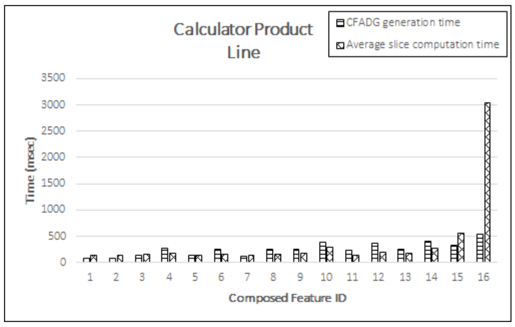


Figure 13: CFADG generation time and Average slice computation time for Calculator Product Line

5.1 Case studies and experimental results

We have applied our algorithm to some product lines [6,7]. We have also taken some open-source Java programs[8] [9] [10]. We have developed few product lines by identifying various features and converting these available Java programs into corresponding Jak programs. It may be noted that Jak is one of the feature-oriented programming languages. We have also taken the models of few product lines (such as calculator product line, stack product line, graph product line) from the work of different researchers [45, 44, 43, 46] and developed the corresponding Jak programs. These may be considered as representative feature-oriented programs with aspect-oriented extensions. In all the product lines, we have identified the aspects that are scattered throughout the program. The product lines we have taken as our case studies have various features and aspects which can be used for composing a variety of software product lines. We have taken fifteen product lines as our case studies. The characteristics of our software product lines are depicted in Table 1. These programs are executed for different compositions of features with different aspects weaved for different inputs. Also, the algorithm has been tested for different slicing criteria for different compositions of features and different inputs.

The CFADG generation time and average slice computation time for various compositions of features in different product lines are depicted in Figures 13–27.

It can be inferred from Figures 13–27 that different compositions of features result in different slice computation times. The aspects weaved at more number of join points take more time than the aspects weaved at less number of join points. For example, in *Calculator Product Line* (CPL), the number of join points where the aspect *Print* is weaved is more than that of aspect *Error*. That's why the slice computation time for the program where *Print* aspect

[6]`http://spl2go.cs.ovgu.de/projects`
[7]`http://www.infosun.fim.uni-passau.de/spl/apel/fh`
[8]`http://www.sanfoundry.com/java-program-implement-avl-tree/`
[9]`http://www.geeksforgeeks.org/avl-tree-set-2-deletion/`
[10]`https://ankurm.com/implementing-singly-linked-list-in-ja`

Table 1: List of Case Study Software Product Lines

Sl. No.	Name of Product Line	Description	Total No. of Features Supported		Total No. of Aspects weaved		Total No. of lines in composed AspectJ file
			Mandatory	**Optional**	**Mandatory**	**Optional**	
1	Calculator Product Line (CPL)	Calculates the factorial, square root and logarithmic value with base 10 of a number.	2 (Base, Print)	3 (sqrt, log, fact)	1 (Error)	1 (Print)	85
2	Stack Product Line (SPL)	Models variations of stacks.	1 (Stack)	3 (Counter, Lock, Undo)	1 (Size)	1 (Top)	130
3	Graph Product Line	Models variability for different types of graphs such as colored, weighted etc.	1 (Base)	4 (Weight, Color, Recursive, PrintHeader)	1 (Print)	1 (AddNode)	150
4	AVL Tree Product Line (AVLTPL)	Simulates the various operations on an AVL tree.	2 (Base, Display)	4 (Insert, Delete, Count, Search)	1 (ComputeHeight	1 (Rotate)	315
5	Single Linked List Product Line (SLLPL)	Simulates the various operations on a single linked list.	2 (Base, Display)	7 (InsertBegin, InsertEnd, InsertAfter, DeleteBegin, DeleteEnd, DeleteAfter, Count)	2 (Insert)	0	195
6	DesktopSearcher	Program for indexing and content based searching in files	7	9	3	4	2516
7	TankWar	A Game	12	19	6	7	3746
8	GPL	Graph and algorithm library	12	24	5	12	801
9	MobileMedia	MobileMedia is a Software Product Line (SPL) that manipulates photo, music, and video on mobile devices. It is a multimedia management for phones	10	37	5	10	4669
10	Digraph	A library for representing and manipulating directed graph structures. Beside basic graphs, it supports various operations such as removal, traversal, and transposition, implemented as optional features.	1	3	1	2	1733
11	Elevator	Simulates various operations of an Elevator	3	3	2	1	873
12	Vistex	VisTex is a product line that features graphical manipulation of graphs and their textual representation. It is designed to be easily extendible for specific graph-based applications such as UML.	5	11	2	3	1890
13	Violet	Graphical model editor	15	73	6	9	9203
14	Notepad	Graphical text editor	6	7	3	3	1672
15	PkJab	Instant messaging client for Jabber.	3	5	2	2	3963

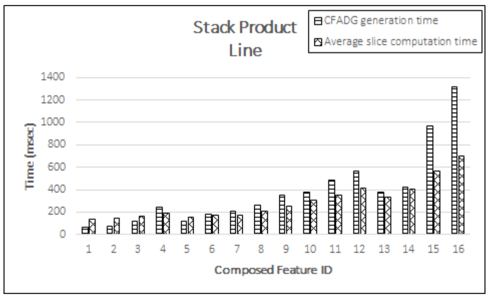


Figure 14: CFADG generation time and Average slice computation time for Stack Product Line

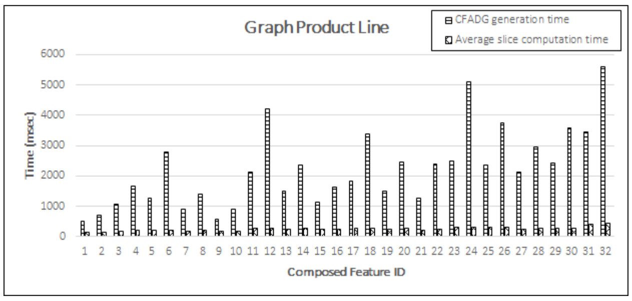


Figure 15: CFADG generation time and Average slice computation time for Graph Product Line

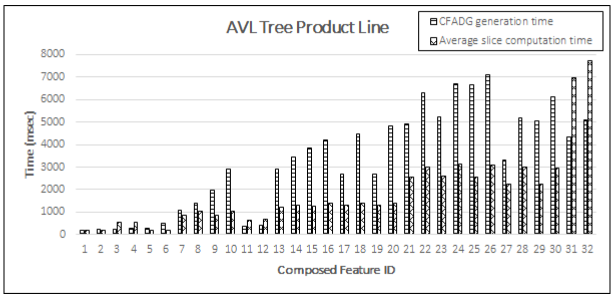


Figure 16: CFADG generation time and Average slice computation time for AVL Tree Product Line

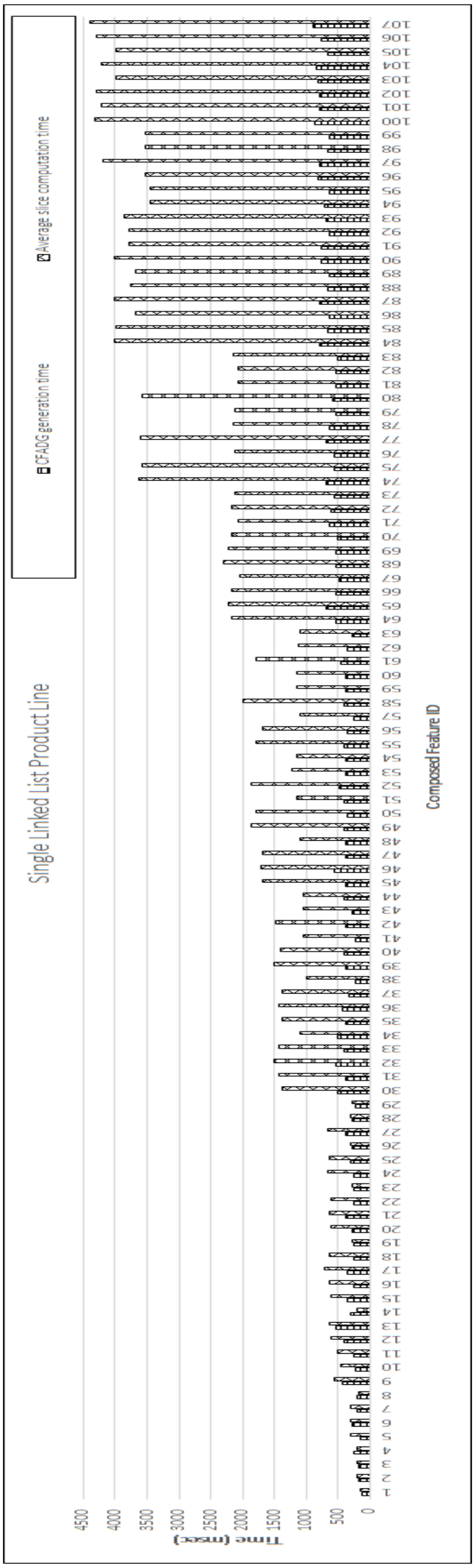


Figure 17: CFADG generation time and Average slice computation time for Single Linked List Product Line

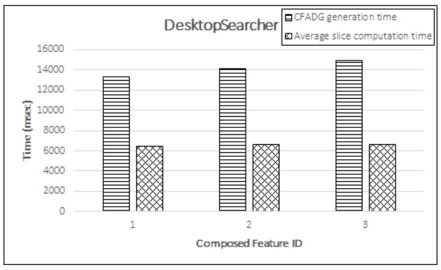


Figure 18: CFADG generation time and Average slice computation time for DesktopSearcher

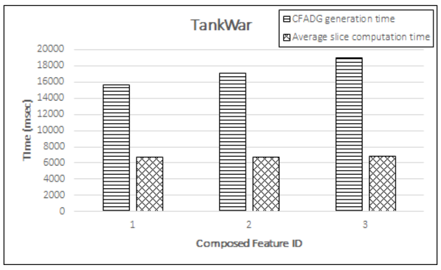


Figure 19: CFADG generation time and Average slice computation time for TankWar

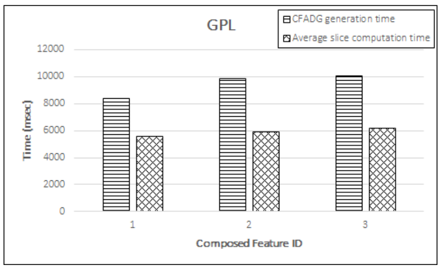


Figure 20: CFADG generation time and Average slice computation time for GPL

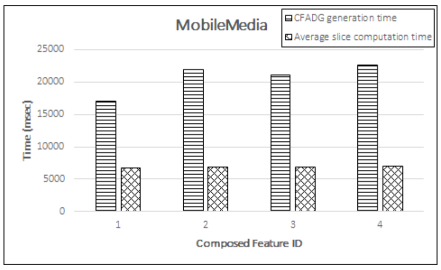


Figure 21: CFADG generation time and Average slice computation time for MobileMedia

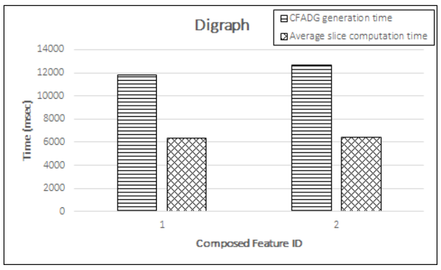


Figure 22: CFADG generation time and Average slice computation time for Digraph

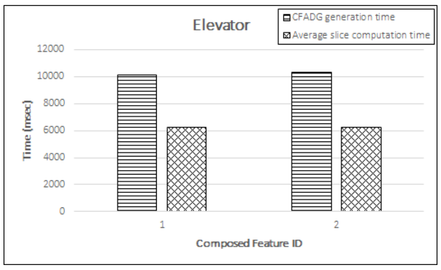


Figure 23: CFADG generation time and Average slice computation time for Elevator

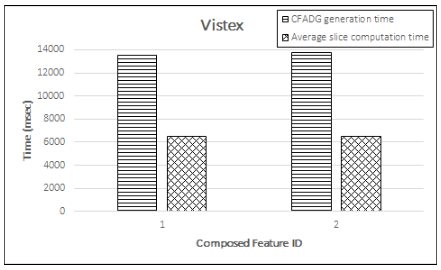


Figure 24: CFADG generation time and Average slice computation time for Vistex

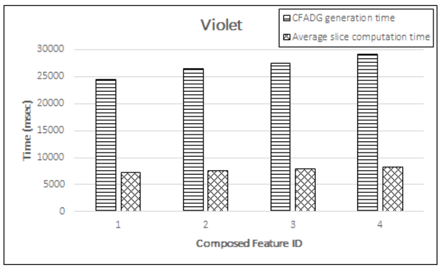


Figure 25: CFADG generation time and Average slice computation time for Violet

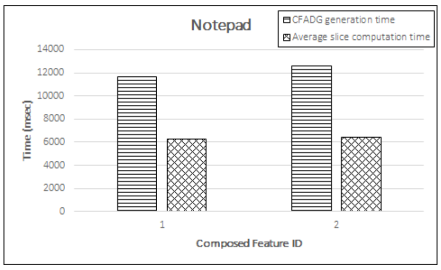


Figure 26: CFADG generation time and Average slice computation time for Notepad

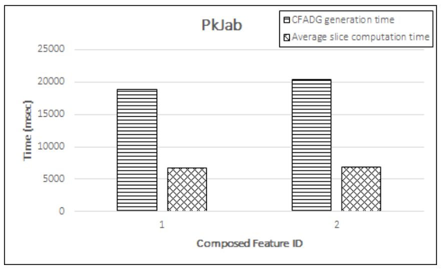


Figure 27: CFADG generation time and Average slice computation time for PkJab

is weaved is more than that of the program where *Error* aspect is weaved. The features containing more number of loops take more time. The composed features containing less number of executable statements take less time compared to those containing more number of executable statements.

6 Comparison with related work

Several works have been carried out on slicing of procedure-oriented programs [34, 32, 33, 30, 47], object-oriented programs [11, 21, 39, 22, 15], aspect-oriented programs [37, 9, 10, 16, 18, 23]. But very, few work have been carried out on slicing of feature-oriented programs [35].

Zhao [9] was the first to develop a two-phase slicing algorithm to compute static slices of aspect-oriented programs. Later, Zhao et al. [10] developed an efficient algorithm for constructing system dependence graph for aspect-oriented programs. Ray et al. [16] developed an algorithm to compute dynamic slices of aspect-oriented programs by constructing *Aspect System Dependence Graph* (AOSG). They had introduced a new logical node called *C-node* to capture communication dependencies among the non-aspect code and aspect code. They had also introduced a new arc called *aspect-membership arc* to connect the dependence graphs of the non-aspect code and aspect code. They had not shown the actual parameters in the pointcuts. Singh et al. [18] proposed a method to compute slices depending upon the slice point location in the program. Their computed slice was an executable slice. Munjal et al. [23] automated the generation of system dependence graphs (SDG) for aspect-oriented programs by analysing the bytecode of aspect-oriented programs. Then, they proposed a three-phase slicing algorithm to compute static slices using the intermediate graph for a given aspect-oriented program. All the above works [9, 15, 16, 18, 23] have not considered feature-oriented programs.

Apel et al. [3] presented a novel language for FOP in C++ namely FeatureC++. They also mentioned few problems of FOP languages. Apel et al. [4] demonstrated FeatureC++ along with its adaptation to Aspect-Oriented Programming (AOP) concepts. They discussed the use of FeatureC++ in solving various problems related to incremental software development using AOP concepts. They also discussed the weaknesses of FOP for modularization of crosscutting concerns. Apel et al. [5] discussed the limitations of crosscutting modularity and the missing support of C++. They also focused on solutions for ease evolvability of software. Batory [2] presented basic concepts of FOP and a subset of the tools of the *Algebraic Hierarchical Equations for Application Design* (AHEAD) tool suite. Apel et al. [7] presented an overview of feature-oriented software development (FOSD) process. They had identified various key issues in different phases of FOSD. Thum et al. [6] developed an open source framework for FOSD namely FeatureIDE that supported all phases of FOSD along with support for feature-oriented programming languages, and delta-oriented programming languages, aspect-oriented programming languages. Pereira et al. [20] discussed the findings of SPL management tools from a Systematic Literature Review (SLR). These works [7, 5, 3, 4, 2, 20, 6] discussed only the programming and development aspects of FOP and did not consider the slicing aspects. We have presented a technique for dynamic slicing of feature-oriented programs with aspect-oriented extensions using Jak as the FOP language.

Very few work have been carried out on slicing of feature-oriented programs [35]. Sahu et al. [35] suggested a technique to compute dynamic slices of feature-oriented programs. Their technique first composed the selected features of feature-oriented programs. Then, they used an execution trace file and a dependence-based program representation namely *dynamic feature-oriented dependence graph* (DFDG). The dynamic slice was computed by traversing DFDG in breadth-first or depth-first manner and mapping the traversed vertices to the program statements. They had missed some of the dependences such as mixn call edge, refinement edge, and mixin return dependence edge, etc. that might arise in feature-oriented programs. The drawback of their approach is the use of execution trace file which may lead to more slice computation time. They had not considered the aspect-oriented extensions of feature-oriented programs. In our approach, we have not used any execution trace file. Usually, the execution trace file is used to store the execution history of each executed statement for a given input. Much time is required to store and retrieve the executed statements. The statements are then used for calculation of dynamic slice for each statement. Thus, extra time is required to perform I/O operations on an execution trace file. We do not use any execution trace file. During execution of the program for a given input, the dynamic slice for each statement is computed by marking and unmarking process. Thus, there is no requirement of any execution trace file for storing the executed statements. So, our proposed approach does not take any extra time to read from or write into the execution trace file, thereby reducing the slice extraction time. Also, we have considered the aspects that are scattered throughout the code. Our algorithm does not create any new node in the intermediate representation CFADG during runtime. This results in faster computation of slices.

7 Conclusion and future work

We have presented an approach to compute dynamic slices of feature-oriented programs with aspect-oriented extensions. The features required for composition are first selected and composed using *Algebraic Hierarchical Equations for Application Design (AHEAD)* composer. Then, the aspects are weaved into the generated composed Java program using *AspectJ* composer to produce the resultant AspectJ program. The intermediate dependence based representation of the program containing Jak code and AspectJ code is constructed and it is called *Composite Feature-Aspect Dependence Graph (CFADG)*. The program is executed for an input. During execution, the nodes of CFADG are marked and unmarked according to our *feature-aspect node marking dynamic slicing* (FANMDS) algorithm. We have developed a tool to implement our FANMDS algorithm and named it FADST. Our tool FADST computes the dynamic slices and the average slice extraction times for various compositions of features and aspects weaved for various product lines. Currently, our tool is able to handle various compositions for few product lines with few aspects captured. Also, current evaluation only uses primitive feature-oriented programs. In future, we will extend our tool to handle more number of product lines with more number of compositions.

Our algorithm may easily be extended to compute dynamic slices of other feature-oriented languages like FeatureC++, FeatureRuby, FeatureHouse, Fuji, etc. Also, the extension of the algorithm can be used to compute conditioned slices, amorphous slices for feature-oriented programs with various aspects captured. We will also find out the differences in the performance of different aspects.

References

[1] Christian Prehofer (1997) Feature-Oriented Programming: A Fresh Look at Objects, *Proceedings of 11th European Conference on Object-Oriented Programming (ECOOP)*, Springer, Berlin, Heidelberg, pp. 419–443. https://doi.org/10.1007/bfb0053389

[2] Don Batory (2006) A Tutorial on Feature–Oriented Programming and the AHEAD Tool Suite, *Proceedings of the 2005 International Conference on Generative and Transformational Techniques in Software Engineering (GTTSE'05)*, Springer–Verlag, Berlin, Heidelberg, pp. 3–35. https://doi.org/10.1007/11877028_1

[3] Sven Apel and Thomas Leich and Marko Rosenmuller and Gunter Saake (2005) FeatureC++: Feature-Oriented and Aspect-Oriented Programming in C++, Tech. rep., Department of Computer Science, Otto-von-Guericke University, Magdeburg, Germany.

[4] Sven Apel and Thomas Leich and Marko Rosenmuller and Gunter Saake (2005) FeatureC++: On the Symbiosis of Feature-Oriented and Aspect-Oriented Programming *Proceedings of the International Conference on Generative Programming and Component Engineering (GPCE'05)*, Springer, pp. 125–140. https://doi.org/10.1007/11561347_10

[5] Sven Apel and Thomas Leich and Marko Rosenmuller and Gunter Saake (2005) Combining Feature-Oriented and Aspect-Oriented Programming to Support Software Evolution, *Proceedings of the 2nd ECOOP Workshop on Reflection, AOP and Meta-Data for Software Evolution (RAM-SE)*, School of Computer Science, University of Magdeburg, July, pp. 3–16.

[6] Thomas Thum and Christian Kastner and Fabian Benduhn and Jens Meinicke and Gunter Saake and Thomas Leich (2014) FeatureIDE: An extensible framework for feature-oriented software development, *Science of Computer Programming*, 79, pp. 70–85. https://doi.org/10.1016/j.scico.2012.06.002

[7] Sven Apel and Christian Kastner (2009) An Overview of Feature-Oriented Software Development. *Journal of Object Technology*, 8(5), pp. 49–84, July–August. https://doi.org/10.5381/jot.2009.8.5.c5

[8] Gregor Kiczales and John Irwin and John Lamping and Jean Marc Loingtier and Cristiana Videira Lopes and Chris Maeda and Anurag Mendhekar (1997) Aspect-Oriented Programming, *Proceedings of the European Conference on Object-Oriented Programming (ECOOP)*, Finland, June, pp. 220–242. https://doi.org/10.1007/bfb0053381

[9] Jianjun Zhao (2002) Slicing Aspect-Oriented Software, *Proceedings of 10th International Workshop on Program Comprehension*, pp. 251–260, June. https://doi.org/10.1109/wpc.2002.1021346

[10] Jianjun Zhao and Martin Rinard (2003) System Dependence Graph Construction for Aspect-Oriented Programs. Technical report, Laboratory for Computer Science, Massachusetts Institute of Technology, USA, March.

[11] Loren Larsen and Mary Jean Harrold (1996) Slicing Object-Oriented Software, *Proceedings of 18th International Conference on Software Engineering*, pp. 495–505, March. https://doi.org/10.1109/icse.1996.493444

[12] Timon Ter Braak (2006) Extending Program Slicing in Aspect–Oriented Programming With Inter–Type Declarations, *5th TSConIT Program*, June.

[13] Aspect Oriented Programming. www.wikipedia.org.

[14] Hiralal Agrawal and Joseph R. Horgan (1990) Dynamic Program Slicing, ACM SIGPLAN Notices, *Proceedings of the ACM SIGPLAN 1990 conference on Programming language design and implementation PLDI'90*, 25(6), pp. 246–256, June. https://doi.org/10.1145/93542.93576

[15] Durga Prasad Mohapatra and Rajib Mall and Rajeev Kumar (2004) An Edge Marking Technique for Dynamic Slicing of Object-Oriented Programs, *Proceedings of the 28th Annual International Computer Software and Applications Conference (COMPSAC'04)*. https://doi.org/10.1109/cmpsac.2004.1342806

[16] Abhisek Ray and Siba Mishra and Durga Prasad Mohapatra (2012) A Novel Approach for Computing Dynamic Slices of Aspect-Oriented Programs, *International Journal of Computer Information Systems*, 5(3),pp. 6–12, September.

[17] Abhisek Ray and Siba Mishra and Durga Prasad Mohapatra (2013) An Approach for Computing Dynamic Slice of Concurrent Aspect-Oriented Programs *International Journal of Software Engineering and Its Applications*, 7(1), pp. 13–32, January.

[18] Jagannath Singh and Durga Prasad Mohapatra (2013) A Unique Aspect-Oriented Program Slicing Technique, *Proceedings of International Conference on Advances in Computing, Communications and Informatics (ICACCI'13)*, pp. 159–164. https://doi.org/10.1109/icacci.2013.6637164

[19] Jagannath Singh and Dishant Munjal and Durga Prasad Mohapatra (2014) Context Sensitive Dynamic slicing of Concurrent Aspect-Oriented Programs, *Proceedings of 21st Asia-Pacific Software Engineering Conference (APSEC'14)*, pp. 167–174. https://doi.org/10.1109/apsec.2014.35

[20] Juliana Alves Pereira and Kattiana Constantino and Eduardo Figueiredo (2015) A Systematic Literature Review of Software Product Line Management Tools, *Proceeddings of 14th International Conference on Software Reuse (ICSR'15)*, Berlin Heidelberg, pp. 73–89. https://doi.org/10.1007/978-3-319-14130-5_6

[21] Durga Prasad Mohapatra and Rajeev Kumar and Rajib Mall and D. S. Kumar and Mayank Bhasin (2006) Distributed dynamic slicing of Java programs, *Journal of Systems and Software*, 79(12), pp. 1661–1678. https://doi.org/10.1016/j.jss.2006.01.009

[22] Durga Prasad Mohapatra and Rajib Mall and Rajeev Kumar (2005) Computing dynamic slices of concurrent object-oriented programs, *Information & Software Technology*, 47(12), pp. 805–817. https://doi.org/10.1016/j.infsof.2005.02.002

[23] Dishant Munjal and Jagannath Singh and Subhrakanta Panda and Durga Prasad Mohapatra (2015) Automated Slicing of Aspect-Oriented Programs using Bytecode Analysis, *Proceedings of IEEE 39th Annual International Computers, Software & Applications Conference (COMPSAC 2015)*, pp. 191–199. https://doi.org/10.1109/compsac.2015.98

[24] Madhusmita Sahu and Durga Prasad Mohapatra (2007) A Node-Marking Technique for Dynamic Slicing of Aspect-Oriented Programs, *Proceedings of 10th International Conference on Information Technology (ICIT 2007)*, pp. 155–160. https://doi.org/10.1109/icit.2007.70

[25] Diganta Goswami and Rajib Mall (1999) Fast Slicing of Concurrent Programs, *Proceedings of 6th International Conference on High Performance Computing (HiPC 1999)*, pp. 38–42. https://doi.org/10.1007/978-3-540-46642-0_6

[26] Diganta Goswami and Rajib Mall (2000) Dynamic Slicing of Concurrent Programs, *Proceedings of 7th International Conference on High Performance Computing (HiPC 2000)*, pp. 15–26. https://doi.org/10.1007/3-540-44467-x_2

[27] Jaiprakash T. Lallchandani and Rajib Mall (2011) A Dynamic Slicing Technique for UML Architectural Models, *IEEE Transactions on Software Engineering*, 37(6), pp. 737–771. https://doi.org/10.1109/tse.2010.112

[28] Philip Samuel and Rajib Mall (2009) Slicing-based test case generation from UML activity diagrams, *ACM SIGSOFT Software Engineering Notes*, 34(6), pp. 1–14. https://doi.org/10.1145/1640162.1666579

[29] Jaiprakash T. Lallchandani and Rajib Mall (2010) Integrated state-based dynamic slicing technique for UML models, *IET Software*, 4(1), pp. 55–78. https://doi.org/10.1049/iet-sen.2009.0080

[30] G. B. Mund and Rajib Mall (2006) An efficient interprocedural dynamic slicing method, *The Journal of Systems and Software*, 79, pp. 791–806. https://doi.org/10.1016/j.jss.2005.07.024

[31] Diganta Goswami and Rajib Mall (2004) A parallel algorithm for static slicing of concurrent programs, *Concurrency – Practice and Experience*, 16(8), pp. 751–769. https://doi.org/10.1002/cpe.789

[32] G. B. Mund and R. Mall and S. Sarkar (2003) Computation of intraprocedural dynamic program slices, *Information and Software Technology*, 45 (8), pp. 499–512. https://doi.org/10.1016/S0950-5849(03)00029-6

[33] G. B. Mund and R. Mall and S. Sarkar (2002) An efficient dynamic program slicing technique, *Information and Software Technology*, 44 (2), pp. 123–132. https://doi.org/10.1016/s0950-5849(01)00224-5

[34] Diganta Goswami and Rajib Mall (2002) An efficient method for computing dynamic program slices, *Information Processing Letters*, 81(2), pp. 111–117. https://doi.org/10.1016/s0020-0190(01)00202-2

[35] Madhusmita Sahu and Durga Prasad Mohapatra (2016) Dynamic Slicing of Feature-Oriented Programs, *Proceedings of 3rd International Conference on Advanced Computing, Networking and Informatics (ICACNI 2015)*, pp. 381–388. https://doi.org/10.1007/978-81-322-2529-4_40

[36] Mark Weiser (1981) Program Slicing, *Proceedings of the 5th International Conference on Software Engineering (ICSE)*, pp. 439–449. IEEE Computer Society, March.

[37] Durga Prasad Mohapatra and Madhusmita Sahu and Rajib Mall and Rajeev Kumar (2008) Dynamic Slicing of Aspect–Oriented Programs, *Informatica*, 32(3), pp. 261–274.

[38] Jaiprakash T. Lallchandani and Rajib Mall (2005) Computation of Dynamic Slices for Object-Oriented Concurrent Programs, *Proceedings of Asia Pacific Software Engineering Conference (APSEC 2005)*, pp. 341–350. https://doi.org/10.1109/apsec.2005.51

[39] Jianjum Zhao (1998) Dynamic Slicing of Object–Oriented Programs, Technical report, Information Processing Society of Japan, pp. 1–7, May.

[40] Sebastian Gunther and Sagar Sunkle (2009) Feature-Oriented Programming with Ruby. In *Proceedings of the First International Workshop on Feature-Oriented Software Development (FOSD'09)*, pp. 11–18, October. https://doi.org/10.1145/1629716.1629721

[41] Sebastian Gunther and Sagar Sunkle (2012) rbFeatures: Feature-oriented programming with Ruby, *Science of Computer Programming*, 77(3), pp. 152–173, March. https://doi.org/10.1016/j.scico.2010.12.007

[42] Bogdan Korel and Satish Yalamanchili (1994) Forward Computation Of Dynamic Program Slices, *Proceedings of the 1994 ACM SIGSOFT international symposium on Software testing and analysis(ISSTA'94)*, pp. 66–79, August. https://doi.org/10.1145/186258.186514

[43] Sven Apel and Thomas Leich and Gunter Saake(2008) Aspectual Feature Modules, *IEEE Transactions On Software Engineering*, 34(2),pp. 162–180, March/April. https://doi.org/10.1109/tse.2007.70770

[44] Jia Liu and Don Batory and Srinivas Nedunuri (2005) Modeling Interactions in Feature-Oriented Software Designs, *Proceedings of International Conference on Feature Interactions in Telecommunications and Software Systems (ICFI 2005)*, pp. 178–197.

[45] Ian Adams and Sigmon Myers (2009) FOP and AOP: Benefits, Pitfalls and Potential for Interaction, pp. 1–7.

[46] Sagar Sunkle and Marko Rosenmuller and Norbert Siegmund and Syed Saif ur Rahman and Gunter Saake and Sven Apel (2008) Features as First-class Entities-Toward a Better Representation of Features. *Proceedings of Workshop on Modularization, Composition, and Generative Techniques for Product Line Engineering*, pp. 27–34, October.

[47] Susan Horwitz and Thomas Reps and David Binkley (1990) Inter-Procedural Slicing Using Dependence Graphs, *ACM Transactions on Programming Languages and Systems*, vol. 12, no. 1, pp. 26–60, January. https://doi.org/10.1145/77606.77608

8 Appendices

A Construction of CFADG

Algorithm 8 Construction of CFADG

```
    Input: The feature-oriented program containing aspects with selected required features and weaved aspects.
    Output: The composite feature-aspect dependence graph (CFADG).
 1: procedure CONSTRUCTPDG()
 2:     for start of a method do
 3:         Create method entry node.
 4:     for each executable statement in the program do
 5:         Create a node.
 6:     for all nodes created do
 7:         if node $n_2$ is under the scope of node $n_1$ then
 8:             Add control dependence edge from $n_1$ to $n_2$, $n_1 \to n_2$.
 9:         if node $n_1$ controls the execution of node $n_2$ then
10:             Add control dependence edge from $n_1$ to $n_2$, $n_1 \to n_2$.
11:         if node $n_2$ uses the value of a variable that is defined at node $n_1$ then
12:             Add data dependence edge from $n_1$ to $n_2$, $n_1 \to n_2$.
13: procedure CONSTRUCTMXDG()
14:     for all methods in a mixin do
15:         Call ConstructPDG().
16:     for entry of a mixin do
17:         Create mixin entry node.
18:     for each parameter present in the method call do
19:         Create an actual-in parameter node.
20:     for each parameter present in the method definition do
21:         Create a formal-in parameter node.
22:     for each parameter in the method call that is modified inside the method do
23:         Create an actual-out parameter node.
24:     for each actual-out parameter node do
25:         Create corresponding formal-out parameter node.
26:     for all nodes created do
27:         if node $x$ corresponds to mixin entry node and node $y$ is a method entry node then
28:             Add mixin membership edge from $x$ to $y$, $x \to y$.
29:         if node $n_1$ returns a value to the calling method at node $n_2$ within a mixin layer then
30:             Add return dependence edge from $n_1$ to $n_2$, $n_1 \to n_2$.
31:         if node $n_1$ calls a method that is defined at node $n_2$ within a mixin layer then
32:             Add call edge from $n_1$ to $n_2$, $n_1 \to n_2$.
33:         if node $n_1$ calls a method that is defined at node $n_2$ within a mixin layer by passing parameters then
34:             Add call edge from $n_1$ to $n_2$, $n_1 \to n_2$.
35:             Add parameter-in edge from actual-in parameter node to corresponding formal-in parameter node.
36:             Add parameter-out edge from formal-out parameter node to corresponding actual-out parameter node.
37:         if node $n_1$ is an actual-in parameter node and node $n_2$ is an actual-out parameter node such that the value at node $n_1$ affects the value at node $n_2$ then
38:             Add summary edge from $n_1$ to $n_2$, $n_1 \to n_2$.
39: procedure CONSTRUCTSDG()
40:     for all mixins within a mixin layer do
41:         Call ConstructMxDG.
42:     for all nodes created do
43:         if node $x$ is a polymorphic method call then
44:             Create polymorphic choice vertex.
45:             if node $y$ is a polymorphic choice vertex then
46:                 Add a call edge from $x$ to $y$, $x \to y$
47:         if node $x$ is a new operator node and node $y$ is the corresponding constructor node then
48:             Add call edge from $n_1$ to $n_2$, $n_1 \to n_2$.
49:             Add parameter-in edge from actual-in parameter node to corresponding formal-in parameter node.
50:             Add parameter-out edge from formal-out parameter node to corresponding actual-out parameter node.
```

```
51:     Remove *mixin membership edges*.
52:     Remove *mixin entry nodes*.
53: **procedure** CONSTRUCTADG()
54:     **for** start of an advice **do**
55:         Create *advice start vertex*.
56:     **if** advice contains parameters **then**
57:         Create *formal-in* and *formal-out* parameter nodes.
58:     **for all** nodes created **do**
59:         **if** node $n_2$ is under the scope of node $n_1$ **then**
60:             Add *control dependence edge* from $n_1$ to $n_2$, $n_1 \rightarrow n_2$.
61:         **if** node $n_1$ controls the execution of node $n_2$ **then**
62:             Add *control dependence edge* from $n_1$ to $n_2$, $n_1 \rightarrow n_2$.
63:         **if** node $n_2$ uses the value of a variable that is defined at node
            $n_1$ **then**
64:             Add *data dependence edge* from $n_1$ to $n_2$, $n_1 \rightarrow n_2$.
65: **procedure** CONSTRUCTIDG()
66:     **for** entry of introduction **do**
67:         Create *introduction start vertex*.
68:     **if** introduction is a method or constructor **then**
69:         Call *ConstructPDG*.
70:     **if** introduction is a field **then**
71:         Do nothing.
72: **procedure** CONSTRUCTPTDG()
73:     **for** entry of pointcut **do**
74:         Create *pointcut start vertex*.
75:     **if** pointcut contains parameters **then**
76:         Create actual-in and actual-out parameter nodes.
77: **procedure** CONSTRUCTASDG()
78:     **for** entry of an aspect **do**
79:         Create *aspect start vertex*.
80:     **for all** advices in an aspect **do**
81:         Call *ConstructADG()*.
82:     **for all** pointcuts in an aspect **do**
83:         Call *ConstructPtDG()*.
84:     **for all** introductions in an aspect **do**
85:         Call *ConstructIDG()*.
86:     **for all** nodes created **do**
87:         **if** node $x$ is aspect start vertex **then**
88:             **if** node $y$ is advice start vertex **then**
89:                 Create *aspect membership edge* from $x$ to $y$, $x \rightarrow y$.
90:             **if** node $y$ is pointcut start vertex **then**
91:                 Create *aspect membership edge* from $x$ to $y$, $x \rightarrow y$.
92:             **if** node $y$ is introduction start vertex **then**
93:                 Create *aspect membership edge* from $x$ to $y$, $x \rightarrow y$.
94:         **if** node $x$ is pointcut start node and node $y$ is advice start node
        **then**
95:             Create *data dependence edge* from $x$ to $y$, $x \rightarrow y$.
96:             Add *parameter-in edge* from actual-in parameter node to
        corresponding formal-in parameter node.
97:             Add *parameter-out edge* from formal-out parameter node
        to corresponding actual-out parameter node.
98: **procedure** CONSTRUCTCFADG()
99:     **for** each mixin layer **do**
100:        Call *ConstructSDG()*.
101:    **for** each aspect **do**
102:        Call *ConstructAsDG*.
103:    **for all** nodes created **do**
104:        **if** node $n_2$ in one mixin layer uses the value of a variable that
        is defined at node $n_1$ in different mixin layer **then**
105:            Add *mixin data dependence edge* from $n_1$ to $n_2$, $n_1 \rightarrow$
        $n_2$.
106:        **if** node $n_2$ in an aspect uses the value of a variable that is
        defined at node $n_1$ in a mixin **then**
107:            Add *aspect data dependence edge* from $n_1$ to $n_2$, $n_1 \rightarrow$
        $n_2$.
108:        **if** node $n_1$ in one mixin layer returns a value to the calling
        method at node $n_2$ in different mixin layer **then**
109:            Add *mixin return dependence edge* from $n_1$ to $n_2$, $n_1 \rightarrow$
        $n_2$.
110:        **if** node $n_1$ in one mixin layer calls a method that is defined
        at node $n_2$ in different mixin layer **then**
111:            Add *mixin call edge* from $n_1$ to $n_2$, $n_1 \rightarrow n_2$.
112:            Add *parameter-in* and *parameter-out edges*.
113:        **if** node $n_1$ calls a method that is defined at node $n_2$ using
        *Super()* method **then**
114:            Add *refinement edge* from $n_1$ to $n_2$, $n_1 \rightarrow n_2$.
115:        **if** node $n_1$ is an output statement followed by node $n_2$ and
        node $n_2$ is an input, a computation, a predicate, or a method call
        statement **then**
116:            Add *message dependence edge* from $n_1$ to $n_2$, $n_1 \rightarrow$
        $n_2$.
117:        **if** node $n_1$ is a method call node and node $n_2$ is a *before*
        advice node capturing the method called at $n_1$ **then**
118:            Add *weaving edge* from $n_1$ to $n_2$, $n_1 \rightarrow n_2$.
119:        **if** node $n_1$ is the last statement in a *before* advice and node
        $n_2$ is the method entry node of the method captured by the advice
        **then**
120:            Add *weaving edge* from $n_1$ to $n_2$, $n_1 \rightarrow n_2$.
121:        **if** node $y$ is an *after* advice node and node $n_1$ is the last
        statement in the method captured by node $n_2$ **then**
122:            Add *weaving edge* from $n_1$ to $n_2$, $n_1 \rightarrow n_2$.
123:        **if** node $n_1$ is the last statement in an *after* advice and node
        $n_2$ is the statement followed by method call node and the method is
        captured by the advice **then**
124:            Add *weaving edge* from $n_1$ to $n_2$, $n_1 \rightarrow n_2$.
125:    Remove aspect membership edges.
126:    Remove aspect entry vertices.
```

B Feature-aspect node-marking dynamic slicing (FANMDS) algorithm

Algorithm 9 Feature-Aspect Node Marking Dynamic Slicing (FANMDS) Algorithm

```
INPUT: Composite Feature-Aspect Dependence Graph (CFADG) of the program $FP$, Slicing criterion $< i, s, v >$.
OUTPUT: List of nodes contained in required dynamic slice.
1: $Marked = \phi$    ▷ Initially, unmark all nodes of CFADG.
2: Set $dyn\_slice(u) = \phi$    ▷ u is a node in CFADG.
3: Set $RecDefn(v) = NULL$    ▷ v is a variable.
4: Execute the program $FP$ for input $i$.
5: while $FP$ does not terminate do
6:     Update $dyn\_slice(u) = \{u, e_1, e_2, \ldots, e_k\} \cup dyn\_slice(e_1) \cup dyn\_slice(e_2) \cup \ldots \cup dyn\_slice(e_k)$
7:     $Marked = Marked \cup \{u\}$.    ▷ Mark node u.
8:     if $u$ is a $Defn(v)$ node then
9:         $Marked = Marked \setminus \{RecDefn(v)\}$.    ▷ Unmark the node RecDefn(v).
10:        $RecDefn(v) = u$.    ▷ Update RecDefn(v).
11:    if $u$ is a method call node for a method $M$ then
12:        $panode_M = f(M, panode)$.
13:        $Me_M = g(M, Me)$.
14:        $pfnode_M = h(M, pfnode)$.
15:        $Marked = Marked \cup \{u\}$.    ▷ Mark node u.
16:        $Marked = Marked \cup panode_M$.    ▷ Mark associated actual parameter nodes.
17:        $Marked = Marked \cup \{Me_M\}$.    ▷ Mark corresponding method entry node.
18:        $Marked = Marked \cup pfnode_M$.    ▷ Mark associated formal parameter nodes.
19:    if $u$ is a $new$ operator node for a constructor $M$ then
20:        $panode_M = f(M, panode)$.
21:        $Me_M = g(M, Me)$.
22:        $pfnode_M = h(M, pfnode)$.
23:        $Marked = Marked \cup \{u\}$.    ▷ Mark node u.
24:        $Marked = Marked \cup panode_M$.    ▷ Mark associated actual parameter nodes.
25:        $Marked = Marked \cup \{Me_M\}$.    ▷ Mark corresponding method entry node.
26:        $Marked = Marked \cup pfnode_M$.    ▷ Mark associated formal parameter nodes.
27:    if $u$ is a polymorphic node for a virtual method $M$ then
28:        $panode_M = f(M, panode)$.
29:        $Me_M = g(M, Me)$.
30:        $pfnode_M = h(M, pfnode)$.
31:        $Marked = Marked \cup \{u\}$.    ▷ Mark node u.
32:        $Marked = Marked \cup panode_M$.    ▷ Mark associated actual parameter nodes.
33:        $Marked = Marked \cup \{Me_M\}$.    ▷ Mark corresponding method entry node.
34:        $Marked = Marked \cup pfnode_M$.    ▷ Mark associated formal parameter nodes.
35:    if $u$ is a mixin call node for a method $M$ then
36:        $panode_M = f(M, panode)$.
37:        $Me_M = g(M, Me)$.
38:        $pfnode_M = h(M, pfnode)$.
39:        $Marked = Marked \cup \{u\}$.    ▷ Mark node u.
40:        $Marked = Marked \cup panode_M$.    ▷ Mark associated actual parameter nodes.
41:        $Marked = Marked \cup \{Me_M\}$.    ▷ Mark corresponding method entry node.
42:        $Marked = Marked \cup pfnode_M$.    ▷ Mark associated formal parameter nodes.
43:    if $u$ is a Super() method call node for a method $M$ then
44:        $Me_M = h(M, Me)$.
45:        $Marked = Marked \cup \{u\}$.    ▷ Mark node u.
46:        $Marked = Marked \cup \{Me_M\}$.    ▷ Mark corresponding method entry node.
47:    if $u$ is a pointcut node then
48:        $badv_P = x(P, badv)$.
49:        $aadv_P = y(P, aadv)$.
50:        $panode_P = f(P, panode)$.
51:        $pfnode_M = g(M, pfnode)$.
52:        $Marked = Marked \cup \{u\}$.    ▷ Mark node u.
53:        $Marked = Marked \cup panode_M$.    ▷ Mark corresponding actual parameter nodes.
54:        $Marked = Marked \cup pfnode_M$.    ▷ Mark corresponding formal parameter nodes.
55:        $Marked = Marked \cup badv_P$.    ▷ Mark the corresponding before advice entry node.
56:        $Marked = Marked \cup aadv_P$.    ▷ Mark the corresponding after advice entry node.
57:    if $u$ is an advice entry node for an advice $A$ corresponding to pointcut $P$ then
58:        $bbadv_A = z(A, badv_P)$.
59:        $baadv_A = z(A, aadv_P)$.
60:        $Marked = Marked \setminus bbadv_A$.
61:        $Marked = Marked \setminus baadv_A$.    ▷ Unmark all nodes in body of advice corresponding to previous execution of u.
62:        $pfnode_M = g(M, pfnode)$.
63:        $Marked = Marked \setminus pfnode_M$.    ▷ Unmark all the formal parameter nodes associated with u corresponding to previous execution of u.
```

```
64:     if $u$ is an introduction node such that $u$ is a method then
65:         $Mb_M = k(M, Mb)$.
66:         $Marked = Marked \setminus Mb_M$.  ▷ Unmark all the nodes in the method body corresponding to previous execution of u.
67:         $pfnode_M = g(M, pfnode)$.
68:         $Marked = Marked \setminus pfnode_M$.  ▷ Unmark all the formal parameter nodes associated with u corresponding to previous execution of u.
69:         if $v$ is method call node corresponding to previous execution of $u$ then
70:             $Marked = Marked \setminus v$.  ▷ Unmark the method call node corresponding to previous execution of u.
71:             $panode_v = f(v, panode)$.
72:             $Marked = Marked \setminus panode_v$.  ▷ Unmark the associated actual parameter nodes for a method call node corresponding to previous execution of u.
73:         $pfnode_M = h(M, pfnode)$.
74:         $Marked = Marked \cup \{u\}$.  ▷ Mark node u.
75:         $Marked = Marked \cup pfnode_M$.  ▷ Mark associated formal parameter nodes.
76:     if $u$ is an introduction node such that $u$ is a field then
77:         $Marked = Marked \cup \{Defn(v)\}$.  ▷ Mark Defn(v) node.
78:         $Marked = Marked \cup \{Usage(v)\}$.  ▷ Mark Usage(v) node.
79:     if $u$ is a method entry node for a method $M$ then
80:         $Mb_M = k(M, Mb)$.
81:         $Marked = Marked \setminus Mb_M$.  ▷ Unmark all the nodes in the method body corresponding to previous execution of u.
82:         $pfnode_M = g(M, pfnode)$.
83:         $Marked = Marked \setminus pfnode_M$.  ▷ Unmark all the formal parameter nodes associated with u corresponding to previous execution of u.
84:         if $v$ is method call node corresponding to previous execution of $u$ then
85:             $Marked = Marked \setminus v$.  ▷ Unmark the method call node corresponding to previous execution of u.
86:             $panode_v = f(v, panode)$.
87:             $Marked = Marked \setminus panode_v$.  ▷ Unmark the associated actual parameter nodes for a method call node corresponding to previous execution of u.
88:     if $u$ is a mixin entry node for a method $M$ then
89:         $Mb_M = k(M, Mb)$.
90:         $Marked = Marked \setminus Mb_M$.  ▷ Unmark all the nodes in the method body corresponding to previous execution of u.
91:         $pfnode_M = g(M, pfnode)$.
92:         $Marked = Marked \setminus pfnode_M$.  ▷ Unmark all the formal parameter nodes associated with u corresponding to previous execution of u.
93:         if $v$ is method call node corresponding to previous execution of $u$ then
94:             $Marked = Marked \setminus v$.  ▷ Unmark the method call node corresponding to previous execution of u.
95:             $panode_v = f(v, panode)$.
96:         $Marked = Marked \setminus panode_v$.  ▷ Unmark the associated actual parameter nodes for a method call node corresponding to previous execution of u.
97:     if $u$ is $new$ operator entry node for a constructor $M$ then
98:         $Mb_M = k(M, Mb)$.
99:         $Marked = Marked \setminus Mb_M$.  ▷ Unmark all the nodes in the method body corresponding to previous execution of u.
100:        $pfnode_M = g(M, pfnode)$.
101:        $Marked = Marked \setminus pfnode_M$.  ▷ Unmark all the formal parameter nodes associated with u corresponding to previous execution of u.
102:        if $v$ is method call node corresponding to previous execution of $u$ then
103:            $Marked = Marked \setminus v$.  ▷ Unmark the method call node corresponding to previous execution of u.
104:            $panode_v = f(v, panode)$.
105:            $Marked = Marked \setminus panode_v$.  ▷ Unmark the associated actual parameter nodes for a method call node corresponding to previous execution of u.
106:    for a given slicing command $< i, s, v >$ do
107:        Look up $dyn\_slice(u)$ for variable $v$.
108:        Display $dyn\_slice(u)$.
109:        Map nodes in $dyn\_slice(u)$ to corresponding statements in composed Java program.
110:        Map statements included in $dyn\_slice(u)$ in composed Java program to corresponding statements in composed Jak program.
111:        Display statements included in $dyn\_slice(u)$ from composed Jak program.
```

Colour-Range Histogram Technique for Automatic Image Source Detection

Nancy C. Woods and Abiodun B. C. Robert
Department of Computer Science, University of Ibadan, Ibadan, Nigeria
E-mail: Chyn.woods@gmail.com and abc.robert@live.com

Keywords: natural images, computer generated images, colour histogram

Received: November 28, 2018

Computer generated images are visually becoming increasingly genuine, due to advances in technology as well as good graphic applications. Consequently, making distinction between computer generated images and natural images is no longer a simple task. Manual identification of computer generated images have failed to resolve the problems associated with legal issues on exact qualification of images. In this work, a colour range histogram was developed to categorise colours in computer generated images and natural images from a point of reference. Four groups were selected, using the algorithm, consisting of exact Red-Green-Blue (RGB) code (group 1), colour code within a range of 10 (group 2), colour code within a range of 20 (group 3) and colour code within a range of 30 (group 4) from the point of reference. An optimised equation for the four Colour Code Groups (CCG) was developed. The computer generated images categorised an average of 69.8%, 92.9%, 96.9% and 98.6%, of any colour code for groups 1, 2, 3 and 4, respectively. The categorised colours for natural images were 31.1%, 82.6%, 90.8% and 95.0% for groups 1, 2, 3 and 4, respectively. The results showed that natural images contain a wide range of RGB colours which makes them different. Consequently, the disparity in the percentage of colours categorised can be used to differentiate computer generated images from natural images.

Povzetek: Razvit je sistem za razločevanje naravnih od računalniško generiranih umetnih slik na osnovi barvnega histograma.

1 Introduction

Digital images have become a commonplace in the lives of individuals nowadays, because of the ease of acquisition using mobile phones and other electronic devices. A digital image can be described as a rectangular two-dimensional array of pixels, where each pixel (usually a square) represents the image colour at that position and where the dimensions represent the width and height of the image as it is displayed [1]. With advances in technology and the proliferation of imaging software, digital images are now classified as either *computer generated* or *natural*. With image processing techniques, it is becoming increasingly easy to produce computer-generated images (CGI) that are so realistic with commercially available software packages [2] and these CGI are presently called Photorealistic images. How can one tell if a digital image is natural or computer generated? Usually, a photograph provides an effective and natural communication medium for humans. This is because people do not really need any special training to comprehend the content of an image and they used to believe that photographs represent the truth [3]. Unfortunately, this truth no longer holds with digital images because it is easy to manipulate them [4]. Therefore, being able to verify the credibility of digital images and perform image forensics can protect the truthfulness of digital images. It can be cumbersome and difficult for the human eye to tell the difference between the two types of images [5]. This is what the research carried out under the field of digital image forensics, among other things tries to answer. Digital image forensics is the area of image processing with the main function of assessing the authenticity and the origin of images and is divided into *active* forensics and *passive* forensics, which are further sub divided [6], [3], [4]. Figure 1 shows the classification of digital image forensics. In active forensics, additional information needs to be inserted into the host or source of the image in advance. This requires that the acquisition device should have the corresponding functionality to hold such information, some of which include digital signature [7] or digital watermarking [8]. Passive forensics technology is more practical and attempts to identify the authenticity or source of an image, based only on the characteristics of the image itself without embedded additional information [6]. Passive forensics occurs after the image has been captured and stored. Depending on its applications in different research fields, passive forensics can be broadly classified into ***tampering detection*** [9], [10], [11], ***Steganalysis*** [12] and ***source identification*** [13], [6], which is the art and science of differentiating computer generated images from natural images (NI).

The aim of this work therefore is to develop a model for colour range histogram towards discovering features that differentiate CGI from NI.

2 Literature review

Several research has been carried out in a bid to differentiate computer generated images from natural images using several approaches and features. One of the first approaches offered to differentiate NI from CGI was proposed by [14]. In their statistical approach, the first and higher-order statistics of wavelet transform coefficients are extracted from both CGI and NI to capture their statistical regularities. Another work by [15] proposed an approach using differences in image texture by considering the physical / visual properties of these images. They took into account the differences in surface and object models as well as the differences in the acquisition processes between the CGI and NI, and extracted 192 geometry features by analysing the differences existing between the physical generative process of computer graphics and photographs. Their approach extracted a lot of features in a bid to find the difference between CGI and NI.

A total of 216 features, based on the RGB colour information of CGI and NI were considered and extracted by [12] in their work. As such, their method employed image decomposition based on separable quadrature mirror filters (QMFs) to capture regularities inherent to photographic images [12]. An approach presented by [16] discriminates CGI from NI based on the lack of artifacts due to the use of a digital camera as an acquisition device for NI [16]. Their technique is based on the fact that image acquisition in a digital camera is fundamentally different from the generative algorithms deployed by computer generated imagery. This difference is captured in terms of the properties of the residual image (pattern noise in case of digital camera images) extracted by a wavelet based de-noising filter. An approach proposed by [17] used features that are based on the differences in the acquisition process of images. First they tried to detect the presence of the colour filter array demosaicking from a given image because most consumer cameras use colour filter array which requires the involvement of a demosaicking operation in generating the RGB colour values. The approach by [17] specifically searched for traces of demosaicking and chromatic aberration which were used to differentiate CGI from NI.

Another technique based on the differences in the acquisition process of images was proposed by [18]. The starting point of their research is that the different formation processes, leave distinct intrinsic traces on digital images. In their algorithm, spectral correlations between colour components are exploited efficiently by discrete wavelet transform, block partitioning and normalized cross correlation, and three statistical features are derived to capture the inherent differences between CGI and NI. [6] combined statistical, visual and physical features of digital images to propose features that can differentiate CGI from NI. Their approach amongst other features, extracted the mean and median of the histograms of grayscale image in the spatial and wavelet domain as statistical features. Secondly, the fractal dimensions of grayscale image and wavelet sub-bands were extracted as visual features. And finally, the physical features are calculated from the enhanced photo response non-uniformity noise. Thereafter, a support vector machine (SVM) classifier was used in the classification process.

More recently, the researchers in [19] comparing CGI with NI, extracted and used 9 dimensions of texture features. They argued that NI have higher self-similar and have more delicate and complex texture. The work by [5] extracted textural descriptors from images using binary statistical image features and also used SVM as the classifier. According to them, the textural features are different for CGI and NI as their approach was based on learning of natural image statistic filters and further using that to differentiate the two images.

In this research work, a model is proposed where colour and statistical features are extracted and combined in identifying features that differentiate CGI from NI.

3 Methodology

The proposed model is termed ***Colour Range Histogram*** (CRH). The CRH works by first randomly selecting a pixel $A_{x,y}$ in an image as a point of reference. Next, the

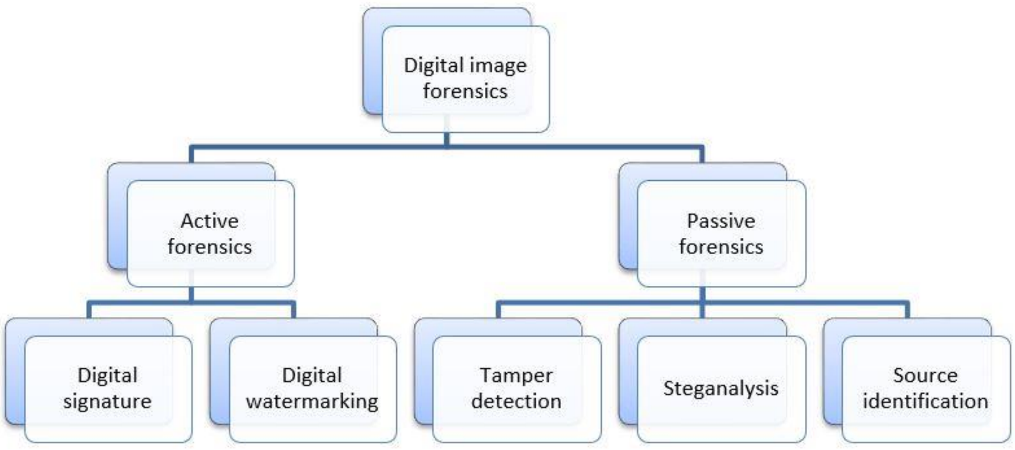


Figure 1: Classification of Digital Image Forensics.

CRH algorithm fetches the RGB colour code for $A_{x,y}$, which is an integer value, due to the programming language used. Then the algorithm checks through all other pixels in the rest of the image to highlight all pixels that have the same RGB colour code as pixel $A_{x,y}$. These pixels were classified as group 1 pixels. The complete steps used in CRH are further elucidated in the pseudo codes in Listing 1. In general, the following steps are proposed:

1. For any image (A), with dimension $(w \times h)$ where h represents the height and w represents the width of the image, select any pair of coordinates (x, y) that represent a pixel position such that $0 \geq x \leq w$ and $0 \geq y \leq h$.
2. Given $A_{x,y}$ fetch $RGB\ (A_{x,y})$ where $A_{x,y}$.represents a pixel in A at the pixel position (x, y)
3. Scan through all other pixels in (A), from $A_{0,0}$. to $A_{w,h}$.
    3.1 Fetch $RGB\ (A_{s,t})$
    3.2 If $RGB(A_{s,t})$ = $RGB(A_{x,y})$, retain its colour
    else
    change the colour of $A_{s,t}$ to white
    3.3 Fetch next pixel
4. Save the resultant image (A_m).

The algorithm above was used to "highlight" group 1 pixels, which are pixels with the exact RGB code as $A_{x,y}$. The algorithm was further extended to highlight seven more groups of pixels (Listing 2). These are pixels in the image that have RGB colour codes within certain ranges from $RGB\ (A_{x,y})$. These groups were:

Group 2:	where $RGB(A_{s,t})$ is within ±10 from $RGB(A_{x,y})$
Group 3:	where $RGB(A_{s,t})$ is within ±20 from $RGB(A_{x,y})$
Group 4:	where $RGB(A_{s,t})$ is within ±30 from $RGB(A_{x,y})$
Group 5:	where $RGB(A_{s,t})$ is within ±40 from $RGB(A_{x,y})$
Group 6:	where $RGB(A_{s,t})$ is within ±50 from $RGB(A_{x,y})$
Group 7:	where $RGB(A_{s,t})$ is within ±60 from $RGB(A_{x,y})$
Group 8:	where $RGB(A_{s,t})$ is within ±70 from $RGB(A_{x,y})$

By 'highlight' we mean that their original colour is retained, while the colour of the rest of the image was changed to white. However, if the randomly selected pixel was white in colour then the colour of the rest of the image was changed to Black as can be seen from figure 2c. This highlight was to enable us have a visual clue of the resultant image.

In addition to saving the image file for a visual presentation of CRH, eight more features which represent the total number of pixels projected in each group was captured. The image dataset used for this work was obtained from the Internet as well as personal picture collections. A total of 1,620 images were obtained which contained 851 CGIs and 769 NIs.

```
    Load imageA = ImageIO.read(new File(path))
      imageA_width = image.getWidth();
      imageA_height = image.getHeight();
      Pick $Pixel_{x,y}$ where $0 \geq x \leq$ imageA_width and $0 \geq y \leq$
imageA_height;
      Pcolour = imageA.getRGB(x,y);
    for (int w = 0; w < imageA_width(); w++)
    for (int h = 0; h < image_height(); h++){
      Pixelcolour = image.getRGB(w,h);
        if (Pcolour = pixelcolour)
                        retain pixel colour
       else
                        change pixel colour to white;
   }
              Save image;
```

Listing 1: Algorithm for exact colour highlight.

```
    Load image = ImageIO.read(new File(path))
         width = image.getWidth();
         height = image.getHeight();
         Pick $Pixel_{x,y}$ where $0 \geq x \leq$ width and $0 \geq y \leq$ height;
      Pcolour = image.getRGB(x,y);
      for(int x = 0; x < image.getWidth(); x++)
    for(int y = 0; y < image.getHeight(); y++){
     current_Pixelcolour = image.getRGB(x,y);
          if current_pixelcolour is within range
                   Project the original colour;
                   else
                   Change colour to white;
          }
    Save image;
    End
```

Listing 2: Algorithm for colour range highlight.

4 Results and discussions

Figure 2(a-d) shows some of the visual outputs of group 1 for both NI (a & b) and CGI (c & d). From figure 2 (a & b), it was observed that in NI, a colour that appears to be the same visually is actually represented by a wide range of RGB codes. This could be largely due to the demosaicking process that NI undergo while being produced or the lighting conditions when the image was captured. Therefore picking a random pixel colour and projecting all pixels with exactly the same colour code yielded a scanty set of pixels visually. However, for CGI, the visual results are considerably different. The CGI visual results show that a higher number of pixels are projected for group 1. This exact colour projection sometimes corresponded with a 'shape' in the CGI as can be seen in figure 2 (c & d). This could be because most CGI are a combination of various shapes, where each shape is "filled" with the same colour and then the colour of some areas "blended".

For the colour range highlight, although eight groups were initially proposed, it was observed that beyond group 4 the number of projected pixels remained almost constant for both CGI and NI. This can be viewed from the projected pixel count result displayed in listing 3 (for a natural image) and listing 4 (for a computer generated image). The listing includes the file chosen, its resolution, the pixel chosen ($A_{x,y}$), the pixel colour $RGB(A_{x,y})$ and finally the various counts of projected pixels by range. Listing 3 showed that 37 pixels were projected for group 1; 98 pixels for group 2, and so on. This result showed that for NI, there is usually a gradual increase in the number of projected pixels from group 1 to group 4. Listing 4 however showed that 3242 pixels were projected for group 1; 3488 pixels for groups 2 and beyond. This showed that computer generated images projected almost a constant number pixels.

```
        File Chosen is C:\...\Natural Images\4Egg.jpg
        Image Width: 1918Image Height: 1077
        Chosen Pixel is: 833,392
        Java RGB code is : -1920995
        the real RGB values are: Alpha: 255, Red: 226, Green: 176, Blue: 29
        For range 0       : ProjectedCount is 37
        For range 10      : ProjectedCount is 98
        For range 20      : ProjectedCount is 246
        For range 30      : ProjectedCount is 267
        For range 40      : ProjectedCount is 267
        For range 50      : ProjectedCount is 267
        For range 60      : ProjectedCount is 267
        For range 70      : ProjectedCount is 267
```

Listing 3: Projected pixel count for a selected natural image.

```
File Chosen is C:\...\CGI\ tamar8.jpg
        Image Width: 564  Image Height: 942
        Chosen Pixel is: 114,194
        Java RGB code is : -13171452
        the real RGB values are: Alpha: 255, Red: 55, Green: 5, Blue: 4
        For range 0       : ProjectedCount is 3242
        For range 10      : ProjectedCount is 3488
        For range 20      : ProjectedCount is 3488
        For range 30      : ProjectedCount is 3488
        For range 40      : ProjectedCount is 3488
        For range 50      : ProjectedCount is 3488
        For range 60      : ProjectedCount is 3488
        For range 70      : ProjectedCount is 3488
```

Listing 4: Projected pixel count for a selected computer generated image.

The projected pixel counts for groups 1 to 4 were saved, processed and analysed. Using equations 1-4, the average percentages, P_1, P_2, P_3, P_4 of projected pixels were calculated for groups 1, 2, 3 and 4 respectively.

$$P_1 = \frac{\sum C_{r0}}{\sum C_{r\pm 30}} \times \frac{100}{1} \qquad \text{Equation (1)}$$

$$P_2 = \frac{\sum C_{r\pm 10}}{\sum C_{r\pm 30}} \times \frac{100}{1} \qquad \text{Equation (2)}$$

$$P_3 = \frac{\sum C_{r\pm 20}}{\sum C_{r\pm 30}} \times \frac{100}{1} \qquad \text{Equation (3)}$$

$$P_4 = \frac{\sum C_{r\pm 30}}{\sum C_{r\pm 30}} \times \frac{100}{1} \qquad \text{Equation (4)}$$

Where:

C_{r0} = count of projected pixels for group 1

$C_{r\pm 10}$ = count of projected pixels for group 2

$C_{r\pm 20}$ = count of projected pixels for group 3

$C_{r\pm 30}$ = count of projected pixels for group 4

These equations were then optimised to give a generalized equation 5

$$P_i = CCG_i \times CCG_j^{-1} \times K \qquad \text{Equation (5)}$$

Where

P_i is the percentage of projected pixel for a group i

$i = 1, 2, 3, 4$; $j = 4$; CCG is count of projected pixels for groups i *or* j and K is a constant.

The summary of the analysed data is presented in table 1. From table 1 it can be observed that the average

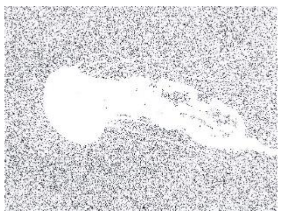
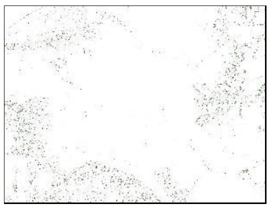
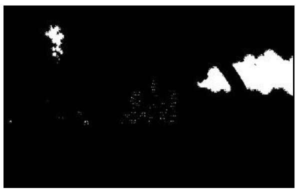
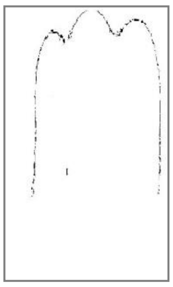
			
a	b	c	d

Figure 2: Results of projected exact colour areas in some images.

percentages of projected pixels for natural images were 31.07%, 82.64%, 90.75% and 95.00% for groups 1, 2, 3 and 4, respectively, while CGI projected an average of

	NI % projected	CGI % projected
P_1	31.07	69.79
P_2	82.64	92.87
P_3	90.75	96.87
P_4	95.00	98.60

Table 1: Average % pixel colour projection.

69.79%, 92.87%, 96.87% and 98.60%, of any colour code for groups 1, 2, 3 and 4, respectively.

This shows that natural images contain a wide range of RGB colour codes for a particular colour that has similar visual colour presentation [20].

For each image, the value of P_1, P_2, P_3, P_4 were further analysed in order to distinguish between NI and CGI. The analysis showed that an image is classified as CGI if:

$$AND(P_2 - P_1 \leq 60;\ P_3 - P_2 \leq 30;\ P_4 - P_3 \leq 15)$$

While an image is classified as NI if:

$$NAND(P_2 - P_1 \leq 25;\ P_3 - P_2 \leq 12;\ P_4 - P_3 \leq 6)$$

Using the above results we achieved the following classification percentages

	CGI	NI
True Positives	81.6%	87.0%
False negatives	18.4%	13.0%

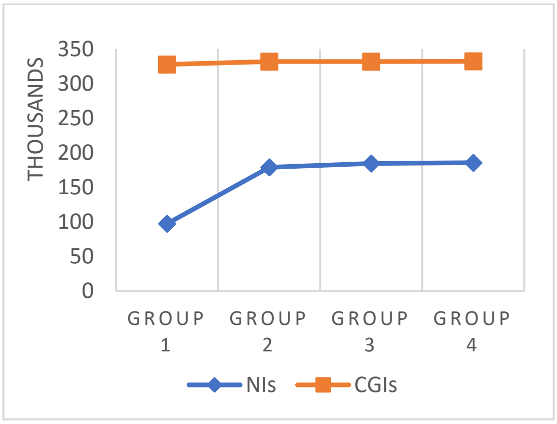


Figure 3: Average number of projected pixels.

Figure 3 shows a graph of the total number of projected pixels for all the computer generated images and natural images in the sample size. This figure shows that irrespective of the random colour chosen, computer generated images projected almost a “constant” number of pixels across the four groups, this can be seen in figure 3 where the red line for computer generated images is almost a horizontal straight line. The pattern of the blue line for the natural images shows a sharp increase in the number pixels emphasizes from group 1 to group 2 and then a gradual increase from group 2 to group 4. The figure also showed that CGI projected a greater percentage of their total pixels than natural images, despite the fact that most natural images had greater number of pixels than the computer generated images. Consequently, the disparity in percentage emphasised can be used to differentiate computer generated images from natural images.

5 Conclusion

In this research work, the RGB colour features of some selected pixels in both natural and computer generated digital images were extracted, grouped and analysed. The analysis revealed that there is a disparity in the percentage selected/emphasized for the two groups of images. Consequently, this disparity in percentage of colours projected, within range 0 to 40 from a point of reference, can be used as a quick method to differentiate computer generated images from natural images.

References

[1] Oracle, "The Java tutorials," 2 Febuary 2012. [Online]. Available: http://www.oracle.com/java-se-7-tutorial-2012-02-28-1536013.html. [Accessed 14 June 2013].

[2] M. K. Johnson, K. Dale, S. Avidan, H. Pfister, W. T. Freeman and W. Matusik, (2011) "CG2Real: Improving the Realism of Computer Generated Images using a Large Collection of Photographs," IEEE Transactions on Visualization and Computer Graphics, vol. XVII, no. 9, pp. 1273 - 1285. https://doi.org/10.1109/tvcg.2010.233

[3] T.-t. Ng, S.-f. Chang, C.-y. Lin and Q. Sun, (2006) "Passive-blind Image Forensics," in In Multimedia Security Technologies for Digital Rights, Elsevier, pp. 383- 412. https://doi.org/10.1016/b978-012369476-8/50017-8

[4] G. K. Birajdar and V. H. Mankar, (2013). "Digital image forgery detection using passive techniques: A survey," Digital Investigation, vol. 10, no. 3, pp. 226-245. https://doi.org/10.1016/j.diin.2013.04.007

[5] G. K. Birajdar and V. H. Mankar, (2017). "Computer Graphic and Photographic Image Classification using Local Image Descriptors," Defence Science Journal, vol. 67, no. 6, pp. 654-663. https://doi.org/10.14429/dsj.67.10079

[6] F. Peng, J. Liu and M. Long, (2012). "Identification of Natural Images and Computer Generated Graphics Based on Hybrid Features," International Journal of Digital Crime and Forensics, vol. IV, no. 1, pp. 1-16. https://doi.org/10.4018/jdcf.2012010101

[7] A. Swaminathan, M. Wu and K. J. R. Liu, (2006). "Component forensics of digital cameras: A non-intrusive approach," in 2006 40th Annual Conference on Information Sciences and Systems. https://doi.org/10.1109/ciss.2006.286646

[8] M. Chandra, S. Pandey and R. Chaudhary, (2010). "Digital watermarking technique for protecting digital images," in 3rd IEEE International

Conference on Computer Science and Information Technology (ICCSIT). https://doi.org/10.1109/iccsit.2010.5565177

[9] A. C. Popescu and H. Farid, (2005). "Exposing digital forgeries in color filter array interpolated images," IEEE Transactions on Signal Processing, vol. 53, pp. 3948-3959. https://doi.org/10.1109/tsp.2005.855406

[10] Wang W., Dong J., Tan T. (2009) A Survey of Passive Image Tampering Detection. In: Ho A.T.S., Shi Y.Q., Kim H.J., Barni M. (eds) Digital Watermarking. IWDW 2009. Lecture Notes in Computer Science, vol 5703. Springer, Berlin, Heidelberg https://doi.org/10.1007/978-3-642-03688-0_27

[11] S. D. Mahalakshmia, K. Vijayalakshmib and S. Priyadharsinia, (2012). "Digital image forgery detection and estimation by exploring basic image manipulations," Digital Investigation, pp. 215-225. https://doi.org/10.1016/j.diin.2011.06.004

[12] S. Lyu and H. Farid, (2005). "How realistic is photorealistic?," IEEE Transactions on Signal Processing, vol. 53, no. 2, pp. 845-850. https://doi.org/10.1109/tsp.2004.839896

[13] J. Lukas, J. Fridrich and M. Goljan, (2005). "Determining digital image origin using sensor imperfections," in SPIE Electronic Imaging, Image and Video Communication and Processing, San Jose, California. https://doi.org/10.1117/12.587105

[14] H. Farid and S. Lyu, (2003). "Higher-order wavelet statistics and their application to digital forensics," in Computer Vision and Pattern Recognition. https://doi.org/10.1109/cvprw.2003.10093

[15] T.-T. Ng, S.-F. Chang, J. Hsu, L. Xie and M.-P. Tsui, (2005). "Physics-motivated features for distinguishing photographic images and computer graphics," in ACM Multimedia, Singapore. https://doi.org/10.1145/1101149.1101192

[16] S. Dehnie, T. Sencar and N. Memon,. (2006). "Digital Image Forensics for Identifying Computer Generated and Digital Camera Images," in IEEE International Conference on image processing. https://doi.org/10.1109/icip.2006.312849

[17] A. E. Dirik, S. Bayram, H. T. Sencar and N. Memon, (2007). "New features to identify computer generated images," in IEEE International Conference on Image Processing 4. https://doi.org/10.1109/icip.2007.4380047

[18] X. Kang, E. Zhang, Y. Chen and Y. Wei, (2011). "Forensic discrimination of computer generated images and photographs using spectral correlations in wavelet domain," Energy Procedia, vol. 13, no. 311, pp. 2174-2182. https://doi.org/10.1016/S1876-6102(14)00454-8

[19] F. Peng, Y. Zhu and M. Long, (2015). "Identification of Natural Images and Computer Generated Graphics using Multi-fractal Differences of PRNU," in ICA3PP 2015: Part II of the 15th International Conference on Algorithms and Architectures for Parallel Processing. https://doi.org/10.1007/978-3-319-27122-4_15

[20] N. C. Woods and C. A. B. Robert, (2017) "A Model for Creating Exact Colour Spectrum for Image Forensic," University of Ibadan Journal of Science and Logics in ICT Research (UIJSLICTR), vol. Volume 1, no. 1, pp. 1-6.

Data Mining Approach to Effort Modeling on Agile Software Projects

Hrvoje Karna, Sven Gotovac and Linda Vicković
University of Split, Poljička cesta 35, 21000, Split, Croatia
E-mail: hrvoje.karna@gmail.com, sven.gotovac@fesb.hr, linda.vickovic@fesb.hr

Keywords: agile scrum, data mining, effort estimation, *k*-nearest neighbor, software engineering, project management

Received: April 26, 2019

Software production is a complex process. Accurate estimation of the effort required to build the product, regardless of its type and applied methodology, is one of the key problems in the field of software engineering. This study presents the approach to effort estimation on agile software project using local data and data mining techniques, in particular k-nearest neighbor clustering algorithm. The applied process is iterative, meaning that in order to build predictive models, sets of data from previously executed project cycles are used. These models are then utilized to generate estimate for the next development cycle. Used data enrichment process, proved to be useful as results of effort prediction indicate decrease in estimation error compared to the estimates produced solely by the estimators. The proposed approach suggests that similar models can be built by other organizations as well, using the local data at hand and this way optimizing the management of the software product development.

Povzetek: V prispevku je predstavljen pristop strojnega rudarjenja za modeliranje agilnih programskih projektov.

1 Introduction

Accurate estimation of work effort required to build the product is a critical activity in software development industry [1] and it is carried out on most projects [2]. Previously a number of approaches have been proposed to reliably estimate the effort, such as theoretical [3], formal [4], analogy-based estimation [5], just to name a few. Despite all, expert estimation [6] remains the most widely used method of effort estimation.

Regardless of its comparative advantages, such as ease of implementation and the validity of the results it produces [7], expert effort estimation can still be improved [8]. Estimation is particularly challenging in large agile projects [9]. One way to achieve this is to use own, locally built, collections of past project data [10], [11]. The emergence of machine learning algorithms and data mining in general, paired with the availability of tools, has led to progress in application of these methods in practice [12].

This paper presents an approach to effort estimation using data mining techniques, particularly *k*-Nearest Neighbor (KNN) clustering algorithm [13], on local collection of telco project data. The approach uses local data [14], extracted from the tracking system implemented on the project. The process itself is iterative, implemented in a way that at first it uses a collection of data from initial project phase in order to build primary predictive model.

Then in the next project phase – an upgrade, this model is being enriched with the data from the recently completed iteration in order to gradually improve its properties, and thus reduce the estimation error.

This research builds upon our previous work [15] now being applied to a large agile project and using different approach to predict effort. Instead of project clustering applied in [15], in this paper KNN is used to cluster work items and for each new instance it finds the nearest neighbors and calculates the model predicted effort.

The proposed approach itself follows on one hand the iterative nature of agile scrum methodology [16] implemented on the project while at the same time fitting it to the cyclicality of the CRISP-DM process [17]. This proved to be efficient way to improve estimation accuracy and therefore can be suggested as a method by which organizations can improve the process of project management.

The remaining part of this paper is organized as follows: Section 2 presents the current state of the research of the areas being discussed in the paper. Section 3 elaborates the design of the study, applied approach and techniques used to model effort estimation. In Section 4 results are presented together with their implication and potential limitations. The concluding section summarizes the findings and gives directions for future work.

2 Related research

Data mining techniques provide a means to analyze and extract patterns from data and through that process produce previously unknown and potentially useful information [18]. They emerged as an interdisciplinary domain with evolution and merging of databases, statistics and machine learning [19].

It can be viewed as a method for discovering knowledge from large sets of data [20]. Data mining consists of a set of techniques applicable for different purposes [21]. Clustering being one among them is particularly useful in prediction [22] and KNN is one of the most widely used algorithms [23].

Research in the field of software development effort estimation is active since the emergence of this industry [24]. During that period this has resulted in the number of approaches intended to estimate the effort required to build the product [25], each with their own advantages and limitations. Up to now, due to its comparative advantages, expert effort estimation remains the most frequently used technique in practice [26]. Paired with modern data analysis techniques it has potential to significantly improve reliability of the estimates [27].

Mining software engineering data raises the interest of researcher for quite some time [28], it also poses specific challenges [29]. It has been applied to different types of data [30], [31] and uses a number of techniques [32]. The application of these techniques is particularly appropriate in software engineering as it is rich in data [33] while, on the other hand, they can be used to optimize the software development process, software itself and support decision making process [34].

Agile development methods emerged from the need to efficiently handle close interaction with the customer, flexibility in requirements definition and the urge to deliver software on time and within the budget [35]. In contrast to sequential, agile development methods propose incremental approach to building of the software product [36]. These practices can also be used to handle the system and team scale issues [37] what is especially important in today’s dynamic business environment.

Agile scrum executes the project in a sequence of iterations called sprints, where each sprint represents a cycle within which development activities occur [38]. During sprint planning, team members determine sprint goal, prioritize and estimate the effort of work items [39].

3 Study design

This empirical study was performed using local data from a complex telco solution development project executed in large international company. Development of the application was based on Java technology and Oracle DB. Data used for the study refers to the tracking system items and descriptive features of the estimators, as these are the entities used to construct the predictive models. The authors implemented these models before [15], [40], so the selection of predictors was based on their relative importance determined in this, our previous [41] and similar studies [2].

The study exclusively used data required to build predictive models for effort estimation and for this it was sufficient that for example, components are identified as Component_1, Component_2, etc. or that estimators are referred to as Estimator_1, Estimator_2, ..., and so on, with matching attributes taking appropriate values.

The average number of estimators per sprint fluctuated around 22, reaching at one point the total of 31. The number of estimation items per Sprint was between 80 and 110, with the total of 1,732 in Phase 1 and 532 in Phase 2. Total actual effort recorded in Phase 1 was 20,814.25 [h] and in Phase 2 sprints 5,344.50 [h].

These numbers indicate that the analyzed project belongs to the class of “large” projects [42]. In the sequence of analyzed sprint data, none of them ended up exactly on the estimated value of effort. Both under and over estimations occurred with relatively same frequency, see Table 1 (sprints 1-19) and Table 3 (sprints 20-24), yet overestimation was more common in early project phase while underestimation was more common in later phase.

Sprint	**Effort [h]**		**Estimation Error**		
	Estimated	Actual	Absolute (Relative)	MMRE	Pred(0.25)
1	1,335.00	1,297.00	+38.00 (+2.93%)	0.652	0.660
2	1,224.00	1,302.00	-78.00 (-5.99%)	0.215	0.738
3	1,294.00	1,223.00	+71.00 (+5.81%)	0.310	0.673
4	1,173.00	1,171.00	+2.00 (+0.17%)	0.359	0.774
5	375.00	358.00	+17.00 (+4.75%)	0.522	0.733
6	1,328.00	1,278.00	+50.00 (+3.91%)	0.378	0.767
7	1,314.00	1,289.00	+25.00 (+1.94%)	0.301	0.663
8	1,262.00	1,239.00	+23.00 (+1.86%)	0.323	0.670
9	1,056.00	1,078.00	-22.00 (-2.04%)	0.254	0.803
10	1,432.50	1,424.25	+8.25 (+0.58%)	0.210	0.779
11	1,120.00	1,146.00	-26.00 (-2.27%)	0.071	0.879
12	1,518.00	1,479.00	+39.00 (+2.64%)	0.383	0.780
13	1,255.00	1,304.00	-49.00 (-3.76%)	0.092	0.893
14	1,089.00	1,081.00	+8.00 (+0.74%)	0.063	0.925
15	991.00	975.00	+16.00 (+1.64%)	0.226	0.861
16	1,182.00	1,149.00	+33.00 (+2.87%)	0.180	0.843
17	970.00	979.00	-9.00 (-0.92%)	0.194	0.813
18	884.00	922.00	-38.00 (-4.12%)	0.128	0.848
19	118.00	120.00	-2.00 (-1.67%)	0.026	0.923

Table 1: Efforts and estimation error values per sprint for the training set.

The analyzed data covers Phase 1 (initial version) and Phase 2 (upgrade) of development project. Each phase was implemented in so called sprints i.e. development cycles as defined by the agile scrum methodology. Phase 1 consists of 19, while Phase 2 covers 5 sprints. Each sprint produces a given set of estimation items i.e. data records. The problem that is being solved was weather it is possible to predict the effort of the upcoming Phase 2 sprints by using the knowledge from those completed. Sprints 1 to 19 (S1-S19) were used as initial data base of items for training and test of predictive model, while sprints S20 to S24 served for validation, see Figure 1.

Upon building of the initial model M1 this model was used to predict effort of sprint S20. In each following iteration the model was enriched by the data from the last sprint, thus the data set (S1-S19+S20) was used to build model M2 and predict effort of S21, data set (S1-S19+S20+S21) was used to build model M3 and predict effort of S22, etc. This process passed through five iterations that could also be presented as follows:

1st iteration: (S1-S19) } M1 → S20
2nd iteration: (S1-S19+S20) } M2 → S21
3rd iteration: (S1-S19+S20+S21) } M3 → S22
4th iteration: (S1-S19+S20+S21+S22) } M4 → S23
5th iteration: (S1-S19+S20+S21+S22+S23) } M5 → S24

Expert estimation heavily relies on the intuition where based on the received input information estimator uses his judgment to come up with the solution [43]. This process can be improved by designing models that support the estimation of effort.

The proposed predictive model targets the agile software development environment. It uses data mining approach that is explained next in more details. This is followed by the description of the entities that represent the sources of data and the fields used as predictors of the effort. Finally, the modeling method, determined by the selected tool itself is described.

3.1 Data mining process

Building of the data mining model considered in this study required the definition of research objective. In this case it was optimization of the software development process through the application of machine learning algorithm in order to provide the way to decrease effort estimation error, thus allowing more efficient management of the project.

The data mining process applied in this study uses de-facto industry standard known as CRISP-DM (CRoss-Industry Standard Process for Data Mining). This is iterative process structured around six phases:

- *Business understanding* – identification of the business problem that has to be solved,
- *Data understanding* – obtaining, exploring and verification of the data that will be used,
- *Data preparation* – retirement of the data before it can be used for modeling,
- *Modeling* – selection of appropriate technique, building and assessment of the model,
- *Evaluation* – evaluation of results and review of the process,
- *Deployment* – use of the model in order to improve the business.

Understanding of the business and data was established prior and during initial prediction iteration: (S1-S19) } M1 → S20. For each next iteration data preparation followed by modeling and evaluation phase was executed. The presented model has academic purpose i.e. evaluation of proposed approach, so currently there is no deployment in real environment. Once the model proves effectiveness, it is possible to recommend its application in practice.

3.2 Entities and data

The study uses following entities and related fields as data sources:

● *Item*: these are the records by which the work is represented and stored in the tracking system implemented on the analyzed project i.e. tickets. Variables used to represent work item entity are: Assignment (representing type of item association to the estimator, taking the form of either "own" or "assigned"), Component (identifying the component

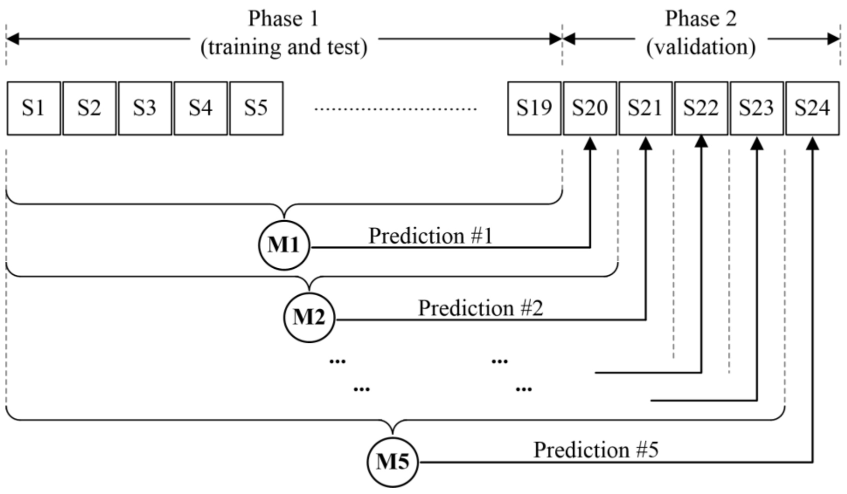


Figure 1: Model building and prediction process used in the study.

within the system that is related to, identified as Component_1, Component_2, …), Area (refers to the area of work with possible values: PM, QM, CM, System, …, Other), Activity (refers to the type of activity with possible values: Management, Quality, Design, Implementation, Test, …, Installation, Documentation, etc.), Type (identifies type of the item according to the applied scrum methodology, being either user story, task, defect, or other) and Priority (or urgency, it indicates the order in which item should be taken into execution in relation to the other items, describe as Prio_1, Prio_2, …, where Prio_1 refers to the highest priority). As it is evident, these are descriptive attributes related to the item at the moment of its creation. Additional fields associated with the item entity used to record the efforts are: Estimated Effort, Remaining Effort and Actual Effort. These were populated at the moment of item creation and later updated as the work progresses until its completion.

- *Estimator*: the estimator is basically the employee engaged on the project, sometimes referred to as a project team member. In the model the estimator is represented with set of variables describing his: Role (representing his primary occupation on the project, with potential values: Project Manager, Solution Architect, Software Engineer, Configuration Manager, etc.), Seniority Level (representing the level of seniority, being either Junior, Mid-Level or Senior), Total Experience (representing the total number of years of work experience), Company Experience (representing number of years of experience within the current company), Number of Projects (representing number of projects employee participated in while working for the current company) and Estimation Competence (representing the level of estimation competence, being either Beginner, Intermediate or Advanced).

The list of fields used as predictors and target, together with associated measurement type is presented in Table 2.

3.3 *k*-Nearest Neighbor algorithm

The model uses *k*-Nearest Neighbor (KNN) algorithm. The nearest neighbor (NN) rule assigns to unclassified incoming observation the class of the nearest sample in the set, the simplest form of KNN when k = 1 [44]. KNN is based on measuring the distance between data to decide the final classification output based on their similarity [45].

KNN is an extension of NN and due to its advantages has been used for solving classification problems in numerous domains, the algorithm procedure can be presented as follows [46]:

$$T = \{(x_i, y_i)\}\,;\; i = 1, \dots, N$$

Let T denote the training set, where $x_i \in \Re^m$ is a training vector in the m-multidimensional feature space, and $y_i$is the corresponding class label. Given a query x', its unknown class y' is assigned in two steps.

First, a set of k similarly labelled target neighbors for the query x' is identified. Denote the set

Entity	Field		
	Name	Measurement	Role
ITEM	Assignment	Flag	Predictor
	Component	Nominal	
	Area	Nominal	
	Activity	Nominal	
	Type	Nominal	
	Priority	Ordinal	
ESTIMATOR	Role	Nominal	
	Seniority Level	Ordinal	
	Total Experience	Continuous	
	Company Experience	Continuous	
	Number of Projects	Continuous	
	Estimation Competence	Ordinal	
	Actual Effort	Continuous	Target

Table 2: Predictors and target of proposed model.

$$T' = \{(x_i^{NN}, y_i^{NN})\}\,;\; i = 1, \dots, k$$

arranged in an increasing order in terms of Euclidian distance $d(x', x_i^{NN})$ between x' and x_i^{NN}

$$d(x', x_i^{NN}) = \sqrt{(x' - x_i^{NN})^T (x' - x_i^{NN})}$$

Secondly, the class label of the query is predicted by the majority voting of its nearest neighbors:

$$y' = \arg \overset{max}{y} \sum_{(x_i^{NN},\; y_i^{NN}) \in T'} \delta(y = y_i^{NN})$$

where y is a class label, y_i^{NN} is the class label for the *i*-th nearest neighbor among its k nearest neighbors. $\delta(y = y_i^{NN})$, the Dirac delta function, takes a value of one if $y = y_i^{NN}$ and zero otherwise.

The quality of *k*-Nearest Neighbor algorithm depends on the choice of k and the distance measure parameter [47]:

- *k*: the selection of k is dependent on the selected data set. There are different recommendations but, instead of having the same number of nearest neighbors, it is good to find the best k automatically [48], the approach used in our study in order to choose the best number of neighbors within the range.
- *Distance*: the distance or dissimilarity measure, between two existing cases x_i and y_i can generally be expressed by Euclidean distance, as presented above. Computed distance is basically the magnitude of the vector obtained by subtracting the training data point from the point to be classified.

Thus, in the space defined by the input fields i.e. predictors, cases positioned near each other are referred to as neighbors. Those dissimilar are more distant from each other. For a new case i.e. target that enters the model, the procedure calculates the predicted value of a continuous measurement type as a mean of k nearest neighbor values [49].

KNN was previously used for estimating effort and provided better results in comparison to other techniques [50], [51]. However, what distinguishes this study is that it is performed on a local set of data on a project driven by agile methodology while applying iterative effort modeling per sprint. This makes it unique according to our knowledge.

3.4 Modeling and evaluation

From the input set 12 variables were used as predictors and single variable (Actual Effort) as a target. The experiment was conducted using IBM SPSS Modeler tool 14.2 [52]. In each iteration for analyzed data sets a stream representing data flow was formed to perform experiment. The modelling element implements the k-Nearest Neighbor algorithm, with k set in range of minimum of 3 and maximum of 5, allowing procedure to choose the best number of neighbors, in order to compute the value of the target variable.

Effort modeling for each validation Phase 2 sprint is performed in the following steps:

1. Predictive model is built using data from previously executed i.e. finished sprints,
2. Existing effort values were removed from input data of the sprint that is estimated in iteration,
3. Sprint data is feed into the prediction stream,
4. Predictive modeling is performed and model estimates are generated,
5. Effort estimates are exported from the stream for subsequent evaluation.

After generation of the model predictions for each Phase 2 iteration, as a part of the evaluation procedure, the comparison of the results of estimations produced by models vs. estimators in relation to the actual values of the total reported effort per sprint was performed. In addition to that criterion, the standard measures of estimation error, MMRE and Pred at level x, are used [53]. They are explained next.

Estimation error is the difference between the estimated effort (EST) and the actual value (ACT):

$$EE = EST - ACT$$

Magnitude of relative error (MRE) is the absolute value of estimation error relative to the actual:

$$MRE = \frac{|EST - ACT|}{ACT}$$

it is the basic metric used to calculate Mean Magnitude of Relative Error (MMRE):

$$MMRE = mean\ (MRE) = \frac{1}{n}\sum_{i=1}^{n}\left|\frac{(EST_i - ACT_i)}{ACT_i}\right|$$

The Pred(x) is a criterion that defines the predictions having a relative error of less than or equal to level x, the set threshold, defined as:

$$Pred(x) = \frac{100}{N}\sum_{i}^{N}\begin{cases}1 & if\ \ MRE_i \le x/100\\ 0 & otherwise\end{cases}$$

The x is typically set to 25 so that it reveals the portion of the estimates that are within the tolerance of 25% from the actuals.

Using these metrics it was possible to conduct a reliable evaluation of the predictive model efficiency.

4 Results and discussion

In this section results of the modeling process are presented and commented. Additionally, the implications and limitations related to the data, model and study are discussed.

4.1 Study results

The results of the effort predictions generated by the models, together with the values of effort estimated by the expert estimators and actuals, for each of the validation sprints are listed in Table 3 and illustrated in Figure 3. To meet the standards used by both industry practitioners and scientific community during evaluation we use a comparison of the estimated and the actual efforts [26] as well as MMRE and Pred(0.25). From the data it can be seen that validation sprints vary in volume, ranging from some 500 [h] up to more than 1,500 [h] of actual effort, making validation set representative.

By reviewing the results of the total values produced by the estimators and predictive models, compared to the actuals of each sprint we can conclude that models provided better estimates in four (S21, S22, S23 and S24) out of five iterations. Given the volume of the iteration S20 the difference in gains that the experts made in relation to the model predictions was practically negligible. Within the last four sprints, in three cases the model's prediction was significantly better than the estimates the experts gave.

Regarding the direction of the estimation error, from the validation set, estimators underestimated effort in three out of five sprints and the same was the outcome of the predictions made by the models. It is interesting that errors from both experts and the predictive models had the same tendency i.e. the models do not show the tendency to either under or overestimate but that results depend exclusively of the properties of the provide data set. It seems that both classify in the similar way, that is, the model in certain way mimics the reasoning process but performs better.

Using this evaluation approach, typical for industry practitioners, and comparing the average estimation error produced by the models it is evident that it was smaller in magnitude as presented in Table 3 and that it had a positive tendency i.e. as the modeling progressed the trend of error correction was better, see Figure 4. This can be attributed to the data mining learning process in which as the quantity of data used to build predictive models was increased from iteration to iteration. As it is evident, this had a positive impact on the accuracy of the

Sprint	Effort [h]			Error[%]		
	Estimated	Actual	Model	Estimators	Model	Correction
20	814.00	866.00	809.00	-6.00%	-6.58%	-0.58%
21	1,688.00	1,549.00	1,647.80	8.97%	6.38%	2.60%
22	1,305.00	1,109.50	1,258.40	17.62%	13.42%	4.20%
23	1,133.00	1,281.00	1,215.60	-11.55%	-5.11%	6.45%
24	497.00	539.00	500.60	-7.79%	-7.12%	0.67%

Table 3: Efforts, Estimation error and correction per sprint for the validation set.

predictions as the tendency of their reliability increased. Therefore, we can assume that a similar trend would have continued if this phase of development consisted of more sprints.

The use of both MMRE and Pred was an option in order to provide more accurate study results as these metrics show different tendencies [54]. MMRE and Pred(0.25) are both measures of relative estimation error in a collection of instances, but quite different. Greater values of MMRE indicate greater magnitudes of error, while higher Pred(0.25) score indicates better estimation efficacy i.e. more predictions within a set tolerance (in this case of 25%) from the actuals.

Comparison of the MMRE and Pred(0.25) values, which are *de facto* standard measures used by the scientific community in the field, generated by the estimators and models for the validation set is provided in Table 5. These results clearly indicate that the model generated estimates produced an overall smaller estimation error. Here again we notice a practically equal score in S20 and improvement in S21, S22, S23 and S24, see Figure 5.

Another observation that can be made is that predictive model, based on *k*-Nearest Neighbor (KNN) algorithm, generally outperformed the expert estimators in their ability to estimate. This indicates that selection of learners used in the model was properly carried out and that the overall modeling process itself was effective.

The results of the performed evaluation confirm the applicability of the proposed approach and suggest that similar models could be built using data mining techniques and local data at hand, that way optimizing the estimation process and management of the agile software projects.

4.2 Implications

Used for the purpose of software development effort estimation data mining methods do not only solve that problem but provide a way for better understanding of the context in which estimation occurs and factors that affect it. This way an additional insight to the problem was achieved.

The study once again confirmed the possibility of application of machine learning algorithms to solve the identified problem, this time on a somewhat different type of project. It encourages the enhanced effort estimation process through the synergy of expert estimation and estimation supported by the use of modern methods of prediction.

4.3 Limitations

Potential limitation of this study is the fact that it was performed using the data from a single large agile project. In regards to that in future it would be desirable to conduct similar experiments using the data sets from other projects and environments.

In order to produce more general models future research could as well include projects driven by other

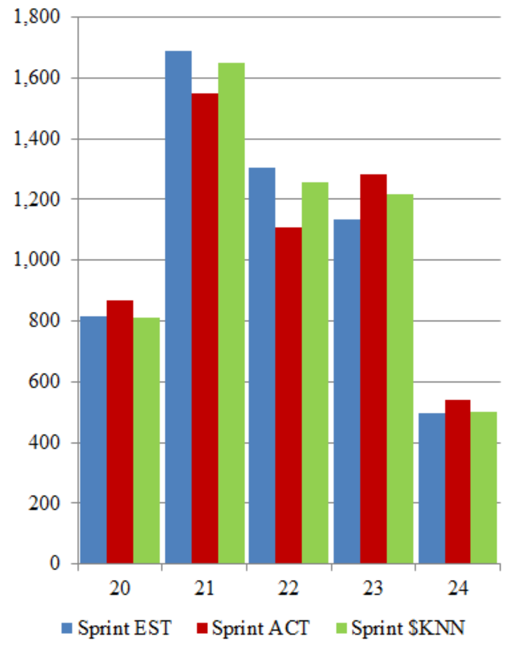


Figure 3: Values of estimated, actual and predicted effort for the validation set.

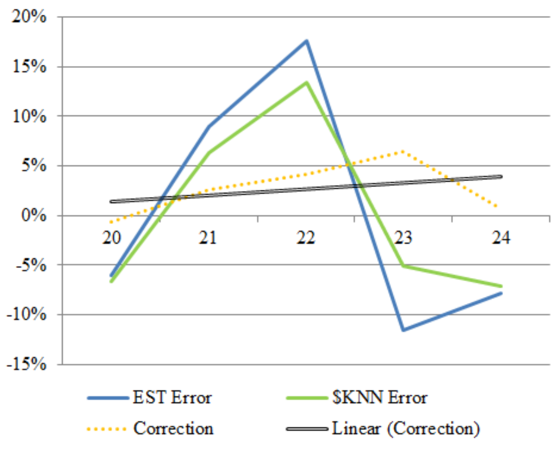


Figure 4: Estimators error, model error and correction in % for validation set with correction trend.

Sprint	Error and Correction					
	Estimators		Model		Correction	
	MMRE	Pred(0.25)	MMRE	Pred(0.25)	MMRE	Pred(0.25)
20	0.086	0.890	0.093	0.878	↓	↓
21	0.501	0.718	0.397	0.835	↑	↑
22	0.700	0.537	0.546	0.734	↑	↑
23	0.183	0.823	0.146	0.872	↑	↑
24	0.106	0.857	0.087	0.915	↑	↑

Table 5: MMRE and Pred(0.25) error and correction indicator per sprint for the validation set.

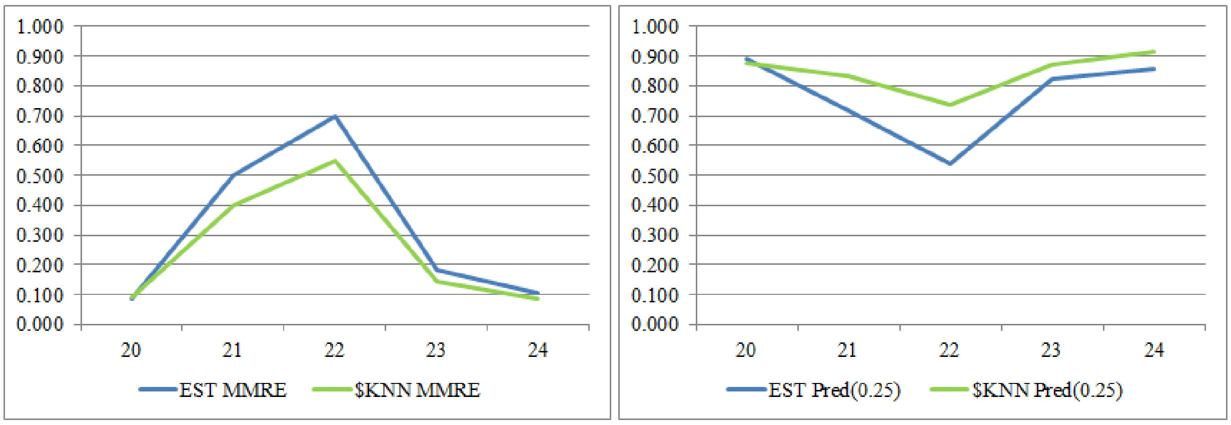


Figure 5: MMRE and Pred(0.25) values generated by estimators and model per sprint for the validation set.

methodologies and those implemented using other technologies. Certainly, relatively straightforward approach used to construct the models, described here, encourages its replication.

5 Conclusions and future work

The paper presented the approach to the effort estimation on agile software project using data mining techniques on the local set of telco project data. It positions the research within the field of software engineering in addition to affirming the actuality of the topic being presented.

Recent research suggests the intensive application of proposed methods to model the effort estimation though, up to our best knowledge, there has been no similar experiment conducted using data set constructed from the mentioned sources and within such environment. This is, among other, the contribution of this work.

The approach proved its validity, providing corrections of the estimated effort generated by the models in most cases in comparison to the experts, and thus can be suggested for use in this or similar forms.

Future work is aimed towards extending the presented model by including data from other entities within the studied environment. Additionally, if possible it would be valuable to include data from other projects of different size and technological basis.

All stated can contribute to the reliability and performance of the predictive models being built and in case of their application in practice support the development process through more optimized project management.

References

[1] M. M. Kirmani. Software Effort Estimation in Early Stages of Software Development A Review, International Journal of Advanced Research in Computer Science (IJARCS), 8(5):1155-1159, 2017. https://doi.org/10.26483/ijarcs.v8i5.3662

[2] T. Haapio and T. Menzies. Exploring the effort of general software project activities with data mining, International Journal of Software Engineering and Knowledge Engineering, 21(5):725-753, 2011. https://doi.org/10.1142/s0218194011005438

[3] L. H. Putnam. A general empirical solution to the macro software sizing and estimating problem, IEEE Transactions on Software Engineering, 4(4):345-361, 1978. https://doi.org/10.1109/tse.1978.231521

[4] A. J. Albrecht and J. E. Gaffney. Software function, source lines of code, and development effort prediction: a software science validation, IEEE Transactions on Software Engineering, 9(6):639-648, 1983. https://doi.org/10.1109/tse.1983.235271

[5] M. Shepperd et al. Effort estimation using analogy, in Proc. of the 18th International Conference on Software Engineering (ICSE), Berlin, Germany, 1996. https://doi.org/10.1109/icse.1996.493413

[6] M. Jørgensen. A review of studies on expert estimation of software development effort, Journal of Systems and Software, 70(1-2):37–60, 2004. https://doi.org/10.1016/s0164-1212(02)00156-5

[7] M. Jørgensen. Practical Guidelines for Expert-Judgment-Based Software Effort Estimation, IEEE Software, 22(3):57–63, 2005. https://doi.org/10.1109/ms.2005.73

[8] B. Tanveer et al. Effort estimation in agile software development: Case study and improvement framework, Journal of Software: Evolution and Practice, Special Issue, 29(11):e1862, 2017. https://doi.org/10.1002/smr.1862

[9] M. Usman et al. Effort Estimation in Large-Scale Software Development: An Industrial Case Study, Information and Software Technology, 99:21-40, 2018. https://doi.org/10.1016/j.infsof.2018.02.009

[10] E. Mendes. Improving Software Effort Estimation Using an Expert-Centered Approach, in Proc. of the 4th international conference on Human-Centered Software Engineering (HCSE '12), Springer-Verlag, Berlin, Heidelberg, 18-33, 2012. https://doi.org/10.1007/978-3-642-34347-6_2

[11] M. Jørgensen. What We Do and Don't Know about Software Development Effort Estimation, IEEE Software, 31(2):37-40, 2014. https://doi.org/10.1109/ms.2014.49

[12] M. Kim et al. Data scientists in software teams: state of the art and challenges, IEEE Transactions on Software Engineering, 44(11):1024-1038, 2017. https://doi.org/10.1109/tse.2017.2754374

[13] M. Bicego and Marco Loog. Weighted K-Nearest Neighbor Revisited, in Proc. of the 23rd International Conference on Pattern Recognition (ICPR '16), Cancún, México, 2016. https://doi.org/10.1109/icpr.2016.7899872

[14] L. Radlinski and W. Hoffmann. On Predicting Software Development Effort Using Machine Learning Techniques and Local Data, International Journal of Software Engineering and Computing (IJSEA), 2(2):123-136, 2010.

[15] H. Karna et al. Application of Data Mining Methods for Effort Estimation of Software Projects, Software: Practice and Experience, 49(2):171-191, 2018. https://doi.org/10.1002/spe.2651

[16] J. Sutherland and K. Schwaber. The scrum guide, Available: https://www.scrumguides.org/

[17] J. Ponce. Data Mining and Knowledge Discovery in Real Life Applications (2nd ed.), Springer, New York, 2009. https://doi.org/10.5772/62143.

[18] Maninderjit Kaur, Sushil Kumar Garg. Survey on Clustering Techniques in Data Mining for Software Engineering, International Journal of Advanced and Innovative Research, 3(4), 2014.

[19] U. Fayyad et al. From Data Mining to Knowledge Discovery in Databases, AI Magazine, 17(3):37-54, 1996. https://doi.org/10.1609/aimag.v17i3.1230

[20] T. Menzies. Practical machine learning for software engineering and knowledge engineering, Handbook of Software Engineering and Knowledge Engineering, 1:837-862, 2001. https://doi.org/10.1142/9789812389718_0035

[21] M. Halkidi et al. Data Mining in Software Engineering, Intelligent Data Analysis Journal (IDA '11), 15(3):413-441, 2011. https://doi.org/10.3233/ida-2010-0475

[22] C. Morbitzer et al. Application of Data Mining Techniques for Building Simulation Performance Prediction Analysis, in Proc. of the 8th International IBPSA Conference, Eindhoven, Netherlands, 2003.

[23] P. Li et al. The Distance-Weighted K-nearest Centroid Neighbor Classification, Journal of Information Hiding and Multimedia Signal Processing, 8(3):611-622, 2017.

[24] B. W. Boehm. Software Engineering Economics, Englewood Cliffs, Prentice Hall, NJ, USA, 1981.

[25] A. K. Bardsiri and S. M. Hashemi. Software Effort Estimation: A Survey of Well-known Approaches, International Journal of Computer Science Engineering (IJCSE), 3(1), 2014.

[26] M. Usman et al. An Effort Estimation Taxonomy for Agile Software Development, International Journal of Software Engineering and Knowledge Engineering (IJSEKE), 27(4):641–674, 2017. https://doi.org/10.1142/s0218194017500243

[27] K. Dejaeger et al. Data Mining Techniques for Software Effort Estimation: A Comparative Study, IEEE Transactions on Software Engineering, 38(2):375-397, 2012. https://doi.org/10.1109/tse.2011.55

[28] A. E. Hassan and Tao Xie. Software intelligence: the future of mining software engineering data, in Proc. of the Workshop on Future of Software Engineering Research (FoSER '10), Santa Fe, NM, USA, 2010. https://doi.org/10.1145/1882362.1882397

[29] T. Xie et al. Data Mining for Software Engineering, IEEE Computer, 42(8):55-62, 2009. https://doi.org/10.1109/mc.2009.256

[30] G. Robles et al. Estimating Development Effort in Free/Open Source Software Projects by Mining Software Repositories, in Proc. of the 11th Working Conference on Mining Software Repositories (MSR 2014), ACM Press New York, NY, USA, 222-231, 2014. https://doi.org/10.1145/2597073.2597107

[31] K. Molokken and M. Jørgensen. Expert Estimation of Web-Development Projects: Are Software Professionals in Technical Roles More Optimistic Than Those in Non-Technical Roles?, Empirical Software Engineering, 10(1):7-29, 2005. https://doi.org/10.1023/b:emse.0000048321.46871.2e

[32] P. K. Suri and P Ranjan. Comparative analysis of software effort estimation techniques, International Journal of Computer Applications (IJCA), 48(21):12-19, 2012. https://doi.org/10.5120/7479-0540

[33] Q. Taylor et al. Applications of data mining in software engineering, International Journal of Data Analysis Techniques and Strategies, 2(3):243-257, 2010. https://doi.org/10.1504/ijdats.2010.034058

[34] L. L. Minku et al. Data mining for software engineering and humans in the loop, Progress in

Artificial Intelligence (PRAI), 5(4):307-314, 2016. https://doi.org/10.1007/s13748-016-0092-2

[35] R. Marques et al. Assessing Agile Software Development Processes with Process Mining: A Case Study, in Proc. of the 20th IEEE Conference on Business Informatics (CBI), 109-118, Vienna, Austria, 2018. https://doi.org/10.1109/cbi.2018.00021

[36] P. Abrahamsson et al. Agile software development methods: Review and analysis, 2002. Available: www.vtt.fi/inf/pdf/publications/2002/P478.pdf

[37] T. Dingsøyr et al. Exploring software development at the very large-scale: a revelatory case study and research agenda for agile method adaptation, Empirical Software Engineering, 23(1):490–520, 2018. https://doi.org/10.1007/s10664-017-9524-2

[38] M. Choetkiertikul et al. A deep learning model for estimating story points, IEEE Transactions on Software Engineering, 2018. https://doi.org/10.1109/tse.2018.2792473

[39] K. S. Rubin. Essential Scrum: a practical guide to the most popular agile process (1st Edition), Addison-Wesley, Upper Saddle River, NJ, USA, 2013.

[40] H. Karna and S. Gotovac. Estimators Characteristics and Effort Estimation of Software Projects, in Proc. of the 9th International Joint Conference on Software Technologies (ICSOFT-EA 2014), Vienna, Austria, 2014. https://doi.org/10.5220/0005002600260035

[41] H. Karna and S. Gotovac. Modeling Expert Effort Estimation of Software Projects, in Proc. of 22nd International Conference on Software, Telecommunications and Computer Networks (SoftCOM 2014), Split, Croatia, 2014. https://doi.org/10.1109/softcom.2014.7039106

[42] J. Aguilar et al. The Size of Software Projects Developed by Mexican Companies, in Proc. of the International Conference on Software Engineering Research and Practice (SERP'14), 2014. Available: https://arxiv.org/abs/1408.1068

[43] M. Jørgensen et al. Human judgement in effort estimation of software projects, Presented at Beg, Borrow, or Steal Workshop, in Proc. of the International Conference on Software Engineering, Limerick, Ireland, 2000.

[44] T. M. Cover and P. E. Hart. Nearest neighbor pattern classification, IEEE Transactions on Information Theory, 13(1):21–27, 1967. https://doi.org/10.1109/tit.1967.1053964

[45] L.-Y. Hu et al. The distance function effect on *k*-nearest neighbor classification for medical datasets, SpringerPlus, 5(1), 2016. https://doi.org/10.1186/s40064-016-2941-7

[46] J. Gou et al. A New Distance-weighted *k*-nearest Neighbor Classifier, Journal of Information & Computational Science, 9(6):1429–1436, 2012.

[47] J. Huang et al. Cross-validation based K nearest neighbor imputation for software quality datasets: An empirical study, The Journal of Systems and Software, 132:226–252, 2017. https://doi.org:10.1016/j.jss.2017.07.012

[48] E. Kocaguneli et al. Exploiting the essential assumptions of analogy-based effort estimation, IEEE Transactions on Software Engineering, 38(2):425–439, 2012. https://doi.org/10.1109/tse.2011.27

[49] IBM SPSS Modeler: KNN Node, Available: https://www.ibm.com/support/knowledgecenter/en/SS3RA7_15.0.0/com.ibm.spss.modeler.help/knn_node_general.htm

[50] R. Olu-Ajayi. An Investigation into the Suitability of *k*-Nearest Neighbor (k-NN) for Software Effort Estimation, International Journal of Advanced Computer Science and Applications (IJACSA), 8(6), 2017. https://doi.org/10.14569/ijacsa.2017.080628

[51] P. Le and V. Nguyen. A *k*-Nearest Neighbors approach for COCOMO calibration, in Proc. of the 4th NAFOSTED Conference on Information and Computer Science, Hanoi, Vietnam, 219-224, 2017. https://doi.org/10.1109/nafosted.2017.8108067

[52] IBM SPSS Modeler 14.2, Available: http://www-01.ibm.com/support/docview.wss?uid=swg27022140

[53] B. Kitchenham et al. Assessing prediction systems, University of Otago Information Science Discussion Paper No. 99/14, 1999. Available: http://hdl.handle.net/10523/1015

[54] C. Tofallis. A better measure of relative prediction accuracy for model selection and model estimation, Journal of the Operational Research Society, 66(3):1352–1362, 2015. https://doi.org/10.1057/jors.2014.103

Multi-Objective Artificial Bee Colony Algorithms and Chaotic-TOPSIS Method for Solving Flowshop Scheduling Problem and Decision Making

Monalisa Panda
Department of Computer Science and Information Technology, Siksha 'O' Anusandhan Deemed to be University
Bhubaneswar, 751030, Odisha, India
E-mail: monalisapanda.iter@gmail.com

Satchidananda Dehuri
Department of Information and Communication Technology, Fakir Mohan University
Vyasa Vihar, Balasore, 756019, Odisha, India
E-mail: satchi.lapa@gmail.com

Alok Kumar Jagadev
School of Computer Engineering, KIIT Deemed to be University
Bhubaneswar, 751024, Odisha, India
E-mail: alok.jagadev@gmail.com

Keywords: flowshop scheduling, multi-objective optimization, local search algorithms, artificial bee colony algorithm, multi-criteria decision making, chaotic-TOPSIS.

Received: December 13, 2018

Retrieval of optimal solution(s) for a Permutation Flow-Shop Scheduling Problem (PFSSP) within a reasonable computational timeframe has been a challenge till yet. The problem includes optimization of various criteria like makespan, total flowtime, earliness, tardiness, etc for obtaining a set of Pareto solutions in the process of Multi-Objective Optimization (MOO). This paper remodels a Discrete Artificial Bee Colony Algorithm (DABC) from a single objective optimization method to a multi-objective optimization one to solve the PFSSP executed and explored through the alternative and combined use of two local search algorithms named as: Iterated Greedy Search Algorithm (IGRS) and Iterated Local Search Algorithm (ILS). The algorithm has been classified into three different scenarios raised with the analysis of time complexity measure of applied local search methods prioritized through the insertion and swap operation of neighborhood structures that intensifies the local optima in the search space. The results of the DABC algorithm are summarized with respect to Total Completion Time (TCT), Mean Weighted Tardiness (MWT), and Mean Weighted Earliness (MWE). Based on the time complexity measure of the obtained results a Multi-Objective Artificial Bee Colony Algorithm (MOABC) has been proposed by adopting the simplest local search method of all in order to reflect the enhanced version of previously remodeled DABC algorithm. Finally, we propose a Chaotic based Technique for Order of Preference by Similarity to Ideal Solution (Chaotic-TOPSIS) using a suitable chaotic map for criteria adaptation in order to enhance the decision accuracy in the multi-Criteria Decision Making (MCDM) domain.

Povzetek: Članek se ukvarja z NP problemom večkriterijske optimizacije izdelave urnika z imenom Permutation Flow-Shop Scheduling Problem (PFSSP). Uvede Multi-Objective Artificial Bee Colony Algorithm (MOABC), tj. več-kriterijski algoritem z umetno čebeljo kolonijo in pokaže izboljšane rezultate.

1 Introduction and related work

The flowshop scheduling problem (FSSP) is a combinatorial optimization problem, inheriting the ideas from Barkers sequencing problem [1] that is based on ordering of jobs to determine a schedule. However, the problem is NP-hard and introduced by Johnson in 1954 [2]. It has a wide application in logistic, industrial, and many other fields. It aims to find the minimal total flow time (TFT) or total completion time (TCT) execution. The permutation flowshop scheduling problem (PFSSP) is a particular case of FSSP, consisting of a set of n jobs which should be processed in the same order as to the available m machines. The goal is to find the best permutation of jobs that would result best minimal TCT execution of all the processes subject to the constraints that each job is independent, and available for processing at time zero. From time zero onwards, each machine is continuously available and is able to process one operation at a time. Each job can be manufactured at a specific moment on a single machine. When a machine is not available, automatically the jobs remaining are queued to a waiting state. An ongoing job, in a machine is not interrupted till completion.

During the last decades, the research attention for combinatorial optimization has turned to hybrid systems. It is observed that combination of different features from various optimization heuristics results in more robust and unique combinatorial optimization tools. Since the pioneering work of Johnson [2], a number of heuristics have been approached for solving FSSP. These proposed heuristics can be specified either as constructive or improvement. Most constructive heuristics [3-7] are the extended version of the Johnson's algorithm [2], based on two or three-machine flowshop problems. In his work Palmer [3] developed a slope order index for sequencing the jobs with some allotted machines and processing times. A little variation to Palmer's algorithm was proposed by Gupta [4] in order to estimate the same slope index. Also a lot many variants of branch and bound algorithms were developed subsequently [8-11] in this regard. Ignall and Scharge [10] applied the branch and bound scheme for the first time, based on two lower bounds in the two-machine FSSP. Bansal [8] extended the proposed idea to an *m*-machine case.

Due to the essence of optimizing multiple objectives in PFSSP, it is also extended to the multi-objective domain with many challenging approaches (non-heuristic and meta-heuristic). Selen and Hotts [12] solved a multi-objective flowshop scheduling problem (MOPFSSP) with m-machines by formulating a mixed-integer goal programming model with two objectives that is makespan and mean flowtime. Wilson [13] proposed an alternative model for it, by considering a fewer number of variables but at the same time he has added large number of constraints to it. Both the models have included same number of integer variables. Daniels and Chambers [14] proposed a branch and bound approach with two objectives (makespan and maximum tardiness) where they computed the Pareto solution for a 2-machine flowshop scheduling problem. Rajendran [15] also presented a similar procedure along with two heuristic approaches for the 2-machine flowshop scheduling problem with two objectives: minimization of TFT subject to optimal makespan. Similarly two different methodologies (one is based on a Branch and Bound (B&B) technique of exact algorithms and other one is based on Palmer approach of heuristic algorithms) are used [16] to find the optimum solution for minimization of bi-criterion (makespan, weighted mean flowtime) objective function of three machines FSSP with transportation times and weight of the jobs. Recently a production scheduling problem in hybrid shops has been solved by Mousavi et al.[17], by assuming some realistic assumptions.

Like the non-heuristics, many meta-heuristic methods like trajectory based and population based methods have also been proposed to solve MOPFSSPs. Chakravarthy and Rajendran [18] proposed a simulated annealing (SA) algorithm for resolving the *m*-machine FSSP to minimize makespan and maximum tardiness. Similarly many SA algorithms [19-21] were proposed to optimize various objectives like makespan, TFT, and total tardiness. Another SA algorithm was approached by Loukil et al. [22] based on *m*-machine case. The algorithm assumed objective pairs out of a number of objectives such as: average weighted completion time, makespan, average weighted tardiness, maximum earliness, maximum tardiness, and the number of tardy jobs. A novel multi-objective memetic search algorithm (MMSA) [23] is proposed to solve the MOPFSSP with makespan and total flowtime. The performance of the algorithm is validated and compared with the four state-of-the-art algorithms on a number of benchmark problem and provides better solutions than these compared algorithms. Another novel fuzzy multi-objective local search-based decomposition algorithm has been approached for solving a fuzzy-MOPFSSP for two fuzzy objectives, that is, the fuzzy makespan and the fuzzy total flow time. An extensive computational study on Taillard benchmarks has been conducted to compare the proposed algorithm with the fuzzy NSGAII and the results demonstrate the effectiveness of the proposed algorithm [24].

Among meta-heuristics, swarm intelligence has created a class of its own, which models the collective behavior of self-organized models and applies these models to solve many complex problems. Earlier works have adopted ant colony optimization (ACO) and particle swarm optimization (PSO) algorithms to simulate the swarm behavior of ant colonies and flocks of birds, respectively. There are a few researches which implements the PSO and ACO for solving the MOPFSSP [25-29] subject to makespan, TFT and completion time variance**.** Recently, a lot many algorithms have been proposed by modeling the intelligent behaviors of real bee swarms in this regard. The emerging research with artificial bee colony algorithms (ABC) demonstrates that these algorithms outperform and is equally competitive as compared to other population-based algorithms with the advantage of employing fewer control parameters [30-35]. Sharma et al. [36] provided a state art survey of ABC algorithm and its performance analysis with different size of population. Singh [37] has explained how one can solve different optimization problems using ABC algorithm. Recently, Amlan et al.[38] applied a Regional Flood Frequency Analysis (RFFA) to 33 stream gauging stations in the Eastern Black Sea Basin, Turkey. Tereshko [39] proposed a DABC algorithm for the FSSP with intermediate buffers (IBFSP) in order to minimize the maximum completion time. The DABC algorithm uses the effectiveness of the insertion and swap operators to produce neighbourhood solutions at the employed bee phase. From many such articles [40-42] it is clearly understood that, swarm intelligence provides a better algorithmic framework inspired by the intelligent behaviour of the animals, birds and social insects.

The earlier work of PFSSP solved using DABC algorithm, mainly focuses on optimization of TCT criterion. As the DABC algorithm uses many strategies to find the nearest solutions in the search space, no detailed work has been done that counts the time complexity of the algorithm. To deal with this, we have remodelled the DABC algorithm of Tasgetiren et al.[43] for three different cases by the application of some effective strategies. The proposed algorithm is inherited with the hybridization of *swap/insertion* operations and *construction-destruction* procedures for the neighbourhood structures known as *iterated local search* (ILS) and *iterated greedy search algorithm* (IGRS) respectively. Through an experimental analysis, the proposed algorithmic cases are evaluated for best CPU time utilization with respect to three objectives such as: TCT, weighted mean tardiness (WMT), and weighted mean earliness (WME). Again in the same scenario we have tested the results of canonical ABC against DABC algorithm. Genuinely due to multiple objectives, here ABC has been turned to multi-objective ABC (MOABC) with necessary improvements to solve the MOPFSSP.

While working with multiple-objectives it is almost impossible to get a single compromising solution. The situation leads to a multi-criteria decision making (MCDM) scenario. MCDM is the most powerful branch of decision making: generally handles multiple objective functions together and includes a lot many approaches that have been applied to different problem domains to choose the best alternative. But major parameters like criterion weight in these methods are founded on randomness of data. Mareschal [44] has claimed that proper weight assignment to each criterion will lead to a better and more appropriate decision making framework for both qualitative and quantitative data. However, the weight assignment procedure (specifically to qualitative criteria) is completely dependent upon the decision maker's preference and varies remarkably from one decision maker to other. This paper proposes TOPSIS using chaotic maps for generating random numbers during criteria adaptation to improve the decision accuracy. The chaotic number generators emerges a random number each time when needed by the decision maker to define the criterion weight. To maintain the criterion preference, we have sorted the random numbers and assigned them accordingly.

The remaining parts of the paper are assembled as follows. Section 2 presents the problem formulation and assumptions. The canonical ABC and DABC algorithm has been illustrated in Sections 3 and 4 and Section 4 also represents the details of the ILS algorithm and IGRS algorithm applied in MOPFSSP. Section 5 encloses the multi-objective ABC for PFSSP. Section 6 contains the computational results for both algorithms with two different synthetic datasets. Decision making using chaotic-TOPSIS is illustrated in section 7. Section 8 concludes the article with future directions.

2 Problem description and assumptions

A PFSSP is consisting of n jobs ($\pi_1, \pi_2, \pi_3 .... \pi_n$), each having m number of tasks, that have to be processed in separate machines. A schedule for the jobs is the assignment of tasks to time intervals on the available machines. Task T_{ji} must be assigned to machine j where the task belongs to job i, additionally for any job i, the processing of task T_{ji} cannot be started till T_{ji-1} has been completed.

Where,

$i \in (1, n)$ and $j \in (1, m)$.

O_{jk} = processing time of job j on machine k.

Assumptions

(i) Jobs consist of a pre-ordered sequence of operations.

(ii) At a time only one job can be processed on one machine.

(iii) The job orderings are same for all machines.

(iv) Timeslot of different job operations is predetermined.

(v) Once a job starts being processed on the first machine, cannot be interrupted in between either on or between machines.

(vi) Release time of all jobs is zero.

As per above stated assumptions, a dummy PFSSP; having '3' jobs, each with '3' operations having some random processing time can be executed in '3' different machines as follows:

With regard to the above context: $F(\pi_j)$, the flowtime of job π_j is same as the completion time $C(\pi_j, m)$ on the machine m. So the total completion time ($TCT(\pi)$) of all jobs is equal to maximum of flow time or completion time of all jobs and is calculated as:

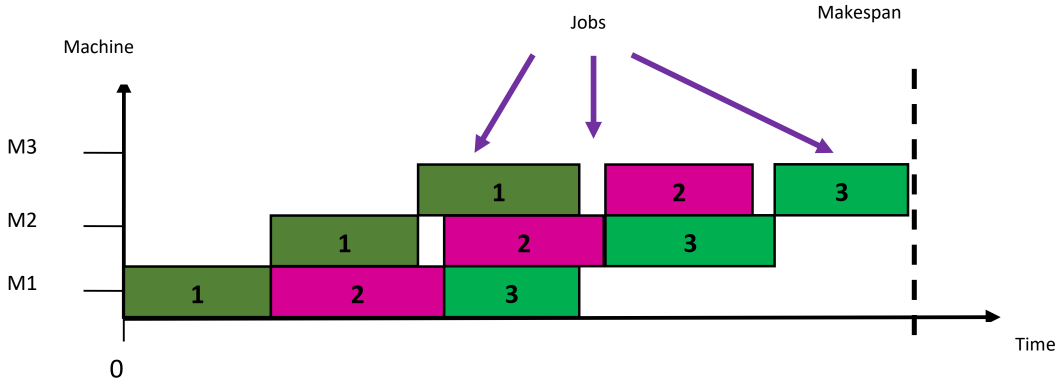


Figure 1: A dummy PFSSP.

$$TCT(\pi) = \max \sum_{j=1}^{n} F(\pi_j) = \max \sum_{j=1}^{n} C(\pi_j, m) \qquad (1)$$

Similarly let D_j, be the due date and C_j the completion time of job *j*, for *jϵn.* The jobs earliness and tardiness can be computed by, $E_j=max\ \{D_j-C_j,\ 0\}$ and $T_j=max\ \{C_j-D_j,\ 0\}$ respectively. Hence the weighted mean tardiness and weighted mean earliness of different job sequence can be calculated as:

$$WME^e = \frac{\sum_{j=1}^{n} \max[D_j - C_j;0]e_j}{\sum_{j=1}^{n} e_j} \qquad (2)$$

$$WMT^R = \frac{\sum_{j=1}^{n} \max[C_j - D_j;0]r_j}{\sum_{j=1}^{n} r_j} \qquad (3)$$

where,

n =number of jobs

j =job index

D_j =due date of job *j.*

C_j = completion time of job *j.*

A_j =arrival time of job *j* in the shop.

e_j = earliness cost per unit time for job *j*

r_j =tardiness of job *j* penalty per unit time.

3 Canonical ABC algorithm

ABC, a member of swarm intelligence is a meta-heuristic algorithm based on the intelligent behavior of honey bees, introduced by Karaboga [30, 34-35, 45]. Due to its simplicity and good performance reported in various fields while optimizing both single and multi-objective problem, we motivated to extend its usage in PFSSP. It is inspired by the nature that is by the foraging behavior of real honey bees, their self-organization capability, and specially division of labor features. The canonical ABC algorithm has some essential components like food sources, nectar-amount in each source, and three kinds of foraging honey bees (employed bee, onlooker bee, and scout bee). Here every food source signifies a candidate solution in the search space and the fitness of these solutions is equivalent to the nectar-amount of those food sources. Employed bees go on searching random food positions; they also share the collected information about food sources among the onlooker bees through the waggle dance. Onlooker bees select the better sources (better solutions) with high nectar amount (high fitness value), based on the information (fitness value) from the employed bees. Scout bees are those employed bees which could not found remarkable food sources. The pseudo-code of canonical ABC is given below.

```
Initialize population (P)
Fitness evaluation (fi)
{
    While (cycle<=maximum number of cycle)
    {
    Employed bee phase
    {
    Produce neighborhoods
    Fitness evaluation selection (fi)
    Probability calculation ($p_i$)
    }
    Onlooker bee phase
    {
    Select a solution based on probability $p_i$
    Produce new solution
    Fitness evaluation
    Greedy selection procedure
    }
    Scout bee phase
    {
    Replace the abandoned one
    }
Memorize the best
cycle$^{++}$
}
```

4 Modified discrete artificial bee colony algorithm

Though ABC algorithm is a proved continuous optimizer for various combinatorial optimization problems, later has also shown its efficiency towards discrete version of it. Here, we use a modified version of the above ABC algorithm to handle discrete decision variables. We have extended the single objective problem of Tasgetiren et al. [43] to a multi-objective one and the detailed of modified DABC has been discussed below:

Initialization:

The population is initialized with a random set of solutions, each consisting with a random permutation of jobs.

$$\pi = (\pi_1, \pi_2, \pi_3 \ldots\ldots \pi_n) \qquad (4)$$

Employed bee phase:

According to the basic ABC algorithm, the employed bees generate their neighborhood nectar sources. Here for obtaining the nearer food sources, we will take the advantage of the adopted strategies from IG_RS algorithm and ILS [43]. From IG_RS algorithm we have borrowed the concept of *construction and destruction* procedure and the two common operators named *insert* and *swap* are being inherited from ILS. Each one of these is used for determining the neighboring solutions in the search space. In order to evaluate their performances, we will adopt three different cases with the alternative and combined use of these operators (*insert* and *swap*) and procedures (*destruction- construction*). For suitability, we named each these cases of DABC algorithm separately as DABC-I, DABC-II and DABC-III respectively. This step attempts to improve the population deterministically by accepting the improved adjacent solutions by examining their fitness values. The solutions to next step are chosen on the basis of equal number of best solutions from each objective respectively to maintain the population diversity.

Case I:
Each nearest solution in the population is determined by any one of the following strategy. The selected strategy is applied two times separately to each permutation *π* in the population, resulting two nearest neighbors and the best one is selected to the next step.

(i) Applying *two-insert* moves to a permutation *π* with *p=2.*
(ii) Applying *three-insert* moves to a permutation *π* with *p=3.*
(iii) Applying *two-swap* moves to a permutation *π* with *p=2.*
(iv) Applying *three-swap* moves to a permutation *π* with *p=3.*

Case II:
Each nearest solution is chosen by applying any of the following strategy.

(i) Applying *two-insert* moves to a permutation *π* with *p=2.*
(ii) Applying *three-insert* moves to a permutation *π* with *p=3.*
(iii) Applying *two-swap* moves to a permutation *π* with *p=2.*
(iv) Applying *three-swap* moves to a permutation *π* with *p=3.*
(v) Applying one *destruct-construct* procedure to a permutation *π* with *destruction size x.*

Case III:
The nearest solutions are determined by using the following strategy.

(i) Applying *destruct-construct* procedure to a permutation *π* with destruction size *x.*

Onlooker bee phase:
This phase selects a food source based on the probabilities obtained from the fitness values during employed bee phase. The aim of this phase is to find further better compromising solutions by applying well devised local search. The probabilistic selection can be described as:

$$p_i = \frac{fit_i}{\sum_j fit_j} \quad (5)$$

Here fit_i is defined as the fitness value of the i^{th} solution compared to other solutions in the solution set. The solutions with a higher probability are always selected to the next cycle. In addition to this, almost an equivalent strategy to that of employed bee phase is employed during the onlooker bee phase to produce a new neighborhood solution. An efficient local search method has to be applied to further improve the candidates of the onlooker bee phase. A better food source has to replace the current one and become a new member in the population; else both are treated as non-dominated to each other.

Scout phase:
In general, the scout bee phase removes the abandoned solutions (worst solutions) from the search space and tries to discover new ones with better fitness value. Therefore, the DABC algorithm removes a defined number of worst solutions and replaces them with new ones by the process of tournament selection in order to deal with local optima by avoiding the trial counter.

During the evolution process, the solutions will be prioritised with respect to TCT, WMT and WME. Also the different cases of the employed bee phase will fall to different CPU utilization of the algorithm. As per the selection of basic ABC algorithm, an old solution is replaced by a new one if it is found to be superior in all objectives by using a greedy selection procedure.

A common framework for DABC-I, DABC-II, and DABC-III as follows:

```
$[Initialization]$
    $\pi = \pi_1, \pi_2, \pi_3, ..............\pi_N$
$[Fitness calculation]$
    $for(i = 1 to N)$
    $Calculate f(\pi_i)$
$[Employed bee phase]$
    $for(i = 1 to N)$
    $\pi^{new} = local\_search(\pi)$
    $for(objective 1 to M)$
    $\pi = propertionate - best sequence(\pi^{new}, \pi)$
$[onlooker bee phase]$
    $for(i = 1 to N)$
    $\pi^{new} = local\_search(\pi)$
    $for(objective 1 to M)$
    $\pi = propertionate - best sequence(\pi^{new}, \pi)$
$[Scout bee phase]$
    $for(\pi = \pi_1, \pi_2, \pi_3, ...............\pi_N)$
    $replace\ abandoned solutions$
$return(\pi)$
```

Figure 2: DABC algorithm

4.1 Local search methods: IG_RS algorithm and ILS algorithm

The *insert* operator eliminates a job from the job pool (position *r*) and reinserts it into another position (*q*) in the same pool that is in the permutation *π*, such that *q*ϵ (*r, r-1*) and the *swap* operator simply interchanges the position of two random jobs in a permutation *π*. Similarly the *destruction- construction* procedure of IG_RS algorithm reconstructs a job pool by assigning best positions to a sub part of the original job sequence. Here the destruction phase randomly removes *x* number of jobs from the permutation *π* without repetition resulting two partial solutions π_x(*x* number of jobs) and $\pi_{x'}$ (*x'=n-x* number of jobs). Then the *construction* phase adds each removed jobs back to the pool in the same order by searching its best position. The motivation of using the above methodologies in our algorithm is inherited from the efficacy of the DABC algorithm. Here the focused parameters are: perturbation strength *p* and the

```
Procedure local_search(π)
$\pi = \pi_1, \pi_2, .........\pi_N$
$\pi^{new} = Null$
for(i = 1 to N)
for(j = 1 to N)
    $\pi' = \pi_i$
    $\pi'' = \begin{cases} insertlocd\_search(\pi) \\ swaplocal\_search(\pi) \\ destructconstructlocal\_search(\pi) \end{cases}$
        if $f(\pi'') < f(\pi')$
            $\pi^{new}{}_j = \pi''$
            else
              $\pi_j^{new} = \pi'$
              $j = j + 1$
        endif
end for
        $i = i + 1$
end for
return($\pi^{new}$)
}
```

Figure 3: Local_search procedure.

```
procedure swaplocal_search(π)
rand(i, j)
swaplocal_search(π, i, j)
return(π)
```

Figure 4: Swap local-search procedure.

```
procedure insertlocd_search(π)
rand(i, j)
insert(π, i, j)
return(π)
end procedure
```

Figure 5: Insert local_search procedure.

```
procedure destructconstruct(π, d)
$\pi^D = \pi^R = destruct_d(\pi)$
$\pi = construct(\pi^D, \pi^R)$
return(π)
end procedure
```

Figure 6: Destruct-construct procedure.

destruction size *x* that has to be carefully chosen. A perturbation is achieved by a random insertion of a job to another position or by swapping of some jobs randomly in a permutation *π*. Similarly choosing a larger destruction size for *x* will lead to a better result and a smaller one will be good for CPU time minimization. Tasgetiren et al. [43] have considered the perturbation values are as 1or 2 and the x values as 8 or 12 for different instances of Taillard [46]. However in our work, we have considered two synthetic datasets for small and large sized systems with variable number of jobs and machines. Here the *p* values are considered as 2 or 3 and the *x* values are considered as 2 (for small sized) and 4 (for large size) respectively.

5 MOABC for MOPFSSP

The above proposed DABC algorithm is the direct extension of single objective DABC proposed by Tasgetiren et al. [43]. The algorithmic framework and search for local optima is much more flexible and effective with the advantages of local search algorithms in the DABC algorithm. To achieve a more accurate and efficient problem solving approach in the field of multi objective optimization; we have simulated these advantages to model a multi objective Pareto-based ABC algorithm with same objectives to solve the FSSP. The proposed MOABC algorithm combines the main idea of ABC with the above local search strategy to search the neighborhood structure. To apply the local search algorithm in the next proposed one, we have adopted one of the simpler one i.e., the swap () local-search instead of using all methods randomly. Firstly the proposed MOABC algorithm initiates a number of randomized job sequences of *n* jobs, and is stored in the population matrix. These sequences represent the random food sources of ABC, with certain quality and diversity. Secondly, an exploitation search procedure for the first two bee phases (employed and onlooker) is designed to best suit the problem and to intensify the local search operation. To record the updated non-dominated sequence emerged in each cycle, it uses a Pareto-based archive set. In addition, the population is well-adjusted to maintain diversity in scout bee phase by eliminating the worst solutions. It is seen that proposed algorithm is able to find the best set of solutions and a proper statistical analysis has also been done to evaluate the proposed algorithm's performance with different inputs. Some important terms related to MOABC can be defined below.

Pareto dominance
Any solution S' is said to be non-dominated to S'' if and only if,

(i) (i)The solution S' is no worse than S'' in all the objectives.
(ii) The solution S' is strictly better than S'' in at least one objective.

Pareto optimal solution set and Pareto optimal front
Pareto optimal solution set is the group of all Pareto optimal solutions, and the respective graphical representation in the objective space is known as the Pareto optimal front.

Archive
An archive records the track of the non-dominated solutions from time to time. It is iteratively updated throughout the search procedure. Once a new non-dominated solution generated, the archive is updated accordingly.

5.1 Problem formulation

The FSSP is rescheduled (fixed to similar assumptions as stated above) with the same three defined criterions (TCT, WMT and WME) and *n* jobs to be solved with ABC. As we know mostly there will be multiple solutions, non-dominated to one another will be emerged during the simultaneous optimization of multiple objectives (known to discover true Pareto front), we have done a straight forward extension of uni-objective ABC as well as above DABC to redesign an MOABC algorithm. In the employed bee phase, an exploitation search procedure is applied on the initialized solutions, to derive the non-dominated solution set. The generated Pareto front is maintained in an archive with the corresponding trial counters and will be updated from time to time. Onlooker bees search for more intensified solutions within the neighborhood of the food source in their memory. Finally, the abandoned solutions are deleted from the archive to stand with a best fitted Pareto front.

5.2 Architecture

As per the problem architecture, '*n*' jobs are divided into '*m*' number of tasks, to be sequenced differently and to be processed in different machines. Each job sequences are evaluated through their fitness values against the individual objective functions. After the problem evaluation, the resulted sequences are listed out that are non-dominating to each other. Figure 7 is representing the MOPFSSP problem architecture which needs to be optimized to a set of optimal job sequences as corresponding non- dominated set.

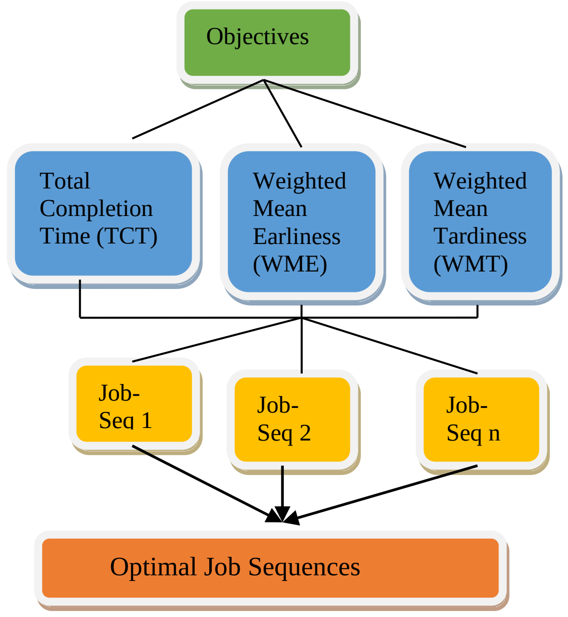

Objectives
Total Completion Time (TCT)
Weighted Mean Earliness (WME)
Weighted Mean Tardiness (WMT)
Job-Seq 1
Job-Seq 2
Job-Seq n
Optimal Job Sequences

Figure 7: MOPFSSP architecture.

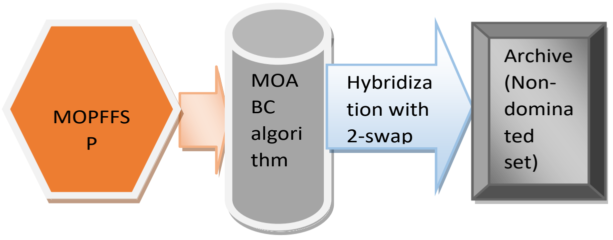


Figure 8: Proposed framework using MOABC.

Figure 8 is represents the proposed solution strategy using MOABC. The proposed model generates multiple Pareto optimal solutions iteratively which are updated in an external archive time to time. Here the algorithm adopts the 2swap () local search strategy to generate the neighborhood structures in the solution space. The selection of the same local search procedure is based upon the time complexity analysis of all considered methods in the remodeled DABC algorithm.

5.3 Proposed MOABC

This section presents the algorithmic representation of proposed MOABC algorithm to solve MOPFSSP.

The derived MOABC algorithm, initializes the population 'п' with 'n' solutions, each consisting of a random number of job sequences similar to the DABC algorithm. Each updated solutions in the population matrix are evaluated for the corresponding fitness value using the objective functions 1- 3. The generated non-dominated set is maintained in an archive with the corresponding trial counters; which is updated in every cycle. Employed bees explore for better sources in the neighborhood by applying *swap* () operation, where two randomly selected jobs *i* and *j* (two random selected dimensions) for a random solution (sequence) *k* are

```
BEGIN
{
        Set parameters;
        Set population size;
        Initialize solutions;
        Archive=Null;
        Trial counter=Null;
                For each solution find the fitness value;
                Generate the non-dominated set;
                Update Archive;
                Do
                 {
                //Employed bee phase//
                Generate all employed bees and check
                their dominance relation to nearby
                solutions by swaplocal_ search
                procedure;
                Compute the Fitness value;
                Compute non-dominated set;
                Update Archive;
                Update trial counter;
                //Onlooker bee phase//
                Update the solutions using swaplocal_search () algorithm;
                Compute the Fitness value;
                Compute non-dominated set;
                Update Archive;
                Update trial counter;
                } While (Stop criterion=Max. no. of iterations);
        //Scout bee phase//
        Delete abandoned solutions
        Update Archive;
    }
        END
```

Figure 9: MOABC pseudo code.

swapped with each other**.** Onlooker bee selects a candidate source depending on its probability values calculated and provided by the employed bees. The solutions with a greater probability are shifted to the archive. Within a defined number of cycles, the employed bees whose solutions cannot be further improved (through a predetermined number of trials) are treated as abandoned ones and are deleted permanently from the archive. These abandoned solutions are calculated by the help of trial counters. If a solution in *S* is improved by the corresponding solution in *S'* then the trial counter is set to zero (0), else it is set to one (1).

6 Numerical simulation

The numerical results represent the performance of both DABC algorithm and MOABC algorithm respectively with respect of TCT, WME^e and WMT^r. Two different datasets have been initialized with little parameter variation. One of this has been initialized with small processing times and due dates named as 'small-size dataset' and the other one is named as 'large-size'. For both the proposed algorithms, we have considered similar input datasets.

6.1 Control parameters

However both the algorithms require same control parameters except the case of abandoned solution. The DABC algorithm removes a defined number of worst solutions and replaces them with new ones in order to remove abandoned solutions from the population where as the MOABC algorithm removes those solutions based on a trial counter limit.

6.1.1 Parameters of DABC

Parameters:	Values:
Population size	10
Maximum iterations	50
Number of onlookers	1/2*(colony size)
Number of employed bees	1/2*(colony size)
Worst solutions to be replaced	2 or 4

6.1.2 Parameters of MOABC

Parameters:	Values:
Colony size	10
Maximum iterations	50
Number of onlookers	1/2*(colony size)
Number of employed bees	1/2*(colony size)
Limit for abandoned solution	20

6.2 Description of the numerical data

To evaluate DABC-I, DABC-II and DABC-III, two instances of datasets are customized with two different combinations of jobs and machines. With a little parameter variation both the datasets consider same population size of 10. The due date of each job is initialized separately with respect to two datasets. We have assigned same weight for both tardiness and earliness in both the input sets, while evaluating WME^e and WMT^r. Again the same datasets are used to characterise the performance of MOABC.

6.2.1 Small-size data

To validate the results at an eye, a small size dataset is randomized with a combination of 4 jobs and 3 machines with an ideal parameter setting. The processing time (O_{ik}) of the jobs are set within [0, 5] and the due times are set in [10, 15]. The earliness and tardiness weights are considered in the range [1, 10]. Based on the due time, the calculated TCT and the weights (earliness and tardiness), the MWT and MWE of the jobs have to be calculated. The destruction size has been assumed as 2. All these parameters, input data, corresponding statistics and the initial population sequence are tabulated below.

	Job 1	Job 2	Job 3	Job 4
Machine 1	$O_{11}=4$	$O_{12}=1$	$O_{13}=5$	$O_{14}=2$
Machine 2	$O_{21}=3$	$O_{22}=2$	$O_{23}=4$	$O_{24}=3$
Machine 3	$O_{31}=5$	$O_{32}=2$	$O_{33}=3$	$O_{34}=4$
Due time	10	12	30	15
Weight	2	3	4	2

Table 1: Processing time of machine vs task of each job.

Jobs	Minimum	Maximum	Standard deviation
1	3	5	0.81
2	1	2	0.47
3	3	5	0.81
4	2	4	0.81

Table 2: Statistics of the small-size dataset.

Parameter setting

(i) O_{jk} :[1-5]

(ii) Weight (tardiness and earliness): [1, 10]

(iii) Population size: 10

(iv) Destruction size: 2

(v) Due time: [10, 15]

Table 1 represents the processing time of 4 different jobs with respect to 3 machines. Also it initializes the expected finish time of each job and an assigned weight which will be further used to calculate the fitness value of the defined objective functions.

Table 2 contains the statistical analysis of standard deviation for each job(small-sized dataset).To calculate the standard deviation we have summarized the minimum and maximum processing time of each job from the pool. The result shows that, standard deviation of each job ranges between [0, 1].

Population sequence	Job sequence			
1	J_0	J_1	J_2	J_3
2	J_1	J_2	J_3	J_0
3	J_2	J_3	J_0	J_1
4	J_3	J_0	J_1	J_2
5	J_3	J_2	J_1	J_0
6	J_0	J_3	J_2	J_1
7	J_1	J_0	J_3	J_2
8	J_2	J_1	J_0	J_3
9	J_0	J_1	J_2	J_3
10	J_1	J_2	J_3	J_0

Table 3: Initial population.

Using the above information a randomized job sequence is initialized with population size 10. As we have considered 4 jobs here, it can have 24 numbers of different possible sequences. Table 3 contains a random selection of 10 sequences out of these. These sequences will be the initial input for the proposed algorithm and the resulted intermediate sequences will be the further inputs for different iterations and bee phases.

6.2.2 Large-size data

The other synthetic large size dataset with population size 10 is generated with 10 jobs and 9 machines are set with the following parameter setting. Here the processing time of jobs are set to [0, 10]. The weights and due times

	Job 0	Job 1	Job 2	Job 3	Job 4	Job 5	Job 6	Job 7	Job 8	Job 9
Machine 1	$O_{11}=10$	$O_{12}=5$	$O_{13}=8$	$O_{14}=5$	$O_{15}=1$	$O_{16}=7$	$O_{17}=2$	$O_{18}=0$	$O_{19}=9$	$O_{1\,10}=3$
Machine 2	$O_{21}=3$	$O_{22}=4$	$O_{23}=5$	$O_{24}=8$	$O_{25}=3$	$O_{26}=6$	$O_{27}=2$	$O_{28}=5$	$O_{29}=7$	$O_{2\,10}=4$
Machine 3	$O_{31}=5$	$O_{32}=3$	$O_{33}=0$	$O_{34}=3$	$O_{35}=5$	$O_{36}=9$	$O_{37}=0$	$O_{38}=0$	$O_{39}=2$	$O_{3\,10}=3$
Machine 4	$O_{41}=4$	$O_{42}=2$	$O_{43}=4$	$O_{44}=7$	$O_{45}=2$	$O_{46}=3$	$O_{47}=4$	$O_{48}=9$	$O_{49}=0$	$O_{4\,10}=3$
Machine 5	$O_{51}=1$	$O_{52}=2$	$O_{53}=1$	$O_{54}=5$	$O_{55}=4$	$O_{56}=7$	$O_{57}=2$	$O_{58}=3$	$O_{59}=4$	$O_{5\,10}=6$
Machine 6	$O_{61}=7$	$O_{62}=3$	$O_{63}=2$	$O_{64}=5$	$O_{65}=3$	$O_{66}=6$	$O_{67}=9$	$O_{68}=4$	$O_{69}=4$	$O_{6\,10}=2$
Machine 7	$O_{71}=3$	$O_{72}=0$	$O_{73}=2$	$O_{74}=0$	$O_{75}=5$	$O_{76}=5$	$O_{77}=5$	$O_{78}=4$	$O_{79}=2$	$O_{7\,10}=2$
Machine 8	$O_{81}=8$	$O_{82}=4$	$O_{83}=1$	$O_{84}=0$	$O_{85}=5$	$O_{86}=9$	$O_{87}=4$	$O_{88}=5$	$O_{89}=4$	$O_{8\,10}=6$
Machine 9	$O_{91}=2$	$O_{92}=4$	$O_{93}=1$	$O_{94}=10$	$O_{95}=8$	$O_{96}=3$	$O_{97}=4$	$O_{98}=5$	$O_{99}=4$	$O_{9\,10}=7$
Due time	80	42	75	85	95	60	100	105	90	65
Weight	2	3	4	6	10	1	4	5	7	9

Table 4: Processing time of machine vs job task.

Jobs	Minimum	Maximum	Standard deviation
1	1	10	2.81
2	0	5	1.45
3	0	8	2.40
4	0	10	3.18
5	1	8	1.94
6	3	9	2.07
7	0	9	2.40
8	0	9	2.72
9	0	9	2.53
10	2	7	1.76

Table 5: Statistics of the large-size dataset.

are initialized within [1, 10] and [50, 100] respectively. The destruction size has been assumed as 4. The same required data as per the small sized dataset are also represented using different tabulations in the same sequence.

Parameter setting

(i) O_{jk} :[0-10]

(ii) Weight (tardiness and earliness): [1, 10]

(iii) Population size: 10

(iv) Destruction size: 4

As per the parameter setting, Table 4 is finalized with different processing times for individual jobs with respect to corresponding machines. It also assumes the due times and job weights. Job weights are basically the representative of their priorities.

Similar to the first dataset, we have also done a statistical analysis of the large-size dataset in Table 5.As per the minimum and maximum processing time of each job, the standard deviation of the jobs ranges between [1,3].

Table 6 contains the initial population set consisting of 10 jobs. These jobs can be arranged in 10! number of possible ways and we have selected a random 10 out of it

P	Job sequence									
1	J_0	J_1	J_2	J_3	J_4	J_5	J_6	J_7	J_8	J_9
2	J_1	J_2	J_3	J_4	J_5	J_6	J_7	J_8	J_9	J_0
3	J_2	J_3	J_4	J_5	J_6	J_7	J_8	J_9	J_0	J_1
4	J_3	J_4	J_5	J_6	J_7	J_8	J_9	J_0	J_1	J_2
5	J_4	J_5	J_6	J_7	J_8	J_9	J_0	J_1	J_2	J_3
6	J_5	J_6	J_7	J_8	J_9	J_0	J_1	J_2	J_3	J_4
7	J_6	J_7	J_8	J_9	J_0	J_1	J_2	J_3	J_4	J_5
8	J_7	J_8	J_9	J_0	J_1	J_2	J_3	J_4	J_5	J_6
9	J_8	J_9	J_0	J_1	J_2	J_3	J_4	J_5	J_6	J_7
10	J_9	J_0	J_1	J2	J_3	J_4	J_5	J_6	J_7	J_8

Table 6: Initial population.

as the initial input. As compared to the small-sized dataset there is a very less chance of repeating the same sequences as the intermediate result sequences, due the application of different tuning operators (insert/swap/construct-destruct).

6.3 Numerical results and analysis

Using the above specified inputs the results are tabulated separately for each algorithm. First, the results of DABC are represented and then that of MOABC. Firstly the results are tabulated then are reflected into corresponding graphical representations through the help of various figures where 'X' and 'Y' dimensions represents the 'performance score' versus 'job sequences' respectively. Each unit of 'X' and 'Y' dimension in the small-size dataset counts as 5 and 1 respectively, similarly it counts as 20 and 1 for the large- size dataset for the same dimensional sequence.

6.3.1 Results of DABC Algorithm

The tabulated results include the performance of DABC algorithms for individual cases with two specified inputs. Table 7-9 represents the final job sequences for small dataset corresponding to TCT, MWT and MWE and table 10-12 includes the results for the other input dataset. Table 7 and 10 includes the results of DABC-I algorithm with swap () and insert () operation having random perturbation values 2 or 3. Table 8 and 11 shows the results of DABC-II, that include another operation construct-destruct () additional to the operations of DABC-I. Only construct-destruct is used in DABC-III and the results are tabulated in Table 9 and12.

6.3.1.1 Small-size dataset

Table 7 contains the resulted TCT, MWT, MWE of the small-sized dataset for DABC-I. As mentioned, DABC-I uses the swap () and insert () algorithms to update the solution vectors. The result includes the completion time of every job in different sequences and TCT of each job sequence is equal to the completion time of the last job of the individual sequence. According to the initialized weight and due time the respective MWT and MWE has been calculated.

The graphical representation of Table 7 has been shown in Figure 10. Each job sequences have been represented individually with its corresponding TCT, MWE, and MWT scores.

The results of DABC-II is tabulated in Table 8.Based on the completion time, weight and due date of individual jobs the corresponding TCT, MWT, and MWE values are summarized and presented here. To update the job sequences here all the local search methods (insert/swap/construction and destruction) have been applied randomly.

P	Final job sequence & completion time				TCT	MWT	MWE
1	J_3	J_1	J_0	J_2	19	0.72	5.54
	9	11	16	19			
2	J_3	J_1	J_2	J_0	20	0.9	5.9
	9	11	15	20			
3	J_0	J_1	J_3	J_2	21	2.0	4.36
	12	14	18	21			
4	J_1	J_0	J_2	J_3	21	1.3	5.6
	5	13	17	21			
5	J_1	J_2	J_0	J_3	22	1.54	5.27
	5	13	18	22			
6	J_1	J_2	J_3	J_0	22	1.54	5.63
	5	13	17	22			
7	J_2	J_1	J_0	J_3	23	2.36	4.0
	12	14	19	23			
8	J_0	J_3	J_2	J_1	21	2.54	4.0
	12	16	19	21			
9	J_0	J_3	J_1	J_2	21	2.54	4.36
	12	16	18	21			
10	J_2	J_3	J_1	J_0	23	2.9	4.36
	12	16	18	23			

Table 7: Results of DABC-I.

Population	Final job sequence & completion time				TCT	MWT	MWE
1	J_1	J_3	J_0	J_2	19	0.72	6.9
	5	10	15	19			
2	J_3	J_0	J_1	J_2	19	1.27	5.27
	9	14	16	19			
3	J_1	J_3	J_2	J_0	20	0.90	6.9
	5	10	15	20			
4	J_3	J_2	J_1	J_0	21	1.63	5.27
	9	14	16	21			
5	J_3	J_2	J_1	J_0	21	1.63	5.27
	9	14	16	21			
6	J_3	J_2	J_1	J_0	21	1.63	5.27
	9	14	16	21			
7	J_3	J_2	J_0	J_1	21	1.63	4.18
	9	14	19	21			
8	J_0	J_2	J_3	J_1	22	2.72	3.63
	12	16	20	22			
9	J_0	J_2	J_3	J_1	22	2.72	3.63
	12	16	20	22			
10	J_2	J_0	J_1	J_3	23	3.18	4.0
	12	17	19	23			

Table 8: Results of DABC-II.

The tabulated results of DABC-II are graphically represented in Figure 11. Like DABC-I, most of the cases have more earliness penalty than the tardiness penalty. While adopting selection of average number of solution sequences from each objective, we found some of the repeating sequences. These have to be treated as one ultimately. Hence, the total numbers of non-dominated sequences are 7 in number but we have represented all repeated sequences also.

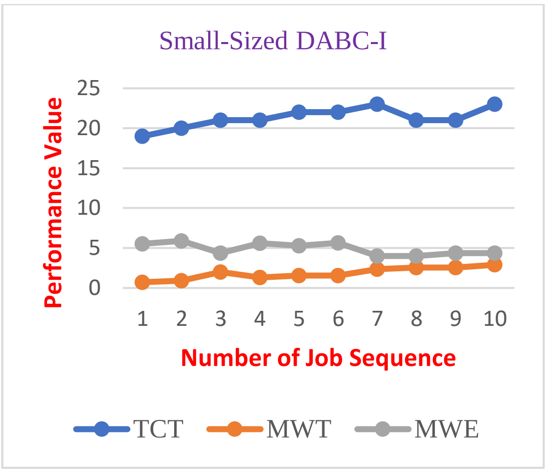


Figure 10 Graphical representation of DABC-I.

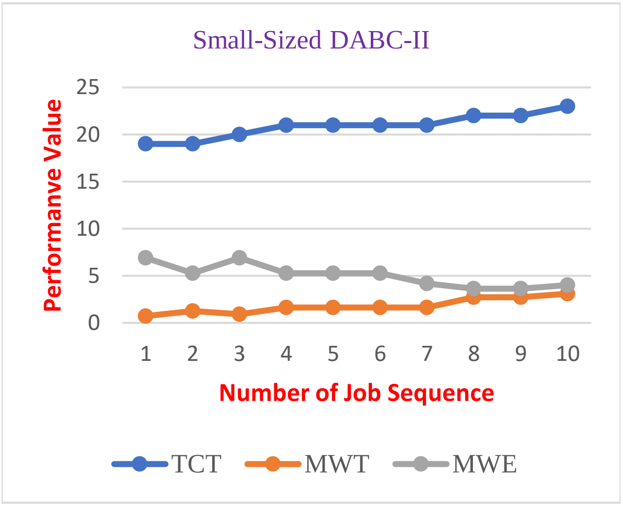


Figure 11: Graphical representation of DABC-II.

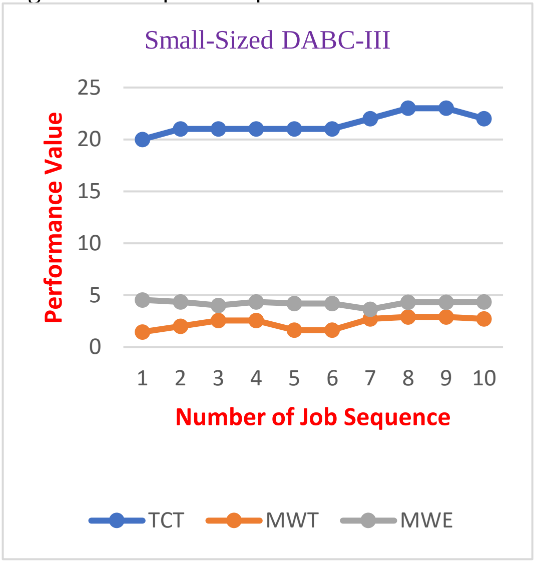


Figure 12: Graphical representation of DABC-III.

The results of DABC-III have been listed in Table 9. The three objective functions are evaluated with a recursive set of sequences and the fitness values are summarized. DABC-III explicitly uses destruct-construct for perturbing the solution sets.

P	Final job sequence & completion time				TCT	MWT	MWE
1	J_3	J_0	J_2	J_1	20	1.45	4.54
	9	14	18	20			
2	J_0	J_1	J_3	J_2	21	2.0	4.36
	12	14	18	21			
3	J_0	J_3	J_2	J_1	21	2.54	4.0
	12	16	19	21			
4	J_0	J_3	J_1	J_2	21	2.54	4.36
	12	16	18	21			
5	J_3	J_2	J_0	J_1	21	1.63	4.18
	9	14	19	21			
6	J_3	J_2	J_0	J_1	21	1.63	4.18
	9	14	19	21			
7	J_0	J_2	J_3	J_1	22	2.72	3.63
	12	16	20	22			
8	J_2	J_3	J_1	J_0	23	2.9	4.3
	12	16	18	23			
9	J_2	J_3	J_1	J_0	23	2.9	4.3
	12	16	18	23			
10	J_0	J_2	J_1	J_3	22	2.72	4.36
	12	16	18	22			

Table 9: Results of DABC-III.

Table 9 results are graphically represented in Figure 12 showing a clear-cut demarcation of TCT, MWE, and MWT for 10 different sequences. Apart from TCT, most of the non-dominated sequences have earliness penalty is more than tardiness penalty.

6.3.1.2 Large-size dataset

Table 10 stores the results of DABC-I for the large-sized dataset. Along with every sequence, the completion time of individual jobs are listed leading to the TCT score of that sequence. Out of 10 random sequences of 10 different jobs, 8 sequences are having greater MWE score than MWT.

Table 10 is graphically represented in Figure 13, showing the ratio of MWE score, MWT score, and TMT score of individual job sequence. Except sequence 10 and 9, other sequences are having more MWE score than MWT score.

Results of DABC-II for the second input are stored in Table 11. The results also show similar efficiency as that before. Here also the earliness cost is more in many sequences.

Figure 14 represents Table 11. We can obtain the better sequences having minimum TCT, MWT, and MWE. As discussed, DABC-II uses all the local search methods to improve pre existing solutions.

DABC-III results for the large-sized input are tabulated in Table 12. It shows construct-destruct () has similar efficiency to search good solutions from the search space. But, out of all local search algorithms this one is having maximum algorithmic complexity.

Figure 15 represents DABC-III, pictorially showing the fitness value of different resulted sequences.

As discussed above the result tables and corresponding graphs have been represented below.

P	Final job sequence & completion time										TCT	MWT	MWE
1	J_7	J_4	J_3	J_5	J_6	J_8	J_9	J_0	J_1	J_2	92	5.21	14.43
	35	43	53	64	69	73	82	85	91	92			
2	J_7	J_2	J_4	J_5	J_6	J_3	J_8	J_9	J_0	J_1	96	6.11	14.6
	35	36	48	64	69	79	83	90	92	96			
3	J_7	J_5	J_8	J_4	J_9	J_0	J_1	J_2	J_3	J_6	98	7.49	11.94
	35	55	60	69	76	78	83	84	94	98			
4	J_0	J_1	J_6	J_3	J_4	J_5	J_2	J_7	J_8	J_9	100	7.25	12.84
	43	49	55	65	73	80	81	88	92	100			
5	J_1	J_2	J_3	J_4	J_8	J_6	J_7	J_5	J_9	J_0	104	8.68	14.0
	27	29	56	67	71	75	81	91	101	104			
6	J_9	J_7	J_0	J_1	J_2	J_8	J_3	J_4	J_5	J_6	106	8.84	17.8
	36	43	49	55	56	62	80	91	101	106			
7	J_0	J_1	J_2	J_9	J_3	J_4	J_5	J_6	J_7	J_8	106	9.7	12.52
	43	49	50	61	71	81	91	96	102	106			
8	J_5	J_6	J_9	J_8	J_0	J_7	J_1	J_2	J_3	J_4	107	10.17	9.0
	55	60	69	73	76	85	88	89	99	107			
9	J_1	J_2	J_3	J_5	J_4	J_6	J_7	J_8	J_9	J_0	106	9.7	9.11
	27	29	56	74	84	88	93	97	104	106			
10	J_8	J_2	J_0	J_3	J_4	J_5	J_6	J_7	J_9	J_1	107	9.66	11.9
	36	37	55	65	74	84	89	95	103	107			

Table 10: Results of DABC-I.

P	Final job sequence & completion time										TCT	MWT	MWE
1	J_9	J_0	J_1	J_7	J_3	J_4	J_5	J_6	J_8	J_2	94	6.07	14.82
	36	46	52	58	68	76	84	89	93	94			
2	J_7	J_8	J_9	J_4	J_1	J_0	J_2	J_3	J_6	J_5	95	5.3	20.49
	35	39	49	57	61	63	64	74	85	95			
3	J_6	J_3	J_4	J_5	J_7	J_2	J_8	J_9	J_0	J_1	95	5.74	13.64
	32	45	56	66	73	74	78	86	89	95			
4	J_3	J_4	J_8	J_6	J_7	J_0	J_9	J_5	J_1	J_2	95	6.8	13.6
	43	54	58	62	68	73	84	89	94	95			
5	J_6	J_2	J_3	J_4	J_5	J_8	J_7	J_1	J_9	J_0	100	7.5	13.47
	32	33	53	64	74	79	85	89	97	100			
6	J_8	J_9	J_0	J_2	J_7	J_1	J_3	J_4	J_5	J_6	109	7.78	15.82
	36	49	54	55	66	70	80	88	96	109			
7	J_8	J_4	J_0	J_5	J_6	J_7	J_9	J_1	J_2	J_3	105	8.7	10.6
	36	51	54	71	76	82	90	94	95	105			
8	J_5	J_6	J_7	J_8	J_9	J_0	J_3	J_2	J_1	J_4	104	9.17	8.76
	55	60	66	70	78	81	91	92	96	104			
9	J_5	J_2	J_6	J_0	J_7	J_8	J_9	J_1	J_3	J4	111	11.13	9.45
	55	56	62	69	77	81	89	93	103	111			
10	J_8	J_2	J_0	J_3	J_4	J_5	J_6	J_7	J_9	J_1	107	9.66	11.9
	36	37	55	65	74	84	89	95	103	107			

Table 11: Results of DABC-II.

Population individual	Final job sequence & completion time										TCT	MWT	MWE
1	J_6	J_3	J_4	J_7	J_5	J_8	J_9	J_1	J_0	J_2	93	5.7	13.6
	32	45	56	61	71	76	85	89	92	93			
2	J_1	J_9	J_3	J_4	J_5	J_6	J_7	J_8	J_2	J_0	94	5.4	14.47
	27	42	52	62	72	77	83	87	88	94			
3	J_6	J_7	J_3	J_9	J_0	J_1	J_2	J_5	J_4	J_8	95	5.54	19.31
	32	39	49	56	60	66	67	81	91	95			
4	J_6	J_7	J_3	J_9	J_0	J_1	J_2	J_5	J_4	J_8	95	5.54	19.31
	32	39	49	56	60	66	67	81	91	95			
5	J_7	J_6	J_3	J_8	J_9	J_0	J_1	J_2	J_5	J_4	96	5.64	19.0
	35	43	53	57	64	66	70	71	86	96			
6	J4	J_6	J_0	J_1	J_2	J_3	J_5	J_7	J_8	J_9	99	6.2	19.17
	36	40	47	53	54	64	80	87	91	99			
7	J_3	J_4	J_7	J_5	J_6	J_8	J_9	J_0	J_1	J_2	97	7.52	11.19
	43	54	59	69	74	78	87	90	96	97			
8	J_5	J_3	J_1	J_6	J_7	J_8	J_9	J_0	J_2	J_4	101	8.41	8.43
	55	65	69	73	78	82	89	91	92	101			
9	J_5	J_7	J_8	J_9	J_0	J_1	J_6	J_2	J_3	J_4	106	9.96	9.19
	55	62	66	74	77	83	87	88	98	106			
10	J_7	J_8	J_5	J_9	J_0	J_1	J_2	J_3	J_4	J_6	106	9.3	9.9
	35	39	64	74	77	83	84	94	102	106			

Table 12: Results of DABC-III.

6.3.1.3 Discussion and time complexity analysis of DABC algorithm

The DABC algorithm is tested under three types of scenarios using the local search algorithms such as two-swap (), three-swap (), two- insert (), three-insert () and destruct-construct () iteratively. Each time a random local search algorithm is used to find out nearest optimal solutions. To achieve the population diversity we have used the selection strategy of selecting proportionately equal number of solutions from each objective function. In the small-sized input data we see that many times the resulted sequences are being repeated because of less number of jobs, which is a rare in the large one. We checked the complexity of these algorithms in terms of number of machines (M) and number of jobs (N).

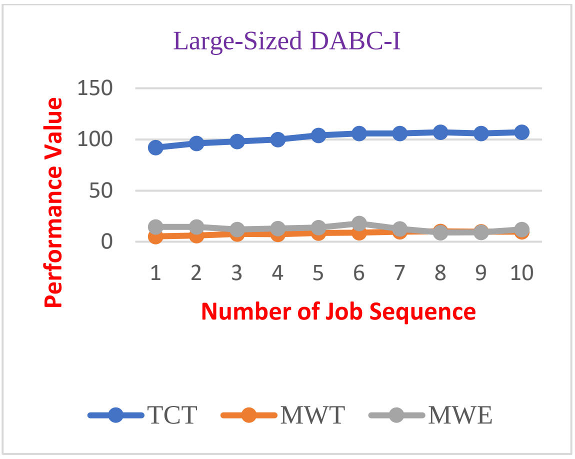


Figure 13: Graphical representation of DABC-I.

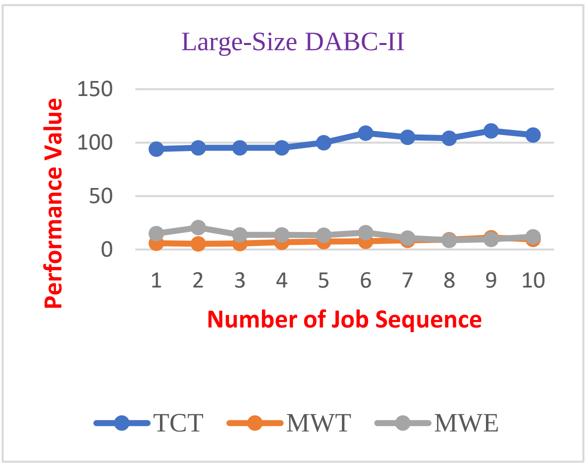


Figure 14: Graphical representation of DABC-II.

Local-search algorithms	Time complexity
Two-swap	O(MN)
Three-swap()	O(MN)
Two-insert()	O(MN)
Three-insert()	O(MN)
Construct-destruct()	O(MN^2)

Table 13: Time complexity analysis of local search algorithms.

We have used these functions alternatively in DABC-I, DABC-II and DABC-III, and see that using construct-destruct (), the algorithm is not giving any significant improvement in the result. Based on the time complexity of different local search methods, we conclude that DABC-I is better than DABC-II and DABC-II is better than DABC-III.

6.3.2 Results and discussions through MOABC

While optimizing three objectives through MOABC, a number of non-dominated solutions are resulted and are listed below with their respective Pareto fronts. Here we do not find any abandoned solutions as there was no solution in the final archive having trial counter value more than 20. In the small-size dataset there are '7' non-dominated sets and the large one is resulting 10 such solutions in the resulting Pareto front.

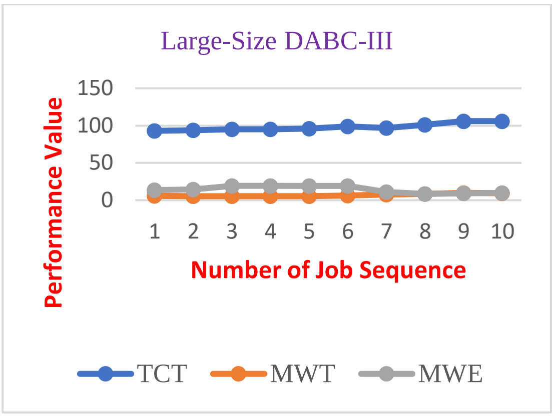


Figure 15: Graphical representation of DABC-III.

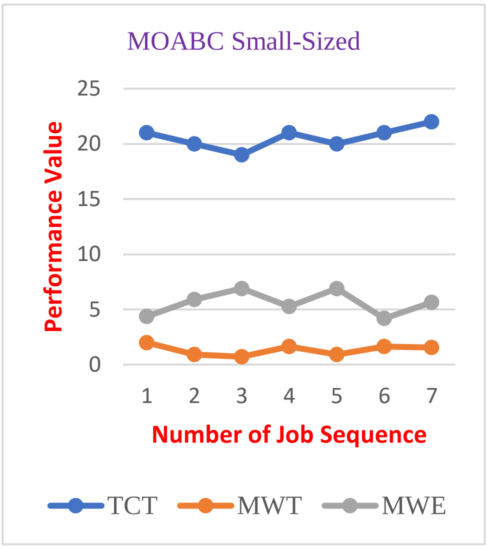


Figure 16: Pareto front (small-size).

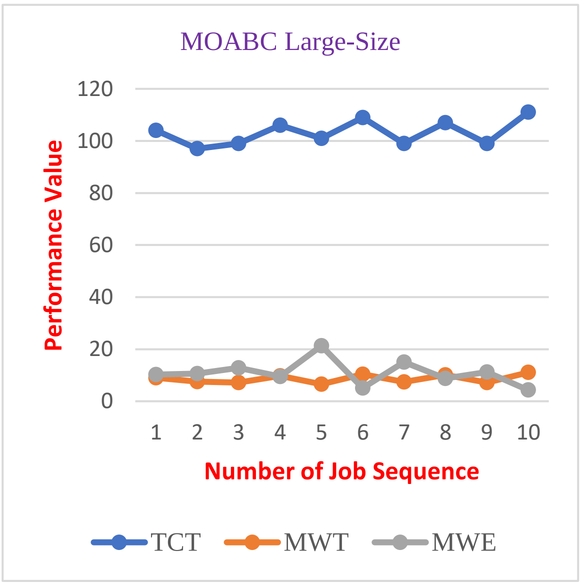


Figure 17: Pareto front (large-size).

P	Final job sequence & completion time				TCT	MWT	MWE
1	J_0	J_1	J_3	J_2	21	2.0	4.36
	12	14	18	21			
2	J_3	J_1	J_2	J_0	20	0.9	5.9
	9	11	15	20			
3	J_1	J_3	J_0	J_2	19	0.72	6.9
	5	10	15	19			
4	J_3	J_2	J_1	J_0	21	1.63	5.27
	9	14	16	21			
5	J_1	J_3	J_2	J_0	20	0.90	6.9
		10	15	20			
6	J_3	J_2	J_0	J_1	21	1.63	4.18
	9	14	19	21			
7	J_1	J_2	J_3	J_0	22	1.54	5.63
	5	13	17	22			

Table 14: Non-dominated job sequence.

6.3.2.1 Small-size dataset

7 non-dominated solutions emerged from the first dataset and are listed in Table 14. These solutions can be further evaluated by the decision maker to reach at the definite goal.

The resulted non-dominated set of table 14 has been depicted to the corresponding Pareto front in figure 16. The 3 objectives fitness values show a clear graphical visualization of the non-dominated set.

6.3.2.2 Large-size dataset

Table 15 stores the 10 non-dominated solutions emerged from the large-sized dataset. Each solution is represented with individual job completion time and finally the TCT value of the same sequence, followed by MWT and MWE respectively.

Each best fitted solution for the large-sized dataset is captured as its Pareto front and is represented in figure 18, with its respective fitness values.

The MOABC also yields equally compromising optimized solutions as that of DABC algorithm. The results reveal that the proposed algorithms are superior enough to deal with multi-objectives with a little parameter variation to the canonical ABC. It is a straight forward extension of uni-objective ABC with mixing advantages of local search procedure from the proposed DABC algorithm. We have just applied one of the simplest local search procedure that is *two-swap* () procedure to optimize the local optima which definitely helps in reducing the algorithmic complexity.

From the result analysis, apart from the completion time, it is seen that most of time the earliness penalty is more than the tardiness penalty. Hence with a required priority level of all the objectives a decision maker can easily go for making a balanced decision for him by applying a suitable MCDM method.

7 Decision making with chaotic-TOPSIS

After generating successful optimized solution set, we cannot avoid for selecting an appropriate one among these during the decision making process. MCDM is a successful tool for decision making with conflicting

P	Final job sequence & completion time										TCT	MWT	MWE
1	J_2	J_3	J_4	J_5	J_6	J_7	J_8	J_9	J_0	J_1	104	8.96	10.27
	24	51	62	72	77	83	87	95	98	104			
2	J_3	J_4	J_5	J_6	J_7	J_8	J_9	J_0	J_1	J_2	97	7.54	10.60
	43	54	64	69	75	79	87	90	96	97			
3	J_4	J_5	J_6	J_7	J_8	J_9	J_0	J_1	J_2	J_3	99	7.19	12.86
	36	56	61	67	71	79	82	88	89	99			
4	J_5	J_6	J_7	J_8	J_9	J_0	J_1	J_2	J_3	J_4	106	9.8	9.47
	55	60	66	70	78	81	87	88	98	106			
5	J_6	J_7	J_8	J_9	J_0	J_1	J_2	J_3	J_4	J_5	101	6.54	21.29
	32	39	43	51	57	63	64	80	91	101			
6	J_0	J_1	J_5	J_3	J_4	J_2	J_6	J_7	J_8	J_9	109	10.39	5.03
	43	49	70	80	88	89	93	98	102	109			
7	J_1	J_2	J_6	J_4	J_5	J_3	J_7	J_8	J_9	J_0	99	7.37	15.01
	27	29	50	60	71	81	86	90	97	99			
8	J_2	J_3	J_7	J_5	J_6	J_4	J_8	J_9	J_0	J_1	107	10.15	8.76
	24	51	61	74	79	89	93	100	102	107			
9	J_4	J_5	J_9	J_7	J_8	J_6	J_0	J_1	J_2	J_3	99	7.19	11.21
	36	56	66	71	75	79	82	88	89	99			
10	J_0	J_1	J_5	J_4	J_3	J_2	J_6	J_7	J_8	J_9	111	11.05	4.29
	43	49	70	80	90	91	95	100	104	111			

Table 15: Non-dominated job sequence.

criterion. Various methods show their respective efficiency in this regard. By a comparative survey we have concluded to decide the final optimal solution here with in our problem using TOPSIS method which really seems to be fit .We have summarized some of the recent TOPSIS applications followed by the discussions of our motivation.

Li et al. [47] presents a new method based on TOPSIS and response surface method (RSM) for MCDM problems with interval number. Similarly Madi et al. [48] provided a detailed comparison of TOPSIS and Fuzzy-TOPSIS in a systematic and stepwise manner. Sotoudeh-Anvari [49] suggested a stochastic multi-objective optimization model for assigning resource and time in order to search the individuals who are trapped in disaster regions. To reduce the heavy computation of the model, two efficient MCDM techniques, i.e. TOPSIS and COPRAS are employed which tackles the ranking problem. Zavadskas et al. [50] reviewed 105 papers which developed, extended, proposed and presented TOPSIS approach for solving DM problems from 2000 to 2015. Recently Wu et al.[51] proposes an improved methodology for handling ships which uses TOPSIS method to make the final decision.

TOPSIS

TOPSIS was developed by Hwang and Yoon [52] in the year of 1981 as an alternative to the elimination and choice translating reality (ELECTRE) method. The basic idea of TOPSIS is quite simple and it has been originated from a displaced ideal point from which the selected solution has shortest distance [53-54]. Further it is refined [52] to the rank based method by assigning specific orders to the available alternatives. The whole concept is based on the two artificial ideal points; that is the ultimate solution is measured by having longest distance from the positive ideal solution (PIS) and the shortest from the negative ideal solution (NIS). Hence a preference order of all alternatives is generated as per their relative closeness to the ideal solutions. As concluded by Kim et al. [55] and our observations, basic TOPSIS advantages are recorded as:

(i) It is an accepted logic that is focused to rationale of human choice;

(ii) A scalar value justifies both the ideal alternatives together;

(iii) Simple algorithmic framework and can easily be coded to the spreadsheet;

(iv) A straightforward performance evaluation of all alternatives against the defined criteria which can be clearly visualized and represented for two or more dimensions.

The above defined advantages make TOPSIS an omnipresent MCDM technique as compared with rest techniques [52]. In fact it is a utility-based method that evaluates every alternative directly depending on the available data in the decision matrices and weights [56]. Apart from this, the simulation comparison [57] of TOPSIS method signifies that it has the fewest rank reversals apart from rest methods in the category. Thus, TOPSIS is chosen as the backbone of MCDM.

The preliminary issue with the method is the normalized decision matrix operation, where randomness is achieved while assigning the criterion weights. Hence a narrow gap derived between the performed measures due to the weighted normalized value of the decision matrix. It can be advantageous to substitute this randomness with a suitable chaotic map. Chaos has a very similar property to randomness with better statistical and dynamical characteristic. Such a dynamic mixing is truly appreciated to enhance solutions potentiality by touching every mode in a multi-objective landscape. Hence the use of a well-suit chaotic map in TOPSIS can be definitely helpful to enhance the decision making by generating preferred randomness in criterion weight.

Chaotic maps

Simulation of complex phenomena such as: numerical analysis, decision making, sampling, heuristic optimization etc. needs random sequences for a longer period and good uniformity [58]. Chaotic map is a deterministic, discrete-time dynamic system that is considered as source of randomness, which is non-period, bounded and non-converging [59-60]. However the nature of chaotic maps is apparently random, unpredictable and it has a very sensitive dependence on its initial condition and parameter [58].

A chaotic map can be represented as:

$$x_{k+1} = f(x_k),\ 0, x_k < 1,\ k = 0,1,2,3... \qquad (12)$$

Different selected chaotic maps that produce chaotic numbers in [0, 1] are listed below in table 16 [59-60].

Chao Map	Definition
Logistic Map	$x_{n+1} = 4x_n(1-x_n)$
Circle Map	$x_{n+1} = x_n + 1.2 - (0.5/2\pi x_n)\bmod(1)$
Gauss Map	$x_n = \begin{cases} 0, & x_n = 0 \\ 1/x_n \bmod(1), & otherwise \end{cases}$ $1/x_k \bmod(1) = 1/x_k - [1/x_k]$
Henon Map	$x_{n+1} = 1 - 1.4x_n^{\ 2} + 0.3x_{n-1}$
Sinusoidal Map	$x_{n+1} = \sin(\pi x_n)$
Sinus Map	$x_{n+1} = 2.3(x_n)^{2\sin(\pi x_n)}$
Tent Map	$x_{n+1} = \begin{cases} x_n/0.7, & x_n < 0.7 \\ 10/3x_n(1-x_n), & otherwise \end{cases}$

Table 16: Different Chaotic Maps.

Altern-ative	TCT	MWT	MWE	Closeness coefficient	Rank
A_1	19	0.72	5.54	0.1524	10
A_2	20	0.9	5.9	0.2130	9
A_3	21	2.0	4.36	0.5606	5
A_4	21	1.3	5.6	0.3238	8
A_5	22	1.54	5.27	0.4201	7
A_6	22	1.54	5.63	0.4302	6
A_7	23	2.36	4.0	0.7036	4
A_8	21	2.54	4.0	0.7278	3
A_9	21	2.54	4.36	0.7475	2
A_{10}	23	2.9	4.36	0.8476	1

Table 17: Alternatives from DABC-I.

Altern-ative	TCT	MWT	MWE	Closeness coefficient	Rank
A_1	19	0.72	6.9	0.2462	7
A_2	19	1.27	5.27	0.2515	6
A_3	20	0.90	6.9	0.2704	5
A_4(A_5,A_6)	21	1.63	5.27	0.3907	3
A_7	21	1.63	4.18	0.3604	4
A_8 (A_9)	22	2.72	3.63	0.6813	2
A_{10}	23	3.18	4.0	0.7755	1

Table 18: Alternatives from DABC-II.

Altern-ative	TCT	MWT	MWE	Closeness coefficient	Rank
A_1	20	1.45	4.54	0.1796	8
A_2	21	2.0	4.36	0.3967	6
A_3	21	2.54	4.0	0.6688	5
A_4	21	2.54	4.36	0.6873	4
A_5(A_6)	21	1.63	4.18	0.1983	7
A_7	22	2.72	3.63	0.7564	3
A_8(A_9)	23	2.9	4.3	0.9461	1
A_{10}	22	2.72	4.36	0.8356	2

Table 19: Alternatives from DABC-III.

Again it is a challenging task to find out a proper and suitable chaotic function to well fit to our decision making problem. Researchers used a number of chaotic sequences to tune various parameters in various meta-heuristic optimization algorithms such as particle swarm optimization[61-62], genetic algorithms[63], harmony search[60], imperialist competitive algorithm [64], ant and bee colony optimization [65, 59], firefly algorithm [62] and simulated annealing [66]. Each research in different direction has shown some promise once the right set of chaotic maps is applied. Gandomi and Yang [67] founds sinusoidal map is the most suitable for the bat algorithm to replace with loudness and pulse rate

Altern -ative	TCT	MWT	MWE	Closeness coeff	Rank
A_1	92	5.21	14.43	0.2960	10
A_2	96	6.11	14.6	0.3667	9
A_3	98	7.49	11.94	0.4132	8
A_4	100	7.25	12.84	0.4320	7
A_5	104	8.68	14.0	0.6617	3
A_6	106	8.84	17.8	0.8070	1
A_7	106	9.7	12.52	0.6873	2
A_8	107	10.17	9.0	0.5862	5
A_9	106	9.7	9.11	0.5640	6
A_{10}	107	9.66	11.9	0.6613	4

Table 20: Alternatives from DABC-I.

Altern -ative	TCT	MWT	MWE	Closeness coefficient	Rank
A_1	94	6.07	14.82	0.2946	9
A_2	95	5.3	20.49	0.4250	6
A_3	95	5.74	13.64	0.2352	10
A_4	95	6.8	13.6	0.3047	8
A_5	100	7.5	13.47	0.3839	7
A_6	109	7.78	15.82	0.5232	3
A_7	105	8.7	10.6	0.4484	4
A_8	104	9.17	8.76	0.4466	5
A_9	111	11.13	9.45	0.5918	1
A_{10}	107	9.66	11.9	0.5668	2

Table 21: Alternatives from DABC-II.

respectively. Similarly Gandomi et al. [61] have experimented twelve different chaotic maps to tune the major parameters of PSO. They revealed sinusoidal map and singer map perform better result in comparison to the rest. Talatahari et al.[64] proposed in a novel chaotic improved imperialist competitive algorithm by investing seven different chaotic maps and sinusoidal and logistic maps are found as the best choices. Also in Gandomi et al. [62] experimentally revealed sinusoidal map and gauss maps are the best performed chao to be adopted for firefly algorithm. Most experimental results proved sinusoidal as a common better performing random generator. By watching the efficiency of sinusoidal map, we have used the same to find out the random numbers in the TOPSIS weight assignment procedure. Again it is important for the decision maker to maintain the priority level of all criterions. To cope up with this we have sorted the random numbers and assigned them to the respective criterions.

Decision results

To finalize the decision results we have generated a set of three chaotic numbers using sinusoidal map and sorted them to represent different criterion weights. With respect to each decision matrix we have allotted the same criterion weight, in a preference order i.e., {0.5, 0.3, 0.2}. Here we have assumed of TCT with highest preference,

Altern -ative	TCT	MWT	MW E	Closen ess coeffic ient	Rank
A_1	94	6.07	14.82	0.2946	9
A_2	95	5.3	20.49	0.4250	6
A_3	95	5.74	13.64	0.2352	10
A_4	95	6.8	13.6	0.3047	8
A_5	100	7.5	13.47	0.3839	7
A_6	109	7.78	15.82	0.5232	3
A_7	105	8.7	10.6	0.4484	4
A_8	104	9.17	8.76	0.4466	5
A_9	111	11.13	9.45	0.5918	1
A_{10}	107	9.66	11.9	0.5668	2
A_1	93	5.7	13.6	0.2540	9
A_2	94	5.4	14.47	0.2760	8
A_3(A_4)	95	5.54	19.31	0.4281	6
A_5	96	5.64	19.0	0.4291	5
A_6	99	6.2	19.17	0.4806	3
A_7	97	7.52	11.19	0.3873	7
A_8	101	8.41	8.43	0.4526	4
A_9	106	9.96	9.19	0.6012	1
A_{10}	106	9.3	9.9	0.5810	2

Table 22: Alternatives from DABC-III.

Altern -ative	TCT	MWT	MWE	Closenes s coeff	Rank
A_1	21	2.0	4.36	0.7498	1
A_2	20	0.9	5.9	0.2380	7
A_3	19	0.72	6.9	0.2529	6
A_4	21	1.63	5.27	0.6696	2
A_5	20	0.90	6.9	0.3065	5
A_6	21	1.63	4.18	0.6129	4
A_7	22	1.54	5.63	0.6554	3

Table 23: Alternatives from MOABC (Small-sized).

Altern-ative	TCT	MWT	MWE	Closeness coefficient	Rank
A_1	104	8.96	10.27	0.1132	6
A_2	97	7.54	10.60	0.1098	7
A_3	99	7.19	12.86	0.1441	4
A_4	106	9.8	9.47	0.8105	1
A_5	101	6.54	21.29	0.2516	2
A_6	109	10.39	5.03	0.0744	10
A_7	99	7.37	15.01	0.1750	3
A_8	107	10.15	8.76	0.1015	8
A_9	99	7.19	11.21	0.1192	5
A_{10}	111	11.05	4.29	0.0847	9

Table 24: Alternatives from MOABC (Large-sized).

then MWT and lastly MWE. The decision matrices are nothing but various resulted non-dominated sequences of TCT, MWT and MWE. For every individual decision matrix we have generated the closeness coefficient value w.r.t both the ideal solutions and so as the ranks. Firstly we have calculated the ranks of all the alternatives generated from DABC-I, DABC-II and DABC-III for the small-size dataset followed by the large one. Lastly the alternatives from MOABC are evaluated in the same sequence.

DABC (Small-sized)

Table 17 represents the alternatives generated from DABC-I. 10 alternatives are evaluated with the proposed chao-TOPSIS procedure and the ranks are presented. Alternative A_{10} is having highest closeness coefficient value than all, hence is chosen as rank 1 alternative for the decision maker.

The non-dominated sequences of DABC-II (Table 18) are having some of the repeating sequences; hence they are treated as one single alternative. Alternatives A_4, A_5, A_6 are the same sequences and that of alternatives A_8 and A_9. These repeating sequences are the result of selecting the proportionately best fitness values from each objective function and application of local search algorithms repeatedly to a small sized data set. Hence altogether we have evaluated 7 sequences and the last alternative A_{10} is the best ranked.

Similarly table 19 contains the resulting sequences of DABC-III. Out of 10 sequences two pairs ((A_5=A_6) and (A_8=A_9)) are repeated sequences. Hence 8 sequences are evaluated against the three objectives using chaotic-TOPSIS. The calculation shows, the seventh sequence i.e. A_8 (or A_9) is having rank 1.

DABC (Large-size Dataset)

The large-sized synthetic dataset has again 3 decision matrices from DABC-I, DABC-II and DABC-III to be evaluated. Table 20 contains the decision matrix resulted from DABC-I. The 10 different alternatives (sequences) are having different closeness coefficient values and A_6 is the highest ranked alternative.

The following decision matrix of Table 21 is the resulted optimized sequence of DABC-II for the large input data. Each alternative are processed to check the best set of functional values from the calculated closeness coefficient value. Here alternative A_9 is found to be superior one.

The non-dominated sequence of DABC-III is represented as the decision matrix in Table 22 with 10 alternatives. Two alternatives A_3 and $A_4$are having same sequences. Hence altogether 9 different sequences are processed and according to chaotic-TOPSIS, A_9 is the best one to be chosen by the decision maker.

MOABC

Table 23 contains the non-dominated sequence of MOABC for the small sized data set. It is consisting of 7 alternatives and chaotic-TOPSIS valuates A_1 as the suitable alternative for the decision maker among all.

The non-dominated sequences of MOABC for the large data input is consisting of 10 sequences and are represented in Table 24. After checking the closeness coefficient values A_4 is found as the best alternative among all.

The use of generating random numbers using different chaotic functions has been one of the remarkable techniques to tune the parameters in various algorithms in many fields, and this has become an active research topic in the recent optimization literature. By watching its advantage, we have introduced the concept of chaotic map to the standard TOPSIS, and have checked for the best alternative among a set of non-dominated solutions. The decision makers will be definitely confident enough to take a right decision among the conflicting ones using the approach.

8 Conclusions and future research

The DABC and MOABC algorithms were coded and applied to the multiple instances of dataset ranging from 3 jobs with 3 machines to 10 jobs and 9 machines. In this paper, we considered the MOPFSSP under the multiple (three) criteria. The DABC algorithm is hybridized with a variant of iterated greedy algorithms employing a local search procedure based on *insertion* (), *swap* () and *destruct- construct* () neighborhood structures. In addition, we also presented an extended version of ABC algorithm to the proposed MOABC algorithm employed through a particular local search procedure with reduced complexity. Our proposal is having a significant application of DABC to check the time complexity of different local search procedures. Hence, we are motivated to use simple *swap* () operation in local search procedure in the MOABC algorithm. The performances of both the proposed algorithms were tested by using different instances of datasets and it has been shown that the performances of both DABC and MOABC algorithms are highly competitive with the best performing existing literature. Also we have extended our work to optimize the non-dominated solutions to a single optimal solution using chaotic-TOPSIS method to derive the optimal decision in the field of MCDM. The proposed approach will definitely help the decision makers to solve various MCDM problems in future. Further the problem of FSSP can be extended with no-wait flowshop, blocking flowshop and no-idle flowshop, etc. Apart from three criteria we may practically have a many objective (MaO) PFSSP, which will obviously increase the number of non-dominated solutions in the search space. We may further work to find other effective ways to make a right decision for the decision makers to reach at a definite goal.

9 Acknowledgment

The data applied in this study is consisting of synthetic datasets ranges from small-size to large-sized ones.

10 References

[1] K R Baker. Introduction to sequencing and scheduling. John Wiley & Sons Inc. New York,1974.

[2] S. M. Johnson. Optimal two- and three-stage production schedules with setup times included. Naval Research Logistics Quarterly, 1(1): 61–68, 1954. https://doi.org/10.1002/nav.3800010110

[3] D. S. Palmer. Sequencing jobs through a multi-stage process in the minimum total time-a quick method of obtaining a near optimum. Operations Research Society, 16(1): 101–107, 1965. https://doi.org/10.2307/3006688

[4] Jatinder N. D. Gupta. A functional heuristic algorithm for the flow shop scheduling problem. Operations Research Quarterly, 22(1)39–47, 1971. https://doi.org/10.2307/3008015

[5] Herbert G. Campbell, Richard A. Dudek and Milton L. Smith. A heuristic algorithm for the n-job, m-machine sequencing problem. Management Science, 16(10): .630–637, 1970. https://doi.org/10.1287/mnsc.16.10.b630

[6] David G. Dannenbring. An evaluation of flow shop sequencing heuristics. Management Science, 23(11):1174–1182, 1977. https://doi.org/10.1287/mnsc.23.11.1174

[7] Nawaz, Muhammad, Enscore Jr, E Emory and Ham, Inyong. A heuristic for the m-machine n-job flow shop sequencing problem. Omega, 11(1): 91–95, 1983.

[8] S P Bansal. Minimizing the sum of completion times of n-jobs over m-machines in a flowshop: a branch and bound approach. AIIE Transactions, 9(3):306–311, 1977. https://doi.org/10.1080/05695557708975160

[9] Chia-Shin Chung, James Flynn and Omer Kirca. A branch and bound algorithm to minimize the total flow time for m-machine permutation flowshop problems. International Journal of Production Economics, 79(3): 185–196, 2002. https://doi.org/10.1016/s0925-5273(02)00234-7

[10] Edward Ignall and Linus Schrage. Application of the branch and bound technique to some flow-shop scheduling problems. Operations Research, 13(3): 400–412, 1965. https://doi.org/10.1287/opre.13.3.400

[11] S.L. van de Velde. Minimizing the sum of the job completion times in the two-machine flow shop by Lagrangian relaxation. Annals of Operations Research, 26(1-4):257–268, 1990. https://doi.org/10.1007/bf03500931

[12] Willem J. Selen and David D. Hott. A mixed-integer goal-programming formulation of the standart flow-shop scheduling problem. Operation Research Society, 37(12) :1121–1128, 1986. https://doi.org/10.2307/2582302

[13] J. M. Wilson. Alternative formulations of a flowshop scheduling problem. Operation Research Society, 40(4): 395–399, 1989. https://doi.org/10.1057/jors.1989.58

[14] Richard L. Daniels and Robert J. Chambers. Multi-objective flowshop scheduling. Naval Research Logistics, 37(6): 981–995, 1990. https://doi.org/10.1002/1520-6750(199012)37:6%3C981::aid-nav3220370617%3E3.0.co;2-h

[15] Chandrasekharan Rajendran. Heuristics for scheduling in flowshop with multiple objectives. European Journal of Operation Research, 82(3):540–555, 1995. https://doi.org/10.1016/0377-2217(93)e0212-g

[16] Neelam Tyagi, R. P. Tripathi and A. B. Chandramouli. Three Machines Flowshop Scheduling Model with Bicriterion Objective Function, 9(48): 1-14, 2016. https://doi.org/10.17485/ijst/2016/v9i48/103012

[17] S.M. Mousavi, I. Mahdavi, J. Rezaeian and M. Zandieh. Bi-objective scheduling for the re-entrant hybrid flow shop with learning effect and setup times. Scientia Iranica, 25(4): 2233-2253, 2017. https://doi.org/10.24200/sci.2017.4451

[18] Karunakaran Chakravarthy and Chandrasekharan Rajendran. A heuristic for scheduling in flowshop with bi-criteria of makespan and maximum tardiness minimization. Production Planning & Control, 10(7): 707–714, 1999. https://doi.org/10.1080/095372899232777

[19] R.K. Suresh and K.M. Mohanasundaram. Pareto archived simulated annealing for permutation flow shop scheduling with multiple objective. Proceedings of the 2004 IEEE Conference on Cybernetics and Intelligent Systems, 712–717, 2004. https://doi.org/10.1109/iccis.2004.1460675

[20] T.K. Varadharajan and Chandrasekharan Rajendran. A multi-objective simulated-annealing algorithm for scheduling in flowshops to minimize the makespan and total flowtime of jobs. Europian Journal of Operation Research, 167(3): 772–795, 2005. https://doi.org/10.1016/j.ejor.2004.07.020

[21] B. Shahul Hamid Khan and Kannan Govindan. Multi-objective simulated annealing algorithm for permutation flow shop scheduling problem. International Journal of Advanced Operations Management, 3(1):88–100, 2011. https://doi.org/10.1504/ijaom.2011.040661

[22] T. Loukil, J. Teghem and D. Tuyttens. Solving multi-objective production scheduling problems using metaheuristics. European Journal Operation Research, 161(1):42–61, 2005. https://doi.org/10.1016/j.ejor.2003.08.029

[23] Xiangtao Li and Shijing Ma. Multi-objective memetic search algorithm for multi-objective permutation flow shop scheduling problem. IEEE Access, 4: 2154-2165, 2016. https://doi.org/10.1109/access.2016.2565622

[24] Fuyu Yuan, Xin Xu and Minghao Yin. A novel fuzzy model for multi-objective permutation flow shop scheduling problem with fuzzy processing time. Advances in Mechanical Engineering, 11(4):1–9, 2019. https://doi.org/10.1177/1687814019843699

[25] S. Chandrasekaran, S. G. Ponnambalam, R. K. Suresh and N. Vijayakumar. Multi-objective particle swarm optimization algorithm for scheduling in flowshops to minimize makespan, total flowtime and completion time variance. IEEE Congress on Evolutionary Computation, 4012–4018, 2007. https://doi.org/10.1109/cec.2007.4424994

[26] Vincent T'kindt, Nicolas Monmarché, Fabrice Tercinet and Daniel Laügt. An ant colony optimization algorithm to solve a 2-machine bicriteria flowshop scheduling problem. European Journal of Operation Research, 142(2):250–257, (2002). https://doi.org/10.1016/s0377-2217(02)00265-5

[27] Betul Yagmahan and Mehmet Mutlu Yenisey. Ant colony optimization for multi-objective flow shop scheduling problem. Computers and Industrial Engineering, 54(3):411–420, 2008. https://doi.org/10.1016/j.cie.2007.08.003

[28] B.M.T. Lin, C.Y. Lu, S.J. Shyu and C.Y. Tsai. Development of new features of ant colony optimization for flowshop scheduling. International Journal of Production Economics, 112(2) :742–755, 2008. https://doi.org/10.1016/j.ijpe.2007.06.007

[29] M. Ziaee, S.J. Sadjadi, J.L. Bouquard. An Ant Colony Algorithm for the Flowshop Scheduling Problem. Journal of Applied Sciences. 8(21): 3938–3944, 2008. https://doi.org/10.3923/jas.2008.3938.3944

[30] Dervis Karaboga. An idea based on honey bee swarm for numerical optimization. Technical Report TR06. Computer Engineering Department. Erciyes University. Turkey, 2005.

[31] Dervis Karaboga and Bahriye Basturk. A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm. Journal of Global Optimization, 39(3):459–471, 2007. https://doi.org/10.1007/s10898-007-9149-x

[32] Dervis Karaboga and B. Basturk. On the performance of artificial bee colony (ABC) algorithm. Applied Soft Computing, 8(1): 687–697, 2008. https://doi.org/10.1016/j.asoc.2007.05.007

[33] Nurhan Karaboga. A new design method based on artificial bee colony algorithm for digital IIR filters. Journal of the Franklin Institute, 346 (4): 328–348, 2009. https://doi.org/10.1016/j.jfranklin.2008.11.003

[34] Dervis Karaboga and Bahriye Akay. A comparative study of artificial bee colony algorithm. Applied Mathematics and Computation, 214 (1):108-132, 2009. https://doi.org/10.1016/j.amc.2009.03.090

[35] Dervis Karaboga and Bahriye Akay. A survey: Algorithms simulating bee swarm intelligence.

Artificial Intelligence Review, 31(1-4):68-85, 2009. https://doi.org/10.1007/s10462-009-9127-4

[36] Sangeeta Sharma and Pawan Bhambu. Artificial bee colony algorithm: A survey. International Journal of Computer Applications, 149(4):11-19, 2016.
https://doi.org/10.5120/ijca2016911384.

[37] Pradeep Kumar Singh. A systematic review on artificial bee colony optimization technique. International Journal of Control Theory and Applications, 9(11): 5487-5500, 2016.

[38] Tuğçe Anılan, Ergun Uzlu, Murat Kankal and Omer Yuksek. The estimation of flood quantiles in ungauged sites using teaching-learning based optimization and artifcial bee colony algorithms. Scientia Iranica, 2017.
https://doi.org/10.24200/sci.2017.4185

[39] Valery Tereshko. Reaction-diffusion model of a honeybee colony's foraging behaviour, in: PPSN VI. Proceedings of the Sixth International Conference on Parallel Problem Solving from Nature, Springer-Verlag, 807–816, 2000.
https://doi.org/10.1007/3-540-45356-3_79

[40] Su-jun Zhang and Xing-sheng Gu. An effective discrete artificial bee colony algorithm for flow shop scheduling problem with intermediate buffers. Journal of Central South University, 22(9):3471−3484, 2015.
https://doi.org/10.1007/s11771-015-2887-x

[41] Hoon-Shik Woo and Dong-Soon Yim. A heuristic algorithm for mean flowtime objective in flowshop scheduling. Computers and Operations Research, 25(3):175–182.
https://doi.org/10.1016/s0305-0548(97)00050-6

[42] I. Kassabalidis, M.A. El-Sharkawi, R.J. Marks, P. Arabshahi and A.A. Gray. Swarm intelligence for routing in communication networks. Global Telecommunications Conference, 3613–3617, 2001.
https://doi.org/10.1109/glocom.2001.966355

[43] M. Fatih Tasgetiren, Quan-Ke Pan, P.N. Suganthan and Angela H-L Chen. A discrete artificial bee colony algorithm for the total flowtime minimization in permutation flowshops. Information Sciences, 181(16): 3459–3475, 2011. https://doi.org/10.1016/j.ins.2011.04.018

[44] Bertrand Mareschal. Weight stability intervals in multicriteria decision aid. Europian Journal of Operation Research, 33(1) :54–64,1988.
https://doi.org/10.1016/0377-2217(88)90254-8

[45] Dervis Karaboga, Beyza Gorkemli, Celal Ozturk and Nurhan Karaboga. A comprehensive survey: Artificial bee colony (ABC) algorithm and applications. Artificial Intelligence Review, 42(1): 21-57, 2014.
https://doi.org/10.1007/s10462-012-9328-0

[46] E. Taillard, E. Benchmarks for basic scheduling problems. European Journal of Operational Research, 64(2):278–285, 1993.
https://doi.org/10.1016/0377-2217(93)90182-m

[47] Peng Wang, Yang Li, Yong-Hu Wang and Zhou-Quan Zhu. A new method Based on TOPSIS and Response Surface Method for MCDM problems with interval numbers. Mathematical Problems in Engineering. Article ID 938535, 2015:1-11, 2015. https://doi.org/10.1155/2015/938535

[48] Elissa Nadia Madi, Jonathan M. Garibaldi and Christian Wagner. An exploration of issues and limitations in current methods of TOPSIS and fuzzy TOPSIS. *IEEE International Conference on Fuzzy Systems*, 2098-2105, 2016.
https://doi.org/10.1109/fuzz-ieee.2016.7737950

[49] Alireza Sotoudeh-Anvari, Seyed Jafar Sadjadi, Seyed Mohammad Hadji Molana and Soheil Sadi-Nezhad. A stochastic multi-objective model based on the classical optimal search model for searching for the people who are lost in response stage of earthquake. Scientia Iranica, 26(3):1842:1864, 2019.
https://doi.org/10.24200/sci.2018.20226

[50] Edmundas Kazimieras Zavadskas, Abbas Mardani, Zenonas Turskis, Ahmad Jusoh and Khalil MD Nor. Development of TOPSIS method to solve complicated decision-making problems: An overview on developments from 2000 to 2015. International Journal of Information Technology & Decision Making, 15 (3): 1-38, 2016.
https://doi.org/10.1142/s0219622016300019

[51] Bing Wu, Likang Zong, Xinping Yan and C. Guedes Soares. Incorporating evidential reasoning and TOPSIS into group decision-making under uncertainty for handling ship without command. Ocean Engineering,164: 590-603, 2018.
https://doi.org/10.1016/j.oceaneng.2018.06.054

[52] Ching-Lai Hwang and Kwangsun Yoon. Multiple Attribute Decision Making. Springer-Verlag, Berlin, 58-191, 1981.
https://doi.org/10.1007/978-3-642-48318-9_3

[53] Sheldon M. Belenson and Kailash C. Kapur. An algorithm for solving multi-criterion linear programming problems with examples. Operational Research Quarterly, 24(1): 65-77, 1973.
https://doi.org/10.2307/3008036

[54] Milan Zelany. A concept of compromise solutions and the method of the displaced ideal' Computers and Operations Research, 1(3-4): 479-496,1974. https://doi.org/10.1016/0305-0548(74)90064-1

[55] Gyutai Kim, Chan S Park and K.Paul Yoon. Identifying investment opportunities for advanced manufacturing systems with comparative-integrated performance measurement. International Journal of Production Economics, 50(1): 23-33, 1997.
https://doi.org/10.1016/s0925-5273(97)00014-5

[56] Steven Cheng, Christine W. Chan and Guo H. Huang. Using multiple criteria decision analysis for supporting decision of solid waste management. Journal of Environmental Science and Health. Part A, 37(6): 975-990, 2002.
https://doi.org/10.1081/ese-120004517

[57] Stelios H. Zanakis, Anthony Solomon, Nicole Wishart and Sandipa Dublish. Multi-attribute

decision making: A simulation comparison of selection methods. European Journal of Operational Research, 107(3):507–529, 1998.
https://doi.org/10.1016/s0377-2217(97)00147-1

[58] Leandro dos Santos Coelho and Viviana Cocco. Use of chaotic sequences in a biologically inspired algorithm for engineering design and optimization. Expert Systems with Applications, 34(3):1905-1913, 2008.
https://doi.org/10.1016/j.eswa.2007.02.002

[59] Bilal Altlas. Chaotic bee colony algorithms for global numerical optimization. Expert systems with applications, 37(8): 5682-5687, 2010.
https://doi.org/10.1016/j.eswa.2010.02.042

[60] Bilal Altlas. Chaotic harmony search algorithms. Applied mathematics and computation, 216(9): 2687–2699, 2010.
https://doi.org/10.1016/j.amc.2010.03.114

[61] Amir Hossein Gandomi, Gun Jin Yun, Xin-She Yang and Siamak Talatahari. Chaos-enhanced accelerated particle swarm algorithm. Communications in Nonlinear Science and Numerical Simulation, 18(2):327–340, 2013. https://doi.org/10.1016/j.cnsns.2012.07.017

[62] Amir Hossein Gandomi, X.-S. Yang, S. Talatahari and A.H. Alavi. Firefly algorithm with chaos. Communications in Nonlinear Science and Numerical Simulation, 18(1): 89–98, 2013.
https://doi.org/10.1016/j.cnsns.2012.06.009

[63] Golnar Gharooni-fard, Fahime Moein-darbari, Hossein Deldari and Anahita Morvaridi. Scheduling of scientific workflows using a chaos-genetic algorithm. Procedia Computer Science, 1(1): 1445–1454, 2010.
https://doi.org/10.1016/j.procs.2010.04.160

[64] S. Talatahari, B. Farahmand Azar, R. Sheikholeslami and A.H. Gandomi. Imperialist competitive algorithm combined with chaos for global optimization. Communications in Nonlinear Science and Numerical Simulations, 17(3): 1312–1319, 2012.
https://doi.org/10.1016/j.cnsns.2011.08.021

[65] Wei Gong and Shoubin Wang. Chaos ant colony optimization and application. 4th Inter-national Conference on Internet Computing for Science and Engineering, 301–303, 2009.
https://doi.org/10.1109/icicse.2009.38

[66] Ji Mingjun and Tang Huanwen. Application of chaos in simulated annealing. Chaos, Solitons & Fractals, 21(4): 933–941, 2004.
https://doi.org/10.1016/j.chaos.2003.12.032

[67] Amir H. Gandomi and Xin-She Yang. Chaotic bat algorithm. Journal of Computational Science, 5(2):224-234, 2014.
https://doi.org/10.1016/j.jocs.2013.10.002

Research on Resource Allocation and Management of Mobile Edge Computing Network

Rui Zhang and Wenyu Shi
Anhui Xinhua University, Hefei, Anhui 230087, China
E-mail: nw2094@163.com

Keywords: edge network, resource allocation, transmitting power, system utility

Received: May 14, 2020

The popularity of mobile Internet makes the application of mobile terminals need more computing resources, and cloud computing enables mobile terminals to handle application tasks that need high computing resources under the premise of maintaining small specifications. However, it is difficult to obtain high-quality low latency services as the mobile Internet edge is far away from the cloud computing center; hence mobile edge computing (MEC) is proposed. This study introduced computing resource allocation methods based on power iteration and system utility, applied them to the mobile edge computing network, and carried out simulation experiments in MATLAB software. The experimental results showed that the network throughput and system utility under two resource allocation methods increased and the average transfer rate decreased with the increase of users in the mobile edge network; under the same number of access users, the edge network based on the system utility allocation method had higher throughput, average transfer rate and system utility.

Povzetek: Robno računalništvo (edge computing) omogoča boljše delovanje mrež, ker podatke v oblaku prestavi na rob mreže. Prispevek se ukvarja z analizo in izdelavo tovrstnih metod.

1 Introduction

The development of mobile Internet technology has further facilitated people's life, and the popularity of mobile terminals such as mobile phones and IPADS has greatly promoted mobile Internet technology [1]. The emergence of cloud computing [2] greatly reduces the requirements of the mobile terminal for computing and storage performance, thus reducing the manufacturing cost. However, even if the mobile Internet has a large coverage, there is only one data center that can provide cloud computing services. When data are transmitted in the nodes of the mobile Internet, there will inevitably be a delay. The further the distance with the data center is, the more serious the delay is. High delay will seriously affect the service reliability of various mobile applications. The emergence of mobile edge computing [3] solves the above problems. Compared with the cloud server in the data center, edge devices are closer to users and have shorter time delay. The combination of cloud server and edge device can provide more reliable services to users on the edge. Liu et al. [4] proposed a deep learning architecture based on the close connection network, transplanted it into the mobile edge algorithm, and found through the simulation that the algorithm had obvious overall efficiency advantage. Zhang et al. [5] proposed the weight based algorithm and mobile prediction based heuristic algorithm for users with certain and uncertain mobile tracks to reduce the network overhead caused by task migration and found through experiments that the two algorithms could effectively reduce the network overhead caused by task migration. In order to solve the joint optimization problem of task caching and offloading in edge cloud computing, Hao et al. [6] proposed an efficient task caching and offloading algorithm based on the alternative iterative algorithm and found through the simulation that the algorithm had lower energy consumption. This study introduced a computing resource allocation method based on power iteration and a computer resource allocation method based on system utility, applied them to the mobile edge computing network, and carried out a simulation experiment on the two computing resource allocation methods in MATLAB software.

2 Mobile edge computing (MEC)

In recent years, the configuration performance of mobile terminals has been greatly improved with the progress of science and technology, mainly reflected in small volume and large amount of computing. However, in order to maintain its mobile convenience, the size of mobile terminal itself must be portable. Unless there is a breakthrough in the existing materials and technologies, the amount of computing must be limited and lower than that of large-scale computing equipment. In the face of today’s huge mobile network applications, mobile terminals with limited computing and energy are gradually difficult to provide good services. The development of cloud computing technology has greatly liberated the computing burden of mobile terminals, but the increase of terminals which are accessed to mobile network for cloud computing has increased the network burden and delay. Cloud computing services usually give priority to meet the service request of mobile terminals near the data center;

therefore terminals on the edge of the network will have network delay.

In order to solve the problem that the service quality of edge network cloud computing reduces due to the excessive access of mobile terminals, mobile edge computing is proposed. The basic structure of MEC system includes the bottom structure, functional

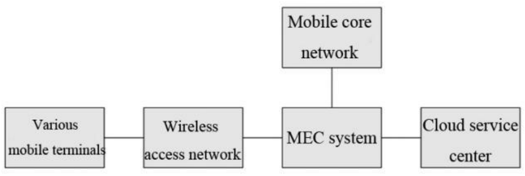


Figure 1: The basic architecture of MEC system.

components and application layer. The bottom structure mainly includes the virtual layer structure and hardware facilities which are used for generating virtual computing resources [7]; the functional components in MEC system play the role of interaction interface between internal and external data, assisting the system to access the mobile network without obstacles; the application layer is the application interface of the system, which is used for interacting information with users. After connecting MEC system to cloud computing service, its basic architecture is shown in Figure 1. The access location of MEC system is between the wireless access network and mobile core network. The wireless access network contains MEC servers, and they constitute the edge cloud and connect with various kinds of mobile terminals through the base station. The mobile core network is the center of the whole mobile Internet, which is used for realizing the large area transmission of communication data. The cloud service center is an important part of providing cloud computing.

The principle of MEC system improving the service quality of cloud computing can be generally described as lowering the task that needs to be executed in the cloud computing center to the edge server. In the traditional cloud computing data interaction, the request is first sent to the base station of wireless access network, and then the base station transmits the request to the core network for data protection. The request of any user follows the above process even though the request is the same. Once the number of access users increases, not only channel resources will be wasted, but also network congestion will cause delay [8]. After adding MEC, cloud computing resources are distributed to MEC server. When users repeat the same request, they can directly obtain resources from MEC server, which greatly improves the speed of service response.

3 Resource management based on power iteration

For the convenience of explanation, it is assumed that the mobile edge network has one base station (BS) and several mobile terminals (MT). MEC server is set in BS, which can directly carry out data interaction. Then all MTs in the edge network can be expressed as $N=\{1,2,3,\ldots,n\}$, all communication channel resources can be expressed as $M=\{1,2,3,\ldots,m\}$, the tasks to be performed by the i-th MT can be expressed as $X_i=(D_i,C_i,T_i^{\max})$, where D_i, C_i and $T_i^{\max}$ are the size of calculation data, the required calculation power and maximum time delay respectively.

If the transmission power is used to control resource allocation, the resource allocation problem of the edge computing network can be expressed as:

objective function is:

$$\max\left(\sum_{i=1}^{N}\sum_{m=1}^{M}a_{i,m}B_m\log_2\frac{\sigma_m^2+I_{i,m}+p_{i,m}g_{i,m}}{\sigma_m^2+I_{i,m}}-R_i^{\min}\right); \quad (1)$$

condition is:

$$\begin{cases}\sum_{m=1}^{M}B_m\log_2\frac{\sigma_m^2+I_{i,m}+p_{i,m}g_{i,m}}{\sigma_m^2+I_{i,m}}\geq R_i^{\min}\\ \sum_{m=1}^{M}a_{i,m}=1\\ \sum_{m\in M}a_{i,m}p_{i,m}\in[0,p_i^{\max}]\\ SNR=\frac{\sum_{m\in M}a_{i,m}p_{i,m}g_{i,m}}{\sigma_m^2+I_{i,m}}\geq\lambda_D\end{cases}, \quad (2)$$

where $a_{i,m}$ indicates whether the i-th MT passes the m-th channel migration tasks or not, 1 for yes, and 0 for no; B_m indicates the bandwidth of the m-th channel, σ_m^2 stands for the Gaussian white noise on channel m [9], $I_{i,m}$ indicates the interference of other MT in channel m to the i-th MT, $p_{i,m}$ indicates the transmitting power of the i-th MT in channel m, $g_{i,m}$ indicates the channel gain of the i-th MT in channel m, $R_i^{\min}$ indicates the minimum transmission rate of the i-th MT in channel m to ensure the transmission quality, SNR indicates the signal to noise ratio [10], and λ_D indicates the threshold of the signal to noise ratio.

It is seen from equation (1) and (2) that the corresponding allocation scheme is optimal when $p_{i,m}$ is optimal, then the solution step of optimal $p_{i,m}$ is:

① Related iteration update parameters including number of iterations t and Lagrange multiplier λ and μ are initialized.

② Let $t=t+1$, indicating one time of iteration, and then the Lagrange multiplier is updated according to the following formula:

$$\begin{cases} \lambda_m(t+1) = \max\left[\lambda_m(t) + \alpha\left(\sum_{m\in M} a_{i,m}p_{i,m} - p_i^{\max}\right), 0\right] \\ \mu_m(t+1) = \max\left[\mu_m(t) + \beta\left(\lambda_D - \frac{\sum_{m\in M} a_{i,m}p_{i,m}g_{i,m}}{\sigma_m^2 + I_{i,m}}\right), 0\right] \end{cases} \quad (3)$$

③ Transmission power $p_{i,m}$ is calculated according to the Lagrange multiplier [11] obtained from the previous update, and the formula is:

$$p_{i,m} = \frac{B_m}{[\mu g_{i.m} - \lambda(I_{i,m} + \sigma_m^2)]\ln 2} - \frac{I_{i,m} + \sigma_m^2}{g_{i.m}} \quad (4)$$

④ $p_{i,m}$ which is calculated after updating and before updating is compared. If the difference between them is smaller than the set threshold value, iteration updating stops, otherwise step ② and ③ repeat. After several times of iterations, optimal $p_{i,m}$ is obtained.

4 Resource management based on system utility

First of all, the model structure of MEC system is, same as above, set as multi-user MT-single BS. The resource allocation of the edge computing network is designed based on the utility function of the system. Then the allocation can be expressed as follows:
objective function is:

$$\max\left(\sum_{i=1}^{N} \rho_i s_i (\gamma_i^T \frac{T_i^l - T_i^r}{T_i^l} + \gamma_i^E \frac{E_i^l - E_i^r}{E_i^l})\right) \quad ; (5)$$

condition is:

$$\begin{cases} s_i = \{0,1\} \\ p_i \in (0, p_{\max}] \\ f_i > 0 \\ \sum_{i\in S} f_i \le f_{\max} \\ \sum_{i\in N} s_i \le M \end{cases} \quad (6)$$

where ρ_i indicates the priority of the i-th MT receiving MEC service, [0,1] , s_i indicates whether terminal equipment I selects task immigration or not, 1 for migrating to MEC server and 0 for processing tasks locally, γ_i^T and γ_i^E indicate the preference of equipment I for improving efficiency and reducing energy consumption respectively [12], i.e., the intention of users to efficiently solve problems and save energy, [0,1], T_i^l and T_i^r indicate the time delay of processing tasks locally and processing tasks by migrating to MEC server respectively, the former depends on the CPU performance of equipment and the latter depends on computation resource f_i allocated by MEC server and the time delay of information transmission, E_i^l and E_i^r indicate the energy consumption of processing tasks locally and processing tasks by migrating to MEC server respectively, the former depends on the power and processing time of equipment and the latter depends on the energy during data transmission, p_i indicates the transmitting power of equipment i, which cannot exceed its maximum transmitting power, S indicates a set of equipment participating in task migration.

Steps to solve objective function (5) are as follows.

① First, the optimal transmission power of the equipment in each mobile edge network was calculated using the dichotomy method [13].

② Then whether task migration is required for each equipment is determined. If necessary, a task migration request is issued; if not, a NULL message is issued.

③ After receiving the message from the device, the equipment is classified according to the following formula:

$$\begin{cases} S_l = \{i \mid \rho_i(\gamma_i^T + \gamma_i^E) - \Delta(i) \le 0\} \\ S_r = \{i \mid \rho_i(\gamma_i^T + \gamma_i^E) - (\Delta(i) + \Delta(i \mid N - S_l - \{i\})) \ge 0\} \\ S_s = N - S_l - S_r \\ \Delta(i) = \frac{\eta_i + \gamma_i p_i}{\log_2(1 + a_i p_i)} + \frac{\tau_i F_i^l}{f_{\max}} \\ \Delta(i \mid N - S_l - \{i\}) = \frac{2\sqrt{\tau_i F_i^l}\sum_{j\in N - S_l - \{i\}}\sqrt{\tau_j F_j^l}}{f_{\max}} \\ \eta_i = \frac{\rho_i \gamma_i^T D_i}{B_m T_i^l} \\ \gamma_i = \frac{\rho_i \gamma_i^E D_i}{B_m T_i^l \zeta} \\ \tau_i = \rho_i \gamma_i^T \end{cases} \quad , (7)$$

where S_l , S_r and S_s are the equipment set of locally processed tasks, the equipment set of task migration and the equipment set to be allocated respectively, $\Delta(i), \Delta(i \mid N - S_l - \{i\})$ are the intermediate quantity for calculating the system marginal utility value in the original migration equipment set after adding equipment i, η_i, γ_i and τ_i are intermediate variables for calculating $\Delta(i), \Delta(i \mid N - S_l - \{i\})$, F_i^l is the working frequency of mobile equipment CPU, a_i is the ratio of channel gain to channel noise power during the transmission of equipment i, and ζ is the working efficiency of transmission power amplifier.

④ Whether the number of equipment in S_r exceeds the total number of channels (K) in the edge network is determined. If it exceeds, then a equipment with the smallest system utility is selected from S_r and moved to S_l . The cycle stops until the number of equipment in S_r does not exceed K . Then S_r is output, and task migration and allocation of computation resources were performed according to S_r .

⑤ If the number of equipment in S_r is smaller than the total number of channels in the edge network, then the equipment with the largest system utility is selected from S_s and added to S_r . S_r does not shrink after adding the new equipment, i.e., the system utility of the set does not decrease. The step repeats until the number of equipment in S_r reaches the largest number of channels in the network or S_r does not extend. After the cycle stops, S_r is output, and the task migration and allocation of computation resources were performed according to S_r .

5 Simulation experiment

5.1 Experimental environment

In this study, the two edge computing network resource allocation methods mentioned above were simulated and analyzed by MATLAB software [14]. The experiment was carried out in a laboratory service. The server configuration included Windows7 system, i7 processor and 16 G memory.

5.2 Experiment setup

In this study, a mobile edge computing network area was established using MATLAB, and the basic parameters are shown in Table 1. In the simulated mobile edge network, the effective coverage of the network was 500 m. In the network, there was a base station and a MEC server. The base station and MEC server were connected. The total channel bandwidth provided by the base station of the edge network was 30 MHz, and its subchannel bandwidth was 1.5 MHz. The maximum number of subchannel that could be provided was 20. The maximum transmission power of MT held by users in the edge network was set as 25 dbm. The computing performance was 1.3 GHz. The user's preference for improving task processing efficiency and equipment energy saving was randomly distributed between 0.3 and 0.8. For the convenience of simulation calculation, the task data size of the MT which was needed to be processed by user was set as 450 kB, and the computing power required to process the task was 1200 Mcycles. The MEC server used for remote processing of tasks had a computing performance (computing resources) of 25 GHz.

The simulation network was was set as described above. Then grouping experiments were carried out according to the number of users in the edge network. There were seven groups in total, 5 users in the 1st group, 10 users in the 2nd group, 15 users in the 3rd group, 20 users in the 4th group, 25 users in the 5th group, 30 users in the 6th group and 35 users in the 7th group. The above two resource management methods are used in each group of simulation experiments. The indicators of resource management method are network throughput, average transmission rate and system utility. Network throughput [15] refers to the number of successfully transmitted data in unit time, while System utility is the effective utilization of network computing resources.

Parameter	Radius of edge network area	Total channel bandwidth	Subchannel bandwidth
Numerical value	500 m	30 MHz	1.5 MHz
Parameter	Channel gain	Channel interference noise	Maximum transmitting power of MT
Numerical value	$128.1+37.5\lg(r)$	-175 dBm/Hz	25 dBm
Parameter	Task size	Resources required to process tasks	CPU performance of MT
Numerical value	450 kB	1200 MCycles	1.3 GHz
Parameter	User preferences (γ_i^T,γ_i^E)	MEC server performance	
Numerical value	0.3~0.8	25 GHz	

Table 1: Basic parameters of simulated mobile edge network.

5.3 Experimental results

Changes of throughput in the edge computing network with the increase of the number of access network users under the two resource management methods are shown in Figure 2. It was seen intuitively from Figure 2 that the network throughput under the two resource management methods was on the rise with the increase of the number of access users in the edge network, and the rise amplitude reduced when the number of users was larger than 20. Generally speaking, the throughput of the edge network based on power iteration was lower than that of the system utility based network under the same number of access

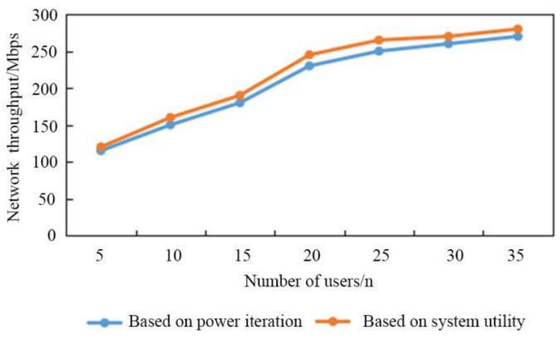


Figure 2: Changes of network throughput with the number of users under the two resource management methods.

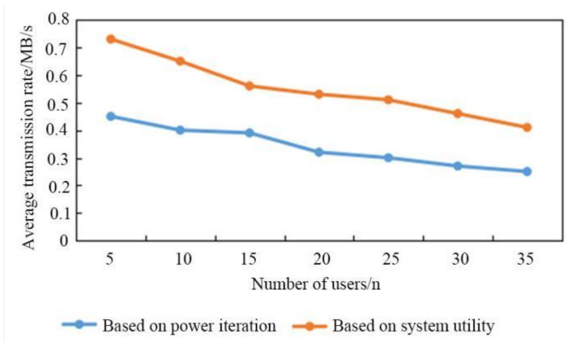


Figure 3: Changes of average transmission rate with number of users under two resource management methods.

users, and the network throughput performance of the system utility based network was better.

Changes of the average transmission rate of the edge network under the two resource allocation methods with the increase of the number of access users are as shown in Figure 3. In the edge network using the power iterative based allocation method, the average transmission rate was 0.45 MB/s when the number of users was 5, 0.4 MB/s when the number of users was 10, 0.39 mb/s when the number of users was 15, 0.32 MB/s when the number of users was 20, 0.30 MB/s when the number of users was 25, 0.27 MB/s when the number of users was 30, and 0.25 MB/s when the number of users was 35. In the edge network using the system utility based allocation method, the average transmission rate was 0.73 mb/s for 5 users, 0.65 mb/s for 10 users, 0.56 mb/s for 15 users, 0.53 mb/s for 20 users, 0.51 mb/s for 25 users, 0.46 mb/s for 30 users, and 0.41 mb/s for 35 users. It was seen from Figure 3 that the average transmission rate of the edge network under the two resource allocation methods decreased with the increase of the access users in the edge network. The reason for the decrease was that the increase of users occupied more subchannels, and moreover the interference between the transmitted signals strengthened. In addition, under the same number of access users, the average transmission rate of the edge network under the system utility based allocation method was higher.

Changes of the system utility of the edge network under the power iteration and system utility with the increase of the number of access users are shown in Figure 4. As shown in Figure 4, when the number of access users was 5, the network system utility of the former was 1.9, and the system utility of the latter was 2.0; when the number of access users was 10, the system utility of the former was 2.8, and the system utility of the latter was 3.0; when the number of access users was 15, the system utility of the former was 3.7, and the system utility of the latter was 4.0; when the number of access users was 20, the system utility of the former was 4.5, and the system utility of the latter was 4.9; when the number of access users was 25, the system utility of the former was 5.0, and the system utility of the latter was 5.5; when the number of access users was 30, the system utility of the former was 5.6, and the system utility of the latter was 6.0; when the number of access users was 35, the system utility of the former was 5.0, and the system utility of the latter was 6.3. It was seen

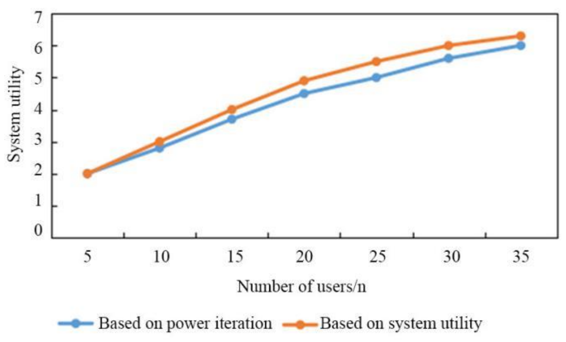


Figure 4: The system utility of the edge network changes with the number of users under two resource management methods.

from Figure 4 that the system utility of the edge network under the two resource allocation methods increased with the increase of the number of access users in the edge network, and the increase amplitude decreased gradually; under the same number of access users, the edge network under the system utility based allocation method had higher system utility.

6 Conclusion

This study introduced computing resource allocation methods based on power iteration and system utility, applied them to the mobile edge computing network, and carried out the simulation experiment on the two methods in MATLAB software. The experimental results are as follows: (1) with the increase of users in the edge network, the network throughput under the two computing resource allocation methods showed an increasing tendency, and the edge network under the system utility based allocation method had higher throughput; (2) the average transmission rate of the edge network decreased with the increase of the number of access users, and the edge network under the system utility based allocation method had a higher average transmission rate; (3) with the increase of the number of users in the edge network, the system utility of the edge network under the two methods of computing resource allocation was on the rise, and the edge network under the system utility based allocation method had higher system utility.

7 References

[1] Al-Shuwaili A, Simeone O (2016). Energy-Efficient Resource Allocation for Mobile Edge Computing-Based Augmented Reality Applications. *IEEE Wireless Communication Letters*, PP(99). https://doi.org/10.1109/LWC.2017.2696539

[2] Ahmed E, Rehmani MH (2016). Mobile Edge Computing: Opportunities, solutions, and challenges. *Future Generation Computer Systems*, 70. https://doi.org/10.1016/j.future.2016.09.015

[3] Paymard P, Mokari N (2019). Resource allocation in PD-NOMA–based mobile edge computing system: Multiuser and multitask priority. *Transactions on Emerging Telecommunications Technologies*, (1), pp. e3631. https://doi.org/10.1002/ett.3631

[4] Liu ZK, Yang XQ, Shen JX (2019). Optimization of multitask parallel mobile edge computing strategy based on deep learning architecture. *Design Automation for Embedded Systems*, (4). https://doi.org/10.1007/s10617-019-09222-5

[5] Zhang F, Liu G, Zhao B, Fu X, Yahyapour R (2018). Reducing the network overhead of user mobility-induced virtual machine migration in mobile edge computing. *Software Practice and Experience*, (3). https://doi.org/10.1002/spe.2642

[6] Hao Y, Chen M, Hu L, Hossain MS, Ghoneim A (2018). Energy Efficient Task Caching and Offloading for Mobile Edge Computing. *IEEE Access*, 6(99), pp. 11365-11373. https://doi.org/10.1109/ACCESS.2018.2805798

[7] Pham QV, Le LB, Chung SH (2019). Mobile Edge Computing with Wireless Backhaul: Joint Task Offloading and Resource Allocation. *IEEE Access*, PP(99), pp. 1-1. https://doi.org/10.1109/access.2018.2883692

[8] Ma LL, Yi SH, Carter N, Li Q (2018). Efficient Live Migration of Edge Services Leveraging Container Layered Storage. *IEEE Transactions on Mobile Computing*, PP(99), pp. 1-1. https://doi.org/10.1109/TMC.2018.2871842

[9] Farris I, Taleb T, Flinck H (2018). Providing ultra-short latency to user-centric 5G applications at the mobile network edge. *Transactions on Emerging Telecommunications Technologies*, 29. https://doi.org/10.1002/ett.3169

[10] Shahzadi S, Iqbal M, Dagiuklas T, Qayyum ZU (2017). Multi-Access Edge Computing: Open issues, Challenges and Future Perspective. *Journal of Cloud Computing Advances Systems & Applications*, 6(1), pp. 30. https://doi.org/10.1186/s13677-017-0097-9

[11] An N, Yoon S, Ha T, Kim Y, Lim H (2018). Seamless Virtualized Controller Migration for Drone Applications. *IEEE Internet Computing*, PP(99), pp. 1-1. https://doi.org/10.1109/MIC.2018.2884670

[12] Zeng DZ, Gu L, Pan SL, Cai JJ, Guo S (2019). Resource Management at the Network Edge: A Deep Reinforcement Learning Approach. *IEEE Network*, 33(3), pp. 26-33. https://doi.org/10.1109/MNET.2019.1800386

[13] Wang Z, Zhao ZW, Min GY (2018). User mobility aware task assignment for Mobile Edge Computing. *Future Generation Computer Systems*, 85. https://doi.org/10.1016/j.future.2018.02.014

[14] Fang WW, Ding S, Li YY (2019). OKRA: optimal task and resource allocation for energy minimization in mobile edge computing systems. *Wireless Networks*, 25(5). https://doi.org/10.1007/s11276-019-02000-y

[15] Yang X, Chen ZY, Li KK (2018). Communication-Constrained Mobile Edge Computing Systems for Wireless Virtual Reality: Scheduling and Tradeoff. *IEEE Access*, 6, pp. 16665-16677. https://doi.org/10.1109/ACCESS.2018.2817288

Research on the Detection of Network Intrusion Prevention with SVM Based Optimization Algorithm

Debing Wang
Anhui Vocational & Technical College of Industry & Trade, Huainan, Anhui 232007, China
E-mail: bdaie2@163.com

Guangyu Xu
Anhui University of Science & Technology, Huainan, Anhui 232001, China

Keywords: support vector machine, intrusion prevention, intrusion detection, whale optimization algorithm

Received: June 6, 2020

Support vector machine (SVM) has a good application in intrusion detection, but its performance needs to be further improved. This study mainly analyzed the SVM optimization algorithm. The principle of SVM was introduced firstly, then SVM was improved using the improved whale optimization algorithm (WOA), the improved WOA (IWOA)-SVM based intrusion detection method was analyzed, and finally experiments were carried out on KDD CUP99 to verify the effectiveness of the algorithm. The results showed that the IWAO-SVM algorithm was more accurate in attack detection; compared with SVM, PSO-SVM and ant colony optimization (ACO)-SVM algorithms, the performance of the IWAO-SVM algorithm was better, the detection rate was 99.89%, the precision ratio was 99.92%, the accuracy rate was 99.86%, and the detection time was 192 s, showing that it had high precision in intrusion detection. The experimental results verify the reliability of the IWAO-SVM algorithm, and it can be promoted and applied in the detection of network intrusion prevention.

Povzetek: Algoritem SVM je bil prilagojen za iskanje napadov v omrežjih.

1 Introduction

With the development of technology and the further popularization of computer, the use of network has become more extensive [1], which not only changes the way people study and work, but also creates great values for economic development. However, the network security problem is becoming more and more prominent [2], means of intrusion attack is becoming more complex and diverse [3], which means greater and stronger harms, and the difficulty of intrusion prevention is becoming higher. In order to deal with all kinds of network intrusion, more and more methods have been applied in intrusion detection. Li et al. [4] studied relevance vector machine (RVM), determined the parameters of RVM using the cloud particle swarm optimization algorithm (CPSO), and verified its high accuracy through experiments. Sangeetha et al. [5] designed a method based on application layer signature. If the signature did not match the rule base, the system would generate an alarm. The method could effectively reduce the false alarm rate and improve the accuracy. Kannan et al. [6] designed an enhanced C4.5 for intrusion detection in hybrid virtual cloud environment and verified the effectiveness of the method through the data set and feeding. Geng et al. [7] designed an intrusion detection algorithm based on rough set and Bayes and combining with weighted average and found through experiments that the resource consumption of the method was low and it was easy to realize and had higher efficiency. This study optimized support vector machine (SVM), applied it to the detection of network intrusion, carried out an experiment on the data set, and compared the performance of different SVM optimization algorithms to verify the effectiveness of the designed optimization algorithm, which provides some theoretical bases for its further application in the actual network and offers more ideas for the design of intrusion detection methods.

2 Network intrusion prevention detection

Network intrusion refers to the behavior of trying to access or destroy a system without authorization to make it unavailable [8]. Detection of network intrusion is to analyze the key information collected from the inside and outside of the computer, such as security log, etc. [9], find out the characteristics that may generate attacks [10], and give responses such as alarm and network outage [11], and its flow is shown in Figure 1.

Firstly, multiple monitoring points are set in the network to collect data such as system log, firewall log, software information and intrusion information as much as possible and comprehensively to ensure the detection effect. Then, the collected data are normalized to reduce the detection error, and the processed data are analyzed

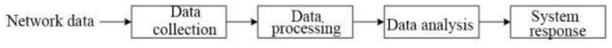


Figure 1: The detection process.

using detection methods to obtain the detection results. Finally, the system makes response to defend according to the detection results.

3 Detection method combined with SVM optimization algorithm

3.1 SVM algorithm

SVM is a machine learning algorithm [12], which has advantages of strong generalization ability, learning ability and applicability. Its classification idea is that two separate categories are on both sides of the hyperplane and have as large an interval as possible (Figure 2).

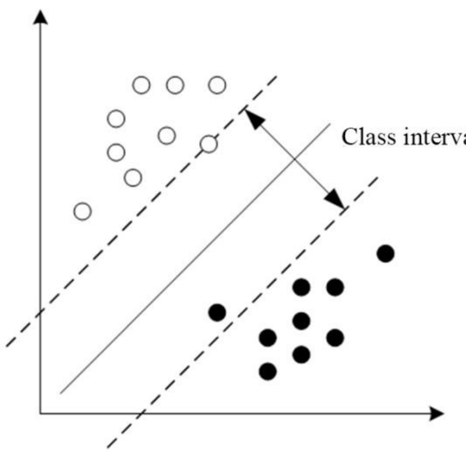


Figure 2: The principle of SVM.

If there is a dataset, $(x_1,y_1),(x_2,y_2),\cdots,(x_i,y_i), i=1,2,\cdots,n, x\in R$, $y\in\{+1,-1\}$, and the hyperplane of its classification can be written as: $y = wx_i + b$, where w stands for weight and b stands for the threshold value. To find the optimal classification plane, the constraints can be written as:

$$\min\frac{\|w\|^2}{2}$$
$$s.t. y_i(wx_i+b)\ge 1 \text{.(1)}$$

In order to improve the modeling speed, slack variable λ is introduced, then:

$$\min\frac{\|w\|^2}{2}+C\sum_{i=1}^{n}\lambda_i$$
$$s.t. y_i(wx_i+b)\ge 1-\lambda_i, \lambda_i\ge 0 \text{,(2)}$$

where C refers to the penalty factor, and then the Lagrange method is introduced to transform it into a dual problem:

$$\max\sum_{i=1}^{n}a_i-\frac{1}{2}\sum_{i,j=1}^{n}a_ia_jy_iy_jk(x_i,x_j) \text{,(3)}$$

where a_i is a Lagrange multiplier and $k(x_i,x_j)$ is a kernel function. The constraint is $\sum_{i=1}^{n}a_iy_i=0, 0\le a_i\le C$. The final classification function can be written as:

$$f(x)=\text{sgn}\left(\sum_{i=1}^{n}a_iy_ik(x_i,x_j)+b\right) \text{.(4)}$$

The kernel function used in this study is RBF kernel function, and the formula is as follows:

$$k(x_i,x_j)=\exp\left(-\frac{\|x_i-x_j\|^2}{r^2}\right) \text{,(5)}$$

where r is a nuclear parameter.

3.2 SVM optimization algorithm

In SVM, penalty factor C and nuclear parameter r has a great impact on the final allocation performance. In order to be able to get optimal values of C and r, the whale optimization algorithm (WOA) [13] was used to obtain the optimal value of parameters in this study, and SVM was optimized. WOA is an optimization algorithm based on the simulation of whale hunting behavior. It is easy to operate and implement, but it also has the problem of slow convergence speed. Therefore, inertia weight σ was introduced to obtain an improved WOA (IWOA).

Suppose that the population size of whales is N, the position of the i -th whale in the d -th space is $X_i=(x_i^1,x_i^2,\cdots,x_i^d)$, $i=1,2,\cdots,N$, and the position of the prey of whale is the optimal solution of problem. In the process of surrounding prey, the formula of the position updating of whale can be written as:

$$X(t+1)=\sigma X_i(t)-A|CX_i(t)-X(t)| \text{, (6)}$$

where t stands for the times of iterations, σ is an inertia weight, $\sigma=\sigma_{\max}-(\sigma_{\max}-\sigma_{\min})\left(\frac{t}{t_{\max}}\right)^{1/t}$, A and C are coefficient vectors, $A=2ar_1-a$, $C=2r_2$, $a=2-\frac{2t}{t_{\max}}$, and r_1 and r_1 are random quantity in $[0,1]$.

The hunting strategy of whales is called bubble-net [14], which means generating bubbles through the spiral path to surround the prey. This process can be expressed as follows:

$$X(t+1)=D'e^{bz}\cos(2\pi z)+\sigma X_i(t) \text{, (7)}$$

where $D'=|X_i(t)-X(t)|$, b stands for the constant defining the spiral shape, and z is a random number in $[-1,1]$.

In addition to bubble-net, whales also conduct random search, which can be expressed as:

$$X(t+1)=\sigma X_{rand}(t)-A|CX_{rand}(t)-X(t)| \text{, (8)}$$

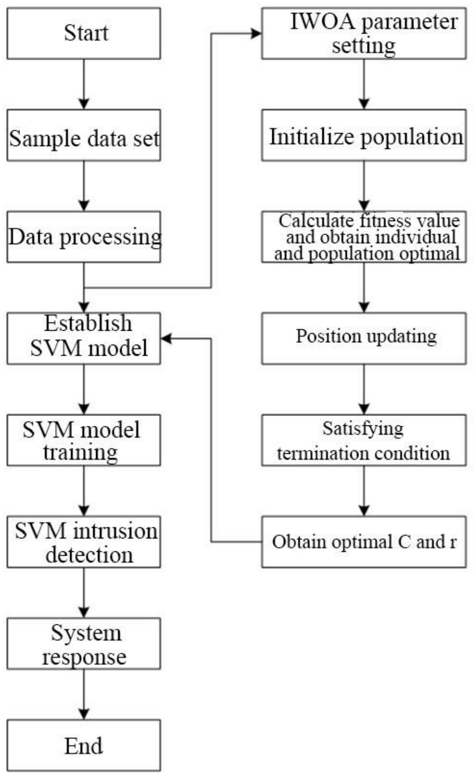


Figure 3: The flow of the intrusion detection algorithm.

where X_{rand} refers to a randomly selected whale position.

3.3 IWOA-SVM intrusion detection algorithm

After optimization with IWOA, the flow of the IWOA-SVM algorithm is shown in Figure 3.

The specific steps of the algorithm are as follows. For the collected sample data set, after preprocessing, the parameters of IWOA are set, and parameter C and r which need optimization in SVM are taken as whale individuals. The population is initialized, and then the fitness value of the individual is calculated to obtain the optimal value of the individual and population. Then, the location is updated by IWOA to obtain new solutions until the termination conditions are met, and optimal value C and r are obtained and regarded as the parameters of SVM. The SVM model is established. After training, the model is tested using the testing samples. Finally, the system responds according to the test results.

4 Experimental results

4.1 Experimental environment and data set

The experiment was carried out on Linux operating system, with Intel Core i7 CPU@2.40GHz, 8 GB memory, and Python language. The size of the IWOA population was 20, $t_{\max}$ was 50, $\sigma_{\min}$ was 0.3, and $\sigma_{\max}$ was 0.9. The experimental data set was KDD CUP99, including probe, DOS, U2R and R2L in addition to Normal. As KDD CUP99 is too large, only a part of data was randomly selected in this study. There were 3500 normal data, 8260 attack data in the training set; there were 1500 normal data and 3540 attack data in the testing set, as shown in Table 1.

Category	Training set	Testing set
Normal	3500	1500
Probe	2100	900
DOS	4900	2100
U2R	700	300
R2L	560	240

Table 1: Experimental data set.

		Detection result	
		Attack data	Normal data
Actual condition	Attack data	A	B
	Normal data	C	D

Table 2: Confusion matrix.

	Normal	Probe	DOS	U2R	R2L
Normal	1497	1	2	0	0
Probe	0	899	1	0	0
DOS	0	1	2198	1	0
U2R	0	0	1	299	0
R2L	0	0	0	0	240

Table 3: Confusion matrix results of the IWAO-SVM algorithm.

4.2 Evaluation index

The detection algorithm was evaluated using the confusion matrix, as shown in Table 2.

In Table 2, A represents that attack data is correctly judged as attack data; B represents that normal data is misjudged as attack data, C represents that attack data is misjudged as normal data, and D represents that normal data is correctly judged as normal data.

(1) Detection rate = A/(A+B)
(2) Precision ratio = A/(A+C)
(3) Accuracy = (A+D) / (A+B+C+D)

4.3 Experimental results

In order to verify the detection effect of the IWOA-SVM algorithm, it was compared with SVM, particle swarm optimization-SVM (PSO-SVM) [15] and ant colony optimization-SVM (ACO-SVM) algorithms [16]. The confusion matrix result of the IWAO-SVM algorithm is shown in Table 3, and the result comparison between different algorithms is shown in Table 4.

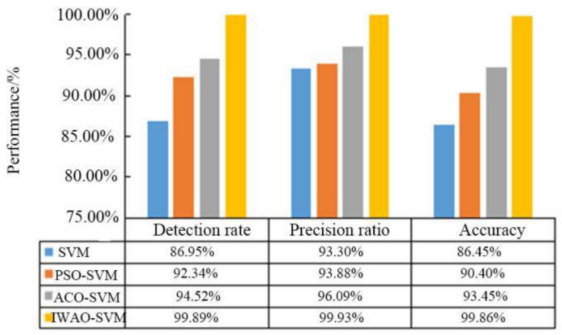
The four numbers separated by slashes in Table 4 represent the results of SVM, PSO-SVM, ACO-SVM and IWOA-SVM algorithms respectively. According to the

			Detection result	
			Attack data	Normal data
Actual condition	Attack data	3540	3320/3478/3486/3536	220/62/54/4
	Normal data	1500	134/38/12/3	1366/1462/1488/1497

Table 4: Comparison results of different algorithms.

	SVM	PSO-SVM	ACO-SVM	IWAO-SVM
Detection time/s	189	197	198	192

Table 5: Comparison of testing time.



	Detection rate	Precision ratio	Accuracy
SVM	86.95%	93.30%	86.45%
PSO-SVM	92.34%	93.88%	90.40%
ACO-SVM	94.52%	96.09%	93.45%
IWAO-SVM	99.89%	99.93%	99.86%

Figure 4: Comparison of the performance between different algorithms.

data in Table 4, the detection rate of the algorithms was calculated, and the results are shown in Figure 4.

According to Figure 4, first of all, the detection rate of the PSO, ACO and IWAO optimized SVM algorithms was 6.21%, 8.71% and 14.88% higher than that of SVM, respectively. It was seen that the detection rate of the IWAO-SVM algorithm significantly improved; the precision ratio of the four algorithms were all over 90%, of which the IWAO-SVM algorithm was the highest, 99.92%; from the perspective of accuracy rate, the optimization by PSO and ACO improved the accuracy rate of the SVM algorithm, but not as significant as IWAO; the accuracy of the IWAO-SVM algorithm was 15.51% higher than that of the SVM algorithm.

The detection time of different algorithms was compared, and the results are shown in Table 5.

It was seen from Table 5 that the time complexity of the SVM optimization algorithms increased compared with the SVM algorithm, the detection time of the PSO-SVM algorithm increased by 4.23% compared with the SVM algorithm, the detection time of the ACO-SVM algorithm increased by 4.76%, and the detection time of the IWAO-SVM algorithm only increased by 1.59%, 2.54% lower than the PSO-SVM algorithm and 3.03% lower than the ACO-SVM algorithm, which showed that the optimization algorithm designed in this study not only had obvious advantages in the detection rate, but also had a good performance in the detection time, i.e., it could provide more excellent service for network intrusion detection.

5 Discussion

It is very important for network protection and control to detect intrusion attacks effectively [17]. In the network intrusion detection, clustering algorithm [18], Apriori algorithm, decision tree [19], Q-learning, neural network [20] and hidden Markov [21] have a wide range of applications. This study mainly analyzed SVM. As a common classification and prediction algorithm, SVM has a good application in many fields, such as face recognition [22], risk assessment [23], electricity price prediction [24] and image classification [25].

In order to improve the effectiveness of SVM in intrusion detection, it was optimized by the WAO algorithm in this study, and then it was verified by KDD CUP99 data set. It was seen from Table 3 that the IWAO-SVM algorithm had excellent accuracy in the classification of intrusion attacks, and only seven data were wrongly classified. Then, it was seen from Table 4 and Figure 4 that the IWAO-SVM algorithm had a better detection performance, with the detection rate reaching 99.89%, 14.88%, 8.18% and 5.68% higher than the other three algorithms respectively; the precision ratio improved by 7.10 %, 6.43% and 3.93% respectively; the accuracy increased by 13.41%, 10.64% and 6.86% respectively, which verified the effectiveness of IWAO in SVM optimization and the good precision of the IWAO-SVM algorithm in the intrusion detection. Finally, the comparison of the detection time showed that the method proposed in this study had a good advantage in time compared to the other optimization algorithms, only 1.59% longer than the SVM algorithm.

Although some achievements have been made in the research of network intrusion prevention and detection, there are still some shortcomings that need to be solved in the future work:

(1) the detection effect of the SVM algorithm should be compared when choosing different kernel functions;
(2) the performance of more optimization algorithms in SVM should be compared;
(3) the performance of the IWAO-SVM algorithm in practical application should be studied.

6 Conclusion

Aiming at the detection of network intrusion prevention, this study analyzed the optimization of SVM, designed an improved WAO algorithm, and compared it with other optimization algorithms on the data set. The results suggested that:

(1) the IWAO-SVM algorithm could detect intrusion attacks accurately;
(2) the detection rate of the IWAO-SVM algorithm was 99.89%, the precision ratio was 99.92%, and the accuracy rate was 99.86%, which were all higher than the other excellent algorithms;
(3) the detection time of the IWAO algorithm was 192s , only 1.59% longer than the SVM algorithm.

7 References

[1] Elekar KS (2015). Combination of data mining techniques for intrusion detection system. International Conference on Computer. IEEE.

[2] Shah AA, Khiyal MSH, Awan MD (2015). Analysis of Machine Learning Techniques for Intrusion Detection System: A Review. International Journal of Computer Applications, 119(3), pp. 19-29.

[3] Keegan N, Ji S Y, Chaudhary A, Concolato C, Yu B, Jeong DH (2016). A survey of cloud-based network intrusion detection analysis. Human-centric Computing and Information Sciences, 6(1), pp. 19.

[4] Li GD, Hu JP, Xia KW (2015). Intrusion detection using relevance vector machine based on cloud particle swarm optimization. Control & Decision, 30(4), pp. 698-702.

[5] Sangeetha S, Devi BG, Ramya R, Dharani MK, Sathya P (2015). Signature Based Semantic Intrusion Detection System on Cloud. Advances in Intelligent Systems and Computing, 339, pp. 657-666.

[6] Kannan A, Venkatesan KG, Stagkopoulou A, Li S (2015). A Novel Cloud Intrusion Detection System Using Feature Selection and Classification. International Journal of Intelligent Information Technologies, 11(4), pp. 1-15.

[7] Geng X, Li Q, Ye D, Wu Z, Jiang Y (2017). Intrusion detection algorithm based on rough weightily averaged one-dependence estimators. Journal of Nanjing University of Science & Technology, 41(4), pp. 420-427.

[8] Milliken M, Bi Y, Galway L, Hawe GI (2015). Ensemble learning utilising feature pairings for intrusion detection. World Congress on Internet Security. IEEE.

[9] Ghosh P, Mandal AK, Kumar R (2015). An Efficient Cloud Network Intrusion Detection System. Advances in Intelligent Systems & Computing, 339, pp. 91-99.

[10] Jinny SV, Kumari JJ (2015). Encrusted CRF in Intrusion Detection System. Advances in Intelligent Systems & Computing, 325, pp. 605-613.

[11] Tedesco G, Aickelin U (2016). Adaptive Alert Throttling for Intrusion Detection Systems. Social Science Electronic Publishing, 730, pp. 194-201.

[12] Abdiansah A, Wardoyo R (2015). Time complexity analysis of support vector machines (SVM) in LibSVM. International Journal of Computer Applications, 128(3), pp. 975-8887.

[13] Aljarah I, Faris H, Mirjalili S (2016). Optimizing connection weights in neural networks using the whale optimization algorithm. Soft Computing, 22(1), pp. 1-15.

[14] Friedlaender A, Weinrich M, Bocconcelli A, et al (2011). Underwater components of humpback whale bubble-net feeding behaviour. Behaviour, 148(5), pp. 575-602.

[15] Wang L, Dong C, Hu J, Li G (2015). Network Intrusion Detection Using Support Vector Machine Based on Particle Swarm Optimization. Plant Biotechnology Reports, 4(3), pp. 237-242.

[16] Zan P, Ai YT, Zhao J, Shao Y (2014). A Prediction Model of Rectum’s Perceptive Function Reconstruction Based on SVM Optimized by ACO. 461, pp. 121-128.

[17] Deng S, Zhou A, Yue D, Hu B, Zhu L (2017). Distributed intrusion detection based on hybrid gene expression programming and cloud computing in cyber physical power system. IET Control Theory and Applications, 11(11), pp. 1822-1829.

[18] Chahal JK, Kaur A (2016). A Hybrid Approach based on Classification and Clustering for Intrusion Detection System. International Journal of Mathematical Sciences & Computing, 2(4), pp. 34-40.

[19] Modinat M, Abimbola A, Abdullateef B, Opeyemi A (2015). Gain Ratio and Decision Tree Classifier for Intrusion Detection. International Journal of Computer Applications, 126(1), pp. 975-8887.

[20] Gautam SK, Om H (2016). Computational Neural Network Regression Model for Host based Intrusion Detection System. Perspectives in Science, 8(C), pp. 93-95.

[21] Sharma SK, Manoria M (2015). Intrusion Detection using Hidden Markov Model. International Journal of Computer Applications, 115(4), pp. 35-38.

[22] Prakash N, Singh Y (2015). Fuzzy Support Vector Machines for Face Recognition: A Review. Maropoulos P G, 131(3), pp. 24-26.

[23] Bui DT, Tuan TA, Klempe H, Pradhan B, Revhaug I (2016). Spatial prediction models for shallow landslide hazards: a comparative assessment of the efficacy of support vector machines, artificial neural networks, kernel logistic regression, and logistic model tree. Landslides, 13(2), pp. 361-378.

[24] Shrivastava NA, Khosravi A, Panigrahi BK (2015). Prediction Interval Estimation of Electricity Prices Using PSO-Tuned Support Vector Machines. Industrial Informatics, IEEE Transactions on, 11(2), pp. 322-331.

[25] Tan K, Zhang J, Du Q, Wang X (2015). GPU Parallel Implementation of Support Vector Machines for Hyperspectral Image Classification. IEEE Journal of Selected Topics in Applied Earth Observations & Remote Sensing, 8(10), pp. 1-10.

A Web Server to Store the Modeled Behavior Data and Zone Information of the Multidisciplinary Product Model in the CAD Systems

Yatish Bathla
Doctoral School of Applied Informatics and Applied Mathematics, Óbuda University, Budapest
E-mail: yatish.bathla@phd.uni-obuda.hu

Sándor Szénási
John von Neumann Faculty of Informatics, Óbuda University, Budapest
E-mail: szenasi.sandor@nik.uni-obuda.hu

Student paper

Keywords: web server, multidisciplinary product modeling, rflp structure, cad systems, information retrieval, information storage

Received: January 23, 2019

This work focuses on Human Computer Interaction (HCI) for multidisciplinary product modeling. Requirement Functional Logical Physical (RFLP) structure has emerged as one of the prominent approaches for modeling the multidisciplinary products. To simplify the HCI of an RFLP structured product model, Information Content (IC) provides effective communication and interaction. It controls the RFLP level by the Multilevel Abstraction based Self-Adaptive Definition (MAAD) structure. However, it needs an application to represent the modeled behavior data and zone information of a multidisciplinary product model. Further, the IC application requires an interface to interact with the Computer-Aided Design (CAD) based multidisciplinary product model application and exchange the information between the database of the servers. As per the knowledge of authors, no work has been done yet on the HCI of the IC. Therefore, this paper proposes a Content Web server, which is used to store the modeled behavior data and zone information of the multidisciplinary product model and represented by the IC web application. Then, the Content database is created to store the Layer Info-Chunk (LiC) entities' information of the multidisciplinary product model. Finally, communication between the Content Server and the CAD server is done to represent the IC application interface in the multidisciplinary product application. The Apache Tomcat server, PostgreSQL database, and RESTful web service are used to explain the operations.

Povzetek: Pristop HCI temelji na izkoriščanju prednosti človeških možganov in računalniške umetne inteligence. Na ta način so avtorji prispevka izboljšali multidisciplinarno modeliranje oblektov.

1 Introduction

A good Human Computer Interaction (HCI) interface in the Computer-Aided Design (CAD) systems deals smartly with the relationship between industrial designers and computer software and hardware, studies the design of man-machine interface model efficiently, the smart design of the virtual interface, multi-user, and multi-sensory interface, and provide a good technical foundation for industrial design [1]. CAD systems simplify the engineering tasks in collecting, using, creating and sharing information, but interface designed without consideration of usability often results in unsatisfied experiences and limited outcomes [2]. Classical Product Models (CPM) in the CAD systems [3] allow product development firms to meet their goals more efficiently. It improves product development time, product quality, productivity and reduces manufacturing as well as product costs. Also, the Requirement Functional Logical Physical (RFLP) structure [4] is applied from the system engineering and offers to handle the multidisciplinary product model as a system. Product assembly is done in the specification tree (white square) of the RFLP structure as shown in Fig. 1. Here, Dassault Systém's CATIA 3DEXPERIENCE [5] is using the RFLP structure for multidisciplinary product modeling. The authors have considered this CAD software for explaining the proposed concepts. There is plenty of research done for improving the HCI of the CAD systems. Some of the appreciated work are as follow:

- A webized interactive CAD review system [12] that uses Super Multi-View (SMV) autostereoscopic displays renders the content through a web browser and handles user interactions via JavaScript. But it is an expensive technology and limited to the CPM.
- A VR (Virtual Reality)-CAD server [13] that embeds

a commercial CAD engine for loading and modifying native CAD data in a CVE (Collaborative Virtual Environments). It is a distributed architecture that allows collaborative modifications on native CAD data from heterogeneous VR platforms. It is based on the management the CAD product data and need improvement in terms of visualizing and representation of 3D product data.

The complexity increases in the case of a multidisciplinary product model as it requires the coordination of huge amounts of model information of the multiple disciplines. Indeed, Information Content (IC) [9] handles the multidisciplinary product model indirectly to record and apply the content of modeled information efficiently [10], which further drives the RFLP level by the Multilevel Abstraction based self Adaptive Definition (MAAD) structure [4]. However, there is a need for Information Content based Web application to represent and store the behavior modeled data [11] from the Process plane [14] and Community Zone information [15] of a multidisciplinary product model.
As a solution, this research work proposes a Content Web Server that consists:

- IC based web application to represent the modeled behavior data and Community zone information of a multidisciplinary product model.
- Content database to store the entities of the modeled behavior data of a multidisciplinary product model.

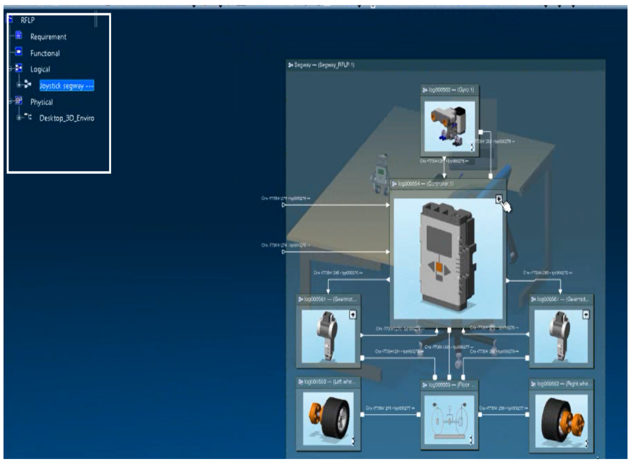


Figure 1: Multidisciplinary Product Model using the RFLP Structure

For the smart interaction of a multidisciplinary product model through the Information Content (IC), an interface is introduced in the multidisciplinary product application through the IC web application. The Apache HTTP Server [16] hosts the IC based web application with the PostgreSQL database [17] to store the multidisciplinary product model data and RESTful web service [18] to exchange the information between the Content and CAD system web server. For the IC web application, the objects collect the Functional and Logical layer information from the InfoChunk [28] entities of the RFLP structure. These objects communicate with the objects of the MAAD structure and collect the modeled behavior data of a multidisciplinary product model. The retrieval of data is according to the process plane of the IC. The objects are based on Object-Oriented Programming (OOP) [8] concepts. The concepts are used frequently in software engineering [29]. The zone and extracted modeled behavior data of a multidisciplinary product are displayed by the IC web application.
This paper starts with the preliminary research where RFLP structured product model, IC, MAAD Structure and InfoChunk entities are discussed. Then human interaction with the IC application and Multidisciplinary application are outlined with the introduction of the Content server. Then, the Content server is explained where the operations and Content database are emphasized. Here, the PostgreSQL database is used for the explanation. Then, Operations of the Content Web server are emphasized. Finally, communication between the Content and CAD system web server is elaborated. Here, the RESTful API web service is used for the explanation.

2 Background

The product modeling is the prominent field. There are plenty of companies like Dassault Systémes [19], Autodesk [20], Robert McNeel [21], Pixologic [22] investing a lot of money in this market. The feature driven CPM (Classical Product Model) [7] is most commonly used for discipline specific product modeling. CPM is limited to the physical level. Handling a complex product model is a challenging task due to the involvement of a large number of engineering objects and their relationship. But, product modeling is not limited to the physical layer. The separated or only slightly integrated mechanical engineering modeling increasingly demanded multidisciplinary integration [30]. Modeling of a multidisciplinary product must have a means for the integration of discipline specific models into a model with a unified structure. Higher abstraction is realized by using of RFLP structure based product model [4]. It is commonly used for multidisciplinary modeling as it models the product as a system. It is compliant with the IEEE 1220 standard. This structure has four layers i.e. Requirement layer for the requirements against the product, Function layer for the functions to fulfill requirements, Logical layer for the product wide logical connections, and Physical layer for the representations of physically existing objects. It accommodates product behavior definitions on its Functional and Logical levels. In the RFLP structure of the Dassault Systém's CATIA 3DEXPERIENCE software, Dymola [6] is used to analyze the dynamic logical behavior of a product and Modelica [7] is used for logical and physical modeling of the technical system. Modelica is a multi-domain modeling language for component-oriented modeling of complex systems and based on the OOP concepts. [30].

Information Content (IC) [9] assists effective communication in the multidisciplinary product modeling. It drives the RFLP level by the MAAD structure. The MAAD modeling [31] methods and model structures are introduced as generalized means for the support of higher level abstraction based generation of RFLP elements. The MAAD modeling was based on the knowledge representation, contextual change propagation, and extended feature definition capabilities for advanced modeling systems. In the IC, the intent is defined by the human to control the definition of engineering objects of a product model[32]. In the *Engineering objectives* layer, the Process plane [14] is used to store the processes performed on a multidisciplinary product model. Also, Community zones [15] are used by the IC to organize the complex product model entities and their relationship as shown in Fig. 2. Here, product model space is divided into community zones based on the discipline, specification or configuration. In this figure, the multidisciplinary product model is divided into community Internal or External based on configuration. The information of the process plane and community zones are shown on the representation plane of the IC.

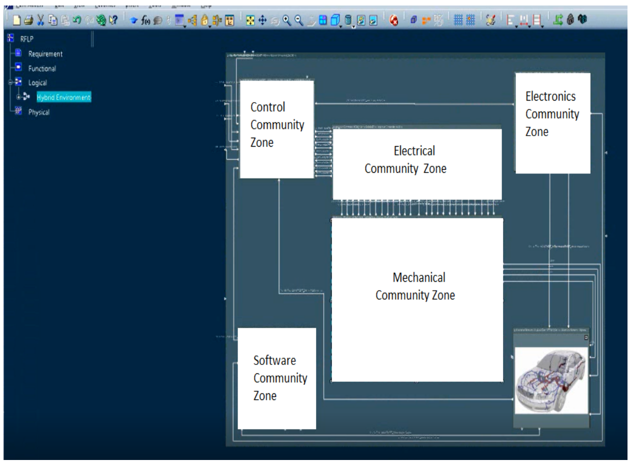


Figure 2: Community Zones in the RFLP Structure

Layer Info-Chunk (LiC) [28][38] entities were introduced in the Functional and Logical layer of the RFLP structure for effective communication with the IC. It controls the behavior data activities of the RFLP structure. The Logical layer Info-Chunk (LiCL) entity stores the information of the Logical layer and the Component Info-Chunk (CiC) entity stores the information of the Logical component. Further, the Functional layer Info-Chunk (LiCF) entity stores the main function information of the Functional layer and the Sub-Function Info-Chunk (SFiC) entity stores the subfunction information of a function. Considering the above mentioned concepts as a base, the authors propose the effective Human Computer Interaction (HCI) for the multidisciplinary product modeling by using the IC application.

3 User interaction and multidisciplinary product application

In this research work, the Multidisciplinary product model is handled and controlled through the IC application. IC application is a web based application in the JSP (JavaServer Pages) format [23]. It resides on a web server called **Content Server**. It is explained in the next section. A multidisciplinary product model application using the RFLP structure is an application in the 3DXML format [24]. The 3DEXPERIENCE [25] CAD software requires ENOVIA [26] Product Lifecycle Management (PLM) system in the backend that allows the data to be stored in one central location, therefore, access from anywhere. ENOVIA V6 uses Microsoft SQL Server 2008 R2 Enterprise platform for database management [39]. Therefore, in this research work, CAD Product server refers to the Microsoft SQL Server [27]. The IC, MAAD Structure and LiC entities in the RFLP structure communicate through the Info-Chunk objects, which is based on the Object-Oriented Programming Principles (OOP) [8]. The advantage of the IC application is a simpler user interface and efficient organization of objects retrieved from the product model. There are two scenarios to be considered:

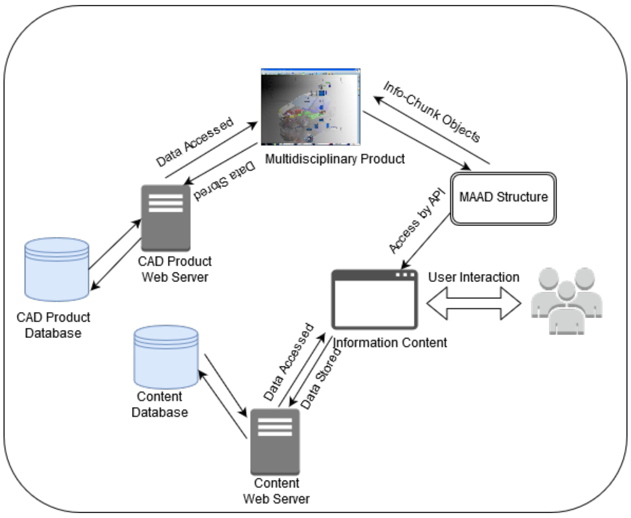


Figure 3: User Interaction from the Information Content application

– The user interacts with the IC application to access the Multidisciplinary product application as shown in Fig. 3. IC drives the RFLP structure through the MAAD structure. Every application has its own web server for the resource management and database to locate the information. The database of the CAD product server is isolated from the database of the content server.

– The user interacts with the Multidisciplinary product application to access the IC application through a separate plane as shown in Fig. 4. There is an interface between the two applications. The database of the CAD product server retrieves the process and zone partition information using the web services from the database of the content server.

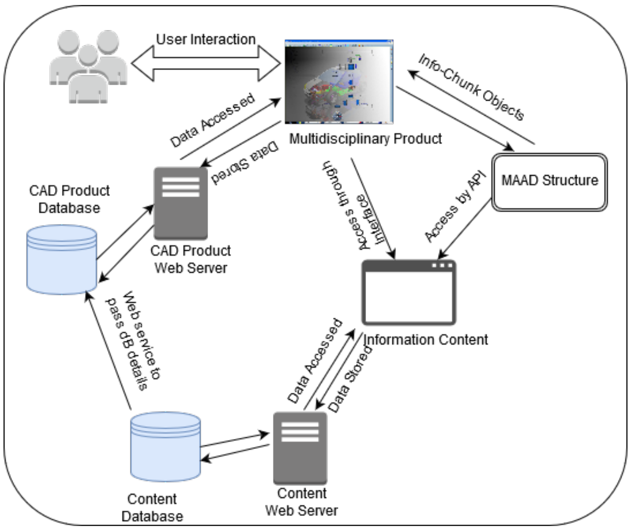


Figure 4: User Interaction from the Multidisciplinary product application

4 Content web server

Content Web server is the Apache Http Server [33] that used to store and display the data of the Information Content (IC) as shown in Fig. 5. The Tomcat Servlet [34] is used for the Information Content web application. Enterprise Management Agent (EMA) is the integral software component responsible for managing and maintaining the IC based Web application. It also allows monitoring the CAD Product database, through management plug-ins and connectors. The Process partition consists of the outcome of the process plane of the IC, the product model after a certain set of the process applied and the files that explain the location of outcomes of the process plane and product model. Similarly, the Zone partition consists of the outcome of the community zone of the IC, the product model after divided into the zones and the files that explain the location of outcomes of the community zone and product model. The outcomes are the graphs obtained from the Process plane and Community zones. The authors store the graphs in the PNG format [35], product model application in the Dassault Systém's 3DEXPERIENCE file format (3DXML) [36], XML file format [37] for the data interchange and SCN file format for the 3D product model (Assembly model or part model) management.

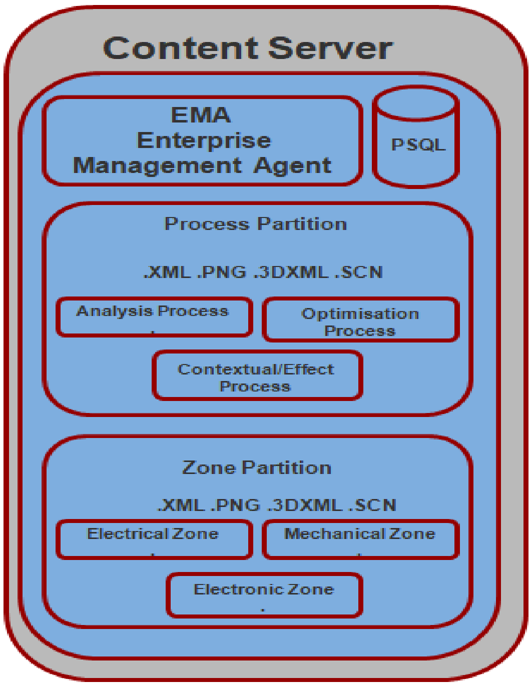


Figure 5: Content Server

The Content database is created by using the PostgreSQL. It stores the data of the Information Content application while handling the behavior modeled data and zone information of the multidisciplinary product application. Entity Relationship (ER) [40] diagram is used for the physical data modeling as shown in Fig. 6. It is required for the schema level for creating a database. There are nine tables created based on the concept of LiC entities of the RFLP structure. During the product modeling using the RFLP structure, there is a set of information transferred from the Requirement layer to the Physical layer. Behaviors of a product model are represented in the Functional and Logical layer of the RFLP structure. LiCF table is used to store the attributes of the Functional layer and the LiCL table is used to store the attributes of the Logical layer of the LiC (Layer Info-Chunk) entity of the RFLP structure. In these tables, some of the data types are built-in while others are user-defined. In the case of the LiCL table,

– LiCLConnector is the Enumerated data type that stores the inner and connector values of the LiCL entity.

– LiCF, CiC, and LiCLDataModel are the composite data type, whose attributes and data types are specified in the Content ER diagram.

– In the CiC table, CiCConnector is the Enumerated data type that stores the inner and stream values of the CiC entity

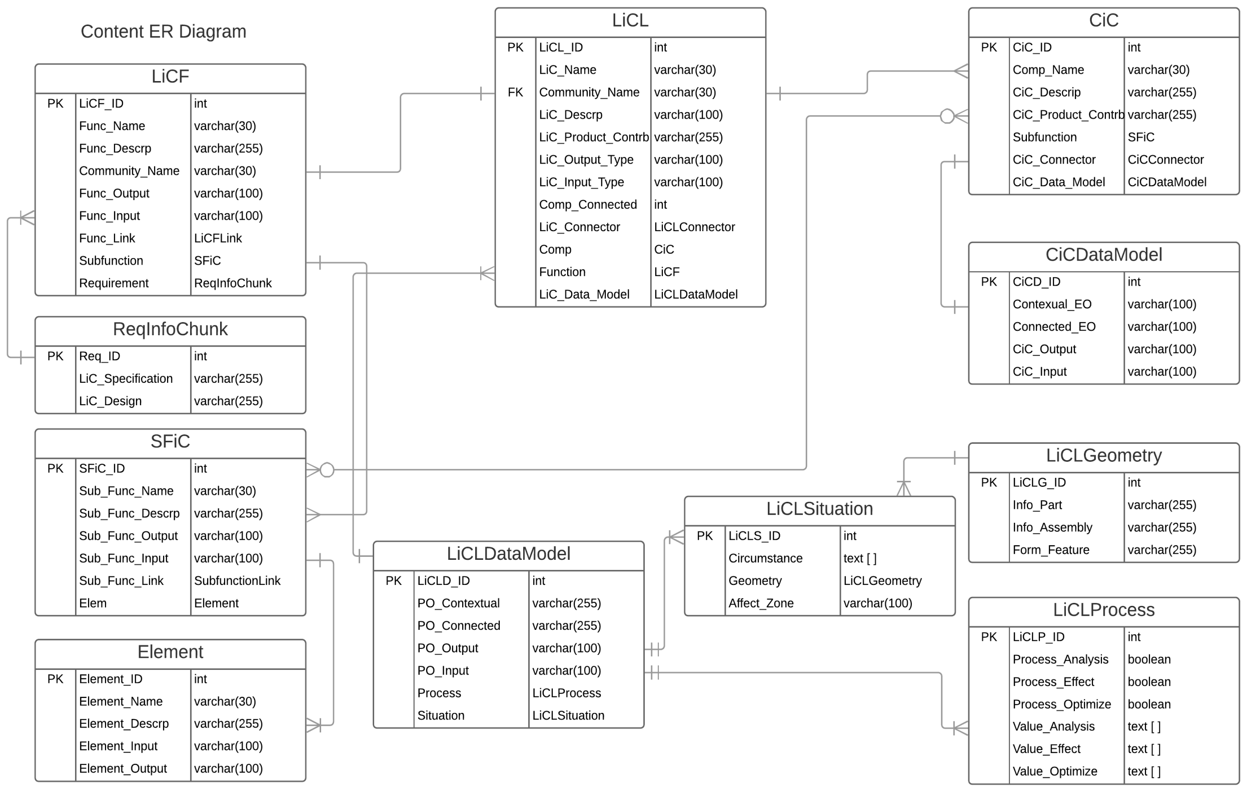


Figure 6: Content Entity Relationship Diagram.

– In the CiC table, SFiC and CiCDataModel is the composite data type, whose attributes and data types are specified in the Content ER diagram.

– In the LiCLDataModel table, LiCLSituation and LiCLProcess are the user-defined composite data types.

The CiC table is used to store the attributes of the CiC (Component Info-Chunk) entity present in a LiCL entity. LiCLDataModel table is used to store the attributes of the detailed description of the Physical layer of the RFLP structure. LiCLProcess table is used to store the attributes of the Process plane of the IC. LiCLSituation table is used to store the attributes of a situation in the logical layer of the RFLP structure. Here, LiCLGeometry composite data type is used to store the information of a part model or assembly model in a situation. In the case of the LiCF table,

– LiCFLink is the Enumerated data type that stores the inner and connector values of the LiCF entity.

– SFiC and ReqInfoChunk are the composite data types, whose attributes and data types are specified in the Content ER diagram.

– In the SFiC table, SubFunctionLink is the Enumerated data type that stores the inner and stream values of the SFiC entity

– In the SFiC table, Element is the composite data type, whose attributes and data types are specified in the Content ER diagram.

– In the ReqInfochunk table, attributes and data types are specified in the Content ER diagram

For reference, LiCLConnector, LiCLDataModel, and LiCLProcess commands are demonstrated using the SQL statements of PostgreSQL as shown below. Here, new tables and data type is created using the CREATE statement.

```
CREATE TYPE LiCLConnector AS ENUM
('inner', 'Stream');

CREATE TYPE LiCLDataModel AS (
    LiCLD_ID INT,
    PO_Contextual VARCHAR(255),
    PO_Connected VARCHAR(255),
    PO_Output VARCHAR(100),
    PO_Input VARCHAR(100),
    Process LiCLProcess,
    Situation LiCLSituation);

CREATE TYPE LiCLProcess AS (
    LiCLP_ID INT,
    Process_Analysis BOOLEAN,
```

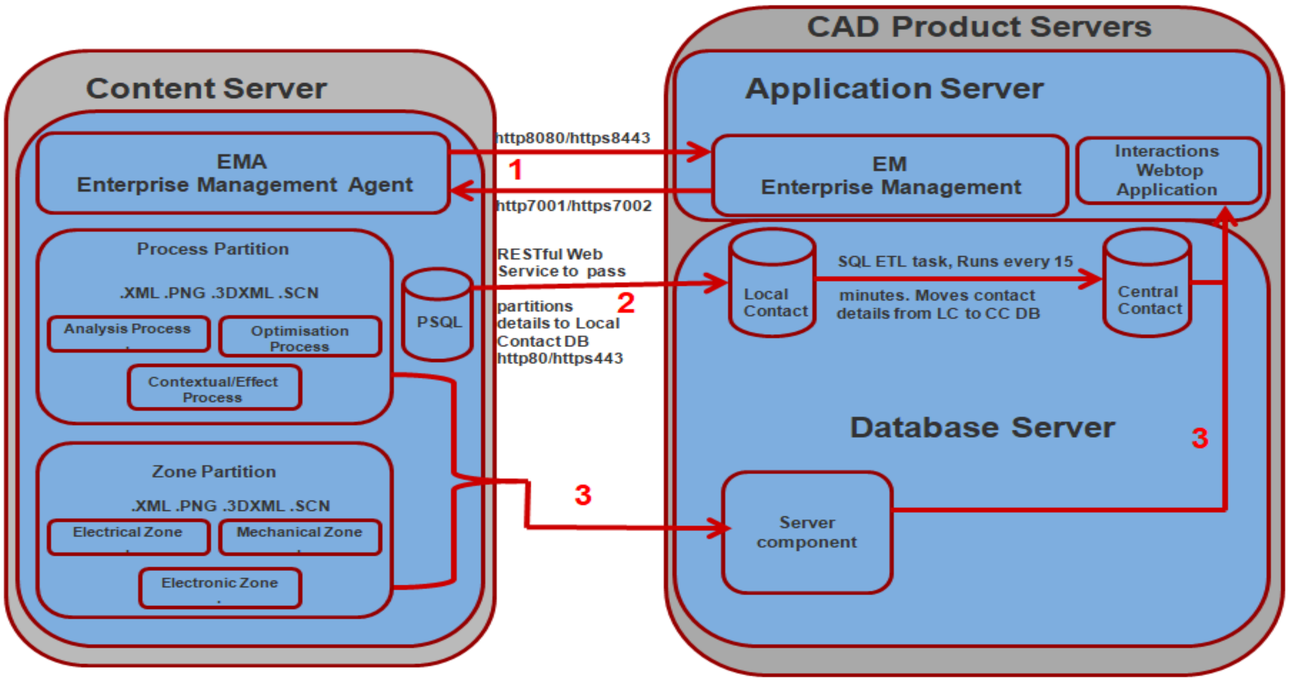


Figure 7: Communication between Content server and CAD server.

```
    Process_Effect BOOLEAN,
    Process_Optimize BOOLEAN,
    Value_Analysis TEXT[],
    Value_Effect TEXT[],
    Value_Optimize TEXT[]);
```

The modeled behavior data of a multidisciplinary product is stored in the entities based on the entities' relationship. The entities are populated by the IC application.

- In the context of the Functional layer, One LiCF entity may have many SFiC entities and one or many LiCF entities may have one ReqInfoChunk entity. Further, one SFiC entity may have one or many Element entities.
- In the context of the Logical layer, One LiCL entity may have one LiCF entity and many CiC entities. Also, one or many LiCL entities may have one LiCLDataModel entity. Further, one and only one LiCLDataModel entity may have one or many LiCLProcess and LiCLSituation entities. Here, one or many LiCLSituation entities may have one LiCLGeometry entity. Also, One CiC entity may have one CiCDataModel.

4.1 Operations

A human expert handles the multidisciplinary application through the IC application. To model the behavior data, the process plane from the *Engineering objectives* layer of the IC interacts with the Info-Chunk objects of the *Product Behaviors* level of the MAAD structure, which further, drives the Info-Chunk objects of the *Functional* and *Logical* layer of the RFLP structure. Here, the Process plane of the IC communicates with LiC entities of the RFLP structure using the Info-Chunk objects to retrieve the modeled behavior data of a multidisciplinary product plane. The data is stored in the Process partition. Also, the Product model is divided into community zone based on the discipline. The outcome is stored in the Zone partition. Then, a human interacts with the results stored in the partition through the representation plane of the Interactive IC application. The outcome could be static or dynamic and represented as graphs, images or animation.

4.2 Communication between content server and CAD product server

The CAD Server pulls process partition and zone partition from the Content server when replaying through the IC interface in the Multidisciplinary web application as shown in Fig. 7. Content server partitions information is saved in CAD server cache and auto deleted almost immediately after a replay. EM-EMA Link handles publishing of configuration between framework and Content server. RESTful Web API Link handles passing of modeled behavior data and zone partition details from Postgres job queue to Local Contact DB which then gets moved on to Central Contact DB by ETL SQL process of the CAD server. The advantage of this API there is no need to install additional software or libraries and provide a great deal of flexibility. Content Server handles retrieval of .XML, .PNG, .3DXML and .SCN content from the Content server to IC Webtop application [41] interface of the Multidisciplinary application for replay. The process and zone partition details are taken from Central Contact DB and converted to .3DXML format for the multidisciplinary application and then it is deleted.

5 Conclusion

This research work proposes the Content server to store zone and modeled behavior information of a multidisciplinary product model. This work starts with the Human Computer Interaction (HCI) of a multidisciplinary product model where the model is handled directly by the Information Content (IC) web application or through an interface in the multidisciplinary product application. The operation and process of IC web application are stored in the Content server. Then, the server is explained in brief, where data is stored in the Zone partition and Process partition based on the communication between the IC and RFLP structure. It is done by the Info-Chunk objects and stored in the Content database. Finally, communication between the Content Server and CAD product server is explained where information of zone partition and process partition pushed temporarily to the CAD product server so that IC webtop application in the main application could handle the multidisciplinary product model. As Modelica and Info-Chunk objects are based on the OOPS concepts, the RFLP structure and IC could be compatible with each other and exchange information easily. This research work is an effort to provide efficient user interaction of a multidisciplinary product model through the Information Content.

Acknowledgement

This study is sponsored by the Doctoral School of Applied Informatics and Applied Mathematics, Óbuda University, Budapest, Hungary and Tempus foundation. The authors gratefully acknowledge to his supervisor Dr. Horváth László for the guidance to write this paper.

References

[1] Z. Liang, Z. Jian, Z. Li-Nan and L. Nan. The Application of Human-Computer Interaction Idea in Computer Aided Industrial Design. International Conference on Computer Network, Electronic and Automation (ICCNEA), 160-164, 2017.https://doi.org/10.1109/iccnea.2017.71

[2] Li, Yujiang, Mikael Hedlind, and Torsten Kjellberg. Usability evaluation of CAD CAM: State of the art. Procedia CIRP, 36, 205-210, 2015. https://doi.org/10.1016/j.procir.2015.01.053

[3] Bloom, R. Getting started with CAD/-CAM. Materials & Design, 17(4): 223–224, 1996.https://doi.org/10.1016/S0261-3069(97)88930-3.

[4] L. Horváth and I. J. Rudas. Systems engineering methods for multidisciplinary product definition. IEEE 12th International Symposium on Intelligent Systems and Informatics (SISY), 293-298, 2014. https://doi.org/10.1109/sisy.2014.6923604

[5] Barfield, W. The law of virtual reality and increasingly smart virtual avatars. Research Handbook on the Law of Virtual and Augmented Reality, 2–4, 2018. https://doi.org/10.4337/9781786438591.00008

[6] Dempsey, M. Dymola for Multi-Engineering Modelling and Simulation. IEEE Vehicle Power and Propulsion Conference, 1-6, 2006. https://doi.org/10.1109/vppc.2006.364294

[7] Peter Fritzson. Principles of Object-Oriented Modeling and Simulation with Modelica 3.3: A Cyber-Physical Approach, Wiley-IEEE Press, John Wiley & Sons Inc, 2015. https://doi.org/10.1002/9781118989166

[8] Baesens, B., Backiel, A., & Broucke, S.vanden (Eds.). Beginning Java Programming, Wrox, 2012. https://doi.org/10.1002/9781119209416

[9] L. Horváth and I. J. Rudas. Towards the Information Content-driven Product Model. Proceedings of the IEEE International Conference on System of Systems Engineering, 1-6, 2008. https://doi.org/10.1109/sysose.2008.4724183

[10] Laszlo Horvath, Imre J. Rudas. Bringing up product model to thinking of engineer. Proceedings of the IEEE International Conference on Systems, Man and Cybernetics, 1355 - 1360, 2008. https://doi.org/10.1109/icsmc.2008.4811474

[11] L. Horváth and I. J. Rudas. Elevated level design intent and behavior driven feature definition for product modeling. 39th Annual Conference of the IEEE Industrial Electronics Society, 1(2), 4374-4379, 2013. https://doi.org/10.1109/iecon.2013.6699839

[12] Seo, Daeil, Yongjae Lee and Byounghyun Yoo. Webizing Interactive CAD Review System Using Super Multiview Autostereoscopic Displays. HCII Posters, Part II, CCIS 714, 62–67, 2017. https://doi.org/10.1007/978-3-319-58753-0-10

[13] Okuya, Yujiro, Nicolas Ladeveze, Olivier Gladin, Cédric Fleury and Patrick Bourdot. Distributed Architecture for Remote Collaborative Modification of Parametric CAD Data. IEEE Fourth VR International Workshop on Collaborative Virtual Environments (3DCVE), 1-4, 2018. https://doi.org/10.1109/3dcve.2018.8637112

[14] Yatish Bathla. Different types of process involved in the information content product model. Proceedings of the IEEE 14th International Symposium on Intelligent Systems and Informatics (SISY), 99-104, 2016. https://doi.org/10.1109/sisy.2016.7601478

[15] Yatish Bathla. Structured organization of Engineering Objects in the information content of PLM system. Proceedings of the IEEE 11th International Symposium on Applied Computational Intelligence and Informatics (SACI), 473 - 478, 2016. https://doi.org/10.1109/saci.2016.7507424

[16] Fielding, R.T., Kaiser, G. The Apache HTTP Server Project, IEEE Internet Computing, 1(4): 88 - 90, 1997. https://doi.org/10.1109/4236.612229.

[17] Neil Matthew, Richard Stones. Introduction to PostgreSQL. Apress, 2005, https://doi.org/10.1007/978-1-4302-0018-5.

[18] Li Li and Wu Chou. Design Patterns for RESTful Communication Web Services. IEEE International Conference on Web Services, 2010. https://doi.org/10.1109/icws.2010.101

[19] Smolek, P., Heinzl, B., Ecker, H., & Breitenecker, F.. Exploring the Possibilities of Co-Simulation with CATIA V6 Dynamic Behavior Modeling, SNE Simulation Notes Europe, 23(3-4), 2013. https://doi.org/10.11128/sne.23.sn.10205

[20] Sha Liu. Sustainable Building Design Optimization Using Building Information Modeling. ICCREM, 2015. https://doi.org/10.1061/9780784479377.038

[21] Chen, H., Lowe, A.A., de Almeida, F.R., Wong, M., Fleetham, J.A., Wang, B.. Three-dimensional computer-assisted study model analysis of long-term oral-appliance wear. Part 1: Methodology. American Journal of Orthodontists & Dentofac. Orthop, 134(3): 393–407, 2008. https://doi.org/10.1016/j.ajodo.2006.10.030

[22] Tim Vernon. Zbrush. Journal of Visual Communication in Medicine, 34(1): 31–35, 2011. https://doi.org/10.3109/17453054.2011.548735

[23] Lennart Jörelid. J2EE FrontEnd Technologies: A Programmer's Guide to Servlets, JavaServer Pages, and Enterprise JavaBeans. Springer, 2002. https://doi.org/10.1007/978-1-4302-1148-8

[24] Jing Chen, Jiawei Li, Mo Li. Progressive Visualization of Complex 3D Models Over the Internet. Transactions in GIS, 20(6): 887–902, 2016. https://doi.org/10.1111/tgis.12185

[25] Adam Suydam, Jason Pyles. Lockheed Martin Conceptual Design Modeling in the Dassault Systemes 3DEXPERIENCE Platform, AIAA Scitech Forum, 2020. https://doi.org/10.2514/6.2020-1391

[26] Ling-Long Lin, Yun-Tao Song, Yu-Xiang Tang, Qing-Qing Du, Yi-Peng Gong. Implementation and application study on 3D collaborative design for CFETR based on ENOVIA VPM. Fusion Engineering and Design, 100, 198–203, 2015. https://doi.org/10.1016/j.fusengdes.2015.05.072

[27] Buffington, J.. Microsoft SQL Server. Data Protection for Virtual Data Centers. 2011, https://doi.org/10.1002/9781118255766.ch8

[28] Yatish Bathla. Conceptual Models of Information Content for Product Modeling. Acta Polytechnica Hungarica, XV (2): 169-188, 2018. https://doi.org/10.12700/aph.15.1.2018.2.9

[29] H. F. Krikorian. Introduction to object-oriented systems engineering.1. Journal of IT Professional. V(2), 38-42, 2003. https://doi.org/10.1109/MITP.2003.1191791

[30] L. Horváth. New method for enhanced driving of entity generation in RFLP structured product model. 12th IEEE Conference on Industrial Electronics and Applications (ICIEA), 541-546, 2017. https://doi.org/10.1109/iciea.2017.8282903

[31] L. Horváth and I. J. Rudas. Multilevel Abstraction Based Self Control Method for Industrial PLM Model. IEEE International Conference on Industrial Technology, 695-700, 2014. https://doi.org/10.1109/icit.2014.6894915

[32] Laszlo Horvath. New methods on the way to intelligent modeling in computer integrated engineering. Proceedings of the 36th Annual Conference on IEEE Industrial Electronics Society(IECON), 1359-1364, 2010. https://doi.org/10.1109/iecon.2010.5675486

[33] Fielding, R.T., Kaiser, G.. The Apache HTTP Server Project. IEEE Internet Computing, 1(4): 88-90, 1997. https://doi.org/10.1109/4236.612229.

[34] Aleksa Vukotic, James Goodwill. Apache Tomcat 7, Springer Nature Switzerland, Apress 2011. https://doi.org/10.1007/978-1-4302-3724-2

[35] Randers-Pehrson, G., Boutell *PNG (Portable Network Graphics) Specification*, Version 1.0. PNG Dev. Gr., 1999. https://doi.org/10.17487/rfc2083

[36] Roberto Riascos, Laurent Levy, Josip Stjepandić, Arnulf Fröhlich. Digital Mock-up. Concurrent Engineering in the 21st Century Foundations, Developments and Challenges, 355-388, 2015. https://doi.org/10.1007/978-3-319-13776-6_13

[37] Clarke, K.S.. Extensible markup language (XML), in: Understanding Information Retrieval Systems. Management, Types, and Standards, 2011. https://doi.org/10.1201/b11499-52

[38] Yatish Bathla. Info-Chunk driven RFLP Structure based Product Model for Multidisciplinary Cyber Physical Systems. IEEE 16th International Symposium on Intelligent Systems and Informatics (SISY), pp. 000327-000332, 2018. https://doi.org/10.1109/sisy.2018.8524653

[39] He Youquan, Zhang Wei, Xie Jianfang, Wang Jian, Qiu Hanxing. Integrated Application of PLM Based ENOVIA Platform in Domestic Manufacturing Industry. International Conference on Information Management, Innovation Management and Industrial Engineering, nov, 2011. https://doi.org/10.1109/iciii.2011.202

[40] Richard Earp, Sikha Bagui. Database Design Using Entity-Relationship Diagrams. Foundations of Database Design, june, 2003. https://doi.org/10.1201/9780203486054

[41] Heller, D., Krenzelok, L., Orr, J. Webtop: Realities in designing a web-application platform. IEEE Potentials, Proceedings of the 2003 Conference on Designing for User Experiences. DUX '03, 2003, https://doi.org/10.1145/997078.997115

Artificial Intelligence Methods for Modelling Tremor Mechanisms

Vida Groznik
University of Primorska, Faculty of Mathematics, Natural Sciences and Information Technologies, Koper, Slovenia
University of Ljubljana, Faculty of Computer and Information Science, Ljubljana, Slovenia
E-mail: vida.groznik@famnit.upr.si

Thesis summary

Keywords: tremor, attribute visualisation, digital spirography

Received: May 28, 2020

The paper summarises a Doctoral Thesis in which we focus on two main goals: (1) building models for differentiation between three most common tremors: Parkinsonian, essential and mixed type tremor and (2) development of a novel method for attribute visualisation on series.

Povzetek: Članek povzema vsebino doktorske disertacije, v kateri se osredotočamo na dva glavna cilja: (1) gradnja modelov za razločevanje med tremi najpogostejšimi tremorji: parkinsonskim, esencialnim in mešanim tipom tremorja ter (2) razvoj nove metode za vizualizacijo atributov na vrstah.

1 Introduction

Tremor is an involuntary movement of the body and is one of the most common movement disorders. It is primarily associated with various diseases of the nervous system, including Parkinson's disease. Since there are more than 20 different types of tremors, differentiation between them is important from the treatment point of view.

Spirography is a diagnostic method where the subject's task is to draw a left-twisted spiral while the doctor observes the process of drawing (speed, hesitation, etc.) and the final drawing. With the development of tablets digitalised spirography emerged, making it possible to store the course of spiral drawing and the analysis of the acquired time series.

In order to increase confidence in such a system, we would need to provide an explanation of the results to doctors. One option is to visualize the anomalies and results onto the drawn spirals. These visualizations must make sense to physicians and, above all, they must be consistent with their medical knowledge of the domain.

This paper summarises a Doctoral Thesis [1] which tries to address the need for automatic differentiation of tremors and visualisation of the decisions such system would give.

2 Diagnostic models for tremor differentiation

In the thesis, we focus on differentiation between three of the most common tremors: Parkinsonian, essential and mixed type of tremor. For the purpose of building the diagnostic models, we used the digitalised spirography for collecting the data needed.

The first diagnostic model distinguishes between the three tremors, based on clinical examination data, family history and digital spirography. The process of building a model was carried out using argument-based machine learning technique which enabled us to build a decision model through the process of knowledge elicitation from the domain expert (a neurologist). The obtained model consists of thirteen rules that are medically sensible. The process of knowledge elicitation itself contributed to the higher classification accuracy of the final model in comparison with the initial one [2, 5].

In the first diagnostic model, attributes derived from the spirography were included in more than half of the rules. This motivated us to build a model based solely on the digital spirography data. For the needs of constructing an understandable model, we first built several attributes which represented domain medical knowledge. We have built more than 500 different attributes which were used in a logistic regression to construct the final diagnostic model. The model is able to distinguish subjects with tremors from those without tremors with 90% classification accuracy. The final diagnostic model is built into the freely available PARKINSONCHECK mobile application [6].

3 Method for attribute visualisation

During the process of attribute construction, we wanted to know what our attributes were detecting. Thus, we have developed a method for attribute visualisation on series. The method not only helped us with attribute construction, but it is also useful for visual interpretation of the diagnostic model's decisions. The visualisation method and consequently the decision model were evaluated with the help of three independent neurology experts. The results show that both the diagnostic model and the visualisation are meaningful and cover medical knowledge of the domain.

Different visualisation approaches and their benefits have been published in several peer reviewed publications [3, 4, 7].

4 Conclusion

The Thesis [1] describes the development of different diagnostic models for digitalised spirography systems. The emphasis is given to elicitation of expert's knowledge and including that knowledge into the built-in attributes. To increase physicians' confidence in such systems, a novel method for attribute visualisation has been proposed. The results were published in several peer-reviewed publications [2, 3, 4, 5, 6, 7].

References

[1] Groznik, V. (2018) Metode umetne inteligence za modeliranje mehanizmov tremorjev: doktorska disertacija. Ljubljana. 139 pages, ilustr.
http://eprints.fri.uni-lj.si/4134/.

[2] Groznik, V., Guid, M., Sadikov, A., Možina, M., Georgiev, D., Kragelj, V., Ribarič, S., Pirtošek, Z., and Bratko, I. (2013) Elicitation of neurological knowledge with argument-based machine learning. *Artificial Intelligence in Medicine*, 57:133–144.
https://doi.org/10.1016/j.artmed.2012.08.003.

[3] Groznik, V., Sadikov, A., Možina, M., Žabkar, J., Georgiev, D., and Bratko, I. (2014) Attribute visualisation for computer-aided diagnosis : a case study. In *ICHI 2014: proceedings, 2014 IEEE International Conference on Healthcare Informatics*, IEEE Computer Society, Conference Publishing Services, pp. 294–299.
https://doi.org/10.1109/ICHI.2014.47.

[4] Groznik, V., Možina, M., Žabkar, J., Georgiev, D., Bratko, I., and Sadikov, A. (2015) Development, debugging, and assessment of ParkinsonCheck attributes through visualisation. In A. Briassouli, J. Benois-Pineau, A. Hauptmann. *Health monitoring and personalized feedback using multimedia data*, Springer, pp. 47–71.
https://doi.org/10.1007/978-3-319-17963-6_4.

[5] Guid, M., Možina, M., Groznik, V., Georgiev, D., Sadikov, A., Pirtošek, Z., and Bratko, I. (2012) ABML knowledge renement loop: a case study. In L. Chen, et al., editors, *Foundations of intelligent systems*, vol. 7661 of Lecture notes in computer science, Lecture notes in artificial intelligence, Springer, pp. 41–50.
https://doi.org/10.1007/978-3-642-34624-8_5.

[6] Sadikov, A., Groznik, V., Žabkar, J., Možina, M., Georgiev, D., Pirtošek, Z., and Bratko, I. (2014) ParkinsonCheck smart phone app.. In T. Schaub, G. Friedrich, B. O'Sullivan, editors, *ECAI 2014: proceedings*, Frontiers in artificial intelligence and applications (Print), vol. 263, IOS Press, pp. 1213–1214.
https://doi.org/10.3233/978-1-61499-419-0-1213.

[7] Sadikov, A., Groznik, V., Možina, M., Žabkar, J., Nyholm, D., Memedi, M., Bratko, I., and Georgiev, D. (2017) Feasibility of spirography features for objective assessment of motor function in Parkinson's disease. *Artificial intelligence in medicine*, vol. 66, pp. 54–62.
https://doi.org/10.1016/j.artmed.2017.03.011.

Call for papers for Special Issue "Artificial Intelligence and Ambient Intelligence"

A special issue of Electronics (ISSN 2079-9292) belongs to the section "Artificial Intelligence".

Deadline for manuscript submissions: December 31, 2020.

Special issue information

Dear Colleagues,

Ambient intelligence (AmI) and artificial intelligence (AI) both rely on AI methods applied to computing devices. Furthermore, their power stems from the same advancement of electronics, sensors, and software development. Yet, AmI is not just an AI application serving humans, and by its definition it is even more aligned to interactions with humans. Be it smart home or autonomous car, it is essential for the AmI system to be familiar with the user's desired performance as well as the current state of the mind. As such, the human-system relation is of a predominant importance, and it represents one of the most fruitful, but often neglected fields of research for AI.

Strategically, the hardware (HW) and software (SW) exponential development was essential not only for AI and AmI, but for the overall human civilization as well. Information society laws such as Moore's Law or Keck's Law describe the progress of electronic computing and data transfer and storage devices. While several limitations of specific HW characteristics (e.g., the speed of the processor chip) have already been met, it is not clear which are the next promising technology fields to enable further HW progress. Is it that we are facing a slow but steady decline following the fast exponential growth? Are there major possibilities of improvements by connecting SW, AI, and AmI methods directly to the chips? Should we integrate the flexibility of SW with the speed of electronic HW and vastly improve the cognitive and computing powers? Will AmI benefit more through this progress, since is it intrinsically devoted to connecting devices and humans? Which one will bring the most benefit to the human society and which one will first achieve superintelligence – general AI or general AmI?

Answers to these and related questions represent the core of this Special Issue. In order to cope with the abovementioned obstacles, original studies and either viewpoints, theoretical analyses or modeling methods can be developed and proposed to foster further progress.

Specific topics

We invite researchers to contribute original research articles as well as review articles to present their proposals, views, and studies in the field of AmI and AI in relation to the overall HW, SW, and human civilization progress. Submissions can focus on the research concept or applied research in topics including, but not limited to, the following:

- Applications in COVID-19: patient monitoring, patient rehabilitation, brain-computer interfaces assisted living and caring, fall detection, elderly care, interventions for psychological crises
- Applications in smart homes and smart buildings
- Mobile/wearable intelligence
- Robotics applied to smart environments
- Applications of combined pervasive / ubiquitous computing with AI
- Use of mobile, wireless, visual, and multi-modal sensor networks in intelligent systems
- Intelligent handling of privacy, security and trust

Manuscript submission information

Manuscripts should be submitted online at www.mdpi.com by registering and logging in to this website. Once you are registered, click here to go to the submission form. Manuscripts can be submitted until the deadline. All papers will be peer-reviewed. Accepted papers will be published continuously in the journal (as soon as accepted) and will be listed together on the special issue website. Research articles, review articles as well as short communications are invited. For planned papers, a title and short abstract (about 100 words) can be sent to the Editorial Office for announcement on this website.

Submitted manuscripts should not have been published previously, nor be under consideration for publication elsewhere (except conference proceedings papers). All manuscripts are thoroughly refereed through a single-blind peer-review process. A guide for authors and other relevant information for submission of manuscripts is available on the Instructions for Authors page. Electronics is an international peer-reviewed open access monthly journal published by MDPI.

Please visit the Instructions for Authors page before submitting a manuscript. The Article Processing Charge (APC) for publication in this open access journal is 1400 CHF (Swiss Francs) (APC: CHF 1500 from 1 July 2020 onwards). Submitted papers should be well formatted and use good English. Authors may use MDPI's English editing service prior to publication or during author revisions.

Keywords

- Ambient intelligence
- Artificial intelligence
- Superintelligence
- Multi-agent systems
- Ambient assisted living
- Computational intelligence methods
- Pervasive Computing
- Mobile computing
- Ubiquitous computing
- Self-adaptation, self-organization, and self-supervised learning
- Intelligent interfaces (user-friendly man–machine interface)
- AI-assisted medical diagnosis
- Mobile sensing applications
- Autonomous and social robots
- IoT and cyber-physical systems

Special issue editors

Prof. Dr. Matjaz Gams Website
Guest Editor
Jozef Stefan Institute, Ljubljana, Slovenia
Interests: artificial intelligence; intelligent systems; ambient intelligence; information society; machine learning

Dr. Martin Gjoreski Website
Guest Editor
Jozef Stefan Institute, Ljubljana, Slovenia
Interests: affective computing; machine learning; deep learning; sensor; mobile computing; mobile health

For further reading, please follow the link to the Special Issue Website at:
https://www.mdpi.com/si/electronics/AIs_electronics

JOŽEF STEFAN INSTITUTE

Jožef Stefan (1835-1893) was one of the most prominent physicists of the 19th century. Born to Slovene parents, he obtained his Ph.D. at Vienna University, where he was later Director of the Physics Institute, Vice-President of the Vienna Academy of Sciences and a member of several scientific institutions in Europe. Stefan explored many areas in hydrodynamics, optics, acoustics, electricity, magnetism and the kinetic theory of gases. Among other things, he originated the law that the total radiation from a black body is proportional to the 4th power of its absolute temperature, known as the Stefan–Boltzmann law.

The Jožef Stefan Institute (JSI) is the leading independent scientific research institution in Slovenia, covering a broad spectrum of fundamental and applied research in the fields of physics, chemistry and biochemistry, electronics and information science, nuclear science technology, energy research and environmental science.

The Jožef Stefan Institute (JSI) is a research organisation for pure and applied research in the natural sciences and technology. Both are closely interconnected in research departments composed of different task teams. Emphasis in basic research is given to the development and education of young scientists, while applied research and development serve for the transfer of advanced knowledge, contributing to the development of the national economy and society in general.

At present the Institute, with a total of about 900 staff, has 700 researchers, about 250 of whom are postgraduates, around 500 of whom have doctorates (Ph.D.), and around 200 of whom have permanent professorships or temporary teaching assignments at the Universities.

In view of its activities and status, the JSI plays the role of a national institute, complementing the role of the universities and bridging the gap between basic science and applications.

Research at the JSI includes the following major fields: physics; chemistry; electronics, informatics and computer sciences; biochemistry; ecology; reactor technology; applied mathematics. Most of the activities are more or less closely connected to information sciences, in particular computer sciences, artificial intelligence, language and speech technologies, computer-aided design, computer architectures, biocybernetics and robotics, computer automation and control, professional electronics, digital communications and networks, and applied mathematics.

The Institute is located in Ljubljana, the capital of the independent state of **Slove**nia (or S♡nia). The capital today is considered a crossroad between East, West and Mediterranean Europe, offering excellent productive capabilities and solid business opportunities, with strong international connections. Ljubljana is connected to important centers such as Prague, Budapest, Vienna, Zagreb, Milan, Rome, Monaco, Nice, Bern and Munich, all within a radius of 600 km.

From the Jožef Stefan Institute, the Technology park “Ljubljana” has been proposed as part of the national strategy for technological development to foster synergies between research and industry, to promote joint ventures between university bodies, research institutes and innovative industry, to act as an incubator for high-tech initiatives and to accelerate the development cycle of innovative products.

Part of the Institute was reorganized into several high-tech units supported by and connected within the Technology park at the Jožef Stefan Institute, established as the beginning of a regional Technology park "Ljubljana". The project was developed at a particularly historical moment, characterized by the process of state reorganisation, privatisation and private initiative. The national Technology Park is a shareholding company hosting an independent venture-capital institution.

The promoters and operational entities of the project are the Republic of Slovenia, Ministry of Higher Education, Science and Technology and the Jožef Stefan Institute. The framework of the operation also includes the University of Ljubljana, the National Institute of Chemistry, the Institute for Electronics and Vacuum Technology and the Institute for Materials and Construction Research among others. In addition, the project is supported by the Ministry of the Economy, the National Chamber of Economy and the City of Ljubljana.

Jožef Stefan Institute
Jamova 39, 1000 Ljubljana, Slovenia
Tel.:+386 1 4773 900, Fax.:+386 1 251 93 85
WWW: http://www.ijs.si
E-mail: matjaz.gams@ijs.si
Public relations: Polona Strnad

INFORMATICA
AN INTERNATIONAL JOURNAL OF COMPUTING AND INFORMATICS
INVITATION, COOPERATION

Submissions and Refereeing

Please register as an author and submit a manuscript at: http://www.informatica.si. At least two referees outside the author's country will examine it, and they are invited to make as many remarks as possible from typing errors to global philosophical disagreements. The chosen editor will send the author the obtained reviews. If the paper is accepted, the editor will also send an email to the managing editor. The executive board will inform the author that the paper has been accepted, and the author will send the paper to the managing editor. The paper will be published within one year of receipt of email with the text in Informatica MS Word format or Informatica LaTeX format and figures in .eps format. Style and examples of papers can be obtained from http://www.informatica.si. Opinions, news, calls for conferences, calls for papers, etc. should be sent directly to the managing editor.

SUBSCRIPTION

Please, complete the order form and send it to Dr. Drago Torkar, Informatica, Institut Jožef Stefan, Jamova 39, 1000 Ljubljana, Slovenia. E-mail: drago.torkar@ijs.si

Since 1977, Informatica has been a major Slovenian scientific journal of computing and informatics, including telecommunications, automation and other related areas. In its 16th year (more than twentysix years ago) it became truly international, although it still remains connected to Central Europe. The basic aim of Informatica is to impose intellectual values (science, engineering) in a distributed organisation.

Informatica is a journal primarily covering intelligent systems in the European computer science, informatics and cognitive community; scientific and educational as well as technical, commercial and industrial. Its basic aim is to enhance communications between different European structures on the basis of equal rights and international refereeing. It publishes scientific papers accepted by at least two referees outside the author's country. In addition, it contains information about conferences, opinions, critical examinations of existing publications and news. Finally, major practical achievements and innovations in the computer and information industry are presented through commercial publications as well as through independent evaluations.

Editing and refereeing are distributed. Each editor can conduct the refereeing process by appointing two new referees or referees from the Board of Referees or Editorial Board. Referees should not be from the author's country. If new referees are appointed, their names will appear in the Refereeing Board.

Informatica web edition is free of charge and accessible at http://www.informatica.si.
Informatica print edition is free of charge for major scientific, educational and governmental institutions. Others should subscribe.

Informatica WWW:

http://www.informatica.si/

Referees from 2008 on:

A. Abraham, S. Abraham, R. Accornero, A. Adhikari, R. Ahmad, G. Alvarez, N. Anciaux, R. Arora, I. Awan, J. Azimi, C. Badica, Z. Balogh, S. Banerjee, G. Barbier, A. Baruzzo, B. Batagelj, T. Beaubouef, N. Beaulieu, M. ter Beek, P. Bellavista, K. Bilal, S. Bishop, J. Bodlaj, M. Bohanec, D. Bolme, Z. Bonikowski, B. Bošković, M. Botta, P. Brazdil, J. Brest, J. Brichau, A. Brodnik, D. Brown, I. Bruha, M. Bruynooghe, W. Buntine, D.D. Burdescu, J. Buys, X. Cai, Y. Cai, J.C. Cano, T. Cao, J.-V. Capella-Hernández, N. Carver, M. Cavazza, R. Ceylan, A. Chebotko, I. Chekalov, J. Chen, L.-M. Cheng, G. Chiola, Y.-C. Chiou, I. Chorbev, S.R. Choudhary, S.S.M. Chow, K.R. Chowdhury, V. Christlein, W. Chu, L. Chung, M. Ciglarič, J.-N. Colin, V. Cortellessa, J. Cui, P. Cui, Z. Cui, D. Cutting, A. Cuzzocrea, V. Cvjetkovic, J. Cypryjanski, L. Čehovin, D. Čerepnalkoski, I. Ćosić, G. Daniele, G. Danoy, M. Dash, S. Datt, A. Datta, M.-Y. Day, F. Debili, C.J. Debono, J. Dedič, P. Degano, A. Dekdouk, H. Demirel, B. Demoen, S. Dendamrongvit, T. Deng, A. Derezinska, J. Dezert, G. Dias, I. Dimitrovski, S. Dobrišek, Q. Dou, J. Doumen, E. Dovgan, B. Dragovich, D. Drajic, O. Drbohlav, M. Drole, J. Dujmović, O. Ebers, J. Eder, S. Elaluf-Calderwood, E. Engström, U. riza Erturk, A. Farago, C. Fei, L. Feng, Y.X. Feng, B. Filipič, I. Fister, I. Fister Jr., D. Fišer, A. Flores, V.A. Fomichov, S. Forli, A. Freitas, J. Fridrich, S. Friedman, C. Fu, X. Fu, T. Fujimoto, G. Fung, S. Gabrielli, D. Galindo, A. Gambarara, M. Gams, M. Ganzha, J. Garbajosa, R. Gennari, G. Georgeson, N. Gligorić, S. Goel, G.H. Gonnet, D.S. Goodsell, S. Gordillo, J. Gore, M. Grčar, M. Grgurović, D. Grosse, Z.-H. Guan, D. Gubiani, M. Guid, C. Guo, B. Gupta, M. Gusev, M. Hahsler, Z. Haiping, A. Hameed, C. Hamzaçebi, Q.-L. Han, H. Hanping, T. Härder, J.N. Hatzopoulos, S. Hazelhurst, K. Hempstalk, J.M.G. Hidalgo, J. Hodgson, M. Holbl, M.P. Hong, G. Howells, M. Hu, J. Hyvärinen, D. Ienco, B. Ionescu, R. Irfan, N. Jaisankar, D. Jakobović, K. Jassem, I. Jawhar, Y. Jia, T. Jin, I. Jureta, Đ. Juričić, S. K, S. Kalajdziski, Y. Kalantidis, B. Kaluža, D. Kanellopoulos, R. Kapoor, D. Karapetyan, A. Kassler, D.S. Katz, A. Kaveh, S.U. Khan, M. Khattak, V. Khomenko, E.S. Khorasani, I. Kitanovski, D. Kocev, J. Kocijan, J. Kollár, A. Kontostathis, P. Korošec, A. Koschmider, D. Košir, J. Kovač, A. Krajnc, M. Krevs, J. Krogstie, P. Krsek, M. Kubat, M. Kukar, A. Kulis, A.P.S. Kumar, H. Kwaśnicka, W.K. Lai, C.-S. Laih, K.-Y. Lam, N. Landwehr, J. Lanir, A. Lavrov, M. Layouni, G. Leban, A. Lee, Y.-C. Lee, U. Legat, A. Leonardis, G. Li, G.-Z. Li, J. Li, X. Li, X. Li, Y. Li, Y. Li, S. Lian, L. Liao, C. Lim, J.-C. Lin, H. Liu, J. Liu, P. Liu, X. Liu, X. Liu, F. Logist, S. Loskovska, H. Lu, Z. Lu, X. Luo, M. Luštrek, I.V. Lyustig, S.A. Madani, M. Mahoney, S.U.R. Malik, Y. Marinakis, D. Marinčič, J. Marques-Silva, A. Martin, D. Marwede, M. Matijašević, T. Matsui, L. McMillan, A. McPherson, A. McPherson, Z. Meng, M.C. Mihaescu, V. Milea, N. Min-Allah, E. Minisci, V. Mišić, A.-H. Mogos, P. Mohapatra, D.D. Monica, A. Montanari, A. Moroni, J. Mosegaard, M. Moškon, L. de M. Mourelle, H. Moustafa, M. Možina, M. Mrak, Y. Mu, J. Mula, D. Nagamalai, M. Di Natale, A. Navarra, P. Navrat, N. Nedjah, R. Nejabati, W. Ng, Z. Ni, E.S. Nielsen, O. Nouali, F. Novak, B. Novikov, P. Nurmi, D. Obrul, B. Oliboni, X. Pan, M. Pančur, W. Pang, G. Papa, M. Paprzycki, M. Paralič, B.-K. Park, P. Patel, T.B. Pedersen, Z. Peng, R.G. Pensa, J. Perš, D. Petcu, B. Petelin, M. Petkovšek, D. Pevec, M. Pičulin, R. Piltaver, E. Pirogova, V. Podpečan, M. Polo, V. Pomponiu, E. Popescu, D. Poshyvanyk, B. Potočnik, R.J. Povinelli, S.R.M. Prasanna, K. Pripužić, G. Puppis, H. Qian, Y. Qian, L. Qiao, C. Qin, J. Que, J.-J. Quisquater, C. Rafe, S. Rahimi, V. Rajkovič, D. Raković, J. Ramaekers, J. Ramon, R. Ravnik, Y. Reddy, W. Reimche, H. Rezankova, D. Rispoli, B. Ristevski, B. Robič, J.A. Rodriguez-Aguilar, P. Rohatgi, W. Rossak, I. Rožanc, J. Rupnik, S.B. Sadkhan, K. Saeed, M. Saeki, K.S.M. Sahari, C. Sakharwade, E. Sakkopoulos, P. Sala, M.H. Samadzadeh, J.S. Sandhu, P. Scaglioso, V. Schau, W. Schempp, J. Seberry, A. Senanayake, M. Senobari, T.C. Seong, S. Shamala, c. shi, Z. Shi, L. Shiguo, N. Shilov, Z.-E.H. Slimane, F. Smith, H. Sneed, P. Sokolowski, T. Song, A. Soppera, A. Sorniotti, M. Stajdohar, L. Stanescu, D. Strnad, X. Sun, L. Šajn, R. Šenkeřík, M.R. Šikonja, J. Šilc, I. Škrjanc, T. Štajner, B. Šter, V. Štruc, H. Takizawa, C. Talcott, N. Tomasev, D. Torkar, S. Torrente, M. Trampuš, C. Tranoris, K. Trojacanec, M. Tschierschke, F. De Turck, J. Twycross, N. Tziritas, W. Vanhoof, P. Vateekul, L.A. Vese, A. Visconti, B. Vlaovič, V. Vojisavljević, M. Vozalis, P. Vračar, V. Vranić, C.-H. Wang, H. Wang, H. Wang, H. Wang, S. Wang, X.-F. Wang, X. Wang, Y. Wang, A. Wasilewska, S. Wenzel, V. Wickramasinghe, J. Wong, S. Wrobel, K. Wrona, B. Wu, L. Xiang, Y. Xiang, D. Xiao, F. Xie, L. Xie, Z. Xing, H. Yang, X. Yang, N.Y. Yen, C. Yong-Sheng, J.J. You, G. Yu, X. Zabulis, A. Zainal, A. Zamuda, M. Zand, Z. Zhang, Z. Zhao, D. Zheng, J. Zheng, X. Zheng, Z.-H. Zhou, F. Zhuang, A. Zimmermann, M.J. Zuo, B. Zupan, M. Zuqiang, B. Žalik, J. Žižka,

Informatica

An International Journal of Computing and Informatics

Web edition of Informatica may be accessed at: http://www.informatica.si.

Subscription Information Informatica (ISSN 0350-5596) is published four times a year in Spring, Summer, Autumn, and Winter (4 issues per year) by the Slovene Society Informatika, Litostrojska cesta 54, 1000 Ljubljana, Slovenia.
The subscription rate for 2020 (Volume 44) is
– 60 EUR for institutions,
– 30 EUR for individuals, and
– 15 EUR for students
Claims for missing issues will be honored free of charge within six months after the publication date of the issue.

Typesetting: Borut Žnidar, borut.znidar@gmail.com.
Printing: ABO grafika d.o.o., Ob železnici 16, 1000 Ljubljana.

Orders may be placed by email (drago.torkar@ijs.si), telephone (+386 1 477 3900) or fax (+386 1 251 93 85). The payment should be made to our bank account no.: 02083-0013014662 at NLB d.d., 1520 Ljubljana, Trg republike 2, Slovenija, IBAN no.: SI56020830013014662, SWIFT Code: LJBASI2X.

Informatica is published by Slovene Society Informatika (president Niko Schlamberger) in cooperation with the following societies (and contact persons):
Slovene Society for Pattern Recognition (Vitomir Štruc)
Slovenian Artificial Intelligence Society (Sašo Džeroski)
Cognitive Science Society (Olga Markič)
Slovenian Society of Mathematicians, Physicists and Astronomers (Dragan Mihailović)
Automatic Control Society of Slovenia (Giovanni Godena)
Slovenian Association of Technical and Natural Sciences / Engineering Academy of Slovenia (Mark Pleško)
ACM Slovenia (Nikolaj Zimic)

Informatica is financially supported by the Slovenian research agency from the Call for co-financing of scientific periodical publications.

Informatica is surveyed by: ACM Digital Library, Citeseer, COBISS, Compendex, Computer & Information Systems Abstracts, Computer Database, Computer Science Index, Current Mathematical Publications, DBLP Computer Science Bibliography, Directory of Open Access Journals, InfoTrac OneFile, Inspec, Linguistic and Language Behaviour Abstracts, Mathematical Reviews, MatSciNet, MatSci on SilverPlatter, Scopus, Zentralblatt Math

Volume 44 Number 2 June 2020 **ISSN 0350-5596**

Informatica

An International Journal of Computing and Informatics

Informatica **44** (2020) Number 2, pp. 113–289