Volume 42 Number 2 June 2018

ISSN 0350-5596

Informatica

An International Journal of Computing and Informatics

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.

Counterexamples in Model Checking – A Survey

Hichem Debbi
Department of Computer Science, University of Mohamed Boudiaf, M'sila, Algeria
E-mail: hichem.debbi@gmail.com

Overview paper

Keywords: model checking, counterexamples, debugging

Received: December 9, 2016

Model checking is a formal method used for the verification of finite-state systems. Given a system model and such specification, which is a set of formal properties, the model checker verifies whether or not the model meets the specification. One of the major advantages of model checking over other formal methods its ability to generate a counterexample when the model falsifies the specification. Although the main purpose of the counterexample is to help the designer to find the source of the error in complex systems design, the counterexample has been also used for many other purposes, either in the context of model checking itself or in other domains in which model checking is used. In this paper, we will survey algorithms for counterexample generation, from classical algorithms in graph theory to novel algorithms for producing small and indicative counterexamples. We will also show how counterexamples are useful for debugging, and how we can benefit from delivering counterexamples for other purposes.

Povzetek: Pregledni članek se ukvarja s protiprimeri v formalni metodi za preverjanje končnih avtomatov, tj. sistemov manjše računske moči kot Turingovi stroji. Protiprimeri koristijo snovalcem na več načinom, predvsem kot način preverjanja pravilnosti delovanja.

1 Introduction

Model checking is a formal method used for the verification of finite-state systems. Given a system model and such specification, which is a set of formal properties in temporal logics like LTL [109] and CTL [28, 52], the model checker verifies whether or not the model meets the specification. One of the major advantages of model checking over other formal methods its ability to generate a counterexample when the model falsifies such specification. The counterexample is an error trace, by analysing it, the user can locate the source of the error. The original algorithm for counterexample generation was proposed by [31], and was implemented in most symbolic model checkers. This algorithm of generating linear counterexamples for ACTL, which is a fragment of CTL, was later extended to handle arbitrary ACTL properties using the notion of tree-like counterexamples [36]. Since then, many works have addressed this issue in model checking. Counterexample generation has its origins in graph theory through the problem of fair cycle and Strongly Connected Component (SCC) detection, because model checking algorithms of temporal logics employ cycle detection and technically a finite system model is determining a transition graph [32]. The original algorithm for fair cycle detection in LTL and CTL model was proposed by [53]. Since then, many variants of this algorithm and new alternatives were proposed for LTL and CTL model checking. In section 3 we will investigate briefly the problem of fair cycles and SCCs detection.

While the early works introduced by [28, 52] have investigated the problem of generating counterexample so widely, which led to practical implementation within well-known model checkers, the open problem that emerged was the quality of the counterexample generated and how it really serves the purpose. Therefore, in the last decade many papers have considered this issue, earlier in terms of structure[36], by proposing the notion of tree-like counterexamples to handle ACTL properties, and followed later by the works investigating the quality of the counterexample mostly in terms of length to be useful later for debugging. In section 3, we will investigate the methods proposed for generating minimal, small and indicative counterexamples in conventional model checking. Model checking algorithms are classified in two main categories, explicit and symbolic. While explicit algorithms are applied directly on the transition system, symbolic algorithms employ specific data structures. Generally, the explicit algorithms are adopted for LTL model checking, whereas symbolic algorithms are adopted for CTL model checking. In this section, the algorithms for generating small counterexamples are presented with respect to each type of algorithms.

However, generating small and indicative counterexamples only is not enough for understanding the error. Therefore, counterexamples analysis is inevitable. Many works in model checking have addressed the analysis of counterexamples to better understand the error. In section 4, we will investigate the approaches that aim to help the designer to localize the source of the error given counterexamples. In this section, we consider that most of these methods range into two main categories: those that are applied on the counterexample itself without any need to other information, and those that require successful runs or witnesses to be compared with the counterexamples.

Probabilistic model checking has appeared as an extension of model checking for analyzing systems that exhibit stochastic behavior. Several case studies in several domains have been addressed from randomized distributed algorithms and network protocols to biological systems and cloud computing environments. These systems are described usually using Discrete-Time Markov Chains (DTMC), Continuous Time Markov Chains (CTMC) or Markov Decision Processes (MDP), and verified against properties specified in Probabilistic Computation Tree Logic (PCTL)[78] or Continuous Stochastic Logic (CSL) [9, 10]. In probabilistic model checking (PMC) counterexample generation has a quantitative aspect. The counterexample is a set of paths in which a path formula holds, and their accumulative probability mass violates the probability bound. Due to its specific nature, we specify section 5 for counterexample generation in probabilistic model checking. As it was done in conventional model checking, addressing the error explanation in the probabilistic model checking is highly required, especially that probabilistic counterexample consists of multiple paths instead of a single path, and it is probabilistic. So, in this section we will also investigate the counterexample analysis in PMC.

The most important thing about counterexample is that it does not just serve as a debugging tool, but it is also used to refine the model checking process itself, through Counterexample Guided Abstraction Refinement(CEGAR)[37]. CEGAR is an automatic verification method mainly proposed to tackle the problem of state-explosion problem, and it is based on the information obtained from the counterexamples generated. In section 6, we will show how counterexample contributes to this famous method of verification.

Testing is an automated method used to verify the quality of software. When we use model checking to generate test cases, this is called model-based testing. This method has known a great success in the industry through the use of famous model checkers such as SPIN, NuSMV and Java Pathfinder. Model checking is used for testing for two main reasons: first, because model checking is fully automated, and secondly and more importantly because it delivers counterexamples when the property is not satisfied. In section 7, we will show how counterexample serves as a good tool for generating test cases.

Although counterexample generation is in the heart of model checking, not all model checkers deliver counterexamples to the user. In the last section, we will review the famous tools that generate counterexamples. Section 9 concludes the paper, where some brief open problems and future directions are presented.

2 Preliminaries and definitions

Kripke Structure. A Kripke structure is a tuple $M = (AP, S, s_0, R, L)$ that consists of a set AP of atomic propositions, a set S of states, $s_0 \in S$ an initial state, a total transition relation $R \subseteq S \times S$ and a labelling function $L : W \to 2^{AP}$ that labels each state with a set of atomic propositions.

Büchi Automaton. A Büchi automaton is a tuple $B = (S, s_0, E, \sum, F)$ where S is a finite set of states, $s_0 \in S$ is the initial state, $E \subseteq S \times S$ is the transition relation, $\sum$ is a finite alphabet, and $F \subseteq S$ is the set of accepting or final states.

We use Büchi automaton to define a set of infinite words of an alphabet. A path is a sequence of states $(s_0 s_1 ... s_k)$, $k \geq 1$ such that $(s_i, s_{i+1}) \in E$ for all $1 \leq i < k$. A path $(s_0 s_1 ... s_k)$ is a cycle if $s_k = s_1$, the cycle is accepting if it contains a state in F. A path $(s_0 s_1 ... s_k .... s_l)$ where $l > k$ is accepting if $s_k ... s_l$ forms an accepting cycle. We call a path that starts at the initial state and reaches an accepting cycle an accepting path or counterexample (see Figure 1). A minimal counterexample is an accepting path with a minimal number of transitions.

Strongly Connected Component. A graph is a pair $G = (V, E)$, where V is a set of states and $E \subseteq V \times V$ is the set of edges. A path is a sequence of states $(v_1, ..., v_k)$, $k \geq 1$ such that $(v_i, v_{i+1}) \in E$ for all $1 < i \leq k$. Let π be a path, the length of π is defined by the number of transitions and is denoted by $[\pi]$. We say that we can reach a vertex u from a vertex v if there exists a path from v to u. We define a Strongly Connected Component (SCC) as a maximal set of states $C \subseteq V$ such that for every pair of vertices $u, v \in C$, u and v are mutually reachable. A SCC C is trivial if $C = \{v\}$, or otherwise C is non-trivial if for every $u, v \in C$ there is a non-trivial path from u to v.

Discrete-Time Markov Chain (DTMC) A Discrete-Time Markov Chain (DTMC) is a tuple $D = (S, s_{init}, P, L)$, such that S is a finite set of states, $s_{init} \in S$ the initial state,$P : S \times S \to [0, 1]$ represents the transition probability matrix, $L : S \to 2^{AP}$ is a labelling function that assigns to each state $s \in S$ the set $L(s)$ of atomic propositions. An infinite path σ is a sequence of states $s_0 s_1 s_2 ...$, where $P(s_i, s_{i+1}) > 0$ for all $i \geq 0$. A finite path is the finite prefix of an infinite path. We define a set of paths starting from a state s_0 by $Paths(s_0)$. The probability of a finite path is calculated as follows:

$$P(\sigma \in Paths(s_0) | s_0 s_1 ... s_n \text{is a prefix of } \sigma) = \\ \prod_{i \leq 0 < n} P(s_i, s_{i+1})$$

Linear Temporal Logic (LTL) The syntax of LTL state formula over the set AP is given as follows :

$$\varphi ::= true | a | \neg\varphi | \varphi_1 \wedge \varphi_2 | \bigcirc \varphi | \varphi_1 U \varphi_2$$

where $a \in AP$ is an atomic proposition. The Other Boolean connectives can be simply derived using the Boolean connectives $\neg$ and $\wedge$. The *eventual* operator F and the

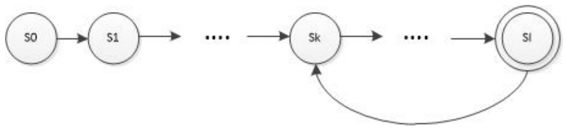


Figure 1: Accepting path (Counterexample).

always operator G can be easily derived using the temporal operator U.

Given a path $\pi = s_0 s_1 ...$ and an integer $j \geq 0$, where $\pi[j] = s_j$, such that $Words(\varphi) = \{\pi \in (2^{AP})^w)\sigma \models \varphi\}$, the semantics of LTL formulas for infinite words over 2^{AP} is given as follows:

$\pi \models true \Leftrightarrow true$
$\pi \models a \Leftrightarrow a \in L(s_0)$
$\pi \models \neg\varphi \Leftrightarrow \pi \not\models \varphi$
$\pi \models \varphi_1 \wedge \varphi_2 \Leftrightarrow s \models \varphi_1 \wedge s \models \varphi_2$
$\pi \models \bigcirc\varphi \Leftrightarrow \pi\,[1..] \models \varphi$
$\pi \models \varphi_1 \mathbf{U} \varphi_2 \Leftrightarrow \exists j \geq 0 . \pi\,[j..] \models \varphi_2 \wedge (\forall 0 \leq k < j . \pi\,[k..] \models \varphi_1)$

The semantics for the derived operators F and G is given as follows :

$\pi \models F\varphi \Leftrightarrow \exists j \geq 0 . \pi\,[j..] \models \varphi$
$\pi \models G\varphi \Leftrightarrow \forall j \geq 0 . \pi\,[j..] \models \varphi$

Verifying whether a finite state system described in Kripke structure A_M satisfies an LTL property φ reduces to the verification that $A = A_M \cap A_{\neg\varphi}$ has no accepting path, where $A_{\neg\varphi}$ refers to the Büchi automaton that violates φ, $L_\omega(A) = Words(\neg\varphi)$. We call this procedure a test of emptiness. So, in case $A_M \cap A_{\neg\varphi} \neq \emptyset$, a counterexample is generated.

Computation Tree Logic (CTL). We use the Computation Tree Logic (CTL) to specify properties of systems described using Kripke Structures. The CTL formulas are evaluated over infinite computations produced by Kripke structure K. A computation of a Kripke structure is an infinite sequence of states $s_0 s_1, ...$ such that $s_i, s_{i+1} \in R$ for all $i \in \mathbb{N}$. We denote by $Paths(s)$ the set of all paths starting at s. The syntax of CTL state formula over the set AP is given as follows:

$$\phi ::= true|a|\neg\phi|\phi_1 \wedge \phi_2|\exists\varphi|\forall\varphi$$

where $a \in AP$ is an atomic proposition and φ is a path formula. The path formulas are formed according to the following grammar:

$$\varphi ::= \bigcirc\phi|\phi_1 U \phi_2$$

We denote by $K, s \models \phi$ the satisfaction of CTL formula at a state s of K. The semantics defined by the satisfaction relation for a state formula is given as follows

$K, s \models true \Leftrightarrow true$
$K, s \models a \Leftrightarrow a \in L(s)$
$K, s \models \neg\phi \Leftrightarrow s \not\models \phi$
$K, s \models \phi_1 \wedge \phi_2 \Leftrightarrow s \models \phi_1 \wedge s \models \phi_2$
$K, s \models \exists\varphi \Leftrightarrow$ for some $\pi \in Paths(s), \pi \models \varphi$
$K, s \models \forall\varphi \Leftrightarrow$ for all $\pi \in Paths(s), \pi \models \varphi$

Given a path $\pi = s_0 s_1 ...$ and an integer $i \geq 0$, where $\pi[i] = s_i$, the semantics of path formulas is given as follows:

$K, \pi \models \bigcirc\phi \Leftrightarrow \pi\,[1] \models \phi$
$K, \pi \models \phi_1 \mathbf{U} \phi_2 \Leftrightarrow \exists j \geq 0 . \pi\,[j] \models \phi_2 \wedge (\forall 0 \leq k < j . \pi\,[k] \models \phi_1)$

In case the Kripke structure violates the specification $K \not\models \phi$, a counterexample will be generated.

Both LTL and CTL are considered as sub-logics or fragments of the logic CTL* [28, 52]. CTL is the subset of CTL* where each path operator $\bigcirc$ and U must be immediately preceded by path quantifiers $\forall$ or $\exists$, whereas LTL is the subset of CTL* that consists of formulas that have the form $\forall f$, where f is a path formula in which the only state formulas are just atomic propositions [32]. $ACTL$ is the analogue fragment of CTL and thus of CTL*, where the only quantifier allowed is $\forall$. Using CTL* we can express formulas of the form $\forall(FGp) \vee \forall G(\exists Fp)$, which is a disjunction of LTL and CTL formula.

Probabilistic Computation Tree Logic (PCTL). Probabilistic Computation Tree Logic (PCTL) [78] has appeared as an extension of CTL for the specification of systems that exhibit stochastic behavior. We use the PCTL to define quantitative properties of DTMCs. PCTL state formulas are formed according to the following grammar:

$$\phi ::= true|a|\neg\phi|\phi_1 \wedge \phi_2|\mathbf{P}_{\sim p}(\varphi)$$

Where $a \in AP$ is an atomic proposition, φ is a path formula, $\mathbf{P}$ is a probability threshold operator, $\sim \in \{<, \leq, >, \geq\}$ is a comparison operator, and p is a probability threshold. The path formulas φ are formed according to the following grammar:

$$\varphi ::= \phi_1 \mathbf{U} \phi_2|\phi_1 \mathbf{W} \phi_2|\phi_1 \mathbf{U}^{\leq n} \phi_2|\phi_1 \mathbf{W}^{\leq n} \phi_2$$

Where ϕ_1 and ϕ_2 are state formulas and $n \in \mathbb{N}$. As in CTL, the temporal operators (**U** for strong until, **W** for weak (unless) until and their bounded variants) are required to be immediately preceded by the operator **P**. The PCTL formula is a state formula, where path formulas only occur inside the operator **P**. The operator **P** can be seen as a quantification operator for both the operators $\forall$ (universal quantification) and $\exists$ (existential quantification), since the properties are representing quantitative requirements.

The semantics of a PCTL formula over a state s (or a path σ) in a DTMC model $D = (S, s_{init}, P, L)$ can be defined by a satisfaction relation denoted by $\models$. The satisfaction of $\mathbf{P}_{\sim p}(\varphi)$ on DTMC depends on the probability mass of a set of paths satisfying φ. This set is considered as a countable union of cylinder sets, so that, its measurability is ensured.

The semantics of PCTL state formulas for DTMC is defined as follows:

$s \models true \Leftrightarrow true$
$s \models a \Leftrightarrow a \in L(s)$
$s \models \neg\phi \Leftrightarrow s \not\models \phi$
$s \models \phi_1 \wedge \phi_2 \Leftrightarrow s \models \phi_1 \wedge s \models \phi_2$
$s \models \mathbf{P}_{\sim p}(\varphi) \Leftrightarrow P(s \models \varphi) \sim p$

Given a path $\sigma = s_0 s_1 ...$ in D and an integer $j \geq 0$, where $\sigma[j] = s_j$, the semantics of PCTL path formulas for DTMC is defined as for CTL as follows:

$\sigma \models \phi_1 \mathbf{U} \phi_2 \Leftrightarrow \exists j \geq 0. \sigma\,[j] \models \phi_2 \wedge (\forall 0 \leq k < j. \sigma\,[k] \models \phi_1)$
$\sigma \models \phi_1 \mathbf{W} \phi_2 \Leftrightarrow \sigma \models \phi_1 \mathbf{U} \phi_2 \vee (\forall k \geq 0. \sigma\,[k] \models \phi_1)$
$\sigma \models \phi_1 \mathbf{U}^{\leq n} \phi_2 \Leftrightarrow \exists 0 \leq j \leq n. \sigma\,[j] \models \phi_2 \wedge (\forall 0 \leq k < j. \sigma\,[k] \models \phi_1)$
$\sigma \models \phi_1 \mathbf{W}^{\leq n} \phi_2 \Leftrightarrow \sigma \models \phi_1 \mathbf{U}^{\leq n} \phi_2 \vee (\forall 0 \leq k \leq n. \sigma\,[k] \models \phi_1)$

For specifying properties of CTMC, we use the Continuous Stochastic Logic (CSL). CSL has the same syntax and semantics as PCTL, except that in CSL the time bound in bounded until formula can be presented as an interval of non-negative reals. Before verifying CSL properties over CTMC, the CTMC has to be transformed to its embedded DTMC. Therefore, further description of CTMC model checking is beyond the scope of this paper. We refer to [9, 10] for further details.

Generally, two types of properties can be expressed using temporal logics: *Safety* and *Liveness*. Safety proprieties state that something bad never happens, a simple example of that is the LTL formula $G\neg error$ that means that error never occurs. Liveness properties state that something good eventually happens, a simple example of that is the CTL formula $(\forall G req \rightarrow \forall F grant)$ that means that every request is eventually granted.

3 Counterexamples generation

Counterexample generation has its origins in graph theory through the problem of cycle detection. Cycle detection is an important issue in the heart of model checking, either explicit or symbolic model checking. To deal with this issue, various algorithms were proposed for both LTL and CTL model checking. Explicit state model checking is based on Büchi automaton, which is a type of ω-automata. The fairness condition relies on several sets of accepting states, where the acceptance condition is visiting the acceptance set infinitely often. So, a run is accepting if only if it contains a state in every accepting set infinitely often. As a result, the emptiness of the language is based on checking the non-existence of the fair cycle or equivalently the fair non-trivial strongly connected component (SCC) that intersects each accepting set. In the case of non-emptiness, the accepting run is a sign of property failure, and as a result it is rendered as an error trace. We call this error trace a counterexample. So, the counterexample is typically presented by a finite stem followed by a finite cycle. Several algorithms were proposed to find counterexamples in reasonable time, where finding the shortest counterexample has been proved to be a NP-Complete problem [31, 82].

To find fair SCCs, Depth First Search (DFS) and Breadth First Search (BFS) algorithms are used. The main algorithm employing DFS is the Tarjan's algorithm [126] that is based on manipulating the states of the graph explicitly. This algorithm is used to generate linear counterexamples in LTL verification and showed promising results [43, 129]. It is also adopted in probabilistic model checking to generate probabilistic counterexamples for lower-bounded properties, through finding bottom strongly connected components (BSCCs)[5]. BSCC is defined as an SCC B from which no state outside B is reachable from B. Finding the set of BSCCs over the probabilistic models is an important issue for the verification of PCTL and CSL properties. Tarjan's algorithm runs in linear time, but as the number of states grows, it simply becomes infeasible. As a result, the symbolic-based algorithms are proposed as a solution.

In contrast to explicit algorithms, symbolic algorithms [17, 19] employ BFS and can describe large sets in a compact manner using characteristic functions. Several symbolic algorithms were proposed for computing the set of states that contains all the fair SCCs, without enumerating them [32, 84, 128]. We refer to these algorithms as SCC-hull algorithms. Currently, most of the symbolic model checkers are employing Emerson's algorithm due to its high performance, and it was proven by [58] that both of the algorithms [52] and [31] can work in a complementary way. Other works [83, 136] proposed algorithms based on enumerating the SCCs, we refer to these algorithms as symbolic SCC-enumeration algorithms.

Different approaches for generating counterexamples are proposed regarding the two types presented before. Clarke et al. [31] proposed a hull-based approach based on Emerson's algorithm by searching a cycle in a fair SCC close to the initial state. Another approach by Hojati [84] was also employed by other works for generating counterexamples that use isolations techniques of the SCCs [95]. Using Emerson's algorithm in a combinatory way with SCC-Enumeration algorithm is possible, but is still not guaranteed to get a counterexample of short length. Ravi et al. [111] introduced a careful analysis of each type of these algorithms. Since there is no guarantee to find terminal SCCs close to the initial state, finding short counterexamples was still a trade-off and an open problem, and thus it was investigated later by many researchers in both explicit and symbolic model checking.

3.1 Short counterexamples in explicit-state model checking

A counterexample in the Büchi automaton is a path $\sigma = \beta\gamma$ where β is a path without loop from the initial state to an accepting state, and γ is a loop around this accepting state. So that, a minimal counterexample is simply a counterexample with a minimal number of transitions. More for-

```
procedure DFS(s)
    Mark($\langle s, 0\rangle$)
    for each successor $t$ of $s$ do
        if $\langle t, 0\rangle$ not marked then
            DFS(t)
        end if
    end for
    if accepting(s) then $seed := s$; NDFS(s)
    end if
end procedure
procedure NDFS(s)
    Mark($\langle s, 1\rangle$)
    for each successor $t$ of $s$ do
        if $\langle t, 1\rangle$ not marked then
            NDFS(t)
        else
            if (t==seed) then report cycle
            end if
        end if
    end for
end procedure
```

Figure 2: Nested Depth First Search Algorithm[130].

mally, a counterexample $\sigma = \beta\gamma$ is minimal if $(|\beta|+|\gamma|) \leq (|\beta'|+|\gamma'|)$ for any path $\sigma' = \beta'\gamma'$. With respect to this definition, a counterexample has at least one transition. Many algorithms consider the issue of generating counterexamples given Büchi automaton [130, 85, 112]. All these works employ Nested-Depth First Search (NDFS), but they are not capable of finding a minimal counterexample. A basic NDFS algorithm proposed by [130] is depicted in Figure 2. The algorithm is based on computing the accepting states by performing a simple search, once an accepting state is found, another search is performed to find an accepting cycle through it.

Although minimal counterexamples can be computed in polynomial time using minimal paths algorithms, the main drawback, in fact, is the memory, where the resulting Büchi automaton to be checked for emptiness is usually very huge, the thing that makes storing all the minimal paths to be compared so difficult.

Recently, new methods were proposed to compute minimal counterexample in Büchi automaton [77, 64, 63]. Hansen and Kervinen [77] proposed a DFS algorithm that runs in $O(n^2)$ and they showed that $O(n \log n)$ is sufficient, although DFS algorithms are memory consuming in general. This is due to the optimizations added using interleaving. Since the algorithms are based on exploring transitions backwards, adapting this method in practice is very difficult, especially by considering some restrictions. While this method requires more memory than the model checker SPIN does, [64, 63] proposed a method that does not use more memory than SPIN does. While the first one uses DFS and its time complexity is exponential [64], Gastin and Moro proposed a BFS algorithm with some optimizations able of computing the minimal counterexample in polynomial time [63]. Hansen et al. [76] also proposed a method for computing minimal counterexamples based on Dijkstra algorithm for detecting strongly connected components. A novel approach was proposed by [93] for generating short counterexamples based on analyzing the entire model and defining which events have more contribution to the error, these events are called crucial. In addition to generating short counterexamples, the technique helps with reducing the state space. The main drawback of this method is how to determine if such set of events are crucial and really led to the error.

3.2 Short counterexamples in symbolic model checking

The original algorithm for counterexample generation in symbolic model checking was proposed by [31] and was implemented in most symbolic model checkers. This algorithm of generating linear counterexamples for the linear fragment of ACTL was later extended to handle arbitrary ACTL properties using the notion of tree-like counterexamples [36]. The authors realized that linear counterexamples are very weak for ACTL, and thus they proposed to generate tree-like Kripke structure instead, which is proven to be a viable counterexample[36, 38]. Formally, a tree-like counterexample is a a directed tree whose SCCs are either cycles or simple nodes. Figure 3 shows an example of a tree-like counterexample for the ACTL property $\forall G \neg a \vee \forall F \neg b$. As we see in the figure, the counterexample consists of two paths refuting both subformulas. The first path leads to a state that satisfies a, whereas the second path, which is expected to be an infinite one, along which b always holds. The generic algorithm for generating tree-like counterexamples as proposed in [36] is depicted in Figure 4.

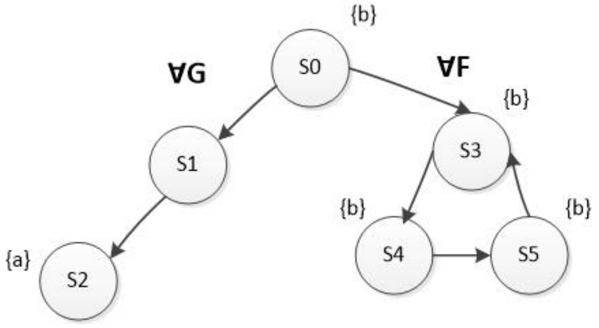


Figure 3: A tree-like counterexample for $\forall G \neg a \vee \forall F \neg b$.

The counterexample is constructed from an indexed Kripke structure K^{ω} that is obtained by creating isomorphic copy for each state in the original Kriple structure K, whereby no repeated state can be found. This process is called path *unraveling*. The algorithm traverses the specification formula in depth manner, where each subformula is evaluated recursively. The symbol O refers to temporal operator, and C is a global variable that is used in unraveling through denoting index of states.

The algorithm outputs a sequence of *descriptors* of the form $< s_0, .., s_n >$(path descriptor) and $< s_0, .., s_n, s_0 >$ (loop descriptor), where $\bigcup\{desc1, desc2\}$ describes a finite path leading to a cycle. The tree-like counterexample will be then $\bigcup\prod$, where $\prod$ refers to the set of descriptors generated by CEX algorithm. The set of descriptors for the example in Figure 3 would be: $< s_0, s_1, s_2 >$, $< s_0, s_3 >$ and $< s_3, s_4, s_5 >^{\omega}$.

```
procedure CEX($K$, $s_0^i$, $\varphi$)
    case $\varphi$ of
    $\underline{\varphi_1 \vee \varphi_2}$:
    CEX(K, $s_0^i$, $\varphi_1$)
    CEX(K, $s_0^i$, $\varphi_2$)
    $\underline{\wedge_{i \geq 1} \varphi_i}$:
    $\underline{\varphi_1 \wedge \varphi_2}$:
    Select $j$ such that $K, s_0 \not\models \varphi j$
    CEX(K, $s_0^i$, $\varphi_j$)
    $\underline{\forall \mathbf{O}(\psi_1, ..., \psi_k)}$:
    Determine $\sigma = s_0, ..., s_N, ..., s_{N+M}$ such that $K, \sigma \not\models \mathbf{O}(\psi_1, ..., \psi_k)$
    desc1:= $\langle s_0^i, unravel(C, s_1, ..., s_N)\rangle$
    desc2:= $\langle unravel(C + N, s_N, ..., s_{N+M})\rangle^{\omega}$
    return desc1 and desc2
    $C := C + N + M + 1$
    for all states $p \in \bigcup\{desc1, desc2\}$ do
        for $j \in \{1, ..., k\}$ do
            if $K, p \not\models \psi_j$ then
                CEX($K, p, \psi_j$)
            end if
        end for
    end for
    end case
end procedure
```

Figure 4: The generic counterexample algorithm for ACTL[36].

After these works of Clarke et al., many works have addressed the issue of computing short counterexamples in symbolic model checking [117, 29, 108]. Schuppan et al. [117] proposed some criteria that should be met for the Büchi automaton to accept shortest counterexamples. They proved that these criteria are satisfied in the approach proposed by [29] just for future time LTL specification, and thus they proposed an approach that meets the criteria proposed for LTL specifications with past. The algorithm proposed employs breadth-first reachability check with Binary Decision Diagrams(BDD)-based symbolic model checker.

The authors in [108] proposed a black-box based technique that masks some parts of the system in order to give an understandable counterexample to the designer. So the work does not just tend to produce minimal counterexamples, but also, it delivers small indicative counterexample of good quality to be analyzed in order to get the source of the error. The major drawback of this method is that the generalization of counterexample generation from symbolic model checking to black box model checking, could lead to non-uniform counterexamples that do not meet the behavior of the system intended. While all of these works are applied to unbounded model checking [117, 108], the works [122, 120, 113] consider bounded model checking, through lifting assignments produced by a SAT solver, where the quality of the counterexample generated depends on the SAT solver in use. Other works have investigated the use of heuristics algorithms for generating counterexamples [124, 50]. Although heuristics were not widely used, they gave pretty good results and were also used later for generating probabilistic counterexamples.

4 Counterexamples analysis and debugging

One of the major advantages of model checking over other formal methods is its ability to generate a counterexample when the model falsifies such specification. The counterexample represents an error trace; by analyzing it the user can locate the source of the error, and as Clarke wrote:"The counterexamples are invaluable in debugging complex systems. Some people use model checking just for this feature" [27].

However, generating small and indicative counterexamples only is not enough for understanding the error. Therefore, counterexamples explanation is inevitable. Error explanation is the task of discovering why the system exhibits this error trace. Many works have addressed the automatic analysis of counterexamples to better understand the failure. Error explanation ranges in two main categories. The first is based on the error trace itself, through considering the small number of changes that have to be made in order to ensure that the given counterexample is no longer exhibited, and thus, these changes represent the sources of the error. The second is based on comparing successful executions with the erroneous one in order to find the differences, and thus those differences are considered as candidate causes for the error. Kumar et al. [97] have introduced a careful analysis of the complexity of each type. For the first type, they showed using three models (Mealy machines, extended finite state machines, and pushdown automaton) that this problem is NP-complete. For the second type, they provided a polynomial algorithm using Mealy machines and pushdown automaton, but solving the problem was difficult with extended finite state machines.

Error explanation methods are successfully integrated into model checkers such as SLAM [12] and Java PathFinder JP [16]. SLAM takes less execution time than JP, and can achieve completeness in finding the causes, but according to Groce [67], this also could be harmful. The error explanation process has many drawbacks; the main one is that the counterexample consists usually of a huge number of states and transitions and involves many variables. The second is that model checker usually floods the designer with multiple counterexamples, without any kind of classification. This makes challenging the task of choosing a helpful counterexample for debugging purposes. Besides, a single counterexample it might not be enough to understand the behavior of the system. Analyzing a set of counterexamples together is an option but the problem is that it requires much effort, and even though, the set of counterexamples to be analyzed could contain the same diagnostic information, which may make analyzing this set of counterexamples a waste of time. The last and the most important problem in error explanation is that not all the events that occur on the error trace are of importance for the designer, so locating critical events is the goal behind error explanation. In this section, we survey some works with respect to the two categories.

4.1 Computing the minimal number of changes

Jin et al. [92] proposed an algorithm for analyzing the counterexamples based on the local information, by segmenting the events of the counterexamples in two main segments, fated and free. The fated segments refer to the events that obviously have to occur in the executions, and the free segments refer to the events that should be avoided for the error not to occur, and thus they are candidate to be causes. Fated and free segments are computed with respect to input variables in the system, where they are classified into controlling and non-controlling. While controlling variables are considered to be critical, and have more control on the environment, the non-controlling variables have less importance. So that, fated segments are determined with respect to controlling variables, whereas free segments are determined with respect to non-controlling ones.

Wang et al. [132] also proposed a method that works just on the failed execution path without considering successful ones. The idea is about looking at the predicates candidate for causing the failure in the error trace. To do so, they use weakest pre-condition computation, the technique that is widely used in predicate abstraction. This computation aims to find the minimal number of conditions that should be met in order to not let the program violate the assertion. This results in a set of predicates that contradict with each other. By comparing how these predicates contradict to each other, we can find the cause for the assertion failed and map it back to the real code. Many similar works also provided error explanation methods in the context of C programs [137, 127, 138].

Using the notion of causality by Halpern and Pearl [74], Beer et al. [88] introduced a polynomial-time algorithm for explaining LTL counterexamples that was implemented as a feature in the IBM formal verification platform RuleBase PE. Given the error trace, the causes for the violation are highlighted visually as red dots on the error trace itself. The question asked was: what values of signals on the trace cause it to falsify the specification? Following the definition of Halpern and Pearl, they refer to such a set of pairs of state-variable as bottom-valued pairs whose values should be switched to make such state-variable pair critical. The pair is said to be critical if changing the value of the variable in this state no longer produces a counterexample. This pair represents the cause for the first failure of the LTL formula given the error trace, where they argue that the first failure is the most relevant to the user. Nevertheless, the algorithm computes an over-approximation of the set of causes not just the first cause that occurred.

Let φ be an LTL formula in Negation Normal Form (NNF) and $\pi = s0, s1, ..., sk$ a counterexample for it. The algorithm for computing the approximate set of causes given φ and π is depicted in Figure 5. The procedure invokes each time a function *val* for evaluating sub-formulas of φ on the path. The procedure is executed recursively given the formula φ until it reaches the proposition level, where the cause is finally rendered as a pair $\langle s_i, p\rangle$/$\langle s_i, \neg p\rangle$, where s_i refers to the current state.

Let us consider the formula : $G((\neg START \wedge \neg STATUS\_VALID \wedge END) \rightarrow [\neg START \mathbf{U} STATUS\_VALID])$. The result of executing the RuleBase PE implementation of the algorithm on this formula is shown in Figure 6. The red dots refer to the relevant causes for the error. Where some variables are not critical for the failure, others can be critical, which means that switching their values alone could result in mitigating the violation. For instance, in state 9, $START$ precedes $STATUS\_VALID$, by switching the value of $START$ from 1 to 0 in state 9, the formula would not fail anymore given this counterexample.

4.2 Comparing counterexamples with successful runs

This is the most adopted method for error explanation that is successfully featured in many model checkers such as SLAM and Java PathFinder. Groce et al. [70] have proposed an approach for counterexamples explanation based on computing a set of faulty runs called negatives, in which the counterexample is included, and comparing it to a set of correct runs called positives. Analyzing the common features and differences could lead to getting a useful diagnostic information. Their algorithms were implemented in JAVA pathfinder. Based on Lewis counterfactual theory of causality [105] and distance metrics, Groce [68] has proposed a semi-automated approach for isolating errors in ANSI C programs, by considering the alternative worlds as programs executions and the events as propositions about those executions. The approach relies on finding causal dependencies between predicates of a program. A predicate a is causally dependent on b given the faulty execution, if only if the executions in which the removal of a cause a also removes the effect b are more likely than the executions where a and b do not appear together. For finding these traces, which are as close as possible to the faulty one, the authors employed distance metric. A description of their

```
procedure CAUSES($\varphi$, $\pi^i$)
    Case $\varphi$ of
    $p$:
    if $p \notin s_i$ then
        return $\langle s_i, p\rangle$
    end if
    $\neg p$:
    if $p \in s_i$ then
        return $\langle s_i, p\rangle$
    end if
    $\mathbf{X}\varphi$:
    if $i < k$ then return Causes($\varphi$, $\pi^{i+1}$)
    end if
    $\varphi \wedge \psi$:
    return Causes($\varphi$, $\pi^i$) $\cup$ Causes($\psi$, $\pi^i$)
    $\varphi \vee \psi$:
    if $val(\varphi, \pi^i) = 0$ and $val(\varphi, \psi^i) = 0$ then
        return Causes($\varphi$, $\pi^i$) $\cup$ Causes($\psi$, $\pi^i$)
    end if
    $\mathbf{G}\varphi$:
    if $val(\varphi, \pi^i) = 0$ then
        return Causes($\varphi$, $\pi^i$)
    else
        if $val(\varphi, \pi^i) = 1$ and $i < k$ and $val(\mathbf{XG}\varphi, \pi^i) = 1$ then
            return Causes($\mathbf{G}\varphi$, $\pi^{i+1}$)
        end if
    end if
    $\phi\mathbf{U}\psi$:
    if $val(\varphi, \pi^i) = 0$ and $val(\psi, \pi^i) = 0$ then
        return Causes($\varphi$, $\pi^i$) $\cup$ Causes($\psi$, $\pi^i$)
        if $val(\varphi, \pi^i) = 1$ and $val(\psi, \pi^i) = 0$ and $i = k$ then
            return Causes($\psi$, $\pi^i$)
        end if
        if $val(\varphi, \pi^i) = 1$ and $val(\psi, \pi^i) = 0$ and $i < k$ and $val(\mathbf{X}[\varphi\mathbf{U}\psi], \pi^i) = 0$ then
            return Causes($\psi$, $\pi^i$) $\cup$ Causes($\pi^{i+1}$, [$\varphi\mathbf{U}\psi$])
        end if
    end if
end procedure
```

Figure 5: Causes generation algorithm given a counterexample[88].

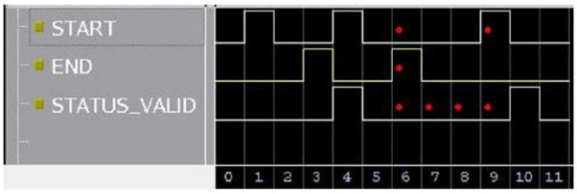


Figure 6: Explanations on counterexample as red dots[88].

approach is depicted in Figure 7.

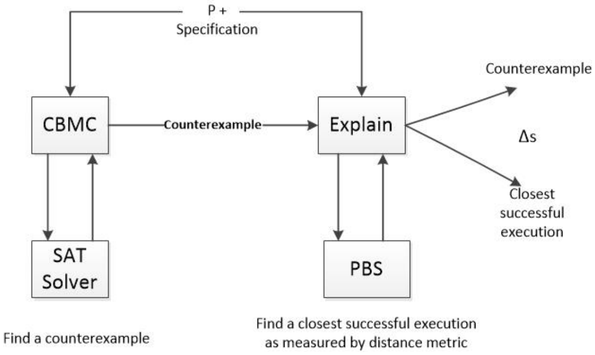
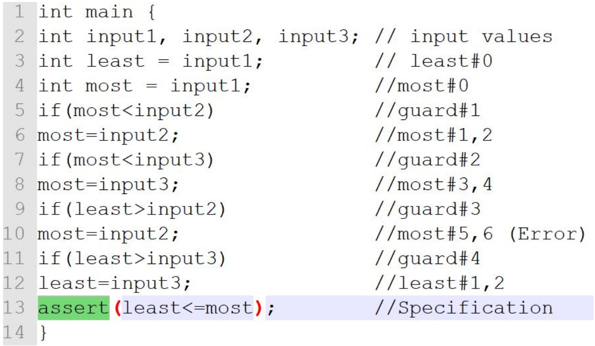


Figure 7: Explanation using distance metric[68].

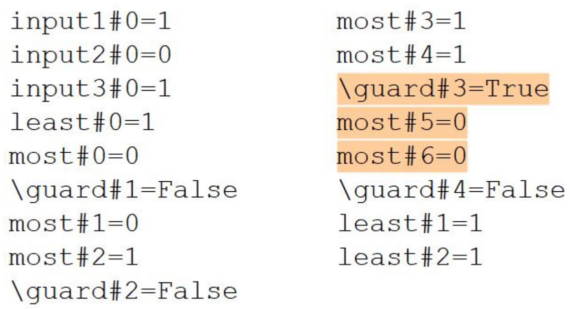
Given a program P and its specification, the model checker CBMC is used to generate a counterexample through using SAT solver, where the counterexample represents a finite execution of P. The explain tool[69] gets the counterexample generated from CBMC together with the P and its specification. It generates first a set of executions that do not violate the specification, and then using PBS solver [7], it tries to find the closest execution to the counterexample. Finally, the distance metric is computed, and a dynamic slicing technique is applied in order to point out to the most relevant assignments in the program that had contributed the most to the error.



```
int main {
int input1, input2, input3; // input values
int least = input1;         // least#0
int most = input1;          //most#0
if(most<input2)             //guard#1
most=input2;                //most#1,2
if(most<input3)             //guard#2
most=input3;                //most#3,4
if(least>input2)            //guard#3
most=input2;                //most#5,6 (Error)
if(least>input3)            //guard#4
least=input3;               //least#1,2
assert(least<=most);        //Specification
}
```

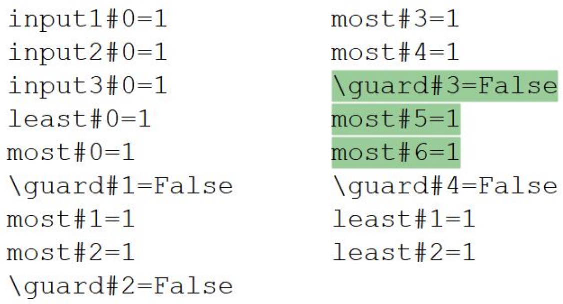
Figure 8: An example of C program.

We introduce here a brief explanation of this approach through a running example on a C code as indicated in [68]. The C program is depicted in Figure 8. For input values (1,0,1), a counterexample is rendered in a set of assignments form named Static Single Assignment(SSA),



```
input1#0=1           most#3=1
input2#0=0           most#4=1
input3#0=1           \guard#3=True
least#0=1            most#5=0
most#0=0             most#6=0
\guard#1=False       \guard#4=False
most#1=0             least#1=1
most#2=1             least#2=1
\guard#2=False
```

Figure 9: Counterexample values.

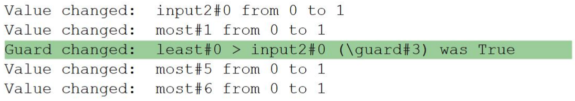


```
input1#0=1           most#3=1
input2#0=1           most#4=1
input3#0=1           \guard#3=False
least#0=1            most#5=1
most#0=1             most#6=1
\guard#1=False       \guard#4=False
most#1=1             least#1=1
most#2=1             least#2=1
\guard#2=False
```

Figure 10: Successful execution values.

which is a representation used by CBMC (See Figure 9). A successful execution so close to the counterexample can be found for the input values (1,1,1) (See Figure 10). The change in the value of $input2$ results in the assertion $least <= most$ being true. The differences between the two executions are presented in Figure 11. The first change is in the value of $input2$, which results in the change of $most\#1$ from 0 to 1. These two changes have of course lower importance to the following change, which concerns the non execution of $guard\#3$ that was executed in the counterexample, since $least\#0$ is no longer greater than $input2\#0$, and thus the value of $\#most6$ has changed to 1, which is considered the last change. The explanation that can be given for this counterexample, is that not executing the instruction at line 10 leads to the satisfaction of the assertion (no error occurring). This shows clearly the causal dependency of the satisfaction/violation of the assertion on executing this line of code. As a result, line 10 will be highlighted by *Explain* tool as an indication for the source of the error. The user will then understand that the this line of code should be corrected as $least = input2$.

In [22], Chaki and Groce extended the original approach for comparing a counterexample with the closest successful run through combining distance metric with predicate abstraction in order to generate explanations for abstract counterexamples. They argue that even for abstract counterexample, abstract state-space makes the explanation more informative. Renieris and Reiss [114] also introduced a method based on distance metric to select the closest correct runs to the faulty one and they provided a quantitative method for evaluating their methods.



```
Value changed:  input2#0 from 0 to 1
Value changed:  most#1 from 0 to 1
Guard changed:  least#0 > input2#0 (\guard#3) was True
Value changed:  most#5 from 0 to 1
Value changed:  most#6 from 0 to 1
```

Figure 11: Differences between the counterexample and the successful execution.

Ball et al. [11] proposed an effective approach that is currently featured in SLAM model checker. Their method is based on the same principle of finding successful runs to be compared with the counterexample. The interesting difference here is that it generates error trace per error cause, which makes the diagnostic easier, since there will not be causal dependencies in the traces generated. It is clear that this method will require the invocation of the model checker each time a cause for the error is found. Finally, the causes are reported as erroneous transitions that do not occur in any correct trace. Copty et al. [41] proposed a framework for debugging counterexamples as they refer to it as counterexample Wizard in the context of symbolic LTL. The technique employs three main capabilities: multi-value counterexample annotation, constraint-based debugging and multiple counterexample generation. But in contrast to the work by Ball et all, the model checker is not invoked each time an error cause is found, but instead, it gets all the data needed together to start the analysis.

Leue and Tabaei Befrouei [104, 103] proposed a novel approach based on computing two datasets, the *bad dataset* that represents the set of counterexamples, and the *good dataset* that represents the successful runs. Both datasets are produced using SPIN model checker. The idea is always about computing the differences between good and bad traces, but this time with the help of data mining technique called sequential pattern mining [49]. The aim behind using this technique is to extract a set of sequences of actions that are mostly to appear in the bad dataset. In concurrent systems, which are usually modeled using interleaving semantics, the unforeseen interleavings resulted from such a set of actions stand as a good indicator for the source of the error.

While all of the previous works addressed safety properties, Kumazawa and Tamai [98] attended to explain errors for liveness properties that involve more computational complexity. For that reason, the counterexample is represented as an infinite trace and not a finite one, and the witnesses to be compared with this counterexample are infinite as well. The method also employs shortest paths algorithms. Many similar works for counterexamples analysis have been done [121, 73, 40, 110, 119, 118, 56, 45].

5 Probabilistic counterexamples

Unlike the previous methods proposed for conventional model checking that generate the counterexample as a single path ending with a bad state representing the failure, the task in PMC is quite different. The counterexample in PMC is a set of evidences or diagnostic paths that satisfy path formula and their probability mass violates the probability threshold. The probabilistic counterexample is generated when a PCTL/CSL property is not satisfied. The probabilistic property $\phi = \mathbf{P}_{\leq p}(\varphi)$ is refuted when the probability mass of the paths satisfying φ exceeds the bound p. Therefore, a probabilistic counterexample for the property ϕ is formed by a set of paths starting at a state s and satisfying the path formula φ. We denote these paths by $Paths(s_0 \models \phi)$. The counterexample can be formed of a set of finite paths where each path $\sigma = s_0 s_1 ... s_n$ is a prefix of an infinite path from $Paths(s_0 \models \phi)$ satisfying the formula φ. We denote these paths by $FinitePaths(s_0 \models \phi)$.

We can get a set of probabilistic counterexamples, noted $PCX(s_0 \models \phi)$, which is a set of any combination from $FinitePaths(s_0 \models \phi)$ that their probability mass exceeds the bound p. Among all these probabilistic counterexamples, we are interested by the most indicative one. The most indicative counterexample is minimal counterexample (has the least number of paths from $FinitePaths(s_0 \models \phi)$) and its probability mass is the highest among all other minimal counterexamples. We denote the most indicative probabilistic counterexample by $MIPCX(s_0 \models \phi)$. We should note that the most indicative probabilistic counterexample may not be unique.

For the counterexample to have a high probability, it should consist of paths that carry high probabilities from $FinitePaths(s_0 \models \phi)$. The path σ having the highest probability over all these paths is called strongest path and is defined as follows: for every path $\sigma' \in FinitePaths(s_0 \models \phi) : P(\sigma) \geq P(\sigma')$. The strongest path also may not be unique.

Example Let us consider the example of DTMC shown in Figure 12 and the property $P_{\leq 0.5}(\varphi)$, where $\varphi = (a \vee b) \cup (c \wedge d)$. The property above is violated in this model ($s_0 \not\models \mathbf{P}_{\leq 0.5}(\varphi)$), since there exists a set of paths satisfying φ whose probability mass is higher than the probability bound (0.5). Any combination from $FinitePaths(s_0 \models \phi)$ having probability mass higher than 0.5, is a valid counterexample including the whole set. For instance, we can find three counterexamples:

$$P(CX_1) =$$
$$P(\{s_0s_1, s_0s_2s_3, s_0s_2s_4s_3, s_0s_2s_4s_5, s_0s_4s_5\})$$
$$= 0.25 + 0.2 + 0.09 + 0.15 + 0.12 = 0.81$$
$$P(CX_2) = P(\{s_0s_1, s_0s_2s_4s_5, s_0s_4s_5\})$$
$$= 0.25 + 0.15 + 0.12 = 0.52$$
$$P(CX_3) = P(\{s_0s_1, s_0s_2s_3, s_0s_2s_4s_5\})$$
$$= 0.25 + 0.2 + 0.15 = 0.6$$

The last probabilistic counterexample is the most indicative since it is minimal and its probability is higher than the other minimal counterexample CX_2,$P(CX_3) = 0.6 > P(CX_2)$. The strongest path is s_0s_1, which is included in the most indicative probabilistic counterexample.

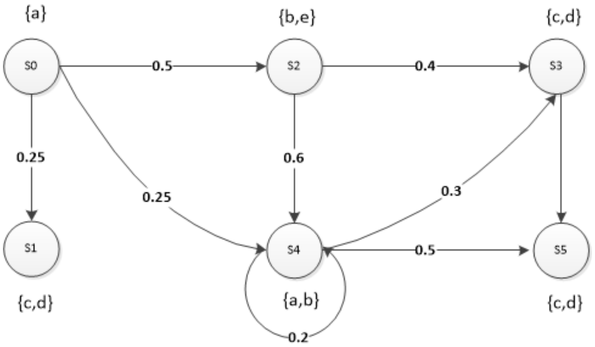


Figure 12: A DTMC.

5.1 Probabilistic counterexample generation

Various approaches for probabilistic counterexamples generation have been proposed. Aljazzar et al. [1, 3] introduced an approach for counterexample generation for DTMC and CTMC against timed reachability proprieties using heuristics and directed explicit state space search. Since resolving nondeterminism in an MDP results in a DTMC, in complementary work [4], Aljazzar and Leue proposed an approach for counterexample generation for MDPs based on existing methods for DTMC. Aljazzar and Leue introduced a complete work in [5] for generating counterexamples for DTMCs and CTMSs as what they refer to as diagnostic sub-graphs. All these works on generating indicative counterexamples have led to the development of the K* algorithm [6], an on-the-fly heuristics guided algorithm for the K shortest path problem. Comparing to classical k-shortest-paths algorithms, K* has two main advantages, it woks on-the- fly in way it avoids exploring the entire graph, and it can be guided using heuristic functions. Based on all the previous works, they built a tool DiPro [2] for generating indicative counterexamples for DTMCs, CTMCs and MDPs. This tool can be jointly used with the model checkers PRISM [81] and MRMC [94], and can render the counterexamples in text format as well as in graphical mode. These heuristic-based algorithms showed a great efficiency in terms of counterexample quality. Nevertheless, with large models, DiPro tool that implements these algorithms takes usually a long time to produce a counterexample. By running DiPro on a PIRSM model of the DTMC presented in Figure 12 against the same property, we obtain the most indicative counterexample $CX3$. The graphical representation of $CX3$ as rendered by DiPro is depicted in Figure 13. The diamonds refer to the final or end states (s_1,s_3,s_5), whereas the circles represent simple nodes s_2 and s_4. The user can navigate through the counterexample and inspect all values.

Similar to the previous works, [75] has proposed the notion of smallest most indicative counterexample that redu-

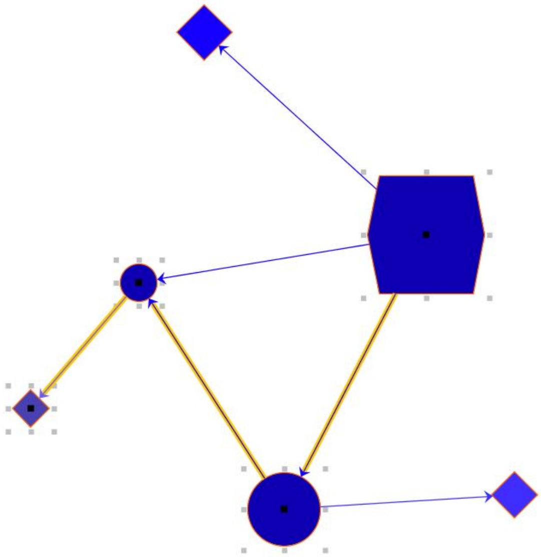


Figure 13: A counterexample generated by DiPro.

ces to the problem of finding K shortest paths. In a weighted digraph transformed from the DTMC model, and given initial state and the target states, the strongest evidences that form the counterexample are selected using extensions of K-shortest paths algorithms for an arbitrary number k. Instead of generating path-based counterexamples, [134] have proposed a novel approach for DTMCs and MDPs based on critical subsystems using SMT solvers and mixed integer linear programming. Critical subsystem is simply a part of the model (states and transitions) that are considered relevant because of its contribution to exceeding the probability bound. The problem has been shown that is NP-Complete. Another work always based on the notion of critical subsystem is proposed to deliver abstract counterexamples with less number of states and transitions using hierarchical refinement method. Based on all of these works, Jansen et al. proposed the COMICS tool for generating the critical subsystems that induce the counterexamples [90].

There are also many other works that addressed special cases for generating counterexamples in PMC. the authors of [8], proposed an approach for finding sets of evidences for bounded probabilistic LTL properties on MDP that behave differently from each other giving significant diagnostic information. While their method is also based on K-shortest path, the main contribution is about selecting the evidences or the witnesses with respect to main five criteria in addition to the high probability. While all of the previous works for counterexample generation are explicit-based, the authors in [133] proposed a symbolic method using bounded model checking. In contrast to the previous methods, this method lacks the selection of the strongest evidences first, since the selection is performed in arbitrary order. Another approach for counterexample generation that uses bounded model checking has been proposed [15]. Unlike the previous work that uses conventional SAT solvers, the authors used a SMT-solving approach in order to put some constraints on the paths selected, in order to get more abstract counterexample that consists of strongest paths. Counterexample generation for probabilistic LTL model checking has been addressed in [116] and probabilistic CEGAR has been also addressed [80]. A comprehensive representation of the counterexamples using regular expressions has been addressed in [44]. Since regular expressions deliver compact representations, they can help to deliver short counterexamples. Besides, they are widely known and easily understandable, so that they will give more benefits as a tool for error explanation.

5.2 Probabilistic counterexample analysis

Instead of relying on the state space search resulted from the parallel composition of the modules, [135] suggests to rely directly on the guarded command language used by the model checker, which is more likely and helpful for debugging purpose. To do so, the authors employ the critical subsystem technique [134] to identify the smallest set of guarded commands contributing to the error.

To analyze probabilistic counterexamples, Debbi and Bourahla [48, 47] proposed a diagnostic method based on the definition of causality by Halpern and Pearl [74] and responsibility [25]. The method proposed takes the probabilistic counterexample generated by DiPro tool and the probabilistic formula as input, and returns a set of pairs (state-variable) as candidate causes for the violation ordered with respect to their contribution to the error. So, in contrast to the previous methods, this method does not tend to generate indicative counterexamples, but it acts directly on indicative counterexamples already generated. Another similar approach for debugging probabilistic counterexamples has been introduced by [46]. It adopts the same definition of causality by Halpern and Pearl to reason formally about the causes, and then transforms the causality model into regression model using Structural Equation Modeling (SEM). SEM is a comprehensive analytical method used for testing and estimating causal relationships between variables embedded in theoretical causal model. This method helps to understand the behavior of the model through quantifying the causal effect of the variables on the violation, and the causal dependencies between them.

The same definition of causality has also been adopted to event orders for generating fault trees from probabilistic counterexamples, where the selection of traces forming the fault tree are restricted to some minimality condition [102]. To do so, Leitner-Fischer and Leue proposed the event order logic to reason about Boolean conditions on the occurrence of events, where the cause of the hazard in their context is presented as an Event Order Logic (EOL) formula, which is a conjunction of events. In [57], they extended their approach by integrating causality in explicit-state model checking algorithm to give a causal interpretation for sub- and super-sets of execution traces, the thing that could help the designer to get a better insight on the behavior of the system. They proved the applicability of their approach to many industrial size PROMELA models. They extended the causality checking approach to probabilistic counterexamples by computing the probabilities of events combination [101], but they still consider the use of causality checking of qualitative PROMELA models.

6 Counterexample guided abstraction refinement (CEGAR)

The main challenge in model checking is the state explosion problem. Dealing with this issue is in the heart of model checking, it was addressed at the beginning of model checking and not finished. Many methods were proposed to tackle this issue, the most famous are: symbolic algorithms, Partial Order Reduction (POR), Bounded Model Checking (BMC) and abstraction. Among these techniques, abstraction is considered as the most general and flexible for handling the state explosion problem [30]. Abstraction is about hiding or simplifying some details about the system to be verified, even removing some parts from it that are considered irrelevant for the property under consideration. The central idea is that verifying a simplified or an abstract model is more efficient than the entire model. Evidently, this abstraction has a price, which is losing some information, and the best abstraction methods are those that control this loss of information. Over-approximation and under-approximation are two main key concepts for this problem. Many abstraction methods have been proposed [42, 65, 106], the last one had the most attention and was adopted in the symbolic model checker NuSMV.

Abstraction can be defined by a set of abstract states $\widehat{S}$, an abstraction mapping function h that maps the states in the concrete model to $\widehat{S}$, and the set of atomic propositions AP labeling these states. Regarding the choice on $\widehat{S}$, h and AP, we distinguish three main types of abstraction : predicate abstraction [66, 115], localization reduction [99] and data abstraction [39]. Predicate abstraction is based on eliminating some variables from the program to be replaced by predicates that still serve the information about these variables. Each predicate has a Boolean variable corresponding to it, where the abstract states $\widehat{S}$ resulted are valuations of these variables. Both the abstraction mapping h between the concrete and abstract states, and the set of atomic propositions AP, are determined with respect to the truth values of these predicates. The entire abstract model can then be defined through existential abstraction. To this end, we can use BDDs, SAT solvers or theorem provers depending on the size of the program. Localization reduction and data abstraction are actually just extensions of predicate abstraction. Localization reduction aims to define a small set of variables that are considered relevant to the property in hand to be verified, these variables are called *visible*, the rest of variables that have no importance with respect to the property to be verified are called *invisible*. We should mention that this technique does not apply any abstraction on the domain of visible variables. Data abstraction deals mainly with the domains of variables by making an abstract domain for each variable. So the abstract model will be built with respect to the abstract values. For more detail on abstraction techniques, we refer to [71].

Given the possible loss of information caused by the abstraction, inventing some refinement methods of the abstract model is necessary. The most known method for abstraction refinement is Counterexample-Guided Abstraction Refinement (CEGAR) that has been proposed by [30] as a generalization of the localization reduction approach. A prototype implementation of this method in NuSMV has also been presented. In this approach, the counterexample plays the crucial role for finding the right abstract model. The process of CEGAR consists of three main steps: the first is to generate an abstract model using one of the abstractions techniques [30, 23, 33] given a formula φ. The second step is about checking the satisfaction of φ, if it is satisfied then the model checker stops and returns that the concrete or the original model satisfies the formula, if it is not satisfied, a counterexample will be generated. The counterexample generated is in the abstract model, so we have to check if it is also a valid counterexample in the concrete model, because the abstract model has different behavior comparing to the concrete one. Otherwise, the counterexample is called spurious and the abstraction must be carried out based on this counterexample. So, a spurious counterexample is an erroneous counterexample that exists only in the abstract model, not the concrete model. The final step is to refine the model until no spurious counterexample is found (see Figure 14). This is how the technique gets its name, refining the abstract model using the spurious counterexample. Refinement is an important task of CEGAR that can make the process faster and gives the appropriate results. To refine the abstract model, different partitioning algorithms are applied to abstract states. Like abstraction, partitioning the abstract states in order to eliminate the spurious counterexample can be carried out in many other ways than BDDs [30]. SAT solvers [24] or linear programming and machine learning [34] can be used to define the most relevant variables to be considered for the next abstraction.

In the literature, we find many extensions for CEGAR depending on the type of predicates and application domains: large program executions [96], non-Disjunctive abstractions [107] and propositional circumscription [89]. The CEGAR technique itself has been used to find bugs in complex and large systems [14]. The idea is based on gathering and saving information during the abstract model checking process in order to generate short counterexamples in the case of failure. This could be helpful for large models that make generating counterexamples using standard BMS intractable. CEGAR currently is implemented in many tools such as NuSMV[26], SLAM and BLAST[13].

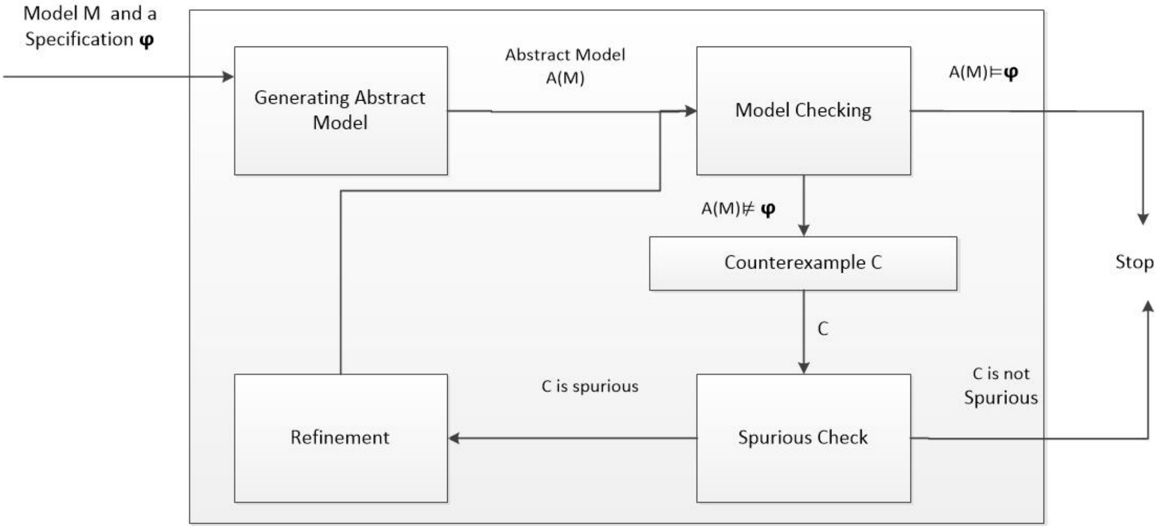


Figure 14: Counterexample Guided Abstraction Refinement Process.

7 Counterexamples for test cases generation

Counterexample generation gives the opportunity for model checking to be adopted and used in different domains, one of the domains in which the model checking has been adopted is test cases generation. Roughly speaking, testing is an automated method used to verify the quality of software. When we use model checking to generate test cases, this is called model-based testing. The use of model checking for testing is mainly subjected to the size of the software to be tested, because a suitable model must be guaranteed. The central idea of using model checking for testing [20, 55] is about interpreting counterexamples generated by the model checkers as test cases, and then test data and some expected results are extracted from these tests using such execution framework. Counterexamples are mainly used to help the designer to find the source of the error. However, they are very useful as test cases. [60].

A test describes the behavior of the test case intended: the final state, the states that should be traversed to reach the final state and so forth. In practice, it might not be possible to execute all test cases, since the software to be tested has usually a large number of behaviors. Nevertheless, there exist some techniques to help us to measure the reliability of testing. These techniques range in two main categories: first, deterministic methods (given initial state and some input, we will be certainty aware about the output), most famous methods for this category are coverage analysis and mutation analysis. Second, statistical analysis, where the reliability of the test is measured with respect to some probability distribution.

In coverage-based testing, the test purpose is specified in temporal logic and then converted to what is called a never-claim by negation; to assert that the test purpose never becomes true. So, the counterexample generated after the verification process will describe how the never-claim is violated, which is a description of how test purpose is fulfilled. Many approaches for creating never-claims based on coverage criterion (called "trap properties") [61] are proposed. Coverage criteria aim to find how such a system is exercised given a specification in order to get the states that were not traversed during the test; in this context, we call this specification a test suit. So, a full coverage is achieved if all the states of the system are covered. To create a test suite that covers all states, we need a trap property for each possible state. For example, claiming that the value of a variable is never 0: $G\neg(a = 0)$. A counterexample to such a trap property is any trace that reaches a state where $(a = 0)$.

With regard to *trap properties*, we find many variations. Gargantini and Heitmeyer addressed the coverage of software cost reduction (SCR) specifications [61]. SCR specifications are defined by tables over the events that represent the change of a value in state and lead to a new state, and conditions defined on the states. Formally, a SCR model is defined as quadruple (S, S_0, E^m, T) where S is the set of states, S_0 is the initial state set, E^m is the set of input events, and T is the transform function that maps an input event and the current state to a new one. SCR requirement properties can be used as never-claims, first by converting SCR into model checkers languages such as SPIN language (PROMELA), or SMV language, and then transform SCR tables into if-else construct in the case of using SPIN, or a case statement in the case of SMV. Another approach by Heimdahl et al. addressed the coverage of transition systems globally [79], where they consider the use of $RSML^{-e}$ as the specification language. A simple example of transition coverage criteria is of the form $G(A \wedge C \rightarrow \neg B)$, where A represents a system's state s, B represents the next state, and C is the condition that guards the transition A to B. So a counterexample for this property could be a trace that reaches a state B when C

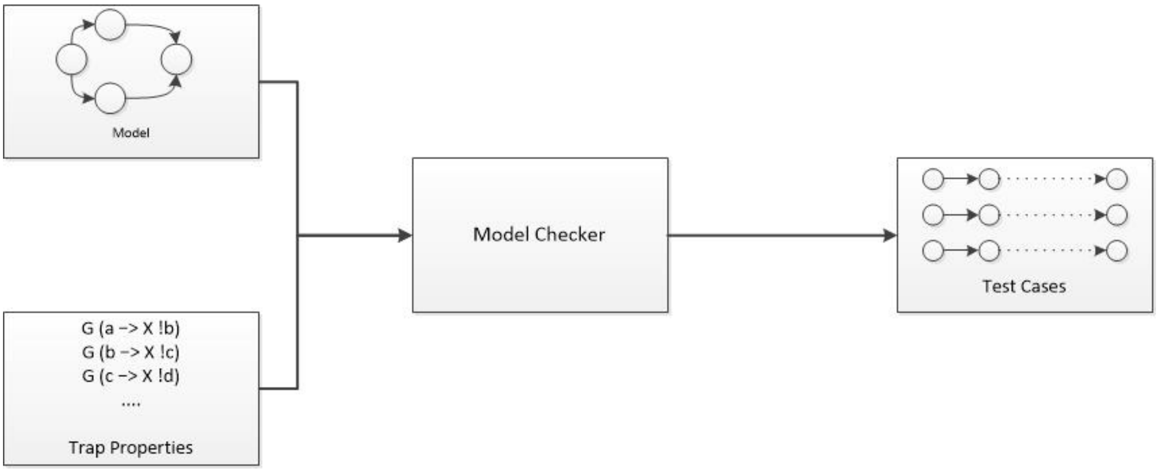


Figure 15: Coverage based test case generation [60].

evaluates to true, or a trace that reaches another state than B when C evaluates to false. Hong and Lee [87] proposed an approach based on control and data flow, where they use SMV model checker to generate counterexamples during the model checking of state-charts. The counterexample generated can be mapped into test sequence that induces information about which initial and stable states are considered. Another approach based on abstract state machines has been introduced [62]. The trap properties here will be defined over a set of rules for guarded function updates. We can see that all coverage-based approaches deal with the same thing, which is trap properties, and defer from each other in the formalism adopted.

Another approach for using requirement properties as test cases has been introduced by [54]. In this approach, each requirement has a set of tests. Trap properties can be easily derived from requirement properties under property-coverage criteria [125]. Another method that is completely different from coverage-based analysis is mutation-based analysis [18]. Mutation analysis consists of creating a set of mutants, which can be obtained by making small modifications on the original program in way these mutants lead to realistic faults. We differ between each mutant by its score, the mutant with the high score indicates high fault sensitivity. It is evident that deriving such mutant that is equivalent to the original program will have a high computational cost [91], because we have to apply all the test cases to each mutant, and all mutants should be considered. And for each mutant the model checker must be invoked. Fraser et al. [59] reviewed in detail most of these techniques and proposed several effective techniques to improve the quality of the test cases generated in model checking-based testing, especially requirements based testing, and apply them on different types of properties in many industrial case studies.

8 Counterexamples generation tools

Practically, all successful model checkers are able to output counterexamples in varying formats [38]. In this section, we will try to survey the tools supporting counterexample generation and study their effectiveness. A set of model checkers with their features are presented in Table 1.

Berkeley Lazy Abstraction Software Verification Tool (BLAST) [13] is a software model checking tool for C programs. BLAST has the ability to generate counterexamples, and furthermore, it employs CEGAR. BLAST is not just a CEGAR-based model checker, but it can be also used for generating test cases. BLAST shows promising results with safety properties of programs with a medium size.

CBMC [35] is a well-known Bounded Model Checker for AINCI C and C++ programs. CBMC performs symbolic execution on the programs and employs a SAT solver in the verification procedure, when the specification is falsified, a counterexample in the form of states with variables valuation leading to these states is rendered to the user.

JavaPathfinder(JPF) [131] is a famous software model checking tool for Java programs. JavaPathfinder is an effective virtual machine-based tool that verifies the program along all the possible executions. Due to its ability to deal with most of JAVA language features, because it runs on byte-code level, JavaPathfinder can generate a detailed report on the error in case that the property is violated. Besides, the tool gives the ability to generate test cases.

SPIN [86] is a model checker mostly known for the verification of systems that exhibit a high interaction between processes. The systems are described using Process Meta Language (PROMELA) language, and verified against properties specified in LTL. By applying a Depth-First Search algorithm on the intersection product of the model and the Büchi automaton representing the LTL formula, a counterexample is generated in case an accepted cycle is detected. SPIN offers an interactive simulator that helps to understand the cause of the failure by showing the processes and their interactions in order.

Table 1: Model checkers with their features.

Name	Model Checking				Counterexample generation	
	Programs, systems	Algorithms, methods	Modeling language	Specification language	Visualization	Form
BLAST	C programs	Predicate lazy abstraction, CEGAR	C	BLAST	No	Set of predicates
CBMC	C programs	BMC, SAT solving	C/C++	Assertions	No	Variables and valuations
JPF	Java programs	Explicit state, POR	Java	No	No	Error report
SPIN	Concurrent, distributed, asynchronous	Nested Depth First Search, POR	PROMELA	LTL	Yes	Execution path
NuSMV	Synchronous, asynchronous	BDD-based, SAT-based BMC	SMV	LTL, CTL	No	States and valuations
UPPAAL	Ral-time	On-the-fly, Symbolic	Timed automata	TCTL	Yes	Sequence of states
PRISM	Probabilistic, real-time	Graph-based, numerical	PRISM	PCTL,CSL, PTCTL	No-By DiPro	Graph
MRMC	Probabilistic	Numerical	PRISM, PEPA	PCTL,CSL, PRCTL,CSRL	No-By DiPro or COMIC	Graph

NuSMV [26] is a symbolic model checker that appeared as an extension of the Binary Decision Diagrams(BDD)-based model checker SMV. NuSMV includes both LTL and CTL for specification analysis, and combines SAT and BDD techniques for the verification. NuSMV can deliver a counterexample in XML format by indicating the states of the trace and the variables with their new values that cause the transitions.

UPPAAL [100] is a verification framework for real-time systems. The systems can be modeled as networks of timed automata extended with data types and synchronization channels, and the properties are specified using a Timed CTL(TCTL). UPPAAL can find and generate counterexamples in graphical mode as message sequence charts that indicate the events with respect to their order.

PRISM [81] is a probabilistic model checker used for the analysis of systems that exhibit stochastic behavior. The systems are described as DTMCs, CTMCs or MDPs, using guarded command language, and verified against probabilistic properties expressed in PCTL and CSL, and can be extended with rewards. Another successful probabilistic model checker extended with rewards is the Markov Reward Model Checker (MRMC) [94]. MRMC is mainly used for performance and dependability analysis. It takes the models as input files in two formats, in PRISM language or Performance Evaluation Process Algebra (PEPA). Although both model checkers have shown high effectiveness, they lack a mechanism for generating probabilistic counterexamples. Nevertheless, they have been used by recent tools (DiPro [2] and COMICS [90]) for generating and visualizing probabilistic counterexamples.

9 Conclusions and future directions

In this paper we surveyed counterexamples in model checking from different aspects. At the beginning of using model checking, counterexamples have not been treated as a particular subject, but they have been treated as a related problem to fair cycle detection algorithms, as presented in section 3. But recently, the quality of the counterexamples generated has been treated as a standalone and a fundamental problem. Many works tried to deliver short and indicative counterexamples to be used for debugging purpose. Concerning their structure, tree-like counterexamples have been proposed for the fragment of ACTL as an alternative for linear counterexamples, however, we see that this approach has not been adopted in model checkers, but instead model checkers are still based on generating simple non-branching counterexamples .

For debugging, we can conclude that approaches that require other successful runs might have some advantages over other methods based on single trace, in way that they compare many good traces to restrict the set of candidate causes. However, these methods take usually much execution time in order to select the appropriate set of traces, and besides, such traces could contain the same diagnostic information. Regardless of the debugging method in use, the challenge of visualizing the error traces and the quality of diagnoses generated to facilitate debugging is still an open issue.

For the case of counterexample generation in PMC, we have seen that the principle of counterexample generation is completely different than conventional model checking, where the presentation of counterexample is different from a work to another, from smallest and indicative set of paths to most critical sub-systems. Despite the notable advancement in generating probabilistic counterexamples that led to inventing important tools like DiPro and COMICS, unfortunately this advancement is still insufficient for debugging. Actually, it is more than important to see the techniques for counterexample generation and counterexample analysis integrated in probabilistic model checkers to get their benefit. All these techniques act on verification results of probabilistic model checkers like PRISM, so making the approaches of counterexample generation and counterexamples analysis to be performed during the model checking process itself is still an open problem. This could really have a great impact on probabilistic model checking.

We have also seen the usefulness of counterexamples for other purposes than debugging, like CEGAR and test cases generation. For CEGAR, we have seen different approaches for both abstraction and refinement. We have seen that we can benefit from using SAT solvers and theorem provers on the both sides, abstraction and refinement, thus they are very useful for CEGAR. Fast elimination of spurious counterexamples is still an active research topic. We also expect to see more works on CEGAR in PMC.

For testing, we have seen that the most useful approaches using model checking are those based on coverage and trap properties. Other approaches for testing like requirement-based analysis and mutation-based analysis have received smaller attention due to the limitations presented. Currently, coverage-based techniques are widely used in the industry. In the future, we expect to see the proposition of new approaches to enable us to test new emerging systems, which require new transformation mechanisms for enabling trap properties to be verified by model checkers to generate the counterexamples.

We should mention that such techniques can benefit from other techniques. For instance, new efficient CEGAR techniques will not only have an impact on conventional model checking, but on probabilistic model checking as well. We can also see in the future the use of probabilistic model checkers like PRISM for testing probabilistic systems. Since PRISM does not generate counterexamples, any advancement in generating indicative counterexamples could be of benefit for testing probabilistic systems. We can also see that techniques based on counterexamples like CEGAR can directly benefit from any advancement in generating small and indicative counterexamples in a considerable time.

In addition to all of this, we expect to see more works in other domains that adapt model checking techniques just for the seek of getting counterexamples. In previous works we have seen for instance that counterexamples can be mapped to UML sequence diagrams, describing states and events in the original model [51], they can be used to generate attack graphs in networks security [123], in fragmentation of services in Service-Based Applications (SBAs) [21], and they have been also used to enforce synchronizability and realizability in distributed services integration [72].

References

[1] Aljazzar, H., Hermanns, H., and Leue, S. Counterexamples for timed probabilistic reachability. In *FORMATS* (2005), LNCS, vol. 3829, Springer, Berlin, Heidelberg, pp. 177–195.

[2] Aljazzar, H., Leitner-Fischer, F., Leue, S., and Simeonov, D. Dipro - a tool for probabilistic counterexample generation. In *18th International SPIN Workshop* (2011), LNCS, vol. 6823, Springer, Berlin, Heidelberg, pp. 183–187.

[3] Aljazzar, H., and Leue, S. Extended directed search for probabilistic timed reachability. In *FORMATS* (2006), LNCS, vol. 4202, Springer, Berlin, Heidelberg, pp. 33–51.

[4] Aljazzar, H., and Leue, S. Generation of counterexamples for model checking of markov decision processes. In *International Conference on Quantitative Evaluation of Systems (QEST)* (2009), pp. 197–206.

[5] Aljazzar, H., and Leue, S. Directed explicit state-space search in the generation of counterexamples for stochastic model checking. *IEEE Trans. on Software Engineering 36*, 1 (2010), 37–60.

[6] Aljazzar, H., and Leue, S. K*: A heuristic search algorithm for finding the k shortest paths. *Artificial Intelligence 175*, 18 (2011), 2129 – 2154.

[7] Aloul, F., Ramani, A., Markov, I., and Sakallah, K. Pbs: A backtrack search pseudo boolean solver. In *Symposium on the Theory and Applications of Satisfiability Testing (SAT)* (2002), pp. 346–353.

[8] Andres, M. E., DArgenio, P., and van Rossum, P. Significant diagnostic counterexamples in probabilistic model checking. In *Haifa Verification Conference* (2008), pp. 129–148.

[9] Aziz, A., Sanwal, K., Singhal, V., and Brayton, R. Model-checking continuous-time markov chains. *ACM Transactions on Computational Logic 1*, 1 (2000), 162–170.

[10] Baier, C., Haverkort, B., Hermanns, H., and Katoen, J.-P. Model checking algorithms for continuous-time markov chains. *IEEE Transactions on Software Engineering 29*, 7 (2003), 524–541.

[11] Ball, T., Naik, M., and Rajamani, S. From symptom to cause: Localizing errors in counterexample traces. In *ACM Symposium on the Principles of Programming Languages* (2003), pp. 97–105.

[12] Ball, T., and Rajamani, S. The slam project: Debugging system software via static analysis. In *ACM Symposium on the Principles of Programming Languages* (2002), pp. 1–3.

[13] Beyer, D., Henzinger, T., Jhala, R., and Majumdar, R. The software model checker blast: Applications to software engineering. *International Journal on Software Tools for Technology Transfer (STTT) 9*, 5 (2007), 505–525.

[14] Bjesse, P., and Kukula, J. Using counterexample guided abstraction refinement to find complex bugs. In *Design, Automation and Test in European Conference and Exhibition* (2004), pp. 156–161.

[15] Braitling, B., and Wimmer, R. Counterexample generation for markov chains using smt-based bounded model checking. In *Formal Techniques for Distributed Systems* (2011), LNCS, vol. 6722, Springer, Berlin, Heidelberg, pp. 75–89.

[16] Brat, G., Havelund, K., Park, S., and Visser, W. Java pathfinder a second generation of a java model checker. In *Workshop on Advances in Verification* (2000).

[17] Bryant, R. E. Graph-based algorithms for boolean function manipulation. *IEEE Trans. Comput 35*, 8 (1986), 677–691.

[18] Budd, T., and Gopal, A. Program testing by specification mutation. *Journal Computer Languages 10*, 1 (1985), 63–73.

[19] Burch, J. R., Clarke, E. M., McMillan, K. L., Dill, D. L., and Hwang, L. J. Symbolic model checking: 1020 states and beyond. *Information and Computation 98*, 2 (1992), 142–170.

[20] Callahan, J., Schneider, F., and Easterbrook, S. Automated software testing using model checking. In *SPIN Workshop* (1996).

[21] Chabane, Y., Hantry, F., and Hacid, M. Querying and splitting techniques for sba: A model checking based approach. In *Emerging Intelligent Technologies in Industry* (2011), SCI 369, Springer-Verlag, Berlin, Heidelberg, pp. 105–122.

[22] Chaki, S., and Groce, A. Explaining abstract counterexamples. In *SIGSOFT04/FSE* (2004), pp. 73–82.

[23] Chauhan, P., Clarke, E., Kukula, J., Sapra, S., Veith, H., and D.Wang. Automated abstraction refinement for model checking large state spaces using sat based conflict analysis. In *Formal Methods in Computer Aided Design(FMCAD)* (2002), LNCS, vol. 2517, Springer, Berlin, Heidelberg, pp. 33–51.

[24] Chauhan, P., Clarke, E., Kukula, J., Sapra, S., Veith, H., and D.Wang. Automated abstraction refinement for model checking large state spaces using sat based conflict analysis. In *FMCAD 2002* (2002), LNCS, vol. 2517, Springer-Verlag, Berlin, Heidelberg, pp. 33–51.

[25] Chockler, H., and Halpern, J. Y. Responsibility and blame: a structural model approach. *Journal of Artificial Intelligence Research (JAIR) 22*, 1 (2004), 93–115.

[26] Cimatti, A., Clarke, E., Giunchiglia, F., and Roveri., M. Nusmv: a new symbolic model verifier. In *Proceedings Eleventh Conference on Computer-Aided Verification (CAV 99)* (1999), LNCS, vol. 1633, Springer, Berlin, Heidelberg, pp. 495–499.

[27] Clarke, E. The birth of model checking. In *Grumberg, O., Veith, H. (eds.) 25 Years of Model Checking* (2008), LNCS, Springer, Berlin, Heidelberg, pp. 1–26.

[28] Clarke, E., and Emerson, A. Design and synthesis of synchronization skeletons using branching-time temporal logic. In *Logic of Programs* (1982), Springer-Verlag, pp. 52–71.

[29] Clarke, E., Grumberg, O., and Hamaguchi, K. Another look at ltl model checking. *Formal Methods in System Design 10*, 1 (1997), 47–71.

[30] Clarke, E., Grumberg, O., Jha, S., Lu, Y., and Veith, H. Counterexample-guided abstraction refinement for symbolic model checking. *Journal of the ACM (JACM) 50*, 5 (2003), 752–794.

[31] Clarke, E., Grumberg, O., McMillan, K., and Zhao, X. Efficient generation of counterexamples and witnesses in symbolic model checking. In *Proc. of the Design Automation Conference* (1995), ACM Press, pp. 427–432.

[32] Clarke, E., Grumberg, O., and Peled, D. *Model Checking*. MIT, 1999.

[33] Clarke, E., Gupta, A., Kukula, J., and Strichman, O. Sat based abstraction refinement using ilp and machine leraning techniques. In *Computer-Aided Verification (CAV)* (2002), LNCS, vol. 2404, Springer, Berlin, Heidelberg, pp. 137–150.

[34] Clarke, E., Gupta, A., Kukula, J., and Strichman, O. Sat based abstraction refinement using ilp and machine leraning techniques. In *CAV 2002* (2002), LNCS, vol. 2404, Springer-Verlag, Berlin, Heidelberg, pp. 265–279.

[35] Clarke, E., Kroening, D., and Lerda, F. A tool for checking ansi-c programs. In *TACAS 2004* (2004), LNCS, vol. 2988, Springer, Berlin, Heidelberg, pp. 168–176.

[36] Clarke, E., Lu, Y., s. Jha, and Veith, H. Tree-like counterexamples in model checking. In *Proc. of the 17th Annual IEEE Symposium on Logic in Computer Science* (2002), pp. 19–29.

[37] Clarke, E., O.Grumberg, Jha, S., Lu, Y., and Veith, H. Counterexample-guided abstraction refinement. In *CAV* (1986), pp. 154–169.

[38] Clarke, E., and Veith, H. Counterexamples revisited: Principles, algorithms and applications. In *In: Grumberg, O., Veith, H. (eds.) 25 Years of Model Checking* (2008), LNCS, Springer, Berlin, Heidelberg, pp. 1–26.

[39] Clarke, E. M., Grumberg, O., and Andlong, D. Model checking and abstraction. *ACM Transactions on Programming Languages and Systems 16*, 5 (1994), 1512–1542.

[40] Cleve, H., and Zeller, A. Locating causes of program failures. In *ACM/IEEE International Conference on Software Engineering (ICSE)* (2005), pp. 342–351.

[41] Copty, F., Irron, A., Weissberg, O., Kropp, N., and Gila, K. Effcient debugging in a formal verification environment. *Int J Softw Tools Technol Transfer 4* (2003), 335–348.

[42] COUSOT, P., and COUSOT, R. Abstract interpretation: A unified lattice model for static analysis of programs by construction or approximation of fixpoints. In *ACM Symposium of Programming Language* (2003), pp. 238–252.

[43] Couvreur, J. On-the-fly verification of linear temporal logic. In *FM* (1999), LNCS, vol. 1708, Springer, Heidelberg, pp. 253–271.

[44] Damman, B., Han, T., and Katoen, J. Regular expressions for pctl counterexamples. In *Quantitative Evaluation of Systems(QEST)* (2008), pp. 179–188.

[45] de Alfaro, L., Henzinger, T., and Mang, F. Detecting errors before reaching them. In *CAV* (2000), LNCS, vol. 2725, Springer, Berlin, Heidelberg, pp. 186–201.

[46] Debbi, H. Diagnosis of probabilistic models using causality and regression. In *in Proceedings of the 8th International Workshop on Verification and Evaluation of Computer and Communication Systems* (2014), pp. 33–44.

[47] Debbi, H. *Systems Analysis using Model Checking with Causality*. PhD thesis, University of M'sila, 2015.

[48] Debbi, H., and Bourahla, M. Causal analysis of probabilistic counterexamples. In *Eleventh ACM-IEEE International Conference on Formal Methods and Models for Codesign (Memocode)* (2008), pp. 77–86.

[49] Dong, G., and Pei, J. *Sequence Data Mining*. Springer, 2007.

[50] Edelkamp, S., Leue, S., and Lluch-Lafuente, A. Directed explicit-state model checking in the validation of communication protocols. *International Journal on Software Tools for Technology Transfer 5*, 2 (2004), 247–267.

[51] Elamkulam, J., Z. Glazberg, I. R., Kowlali, G., Chandra, S., Kohli, S., and Dattathrani, S. Detecting design flaws in uml state charts for embedded software. In *HVC 2006* (2006), LNCS, vol. 4383, Springer-Verlag, Berlin, Heidelberg, pp. 109–121.

[52] Emerson, E., and Halpern, J. Decision procedures and expressiveness in the temporal logic of branching time. In *STOC 82: Proceedings of the fourteenth annual ACM symposium on Theory of computing* (1982), ACM Press, pp. 169–180.

[53] Emerson, E. A., and Lei, C.-L. Efficient model checking in fragments of the propositional mucalculus. In *Proceedings of the First Annual Symposium of Logic in Computer Science* (1986), pp. 267–278.

[54] Engels, A., Feijs, L., and Mauw, S. Test generation for intelligent networks using model checking. In *Third International Workshop on Tools and Algorithms for the Construction and Analysis of Systems. (TACAS97)* (1997), LNCS, vol. 1217, Springer, Berlin, Heidelberg, pp. 384–398.

[55] Engels, A., Feijs, L., and Mauw, S. Test generation for intelligent networks using model checking. In *Third International Workshop on Tools and Algorithms for the Construction and Analysis of Systems(TACAS)* (2010), LNCS, vol. 1217, Springer, Berlin, Heidelberg, pp. 384–398.

[56] Fey, G., and Drechsler, R. Finding good counterexamples to aid design verification. In *First ACM and IEEE International Conference on Formal Methods and Models for Co-Design (MEMOCODE03)* (2003), pp. 51–52.

[57] Fischer, F., and Leue, S. Causality checking for complex system models. In *Verification, Model Checking, and Abstract Interpretation (VMCAI)* (2013), LNCS, vol. 7737, Springer, Berlin, Heidelberg, pp. 248–276.

[58] Fisler, K., Fraer, R., Kamhi, G., Vardi, M., and Yang, Z. Is there a best symbolic cycle-detection algorithm. In *TACAS 2001* (2001), LNCS, vol. 2031, Springer, Berlin, Heidelberg, pp. 420–434.

[59] Fraser, G. *Automated Software Testing with Model Checkers*. PhD thesis, IST - Institute for Software Technology, 2007.

[60] Fraser, G., Wotawa, F., and Ammann, P. E. Testing with model checkers. *Journal of Software Testing, Verification and Reliability 19*, 3 (2009), 215–261.

[61] Gargantini, A., and Heitmeyer, C. Using model checking to generate tests from requirements specifications. In *ESEC/FSE99: 7th European Software Engineering Conference, Held Jointly with the 7th ACM SIGSOFT Symposium on the Foundations of Software Engineering* (1999), LNCS, vol. 1687, Springer, Berlin, Heidelberg, pp. 146–162.

[62] Gargantini, A., Riccobene, E., and Rinzivillo, S. Using spin to generate tests from asm specifications. In *Abstract State Machines 2003. Advances in Theory and Practice: 10th International Workshop, ASM* (2003), LNCS, vol. 2589, Springer, Berlin, Heidelberg, pp. 263–277.

[63] Gastin, P., and Moro, P. Minimal counterexample generation for spin. In *14th International SPIN Workshop 2007* (2007), LNCS, vol. 4595, Springer, Berlin, Heidelberg, pp. 24–38.

[64] Gastin, P., Moro, P., and Zeitoun, M. Minimization of counterexample in spin. In *SPIN 2004* (2004), LNCS, vol. 2989, Springer, Berlin, Heidelberg, pp. 92–108.

[65] GRAF, S., and ANDSADI, H. Construction of abstract state graphs with pvs. In *CAV* (1997), LNCS, vol. 1254, Springer, Berlin, Heidelberg, pp. 72–83.

[66] Graf, S., and Saidi, H. Construction of abstract state graphs with pvs. In *CAV 97* (1997), LNCS, vol. 1254, Springer-Verlag, Berlin, Heidelberg, pp. 72–83.

[67] Groce, A. *Error Explanation and Fault Localization with Distance Metrics*. PhD thesis, School of Computer Science Carnegie Mellon University, 2005.

[68] Groce, A., Chaki, S., Kroening, D., and Strichman, O. Error explanation with distance metrics. *International Journal on Software Tools for Technology 4*, 3 (2006), 229–247.

[69] Groce, A., Kroening, D., and Lerda, F. Understanding counterexamples with explain. In *Alur R., Peled D.A. (eds) Computer Aided Verification. CAV 2004* (2004), vol. 3114 of *Lecture Notes in Computer Science*, Springer, pp. 453–456.

[70] Groce, A., and Visser, W. What went wrong: Explaining counterexamples. In *SPIN Workshop on Model Checking of Software* (2003), pp. 121–135.

[71] Grumberg, O. Abstraction and refinement in model checking. In *FMCO 2005* (2006), LNCS, vol. 4111, Springer-Verlag, Berlin, Heidelberg, pp. 219–242.

[72] Gudemann, M., Salaun, G., and Ouederni, M. Counterexample guided synthesis of monitors for realizability enforcement. In *ATVA 2012* (2012), LNCS, vol. 7561, Springer-Verlag, Berlin, Heidelberg, pp. 238–253.

[73] Guo, L., Roychoudhury, A., and Wang, T. Accurately choosing execution runs for software fault localization. In *15th international conference on Compiler Construction* (2006), LNCS, vol. 3923, Springer, Berlin, Heidelberg, pp. 80–95.

[74] Halpern, J., and Pearl, J. Causes and explanations: A structural-model approach part i: Causes. In *17th UAI* (2001), pp. 194–202.

[75] Han, T., and Katoen, J. Counterexamples generation in probabilistic model checking. *IEEE Trans. on Software Engineering 35*, 2 (2009), 72–86.

[76] Hansen, H., and Geldenhuys, J. Cheap and small counterexamples. In *Software Engineering and Formal Methods, SEFM '08* (2008), IEEE Computer Society Press, pp. 53–62.

[77] Hansen, H., and Kervinen, A. Minimal counterexamples in o(n log n) memory and o(n 2) time. In *ACDC 2006* (2006), IEEE Computer Society Press, pp. 131–142.

[78] Hansson, H., and Jonsson, B. logic for reasoning about time and reliability. *Formal aspects of Computing 6*, 5 (1994), 512–535.

[79] Heimdahl, M., Rayadurgam, S., and Visser, W. Specification centered testing. In *Second International Workshop on Automates Program Analysis, Testing and Verification* (2000).

[80] Hermanns, H., Wachter, B., and Zhang, L. Probabilistic cegar. In *Computer Aided Verification (CAV)* (2008), LNCS, vol. 5123, Springer, Berlin, Heidelberg, pp. 162–175.

[81] Hinton, A., Kwiatkowska, M., Norman, G., and Parker, D. Prism: A tool for automatic verification of probabilistic systems. In *TACAS* (2006), LNCS, vol. 3920, Springer, Berlin, Heidelberg, pp. 441–444.

[82] Hojati, R., Brayton, R. K., and Kurshan, R. P. Bdd-based debugging of designs using language containment and fair ctl. In *Fifth Conference on Computer Aided Verification (CAV 93)* (1993), LNCS, vol. 697, Springer, Berlin, Heidelberg, pp. 41–58.

[83] Hojati, R., Brayton, R. K., and Kurshan, R. P. Bdd-based debugging of designs using language containment and fair ctl. In *CAV 93* (1993), LNCS, vol. 697, Springer, Berlin, Heidelberg, pp. 41–58.

[84] Hojati, R., Touati, H., Kurshan, R. P., and Brayton, R. K. Effcient -regular language containment. In *Computer Aided Verification* (1992), LNCS, vol. 1708, Springer, Berlin, Heidelberg, pp. 371–382.

[85] Holzmann, G., Peled, D., and Yannakakis, M. On nested depth

first search. In *SPIN'96* (1996).

[86] Holzmann, G. J. The model checker spin. *IEEE Transactions on Software Engineering 23*, 5 (1997), 1–17.

[87] Hong, H. S., and Lee, I. Automatic test generation from specifications for controlflow and data-flow coverage criteria. In *International Conference on Software Engineering (ICSE)* (2003).

[88] I.Beer, Ben-David, S., Chockler, H., Orni, A., and Treer, R. Explaining counterexamples using causality. *Formal Methods Systems Design 40*, 1 (2012), 20–40.

[89] Janota, M., Grigore, R., and Marques-Silva, J. Counterexample guided abstraction refinement algorithm for propositional circumscription. In *JELIA'10 Proceedings of the 12th European conference on Logics in artificial intelligence* (2010), LNCS, vol. 6341, Springer, Berlin, Heidelberg, pp. 195–207.

[90] Jansen, N., Abraham, E., Volk, M., Wilmer, R., Katoen, J., and Becker, B. The comics tool - computing minimal counterexamples for dtmcs. In *ATVA* (2012), LNCS, vol. 7561, Springer, Berlin, Heidelberg, pp. 249–253.

[91] Jia, Y., and Harman, M. An analysis and survey of the development of mutation testing. *IEEE Transactions ON Software Engineering 37*, 05 (2011), 649 – 678.

[92] Jin, H., Ravi, K., and F.Somenzi. Fate and free will in error traces. *International Journal on Software Tools for Technology Transfer 6*, 2 (2004), 102–116.

[93] Kashyap, S., and Garg, V. Producing short counterexamples using crucial events. In *CAV 2008* (2008), LNCS, vol. 5123, Springer, Berlin, Heidelberg, pp. 491–503.

[94] Katoen, J.-P., Khattri, M., and Zapreev, I. S. A markov reward model checker. In *QEST* (2005), pp. 243–244.

[95] Kesten, Y., Pnueli, A., and o. Raviv, L. Algorithmic verification of linear temporal logic specifications. In *International Colloquium on Automata, Languages, and Programming (ICALP-98),* (1998), LNCS, vol. 1443, Springer, Berlin, Heidelberg, pp. 1–16.

[96] Kroening, D., Groce, A., and Clarke, E. Counterexample guided abstraction refinement via program execution. In *6th International Conference on Formal Engineering Methods (ICFEM)* (2004), LNCS, vol. 3308, Springer, Berlin, Heidelberg, pp. 224–238.

[97] Kuma, N., Kumar, V., and Viswanathan, M. On the complexity of error explanation. In *Verification, Model Checking, and Abstract Interpretation (VMCAI)* (2005), LNCS, vol. 3385, Springer, Berlin, Heidelberg, pp. 448–464.

[98] Kumazawa, T., and Tamai, T. Counterexample-based error localization of behavior models. In *NASA Formal Methods* (2011), pp. 222–236.

[99] Kurshan, R. P. *Computer-Aided Verification of coordinating processes - the automata theoretic approach.* Princeton University Press, 1994.

[100] Larsen, K. G., Pettersson, P., and Wang, Y. Uppaal in a nutshell. *Int. J. Software Tools for Technology Transfer 1*, 1 (1997), 134–152.

[101] Leitner-Fischer, F., and Leue, S. On the synergy of probabilistic causality computation and causality checking. In *SPIN 2013* (2013), LNCS, vol. 7976, Springer-Verlag, Berlin, Heidelberg, pp. 246–263.

[102] Leitner-Fischer, F., and Leue, S. Probabilistic fault tree synthesis using causality computation. *IJCCBS 4*, 2 (2013), 119–143.

[103] Leue, S., and Befrouei, M. T. Counterexample explanation by anomaly detection. In *SPIN* (2012), vol. 7385 of *Lecture Notes in Computer Science*, Springer, pp. 24–42.

[104] Leue, S., and Befrouei, M. T. Mining sequential patterns to explain concurrent counterexamples. In *SPIN* (2013), vol. 7976 of *Lecture Notes in Computer Science*, Springer, pp. 264–281.

[105] Lewis, D. Causation. *Journal of Philosophy 70* (1973), 556–567.

[106] LONG, D. *Model checking, abstraction and compositional verification.* PhD thesis, School of Computer Science, Carnegie Mellon University, 2005.

[107] McMillan, K., and Zuck, L. Abstract counterexamples for non-disjunctive abstractions. In *Reachability Problems* (2009), LNCS, vol. 5797, Springer, Berlin, Heidelberg, pp. 176–188.

[108] Nopper, T., Scholl, C., and Becker., B. Computation of minimal counterexamples by using black box techniques and symbolic methods. In *Computer-Aided Design (ICCAD)* (2007), IEEE Computer Society Press, pp. 273–280.

[109] Pnueli, A. The temporal logic of programs. In *18th Annual Symposium on Foundations of Computer Science* (1977), IEEE, pp. 46–57.

[110] Pytlik, B., Renieris, M., Krishnamurthi, S., and Reiss, S. P. Automated fault localization using potential invariants. In *AADEBUG'2003, Fifth International Workshop on Automated and Algorithmic Debugging* (2003), pp. 273–276.

[111] Ravi, K., Bloem, R., and Somenzi, F. A comparative study of symbolic algorithms for the computation of fair cycles. In *Third International Conference, FMCAD 2000* (2000), LNCS, vol. 1954, Springer, Berlin, Heidelberg, pp. 162–179.

[112] Ravi, K., Bloem, R., and Somenzi, F. A note on on-the-fly verification algorithms. In *TACAS 2005* (2005), LNCS, vol. 3440, Springer, Berlin, Heidelberg, pp. 174–190.

[113] Ravi, K., and Somenzi, F. Minimal assignments for bounded model checking. In *TACAS* (2004), LNCS, vol. 2988, Springer, Berlin, Heidelberg, pp. 31–45.

[114] Renieris, M., and Reiss, S. Fault localization with nearest neighbor queries. In *ASE* (2003), IEEE Computer Society, pp. 30–39.

[115] Saidi, H., and Shankar, N. Abstract and model check while you prove. In *CAV 99* (1999), LNCS, vol. 4111, Springer-Verlag, Berlin, Heidelberg, pp. 219–242.

[116] Schmalz, M., Varacca, D., and Volzer, H. Counterexamples in probabilistic ltl model checking for markov chains. In *International Conference on Concurrency Theory (CONCUR)* (2009), LNCS, vol. 5710, Springer, Berlin, Heidelberg, pp. 787–602.

[117] Schuppan, V., and Biere, A. Shortest counterexamples for symbolic model checking of ltl with past. In *11th International Conference on Tools and Algorithms for the Construction and Analysis of Systems* (2005), LNCS, vol. 3440, Springer, Berlin, Heidelberg, pp. 493–509.

[118] Shen, S., Qin, Y., and Li, S. Bug localization of hardware system with control flow distance minimization. In *13th IEEE International Workshop on Logic and Synthesis (IWLS 2004)* (2004).

[119] Shen, S., Qin, Y., and Li, S. Localizing errors in counterexample with iteratively witness searching. In *ATVA* (2004), LNCS, vol. 3299, Springer, Berlin, Heidelberg, pp. 456–469.

[120] Shen, S., Qin, Y., and Li, S. Minimizing counterexample with unit core extraction and incremental sat. In *Verification, Model Checking, and Abstract Interpretation* (2005), LNCS, vol. 3385, Springer, Berlin, Heidelberg, pp. 298–312.

[121] Shen, S., and Y. Qin, S. L. Localizing errors in counterexample with iteratively witness searching. In *ATVA 2004* (2004), LNCS, vol. 3299, Springer, Berlin, Heidelberg, pp. 459–464.

[122] Shen, S.-Y., Qin, Y., and Li, S. A fast counterexample minimization approach with refutation analysis and incremental sat. In *Conference on Asia South Pacific Design Automation* (2005), pp. 451–454.

[123] Sheyner, O., Haines, J., , Jha, S., Lippmann, R., and Wing, J. Automated generation and analysis of attack graphs. In *IEEE Symposium on Security and Privecy 2002* (2002), pp. 273–284.

[124] Tan, J., Avrunin, G., and Leue, S. Heuristic-guided counterexample search in flavers. In *12th ACM SIGSOFT international symposium on Foundations of software engineering* (2004), pp. 201–210.

[125] Tan, L., Sokolsky, O., and Lee, I. Specification-based testing with linear temporal logic. In *Proceedings of IEEE International Conference on Information Reuse and Integration* (2004), pp. 493–498.

[126] Tarjan, R. E. Depth-first search and linear graph algorithms. *SIAM Journal of Computing 1*, 2 (1972), 146–160.

[127] Tip, F., and Dinesh, T. A slicing-based approach for locating type errors. *ACM Transactions on Software Engineering and Methodology1 10*, 1 (2001), 5–55.

[128] Touati, H. J., Brayton, R. K., and Kurshan, R. P. Testing language containment for ω automata using bdds. In *International Workshop on Formal Methods in VLSI Design* (1991), pp. 371–382.

[129] Valmari, A., and Geldenhuys, J. Tarjans algorithm makes on-the-fly ltl verification more efficient. In *Jensen, K., Podelski, A. (eds.) TACAS* (2004), LNCS, vol. 2988, Springer, Berlin, Heidelberg, pp. 205–219.

[130] Vardi, M., Wolper, P., and Yannakakis, M. Memory-efficient algorithms for the verification of temporal properties. *Formal Methods in System Design 1*, 2 (1992), 275–288.

[131] Visser, W., Havelund, K., Brat, G., Park, S., and Lerda, F. Model checking programs. *Automated Software Engineering Journal 10*, 2 (2003), 203–222.

[132] Wang, C., Yang, Z., Ivancic, F., and Gupta, A. Whodunit? causal analysis for counterexamples. In *4th International Symposium, ATVA* (2006), LNCS, vol. 4218, Springer, Berlin, Heidelberg, pp. 82–95.

[133] Wimmer, R., Braitling, B., and Becker, B. Counterexample generation for discrete-time markov

chains using bounded model checking. In *Verification, Model Checking, and Abstract Interpretation* (2009), LNCS, vol. 5403, Springer, Berlin, Heidelberg, pp. 366–380.

[134] Wimmer, R., Jansen, N., Abraham, E., Becker, B., and Katoen, J. Minimal critical subsystems for discrete-time markov models. In *TACAS* (2012), LNCS, vol. 7214, Springer, Berlin, Heidelberg, pp. 299–314.

[135] Wimmer, R., Jansen, N., and Vorpahl, A. High-level counterexamples for probabilistic automata. In *Quantitative Evaluation of Systems (QEST)* (2013), LNCS, vol. 8054, Springer, Berlin, Heidelberg, pp. 39–54.

[136] Xie, A., and Beerel, P. A. Implicit enumeration of strongly connected components. In *International Conference on ComputerAided Design* (1999), pp. 37–40.

[137] Zeller, A. Yesterday, my program worked. today, is does not. why? In *ACM Symposium on the Foundations of Software Engineering* (1999), pp. 253–267.

[138] Zeller, A. Isolating cause-effect chains for computer programs. In *ACM Symposium on the Foundations of Software Engineering* (2002), pp. 1–10.

Evaluation of Medical Image Algorithms on Multicore Processors

Damir Demirović
Department of Computer Science and Informatics, Faculty of Electrical Engineering,
University of Tuzla, Bosnia and Herzegovina
E-mail: damir.demirovic@untz.ba, http://www.fe.untz.ba

Zekerijah Šabanović
Medical Faculty, University of Tuzla, Bosnia and Herzegovina
E-mail: zekerijah.sabanovic@untz.ba, http://www.medf.untz.ba

Keywords: medical image processing, multicore processor, GPU, GPGPU, filtering, image registration

Received: April 26, 2016

Introduction: In recent time medical image processing and analysis became an essential component inclinical practice. Medical images contain huge data to process due to increased image resolution. Thesetasks are inherently parallel in nature, so they naturally fit to parallel processors like Graphics Processing Unit (GPU). In this work several commonly used image processing algorithms for 2-D and 3-D were evaluated regarding the computation performance increase using the GPUs and CPUs on a personal computer. For tested algorithms, GPU outperforms CPU from 1.1 to 422 times.

Povzetek: V zadnjem času je obdelava in analiza medicinskih slik postala bistvena sestavina v klinični praksi. Medicinske slike vsebujejo ogromne količine podatkov, vendar je procesiranje slik vzporedne narave, posebej primerno za obdelavo z grafično procesno enoto (GPU). V tem delu smo ocenili več pogosto uporabljenih algoritmov za obdelavo slik za 2-D in 3-D glede povečanja zmogljivosti računanja z grafičnimi procesorji na osebnem računalniku. Za testirane algoritme je grafični procesor omogočil zmanjšanje časa računanja od 1,1 do 422-krat.

1 Introduction

In the last decade parallel processing has become the most dominant for high-performance computing. Increasing the processor clock rate in single-core processors has slowed down due to the problems with heat dissipation. Application developers cannot count on Moore's law to make complex algorithms computationally feasible. Consequences are that they are increasingly shifting the algorithms to parallel computing architectures [1][2]. These architectures are multicore Central Processing Units (CPU), Graphical Processing Units (GPU) and Field-Programmable Gate Array (FPGA).

The amount of data processed in clinical practice is also increasing. Increased resolution of medical images and a huge amount of data for processing is exploding. Trends like 3-D and 4-D imaging technologies used in treatment planning need a lot of computer power. Due to its nature, these tasks are inherently data-parallel, i.e. data from such dataset can be processed in parallel using multiple threads. GPUs originally designed for acceleration of computer graphics, become a versatile platform for running massively parallel computation. This is due to its nature, like high memory bandwidth, high computation throughput etc. [2]. In the year, 2004 programmable GPUs were introduced. Firstly, they could run in parallel custom programs called shaders. This is the first time to accelerate the non-graphical applications with GPUs.

Today GPU become a viable alternative to CPUs in time-consuming tasks. When same computations can be performed on many image elements in parallel, so it can easily fit on GPUs. Two dominant parallel computing platforms are NVidia CUDA and OpenCL.

OpenCL [3] is a software framework for writing programs that run across heterogeneous platforms like CPUs, GPUs, digital signal processors (DSPs) and FPGAs. Heterogeneous refers to systems with more the one kind of processors or cores. Both CUDA and OpenCL support heterogeneous computing. OpenCL is based on a C programming language, and it is an open standard. NVIDIA CUDA [4] is a parallel computing platform and Application Programming Interface (API), which supports programming framework OpenCL.

In [5] authors gave the introduction to the GPU architecture, and its applications in image processing, software development, and numerical applications.

In [2] authors review the principles of GPU computing in the area of medical physics. Segmentation of anatomical structures from image modalities like Computed Tomography (CT) and Magnetic Resonance Imaging (MRI) were given in [6]. Due to its computational complexity most segmentation procedures require vast processing power like GPU. A brief literature review of several segmentation methods is given here.

In [7] authors give a review of applications for GPU in medicine, which covers the past and current trend in this field, like commonly used method and algorithm which are specific to individual image modalities. Also, in the field of medical visualization GPU can be effectively used.

Algorithm Marching Cubes that extract surfaces from volumetric data was presented [8]. Fast extraction in medical applications is necessary, so near real-time applications are very desirable. Their algorithm implementation is completely data-parallel, which is ideal for application on a GPU.

In [10] authors implement widely known Demons algorithm for medical image registration [16] on a GPU, for registering 3-D CT lung images. Speedups of 55 times were reported over non-optimized CPU version.

In [20] authors were using OpenCL to evaluate reconstruction of 3-D volumetric data from C-arm CT projections on a variety of high-performance computing platforms, like FPGAs, graphic cards and multi-core CPUs.

Three-dimensional reconstruction task in cone-beam CT, a computation complex algorithm was implemented using CUDA [21].

Book [9] covers developing data-parallel version of registration algorithms suitable for execution on GPU.

Our main objective was to compare algorithms using CPU and GPU, and their assessment on a different processor architecture. Some of the most used image processing algorithms, which are suitable for algorithm parallelization, were evaluated and speedups were compared to a single core of the CPU. CPU results were used as a base for comparison of the results from the GPU.

2 Methods

In this work, time-consuming algorithms were evaluated on a CPU and GPU. Algorithms for 2D and 3D were tested, and running times were evaluated.

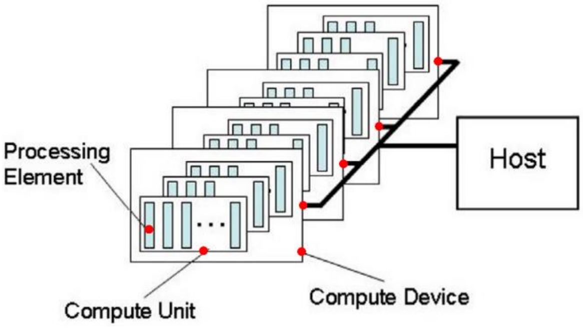


Figure 1: OpenCL platform model [3].

There are several software packages for image processing and analysis of medical images. For the purpose of this research, the different software packages were used, as described as follows.

Plastimatch [11] is an open source software for image computation. The main focus is high-performance volumetric registration of medical images, such as X-ray CT, MRI, and positron emission tomography (PET). Software features include methods for medical image registration, segmentation etc.

Capabilities	Processor	
	GPU GTX 560Ti	CPU Intel i5-2500
OpenCL version	1.1	not available
Compute capability	2.1	not available
Double precision	Yes	Yes
Number of cores	384	4
Max clock freq. (GHz)	1.7	3.7
Global memory (MB)	1023	6
Power rating (W)	170	95

Table 1: Processor specifications.

OpenMP (Open Multi-Processing) [12] is an application programming interface (API) that supports multi-platform shared memory multiprocessing programming in C, C++, and Fortran, on most platforms, processor architectures and operating systems, including Solaris, AIX, HP-UX, Linux, OS X, and Windows.

OpenCL and CUDA allow heterogeneous programming model, so a typical sequence of operations is the same in both of them. In both platforms, host refers to the CPU and its memory, while device refers to GPU and its memory. Kernels are functions executed on the device (GPU) in parallel. A typical program has the following steps: declaring and allocating host and device memory, initialize host data, transfer data from the host to the device, execute one or more kernels, transfer results from device to the host.

OpenCL is portable API, based on the C99 standard of the C programming language. OpenCL platform model (Figure 1) consists of a host of several computing devices which each contain several computing units. Further, a computing unit contains several processing units. The serial code runs on a Host (which is a CPU) thread, and the parallel code executes in many devices (GPUs) threads across multiple processing elements.

Functions executed on OpenCL devices are called kernels. Both CUDA and OpenCL support built-in functions which can take scalar and vector arguments. Native functions are built-in functions with reduced precision which is implementation defined, but with decreased execution time. Built-in functions conform to IEEE 754 compatible rounding for single precision floating point calculations.

OpenCV [13] is a library of functions for computer vision. It is cross platform and released under the BSD license, written in C++ language, and supports Intel Integrated Performance Primitives (IPP) optimized routines, support for GPUs for CUDA and OpenCL.

In this work nine commonly used algorithms were evaluated. First, algorithms in 2D which can be used for filtering medical images were evaluated. Medical image datasets usually come as volumes like CT image. They have usually 100 or mores slices, so running times are exceptionally high, which prevents their clinical usage in real time.

All experiments presented in this work were evaluated on a PC computer with Intel CPU and NVidia GPU with 8GB of RAM memory. For the GPU implementation of algorithms NVIDIA CUDA Toolkit version 7.5 was used. CPU implementations were implemented using Microsoft Visual Studio Express 2013. Specifications of the processors for this research are given in Table 1.

For the purpose of research, we choose nine image processing algorithms with frequent usage in medical practice. We have split the analysis of algorithms for 2-D and 3-D images as described in the following sections.

2.1 2-D algorithms

In medical practice 2-D algorithm can be used on a single image slice or extracted images from 3-D volumes. For the purpose of this research, we choose the rotation, Gaussian filter, Sobel filter, Fast Walsh transform, Farneback method and Horn-Schunk optical flow.

Image rotation is a geometric operation which maps the image pixel in an input image onto the position in an output image by rotating the image around the specified angle about an origin. Rotation is a case of an affine transformation, and it is widely used in image processing (for example image registration).

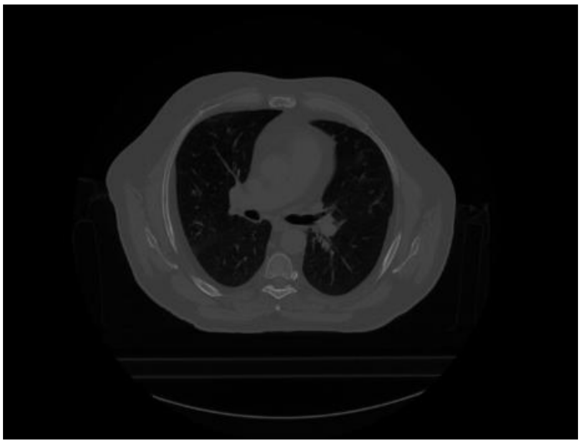


Figure 2: Image used for all 2-D experiments.

Gaussian filter is the most common used in filtering and have significant usage in medical applications (for example in image registration which acts as smoother). Gaussian filter was evaluated for input image of 2048x2048 with parameters sigma 10 pixels and kernel size 81. For purpose of these experiments, Gaussian kernel were implemented on the CPU and the GPU. For CPU, we used up to 4 threads with standard CPU optimizations. For these experiments the image showed in Figure 2 was used.

Fast Walsh or Hadamard transform is a special case of generalized Fourier transforms, which has the same complexity like Fourier transform but without multiplications.

Farneback method for computation of optical flow was presented in [14]. Optical flow was used for the finding of relative motion between two images. It can be used to recover motion for example between two organs. The method is based on approximation of each neighborhood of two frames by quadratic polynomials, using the polynomial expansion transform (images are shown in Figure 3 and Figure 4). Obtained deformation field is shown in Figure 5 and Figure 6, where colors correspond to different values of deformation obtained. Two deformations appear similar but a significant value difference can be seen in the lower and the right part of Figure 5. If we take the CPU implementation as the golden truth, the difference between these two results originates from a loss of computation precision of the GPU.

Horn-Schunck is optical flow method is a classical method for finding the apparent motion in images [15]. The method assumes smoothness in the flow over the whole image and tries to minimize global energy functional which consists of two parts, intensity and regularization. The method employs iterative scheme using Jacobi method. For this experiment, image showed in Figure 3 and Figure 4 were used. Deformation field after registration obtained with this algorithm are showed in Figure 7 and Figure 8. Comparing the obtained deformation fields from two algorithms we found some small differences on the pixels on GPU image (Figure 5 and Figure 6).

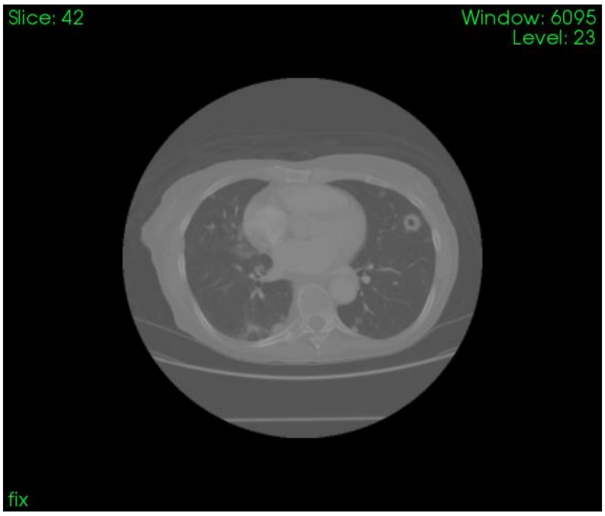


Figure 3: Static image used for all 3-D experiments.

Results for described 2-D algorithms are given in Table 2 and corresponding Figure 9. From the results one can see that almost all algorithms, with exception of image rotation, execute faster on the GPU, and depending on the algorithm speedups are from 10x to 84x compared to one CPU thread. Significant improvements can be also obtained with some loss of the accuracy. Almost all algorithms can be run in a real-time on the GPU, and just one on the CPU.

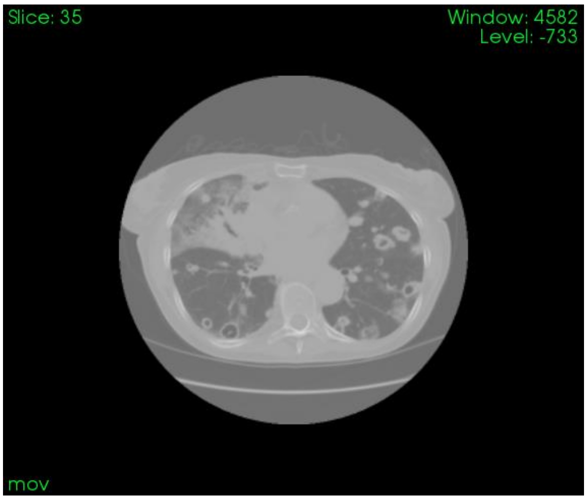


Figure 4: Moving image used for all 3-D experiments.

The Sobel operator is a widely used filter in image processing for edge detection. In 2-D Sobel operator is 3x3 for one dimension, whereas in 3-D 3x3x3 for each of 3 dimensions. The result of Sobel operator is a gradient vector. The filter is separable so it can be written as product of two simpler filters.

For this experiment, we used up to four CPU thread for evaluation. Speedups are given in Table 3 and Figure 10. For the best experiment we can expect the speedup of 38 times for the four CPU cores, or in worst case 141 times compared to one CPU core. From these results can be clearly seen that Sobel algorithm can benefit significantly from implementation on the GPU compared to one CPU thread.

2.2 3-D algorithms

3-D algorithms in medical practice are very important. Most of medical images are 3-D volumes and needs to be preprocessed, analyzed or visualized in some way. In the next part the five widely used algorithms in 3-D were evaluated.

For the purpose of this evaluation, we implemented 3-D Gaussian filter in C programming language. Volume dimensions for tested images were 482x360x141 with kernel size of 5 and sigma 0.5 voxels.

The Sobel operator in 2D has the dimension of 3x3, whereas in 3-D 3x3x3 for each of 3 dimensions. The result of Sobel operator is a gradient vector. The filter is separable so it can be written as the product of two simpler filters, thus reducing the computation time. For this experiment, the same volume was used as in the previous experiment.

All 3-D registration was evaluated for the three resolution levels, with maximal 30, 50 and 50 iterations respectively. Threading in CUDA, OpenMP, and single thread have been used. For registration bspline,

Demons, and affine algorithms from Plastimatch were used. Registration using bsplines falls into a category of Free-Form Deformations (FFD) in which object to be registered is embedded into bspline object [19]. Deformation of bspline object represents the transformation of the registration [17]. Affine image registration falls into a category of linear registration, which is a composition of linear transformations with translations. In this category falls rigid transformations (translating plus rotations), rigid plus scaling and affine. Another category of non-linear registration is non-rigid, deformable, fluid elastic etc. Affine transformation preserves points, straight lines, and planes. After transformation set of parallel lines remains parallel. Affine transformations define translation, scale, shear, and rotation.

Obtained deformations of Horn-Schunk algorithm are shown in Figure 7 and Figure 8, where colors correspond to different values of deformation which was obtained from algorithms running on the GPU and CPU respectively. In contrast to the 2-D Farneback method, some small differences can be spotted between the two deformation fields, which corresponds to very small error for the GPU.

All results obtained with 3-D registration are showed in Table 4 and Figure 11. For these experiments, OpenMP were used with four CPU threads, except for filtering algorithms Gaussian and Sobel. Obtained speedups are from 1x to 422x depending on the algorithm. Lowest speedup is for the affine registration where CPU version of the algorithm is little faster. Highest speedup is for

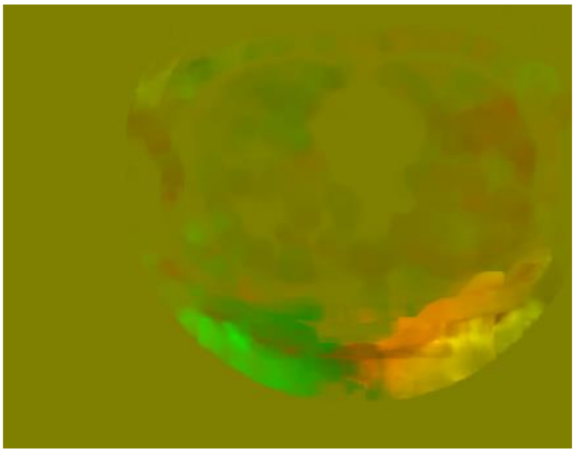


Figure 5: Color representation of deformation field using Farneback algorithm (GPU).

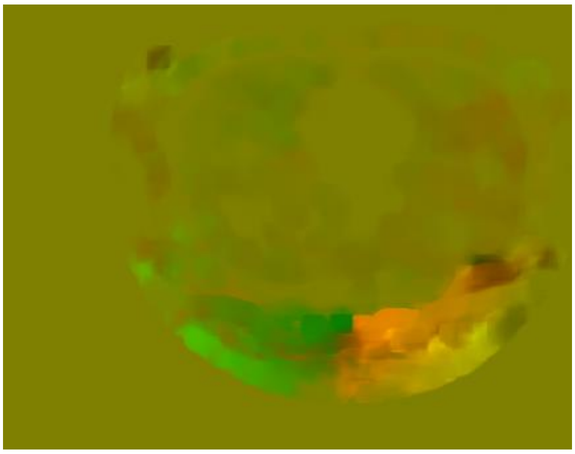


Figure 6: Color representation of deformation field using Farneback algorithm (CPU).

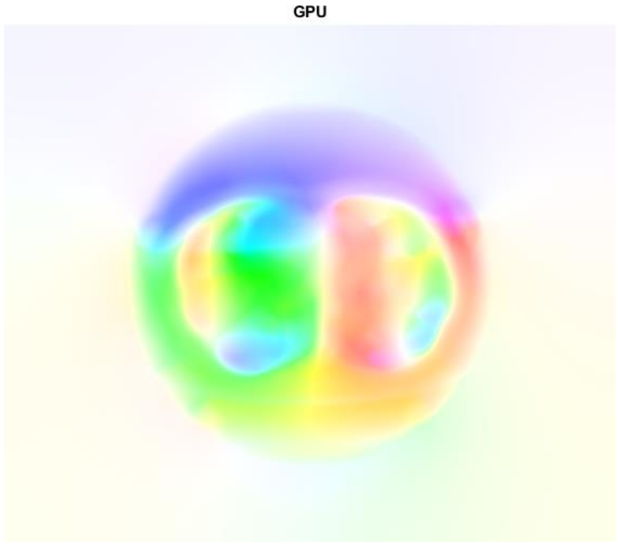


Figure 7: Color representation of deformation field using Horn-Schunk algorithm (GPU).

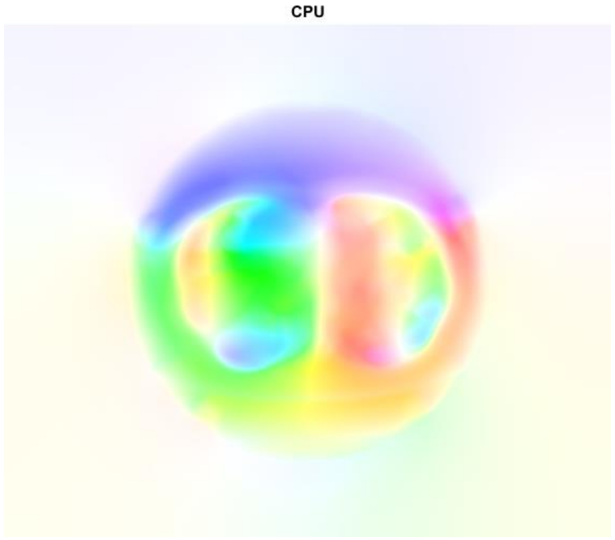


Figure 8: Color representation of deformation field using Horn-Schunk algorithm (CPU).

filtering, from 127x to 422x compared with single CPU thread. Registration algorithm Demons and affine have little or no speedup for 4x, whereas bspline have a significantly lower performance in this case. Algorithms for image registration are highly computing extensive and obtained speedup is from about 1x for affine to 15x for Demons algorithm.

It is worth to mention that Demons algorithm uses Gaussian filter in each iteration to smooth the deformation field. From the running times for Demons, one can see that speedup is almost the same for CPU, which indicates the single thread implementation for this algorithm. Similar to 2-D implementations, there is a trade-off between precision and running time.

3 Conclusions

In this paper was presented an evaluation of speed gain using modern GPU cards compared to the standard CPU. In total, nine common used algorithms on different processors were evaluated using parallel processing for 2-D and 3-D. For the CPU up to 4 threads were used,

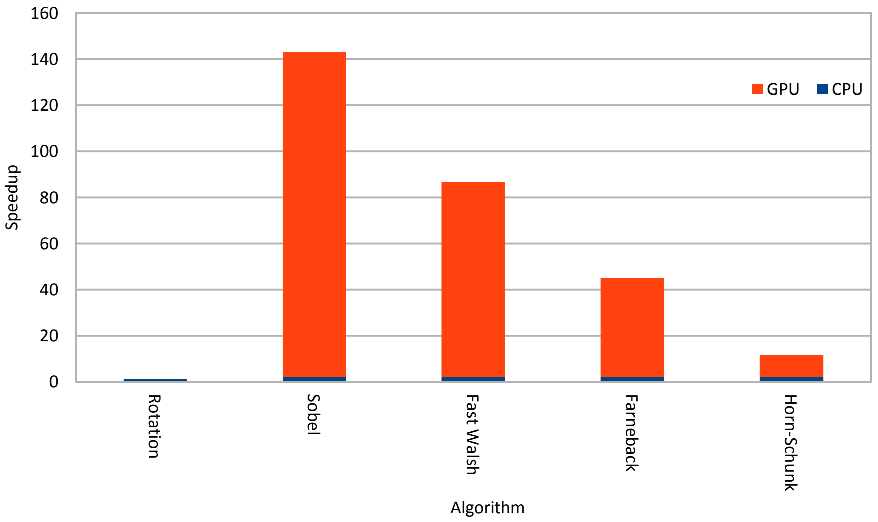


Figure 9: Speedups for 2-D experiments.

Algorithm	**GPU (s)**	**CPU (s)**	**speedup (in times)**
Image rotation	0.0090	0.10	0.01
Fast Walsh transform	0.0399	3.38	**84**
Farneback optical flow [14]	0.0116	0.50	**43**
Horn-Schunk optical flow [15]	1.4200	13.69	**10**

Table 2: Running times and corresponding GPU Speedups for 2-D experiments.

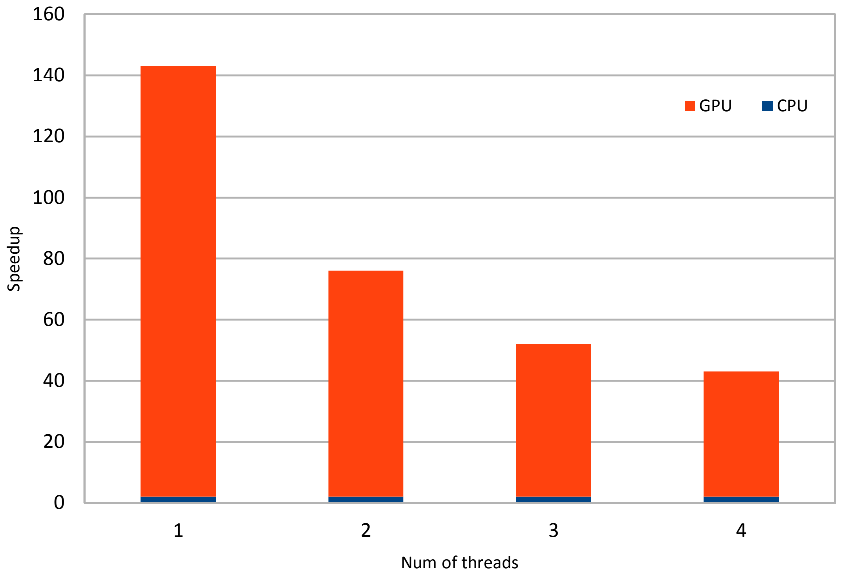


Figure 10: Speedups for Sobel algorithm.

Algorithm	**GPU (s)**	**Number of CPU threads (s)**				**speedup (in times)**			
		1	2	3	4	1	2	3	4
Sobel filter	0.0155	2.2	1.1	0.8	0.6	**141**	**71**	**51**	**38**

Table 3: Running times (in seconds) and corresponding GPU speedups for 2-D Sobel filter.

depending on the algorithm implementation. For the GPU, algorithms were used with simple naïve implementation, without optimization and all available cores.

In almost all cases processing times decrease due to highly parallelizable algorithms. Obtained speedups varied from 1.1x to 422x depending on the algorithm. Some of the tested algorithms was not well suited to parallel implementation, i.e. their running times increased with larger number of threads. Obtained results on a GPU suffers small loss of accuracy, and show near real-time performance.

Future work can evaluate the specific optimizations for CPU and GPU, instructions like SSE, AVX for CPU. Native instructions, determining the optimal local and global block size for CUDA and OpenCL and instructions with lower precision can be analyzed for the GPU. Another possibility for detecting and reducing the bottlenecks in the GPU implementation can be done using a GPU profiler.

4 References

[1] Xue X, Cheryauka A, Tubbs D. Acceleration of fluoro-CT reconstruction for a mobile C-Arm on GPU and FPGA hardware: a simulation study. Proc. SPIE 6142, Medical Imaging 2006: Physics of Medical Imaging, 61424L (2 March 2006); 2006.

[2] Pratx G, Xing L. GPU computing in medical physics: A review. *Medical Physics*. 2011; 38(5): p. 2685-2697.

[3] Khronos. OpenCL. [Online]. [cited 2016 03 21. Available from: https://www.khronos.org/opencl/.

[4] NVidia. NVIDIA CUDA. [Online].; 2016 [cited 2016 03 15. Available from: http://www.nvidia.com/object/cuda_home_new.html.

[5] Couturier R. Designing Scientific Applications on GPUs: Chapman & Hall CRC; 2013.

[6] Smistad E, Falch TL, Bozorgi M, Elster AC, Lindseth F. Medical image segmentation on GPUs – A comprehensive review. *Medical Image Analysis*. 2015; 20(1): p. 1-18.

[7] Eklund A, Dufort P, Forsberg D, LaConte SM. Medical image processing on the GPU - past, present and future. *Medical Image Analysis*. 2013; 17(8)

[8] Smistad E, Elster AC, Lindseth F. Fast surface extraction and visualization of medical images using OpenCL and GPUs. *The Joint Workshop on High Performance and Distributed Computing for Medical Imaging*. 2011; 2011.

[9] Shackleford J, Kandasamy N, Sharp G. High Performance Deformable Image Registration Algorithms for Manycore Processors. 1st ed. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc.; 2013.

[10] Samant P, Muyan-Ozcelik , Owens JD, Xia J, S. S. Fast Deformable Registration on the GPU: A CUDA Implementation of Demons. *In proceedings of the 1st technical session on UnConventional High Performance Computing (UCHPC) in conjunction with the 6th International Conference on Computational Science and Its Applications (ICCSA)*; 2008; Perugia, Italy. p. 223-233.

[11] Plastimatch. Plastimatch. [Online]; 2016 [cited 2016 04 20]. Available from:http://plastimatch.org/.

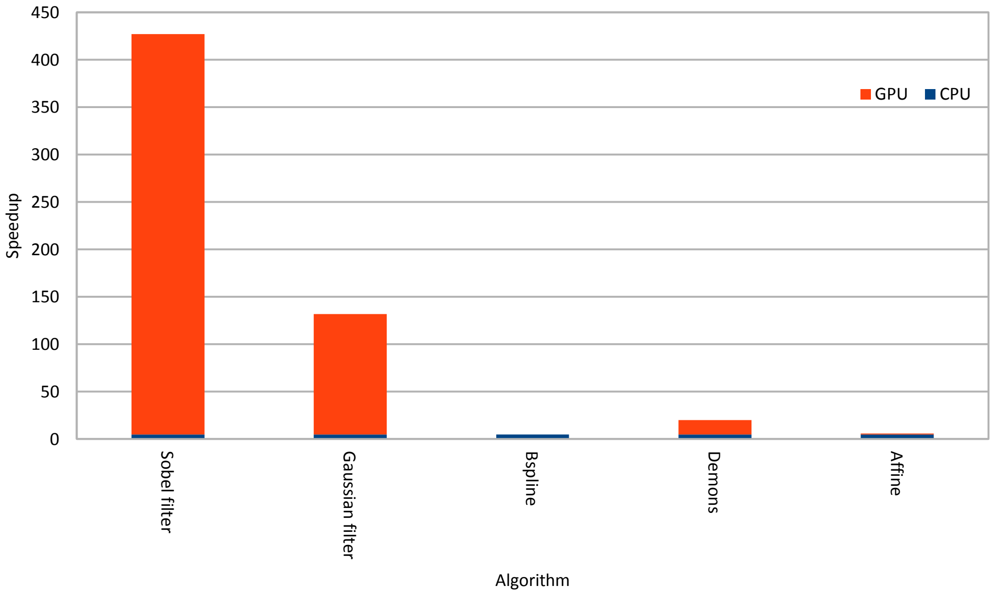


Figure 11: Speedups for 3-D algorithms.

Algorithm	GPU (s)	Number of CPU threads (s)		speedup (in times)	
		1	4	1	4
Sobel filter	0.0557	23.5	-	**422**	-
Gaussian filter	0.7860	100.1	-	**127**	-
Bspline registration [17]	41.4000	323.4	99.6	**8**	**2**
Demons registration [18]	6.5100	98.9	99.0	**15**	**15**
Affine registration	74.7100	69.2	81.2	0.92	**1.1**

Table 4: Running times (in seconds) and corresponding GPU speedups for 3-D algorithms.

[12] OpenMP. http://openmp.org/wp/. [Online].; 2016 [cited 2016 04 02]. Available from: http://openmp.org/wp/.

[13] OpenCV. OpenCV. [Online].; 2016 [cited 2016 04 15. Available from: http://opencv.org/.

[14] Farneb. Image Analysis: 13th Scandinavian Conference, SCIA 2003 Halmstad, Sweden, June 29-July 2, 2003 Proceedings. In Bigun J, Gustavsson T, editors. Berlin, Heidelberg: Springer Berlin Heidelberg; 2003. p. 363-370.

[15] Horn BKP, Schunck BG. Determining Optical Flow. *Tech. rep. Cambridge, MA, USA*; 1980.

[16] Maintz JBA, Viergever MA. A Survey of Medical Image Registration. *Medical Image Analysis, Volume 2 , Issue 1 , 1 - 36* 1998.

[17] Pennec X, Cachier P, Ayache N. Understanding the demon’s algorithm: 3D non-rigid registration by gradient descent. *In: Proc. MICCAI’99*; 1999.

[18] Thirion JP. Image matching as a diffusion process: an analogy with Maxwell's demons. *Medical Image Analysis*. 1998 sep; 2(3): p. 243-260.

[19] Tustison NJ, Avants BA, Gee JC. Improved FFD B-Spline Image Registration. *Computer Vision, IEEE International Conference on*. 2007; 0: p. 1-8.

[20] Siegl C, Hofmann HG, Keck B, Prümmer M, Hornegger J. OpenCL: a viable solution for high-performance medical image reconstruction? *Proceedings of SPIE (Medical Imaging 2011: Physics of Medical Imaging)*, Lake Buena Vista, Florida, USA, 12 - 17 Feb 2011, vol. 7961, pp. 79612Q, 2011

[21] Scherl H, Keck B, Kowarschik M, Hornegger J. Fast GPU-Based CT Reconstruction using the Common Unified Device Architecture (CUDA). *In Nuclear Science Symposium Conference Record*, 2007. NSS '07. IEEE; 2007 Oct. p. 4464-4466.

A Modification of the Lasso Method by Using the Bahadur Representation for the Genome-Wide Association Study

Lev V. Utkin
Peter the Great St.Petersburg Polytechnic University, St.Petersburg, Russia
E-mail: lev.utkin@gmail.com

Yulia A. Zhuk ITMO University, St.Petersburg, Russia
E-mail: zhuk_yua@mail.ru

Keywords: data analysis, feature selection, Lasso, Bahadur representation, genome-wide association study

Received: September 24, 2016

A modification of the Lasso method as a powerful machine learning tool applied to a genome-wide association study is proposed in the paper. From the machine learning point of view, a feature selection problem is solved in the paper, where features are single nucleotide polymorphisms or DNA-markers whose association with a quantitative trait is established. The main idea underlying the modification is to take into account correlations between DNA-markers and peculiarities of phenotype values by using the Bahadur representation of joint probabilities of binary random variables. Interactions of DNA-markers called the epistasis are also considered in the framework of the proposed modification. Various numerical experiments with real datasets illustrate the proposed modification.

Povzetek: Predstavljena je modifikacija metode strojnega učenja, imenovana Lasso.

1 Introduction

One of the important area for a successful application of the artificial intelligence, in particular, machine learning algorithms, is the computational biology which can be regarded as a basis for many engineering problems in biotechnology. An interesting task clearly illustrating the application of artificial intelligence to biotechnology problems is a genome-wide association study (GWAS). GWAS examines the association between phenotypes or quantitative traits and genetic variants or genotypes across the entire genome. In the machine learning framework, it can be regarded as one of the methods for a feature selection problem where features are the so-called single nucleotide polymorphisms (SNPs) or DNA-markers. As pointed out in [12, 15], there are some difficulties of solving this feature selection problem. First of all, the number of SNPs p is typically 10–100 times the number of individuals n in the training sample. This is the so called $p > n$ (or large p small n) problem, which leads to difficulty of an oversaturated model. Another difficulty is that SNPs may affect phenotype in a complicated and unknown manner. For example, some DNA-markers may interact in their effects on phenotype. This interaction is called the epistatic effect.

A huge amount of the statistical models and methods solving the SNP selection problem have been developed the last decades. A part of methods can be referred to as *filter* methods [1, 28] which use statistical properties of SNPs to filter out poorly informative ones. A review of filter methods in GWAS is proposed by Zhang et al. [53]. The t-test, Fisher criterion (F-statistics), χ^2-statistics, ANOVA tests are the well-known statistical methods for detecting differential SNPs between two samples in training data.

Another part of methods called *wrapper* methods generally provides more accurate solutions than the filter methods, but it is computationally demanding [24]. One of the well-known wrapper methods proposed by Guyon et al. [17] and called the Recursive Feature Elimination has been applied to the gene selection problem for cancer classification.

Filter methods and their modifications as well as wrapper methods may be efficient tools for solving the problems of GWAS. At the same time, a lot of methods of the feature selection use regression models. One of the pioneering and the most well-known papers devoted to the use of regression models in SNP selection has been written by Lander and Botstein [27]. Methods for constructing the corresponding regression models can be referred as *embedded* methods [25]. They performs feature selection in the process of model building and cover a lot of well-known approaches, including the Ridge regression, Least Absolute Shrinkage and Lasso techniques [41] which are the most popular and efficient tools in SNP selection problems. The main advantage of using the Lasso method is that it performs variable selection and classification or regression simultaneously. A lot of approaches using the Lasso method and its modifications have been developed for solving the SNP selection problem in the framework of the GWAS [13, 31, 35, 36, 40, 43]. Hayes [18] provided a comprehensive overview of statistical methods for GWAS in animals, plants, and humans. Various approaches to SNP selection with the Lasso algorithm and other methods can be also found in papers [16, 22, 33, 46].

The main aim of GWAS is to identify SNPs that are directly associated with a trait, i.e., the standard GWAS analyzes each SNP separately in order to identify a set of significant SNPs showing genetic variations associated with the trait. However, an important challenge in the analysis of genome-wide data sets is taking into account the so-called epistatic effect when different SNPs interact in their association with phenotype.

Campos et al. [12] explain some shortcomings of the standard GWAS. They write that the currently identified SNPs might not fully describe genetic diversity. For instance, these SNPs may not capture some forms of genetic variability that are due to copy number variation. Moreover, genetic mechanisms might involve complex interactions among genes and between genes and environmental conditions, or epigenetic mechanisms which are not fully captured by additive models. Many statistical approaches make sense under the assumption that only a few genes affect genetic predisposition. However, GWAS may be unsatisfactory for many important traits which may be affected by a large number of small-effect, possibly interacting, genes. Limitations and pitfalls of prediction analysis in the framework of the GWAS have been discussed in detail by Wray et al. [47] where it is shown how naive implementations can lead to severe bias and misinterpretation of results.

In fact, the epistatic effect can be viewed as gene-gene interaction when the action of one locus depends on the genotype of another locus. At the same time, there are different interpretations of the epistatic effect. A fundamental critical review of different definitions and interpretations of epistasis is provided by Cordell [11] where it is pointed out that there are many conflicting definitions of epistasis, which lead to certain problems in interpretations, namely, the statistical interaction may not correspond to the biological models of epistasis. As indicated by Wan et al. [44], there are mainly three different definitions of gene-gene interactions: functional, compositional and statistical epistasis. We consider only the statistical epistasis which can be regarded as the statistical deviation from the joined effects of two SNPs on the phenotype. At that, the individual SNPs may exhibit no marginal effects.

A lot of methods dealing with epistasis effect have been developed last decades [3, 30, 52, 49, 50, 51, 54]. Comprehensive and interesting reviews of methods detecting interacting the epistatic effect were provided by several authors [7, 45]).

Analyzing various modifications of the Lasso method applied to the GWAS problems, we can point out that many efficient modifications are based on applying special forms of the penalty function, which take into account some additional information about SNP markers and the corresponding phenotype values. Some interesting algorithms [33, 42] devoted to various penalty functions will be studied in the next section. The use of a specific additional information allows us to improve the GWAS and is considered in the paper.

In the present study, we modify the Lasso method by taking into account some peculiarities of the double haploid (DH) lines of barley which are very important in the plant biotechnology. According to the DH method, only two types of genotypes occur for a pair of alleles. From a statistical point of view, we solve a linear regression problem with binary explanatory variables. Our method is based on the well-known adaptive Lasso [56] and takes into account additional information about the correlation between SNPs, frequencies of alleles and expected phenotype values. We propose to use the Bahadur representation [2] by partially applying the ideas provided by Lee and Jun [29] where the authors propose to apply the Bahadur representation to classification problems. The Bahadur representation allows us to compute joint probabilities of SNPs by taking into account the correlation between binary random variables. That is another reason why we analyze only DH lines which produce the binary genotypes. In order to modify the adaptive Lasso, we propose to assign penalty weights in accordance with expected values of the phenotype with respect to a probability mass function somehow defined on the genotype values. In other words, computing the expected values of the phenotype in a special way is a main idea of the proposed method. We show that the proposed modification is directly extended on the case taking into account the epistatic effect.

2 The Lasso method

We analyze n double haploid (DH) lines of barley or a population from n individuals. From a statistical point of view, marker genotypes can be treated as qualitative explanatory variables, i.e., $X_j = (x_{1j}, ..., x_{nj})^{\mathrm{T}}$ is a predictor representing the j-th SNP, $j = 1, ..., p$. Here x_{ij} is a binary variable, i.e., $x_{ij} \in \{0, 1\}$. A quantitative trait of interest or a set of the phenotype values $y_i \in \mathbb{R}$, $i = 1, ..., n$, can be regarded as the response vector $Y = (y_1, ..., y_n)^{\mathrm{T}}$. We also denote $\mathbf{X} = [X_1, ..., X_p]$ is a genotype matrix for n lines or individuals or a predictor matrix in terms of statistics; $\mathbf{x}_i^{\mathrm{T}} = (x_{i1}, ..., x_{ip})$ is a vector of alleles corresponding to the i-th line, $i = 1, ..., n$.

First, we focus on the standard linear regression model

$$y = \sum_{i=1}^{p} \beta_i X_i + \beta_0 + \epsilon = \mathbf{X}\beta + \beta_0 + \epsilon. \quad (1)$$

Here ϵ is a noise variable with the zero-valued expectation; β_i is the SNP effect, $\beta = (\beta_1, ..., \beta_p)$.

Without loss of generality, we assume the predictors and the response are centered, and the predictors are standardized, that is

$$\sum_{i=1}^{n} y_i = 0,\ \sum_{i=1}^{n} x_{ij} = 0,\ \sum_{i=1}^{n} x_{ij}^2 = 1,\ X_i \in \mathbb{R}^p.$$

This implies that the intercept is not included in the regression function.

The Lasso is a regularization technique for simultaneous estimation and variable selection [41]. The Lasso estimates are defined from the following quadratic programming problem:

$$\beta = \arg\min_{\beta\in\mathbb{R}^p} \|Y - \mathbf{X}\beta\|^2,$$

subject to

$$\sum_{j=1}^{p} |\beta_j| \leq s$$

for some $s > 0$. The Lagrange formulation is

$$\beta = \arg\min_{\beta\in\mathbb{R}^p} \|Y - \mathbf{X}\beta\|^2 + \lambda\sum_{j=1}^{p} |\beta_j|,$$

where λ is a nonnegative regularization parameter. The second term is the L_1 penalty which is crucial for the success of the Lasso. The Lasso estimator is usually calculated at a grid of tuning parameters of λ, and a cross-validation procedure is subsequently used to select an appropriate value of λ.

The Lasso penalizes the regression coefficients by their L_1 norm. However, in order to improve the performance of the Lasso, the regression coefficients can be penalized individually. As a result, we write the weighted Lasso estimates as follows:

$$\beta = \arg\min_{\beta\in\mathbb{R}^p} \|Y - \mathbf{X}\beta\|^2 + \lambda\sum_{j=1}^{p} w_j |\beta_j|,$$

where $w_j > 0, j = 1, ..., p$, are weights determined a priori in accordance with some rules. A larger weight w_j corresponds to a higher penalty and discourages the j-th predictor from the model. Conversely, a smaller weight w_j exerts less penalty and encourages selection of the corresponding predictor [55].

The penalized Lasso can be reformulated as the standard Lasso problem [6]. If we introduce new covariates and regression coefficients as

$$\widetilde{x}_{ij} = x_{ij}/w_j,\ i = 1, ..., n,\ \widetilde{\beta}_j = \beta_j w_j,$$

then the weighted Lasso problem can be rewritten as follows:

$$\beta = \arg\min_{\beta\in\mathbb{R}^p} \left\|Y - \widetilde{\mathbf{X}}\widetilde{\beta}\right\|^2 + \lambda\sum_{j=1}^{p} \left|\widetilde{\beta}_j\right|,$$

where $\widetilde{\beta}$ and $\widetilde{\mathbf{X}}$ are the vector and the matrix with elements $\widetilde{\beta}_j$ and $\widetilde{x}_{ij}$, respectively.

Zou [56] proposed one of the methods for determining the weights w_j such that $w_j = 1/\left|\beta_{init,j}\right|$, where $\beta_{init,j}$ is a prior estimator of β_j, for example, the least square estimator. The corresponding Lasso problem is referred as the adaptive Lasso, and it has many nice properties improving the performance of the Lasso. Moreover, it can be a basis for constructing the boosting Lasso [6].

The Lasso has many interesting properties which make the method to be very popular. At the same time, Zou and Hastie [57] point out that in spite of success of the Lasso it has some limitations, in particular, if there is a group of variables among which the pairwise correlations are very high, then the lasso tends to select only one variable from the group and does not care which one is selected. In other words, the Lasso tends to put all the weight on the selected variable. On the one hand, this is a shortcoming. Many methods have been proposed to overcome this obstacle, for example, the so-called the elastic net [57] where the estimates are defined by

$$\beta = \arg\min_{\beta\in\mathbb{R}^p} \left\|Y - \widetilde{\mathbf{X}}\widetilde{\beta}\right\|^2 + \lambda_1\sum_{j=1}^{p} \left|\widetilde{\beta}_j\right| + \lambda_2\left\|\widetilde{\beta}\right\|^2.$$

However, the elastic net requires to assign an additional parameter λ_2 whose value is a priori unknown. On the other hand, in contrast to the ridge regression which tends to select all of the correlated variables and make the corresponding coefficients to be equal, the Lasso selects a group of correlated variables and "isolates" it.

A special choice of the penalty term on the basis of some prior information about SNPs or about an exploited genome selection model itself may lead to a series of useful or important properties of the regression or classification model corresponding to the Lasso. Liu et al. [33] tried to apply the observed fact that there exists a natural grouping structure in SNPs and, more importantly, such groups are correlated. The authors proposed a new penalization method for group variable selection which can properly accommodate the correlation between adjacent groups. Their method referred to as smoothed group Lasso is based on a combination of the group Lasso penalty and a quadratic penalty on the difference of regression coefficients of adjacent groups. Liu et al. [33] assume that SNPs are divided into J groups, each with size d_j, $j = 1, ..., J$, according to their physical locations and correlation patterns. As a results, the vector $\beta = (\beta_1, ..., \beta_J)$ is defined for groups of SNPs, but not for separate SNPs, β_j is the vector of parameters corresponding to SNPs from the j-th group. The authors consider the quadratic loss function of the form:

$$l(\beta) = \left\|Y - \sum_{j=1}^{J} \mathbf{X}_j\beta_j\right\|^2.$$

Here $\mathbf{X}_j$ is an $n\times d_j$ matrix corresponding to the j-th group [33]. There are two main difficulties of using the above considered method. First, it is rather hard from the computation point of view. Second, we have to know a priori the grouping structure SNPs.

An interesting approach for dealing with correlated covariates was proposed by Tutz and Ulbricht [42]. Their method utilizes the correlation between predictors explicitly in the penalty term. Coefficients which correspond to pairs of covariates are weighted according to their marginal correlation. The correlation based penalty is given by

$$Q_\lambda(\beta) = \lambda \sum_{i=1}^{p-1} \sum_{j>i} \left\{ \frac{(\beta_i - \beta_j)^2}{1-\rho_{ij}} + \frac{(\beta_i + \beta_j)^2}{1+\rho_{ij}} \right\}.$$

Here ρ_{ij} denotes the empirical correlation between the i-th and the j-th predictors. If we have the positive correlation, i.e., $\rho_{ij} \to 1$, then the first term in the sum becomes dominant. When $\rho_{ij} \to -1$, then the second term becomes dominant. Both these cases lead to the approximate equality $\beta_i \approx \beta_j$. In case of uncorrelated predictors and $\rho_{ij} \to 0$, the corresponding model is reduced to the ridge regression.

Another model proposed by Park and Hastie [38] constructs sets of indicators representing all the available factors and all possible two-way interactions in order to fit gene-interaction models with the data consisting of genotype measurements and a binary response. The obtained grouped variables are used in the path-following algorithm for the group-Lasso method.

In order to take into account different probabilities of feature values, in particular, to take into account the allele frequency, Zhou et al. [55] proposed a weighted Lasso penalty in the Lasso method such that the weights are assigned in accordance with the following sources of prior knowledge. First, Zhou et al. [55] considered genotyping errors such that the unreliable variants should be penalized more. Second, they pointed out that the allele frequencies can be used in accordance with an idea of Madsen and Browning [34] where it was proposed to take the weight $w = 2\sqrt{\pi(1-\pi)}$ for a variant with population frequency π by arguing that this scheme assigns smaller penalties to rarer variants as suggested by classical population genetics theory.

3 The proposed method

3.1 Motivation for a new penalized method

The considered in previous sections modifications of the Lasso are efficient tools for solving the GWAS and SNP selection problems. Their performance has been experimentally shown by many authors [13, 22, 23, 31, 33]. However, every real application problem has some peculiarities whose accounting might improve the regression method. Let us mention these peculiarities.

1. First of all, our aim is not to find the "best" regression model for the given information, but to select SNPs which impact on the smallest (largest) values of the phenotype, for example, on the heading date early flowering of barley in the studied applications. This does not mean that the whole fitted regression model is not important for us. We have to combine two above aims. This can be done by introducing the weighted Lasso penalties of a special form. This form has to take into account in the first place the smallest values of phenotype. Smaller values of the phenotype produce larger weights, whereas larger values of the phenotype should be also considered. It would seem that we can assign the weights to phenotype values with respect to their closeness to the minimal phenotype value. However, the phenotype values are random and depend on environment conditions. Moreover, the smallest phenotype value does not mean that its value is caused by the corresponding genotype. This implies that we cannot assign weights to the available phenotype values. *The main idea underlying the method is to assign weights to expected values of the phenotype with respect to a probability mass function somehow defined on the genotype values.*

2. The genotype values corresponding to every SNP in the studied application make up a binary vector. The dependence of SNPs leads to dependence of the corresponding binary vectors which can be estimated.

3. The allele frequencies and correlations indirectly impact on the smallest values of the phenotype.

3.2 A method for computing weights for the Lasso

By extending the ideas proposed by aforementioned authors [34, 42, 55], we define the weighted Lasso penalty in a new way. The main idea is the following. We define the average contribution of every SNP to the mean phenotype value. These contributions or their function are nothing else but the weights w_k in the adaptive Lasso. They have to take into account the probabilities of alleles, the correlations between SNPs and the phenotype values. The next question is how to determine the average contribution of every SNP. It can be carried out as follows:

1. For every genotype vector $\mathbf{x}_j$ (the j-th line), we compute joint probabilities $\pi(x_{kj}, x_{ij})$ of all pairs (k, i) of SNPs by taking into account correlations between pairs of random variables (SNPs).

2. For every pair (k, i), we compute the mean phenotype value R_{ki} as the expectation of phenotypes with respect to the joint probabilities $\pi(x_{kj}, x_{ij})$ over all lines or individuals.

3. The average contribution of every, say k-th, SNP into the phenotype is computed by averaging the mean phenotype values R_{ki} over all $i = 1, ..., p$.

4. The weights or their function for the adaptive Lasso are defined by the average contributions.

Below we consider every step in detail.

3.3 Bahadur representation

The main idea for using the joint probability $\pi(x_{jk}, x_{ji})$ is to take into account the correlation between SNPs with indices k and i. For every pair of SNPs X_k and X_i, we have to determine the joint probability $\pi(x_{kj}, x_{ij})$, $i = 1, ..., p$, $i \neq k$, of the j-th individual. It can be computed by using the so-called Bahadur representation proposed by Bahadur [2]. The Bahadur representation takes into account the correlation between binary variables, and it can be written in the case of two binary variables with numbers k and i as

$$\begin{aligned}\pi(x_k, x_i) &= \pi_k^{x_k}(1-\pi_k)^{1-x_k} \cdot \pi_i^{x_i}(1-\pi_i)^{1-x_i} \\ &\quad \times (1 + \rho_{ki} u_k u_i). \qquad (2)\end{aligned}$$

Here π_k is the probability of an allele for the k-th SNP or its allele frequency, i.e., $\pi_k = \Pr\{x_k = 1\}$; ρ_{ki} is the correlation coefficient between the k-th and the i-th SNPs which is defined as $\rho_{ki} = \mathbb{E}[U_k U_i]$, where the random standardized variable U_k takes the values u_k such that there hold

$$U_k = \frac{X_k - \pi_k}{\sqrt{\pi_k(1-\pi_k)}},\ u_k = \frac{x_k - \pi_k}{\sqrt{\pi_k(1-\pi_k)}}.$$

Note that the first term in the right-hand side of the expression for $\pi(x_k, x_i)$ represents the joint probability mass function under condition that variables X_k and X_i are statistically independent. The second term includes the interaction from the first order up to the second. Note also that U_k should be evaluated by estimating π_k.

The corresponding estimates of parameters π_k, u_k, ρ_{ki} denoted as $\widehat{\pi}_k$, $\widehat{u}_k$, $\widehat{\rho}_{ki}$ are computed by means of the following expressions:

$$\widehat{\pi}_k = \sum_{l=1}^{n} x_{kl}/n,\ \ \widehat{\rho}_{ki} = \sum_{l=1}^{n} \widehat{u}_{kl}\widehat{u}_{il}/n,$$

where n is the number of individuals and

$$\widehat{u}_{kl} = \frac{(x_{kl} - \widehat{\pi}_k)}{\sqrt{\widehat{\pi}_k(1-\widehat{\pi}_k)}}$$

is the l-th observed value of variable U_k.

It should be noted that the Bahadur representation can be written also for joint probabilities of three, four, etc. variables. [32] mention a property of the Bahadur representation such that the joint probability distribution of any subset $x_1, x_2, ..., x_t$ can be written as follows:

$$\begin{aligned}\pi(x_1, ..., x_t) &= \prod_{i=1}^{t} \pi_i^{x_i}(1-\pi_i)^{1-x_i} \\ &\quad \times \left(1 + \sum_{Q \subset \{1,...,t\},\ |Q| \geq 2} \rho_Q \prod_{k \in Q} u_k\right).\end{aligned}$$

Here ρ_Q represents $\rho_{i_1,...,i_k}$ if $Q = \{i_1, ..., i_k\}$ and $|Q|$ denotes the number of elements in Q. The main disadvantage of the Bahadur representation is the large number of parameters and hard computations required for getting the probabilities. Therefore, we restrict our study only by probabilities of two variables.

It should be noted that the Bahadur representation has been used in some classification models. One of the interesting models for discriminant analysis of binary data was proposed by Lee and Jun [29]. The main contribution of [29] is that they proposed to take into account the correlation between variables or, more exactly, estimates of the correlation by means of the Bahadur representation.

There are pros and cons of using this model when the number of variables is larger than the number of observations. For example, Bickel and Levina [4] suppose that classification rules ignoring the correlation structure often perform better in this case. However, Lee and Jun [29] show by means of various experimental studies that the correlation should be taken into account in all cases.

In spite of arguments of [29] in defense of the correlation analysis for high-dimensional data, there is a risk of incorrect estimates of interactions of the large order. Moreover, it is practically impossible to compute the corresponding joint probabilities when the number of SNPs is rather large. Therefore, we propose an approach which partially uses joint probabilities of variables and partially takes into account the correlation between the variables.

3.4 Average contributions of SNPs

In order to determine the average contribution of the k-th SNP into the mean value of the phenotype, we consider all possible pairs of SNPs such that one of the SNPs in every pair is the k-th SNP, i.e., we are interesting in considering $p-1$ pairs of SNPs with numbers $(k,1), ..., (k, k-1), (k, k+1), ..., (k, p)$. Every pair, say (k, i), determines a mean phenotype value R_{ki} corresponding to this pair of SNPs as follows:

$$R_{ki} = \frac{\sum_{j=1}^{n} \pi(x_{kj}, x_{ij}) y_j}{\sum_{j=1}^{n} \pi(x_{kj}, x_{ij})}. \qquad (3)$$

In other words, we can compute the expected phenotype value under condition that every phenotype value y_j is produced by the subset of the genotypes corresponding to the k-th and the i-th SNPs. The measure R_{ki} can be regarded as a contribution of the k-th and the i-th SNPs to the mean phenotype value.

Then the contribution of the k-th SNP denoted by $\widetilde{R}_k$ into the mean phenotype value can be determined through averaging the measures R_{ki}, i.e., it is computed as

$$\widetilde{R}_k = \frac{1}{p-1} \sum_{i=1, i \neq k}^{p} R_{ki}. \qquad (4)$$

It is obvious that the smaller values of the measure $\widetilde{R}_k$ give us significant or top ranked SNPs and exert less penalty w_k, i.e., we can introduce an increasing function g such that

$$w_k = g\left(1/\left|\beta_{init,k}\right|\right).$$

One of the possible functions which will be used in numerical experiments is

$$w_k=\left(\frac{\widetilde{R}_k-\min_{k=1,...,p}\widetilde{R}_k}{\max_{k=1,...,p}\widetilde{R}_k-\min_{k=1,...,p}\widetilde{R}_k}\right)^{-q}. \qquad (5)$$

Here q is a positive real which defines how changes of the difference between $\widetilde{R}_k$ and $\min_{k=1,...,p}\widetilde{R}_k$ impact on changes of weights w_k. The number q can be regarded as a tuning parameter whose optimal value can be obtained by means of the cross-validation procedure.

In sum, the obtained weights take into account the correlation between SNPs, the allele frequencies, binary data and the fact that the smallest (largest) values of the phenotype are more important in comparison with other values because we are looking for the SNPs which impact on the values of some trait with predefined properties, for example, the heading date of barley should be as small as possible. At the same time, we do not need to directly use the obtained weights and to implement the adaptive Lasso algorithm. It has been mentioned in the previous section that the adaptive Lasso can be transformed to the standard Lasso by means of introducing new covariates $\widetilde{x}_{ij}=x_{ij}/w_j$.

Finally, we write the following SNP selection algorithm.

Algorithm 1 The SNP selection algorithm.

```
Require: $Y=(y_1,...,y_n)^{\mathrm{T}}$ is the response vector (phenotype values), $\mathbf{X}=[X_1,...,X_p]$ is the binary predictor matrix (genotype values).
Ensure: $\beta=(\beta_1,...,\beta_p)$ is the vector of the regression coefficients (degrees of the SNP effect).
  repeat
    $k \leftarrow 1$
    Compute joint probabilities $\pi(x_{jk},x_{ji})$, $i=1,...,p$, $i\neq k$, for all $j=1,...,n$, by means of the Bahadur representation (2)
    Compute the mean phenotype values $R_{ki}$, for all $i=1,...,p$, $i\neq k$, by means of (3)
    Compute the average mean phenotype value $\widetilde{R}_k$ by means of (4)
    Compute the weights $w_k$ by means of (5)
    Compute new variables $\widetilde{x}_{ik}=x_{ik}/w_k,\ i=1,...,n$.
  until $k>p$
  Compute $\widetilde{\beta}^{\mathrm{opt}}$ by using the standard Lasso with $\widetilde{\beta}$ and $\widetilde{\mathbf{X}}$ instead of $\widetilde{\beta}$ and $\mathbf{X}$.
  Compute $\beta_k=\widetilde{\beta}_k/\widetilde{R}_k$, $k=1,...,p$.
```

Let us indicate the main virtues of the proposed method. First of all, it does not require to develop special algorithms for solving the optimization problem for computing the vector of regression coefficients β. The obtained problem is solved as the standard Lasso algorithm after reformulating the penalized Lasso.

Second, the method is rather general because we could change the weights in (5) in accordance with our goal. For example, in one of the applications, we have aimed to minimize the mean heading date of barley as the mean phenotype value. However, we could aim to maximize, for example, the amount of grain protein. In this case, we change (5) by taking decreasing function g as follows:

$$w_k=\left(\frac{\max_{k=1,...,p}\widetilde{R}_k-\widetilde{R}_k}{\max_{k=1,...,p}\widetilde{R}_k-\min_{k=1,...,p}\widetilde{R}_k}\right)^{-q}.$$

Here the larger values of the measure $\widetilde{R}_k$ give us more significant SNPs and exert less penalty w_k.

Third, we consider not only correlations between SNPs, but also joint probabilities accounting for correlations. The joint probabilities are more informative in comparison with the correlation coefficients.

Fourth, we have simplified procedures for computing the joint probabilities. This substantially reduces the computation time.

3.5 The proposed method with epistatic effect

A lot of studies devoted to the epistatic effect (see, for example, [5]) consider extension of the so-called main effect model (1) on the interaction model which can be written as

$$Y=\sum_{i=1}^{p}\beta_iX_i+\sum_{i<j,\ i,j=1,...,p}\beta_{ij}X_iX_j+\beta_0+\epsilon. \qquad (6)$$

The second term in (6) corresponds to pairwise interactions whose number is $p(1-p)/2$. Here β_{ij} is the parameter characterizing the epistatic interaction effect of a pair SNPs with indices i and j. Now the weighted Lasso estimates can be written as follows:

$$(\beta,\beta^*)=\arg\min_{\beta\in\mathbb{R}^p}\left\|Y-\mathbf{X}\beta-\mathbf{X}^*\beta^*\right\|^2$$
$$+\lambda\sum_{j=1}^{p}w_j\left|\beta_j\right|+\lambda\sum_{i<j,\ i,j=1,...,p}w_{ij}\left|\beta_{ij}\right|,$$

where $\beta^*=(\beta_{12},...,\beta_{p-1,p})$ is the additional vector characterizing the epistatic interaction effect of every pair of SNPs; $\mathbf{X}^*=(X_1X_2,...,X_{p-1}X_p)$ is the vector of covariates corresponding to pairwise interactions; $w_{ij}>0$ are weights penalizing the additional parameters β_{ij}, $i<j$, $i,j=1,...,p$, in accordance with the rules of the adaptive Lasso [55].

It can be seen from the previous section that the weight or contribution of the pair of the k-th and the i-th SNPs into the phenotype values can be determined by the mean phenotype value R_{ki} obtained by means of (3). It is interesting to note that, in contrast to the k-th SNP contribution $\widetilde{R}_k$ obtained in a heuristic way (5), the value R_{ki} is the expectation of the phenotype with respect to the probability mass function $\pi(x_{kj},x_{ij})/\sum_{j=1}^{n}\pi(x_{kj},x_{ij})$. So, the weight w_{ij} can be directly computed as

$$w_{ij}=\left(\frac{R_{ij}-\min_{ij}R_{ij}}{\max_{ij}R_{ij}-\min_{ij}R_{ij}}\right)^{-q}.$$

In order to take into account the interactions and to implement the method for epistasis detection, we apply a two-stage procedure (see the Screen and Clean method proposed by Wu et al. [48] for example). The first stage is for constructing the main effect model and searching for marginal significant SNPs by using the proposed penalized Lasso method with weights w_k determined from (5). Then only top ranked SNPs and pairs of SNPs composed from the significant ones are used in the interaction penalized Lasso model. The main idea here is to again use the Bahadur representation, namely, the mean phenotype values R_{ki} computed by means of (3). This is a very important place because we do not need to repeatedly compute the mean phenotype values. They have been computed during construction of the main effect model.

We do not provide here an algorithm for computing the optimal vectors β and β^* because it is just a simple extension of the algorithm given above.

4 Numerical experiments

The Lasso method in numerical experiments is regarded as a special case of a general problem solved by means of the R-package "glmnet" developed by Friedman et al. [14]. The tuning parameter λ is computed by using the function cv.glmnet() with 10-fold cross validation.

Below we use indices of SNPs instead of their full titles for short.

4.1 Data sets

Numerical experiments are carried out on three populations of double haploid (DH) lines of barley:

1. The first dataset consists of 93 DH lines of barley described in [8] and [9]. Phenotyping and genotyping data are available at Oregon Wolfe Barley Data (OWBD) and GrainGenes Tools. The lines are analyzed with respect to seven phenotypic traits: spike length (SL) in cm; grain number (GN); floret number (FS); hundred grain weight (HGW) in g of 100 grains; plant height (PH) in cm; spike number (SN); heading date (HD) in days. The linkage map consists of 1328 markers (SNPs).

2. The second dataset consists of 92 DH lines of barley obtained from the Dicktoo x Morex cross and described by several authors [20, 19, 37]. Phenotyping and genotyping data are available at http://wheat.pw.usda.gov/ggpages/DxM/ . We analyze the lines with respect to two phenotypic traits: heading date with and without vernalization with an 8-h light/16-h dark photoperiod regime. The linkage map consists of 117 markers.

3. The third population dataset includes 150 DH lines of barley obtained from the Steptoe x Morex cross [10, 21]. Phenotyping and genotyping data are available at http://wheat.pw.usda.gov/ggpages/SxM. The linkage map consists of 223 markers. The lines are analyzed with respect to the heading date (HD) trait, which is measured in 16 environments, and grain yield (GY) trait, which is measured in 6 environments.

4.2 Missing data

Missing marker data in all the datasets are estimated by means of the following heuristic procedure which can be regarded as some modification of the well-known method of K-nearest neighbors. Suppose the vector X_i corresponding to the i-th SNP has a missing value at the k-th position, i.e., X_{ik} is missing. By using the specific Hamming distance between the vector X_i and all vectors X_j, $j = 1, ...p$, $j \neq i$, we select K nearest neighbors $X_{i_1}, ..., X_{i_K}$ or K closest vectors. In order to take into account the missing values, they are excluded from computing the Hamming distance. That is why we use the specific Hamming distance in order to compare vectors with different numbers of missing elements, i.e., we compute the distance per one element of X_i. The imputed value is that represents the maximum of the K values at the k-th position of all the nearest neighbors $X_{i_1 k}, ..., X_{i_K k}$.

4.3 Error measure

From each of the (synthetic or real) data sets we randomly select two distinct subsets: a training data set of n examples to learn the model, and a test data set of n_{test} instances to evaluate the performance of the algorithms. The performance is assessed by means of the mean square residual (RMSR), which is defined by

$$\text{RMSR} = \frac{\sum_{i=1}^{n_{test}} (y_i - \hat{f}(\mathbf{x}_i))^2}{n_{test}},$$

where $\hat{f}$ is the function estimated by the proposed method, and $\hat{f}(\mathbf{x}_i)$ is the predicted value of the phenotype value y_i for each $i \in \{1, \dots, n_{test}\}$. The error measure RMSR is computed from repeatedly random drawing training and test data sets and by averaging over the runs. The smaller the values of the average error measure are, the better the corresponding method. We use the one-fold-cross-validation, i.e., $n_{test} = 1$. This is because the number of lines is very small in comparison with the number of SNPs and we cannot reduce them.

4.4 The first dataset

First, we investigate DH lines of barley from OWBD. Values of the RMSR for the first dataset are shown in Table 1, where the first column corresponds to seven traits analyzed, columns 2-5 illustrate the RMSR by using only 40 top ranked SNPs. At that, we study cases when the accuracy is determined for all lines (All lines) and for the first 10 lines with the smallest values of phenotypes (First 10 lines). Abbreviations S.L. and P.L. denote the standard and new proposed Lasso methods, respectively. One can see that the proposed method provides better accuracy for the most traits. It does not mean that it can be successful in all cases. It is seen from Table 1 that the proposed method by traits PH and HD does not outperform the standard Lasso. Perhaps, another function determining the weights w_k from R_k could provide better results, but we did not find it. In addition, we can observe from Table 1 that use only of top ranked SNPs gives outperforming results in comparison with taking all SNPs for modelling GWAS. The same can be said about considering all lines and the first 10 lines.

Table 2 illustrates how the error measures depend on the reduced number of top ranked SNPs which are used for constructing the GWAS for the spike length trait. We take the fixed value of $q = 0.25$. It can be seen from Table 2 that the optimal number of top ranked SNPs is 40. It is interesting to observe also that the standard Lasso weakly depends on the SNP number.

Table 3 is similar to Table 1, but RMSRs in Table 3 are obtained by taking into account the epistatic effect. By comparing Tables 1 and 3, we can see that the consideration of epistasis allows us to construct a more accurate model. Moreover, the proposed method outperforms the standard Lasso even for traits PH and HD which distinguished from other traits and illustrated worse results with the proposed method (see Table 1). This is a very important fact showing that joint probabilities of pairs of SNPs as well as correlations between SNPs may improve the GWAS.

Table 4 shows the top ranked SNPs or their pairs with the largest 10 weights β obtained by means of the standard Lasso and the proposed method. Moreover, Table 4 shows the chromosomes where the corresponding SNPs are located. One can see that the largest weight has a pair of SNPs 997×1279. This implies that impact of the epistatic effect is very significant. It is interesting to note that the both methods select this pair of SNPs as the most significant one.

4.5 The second dataset

Let us study the dataset consisting of 92 DH lines of barley obtained from the Dicktoo x Morex cross. Tables 5 and 6 contain error measures for the Dicktoo x Morex dataset by considering two traits mentioned above. At that, Table 5 is obtained without taking into account the epistatic effect. In Table 6, the results are represented under condition of epistasis. Comparison of the tables shows that the use of condition of epistasis allows us to get outperforming results.

Table 7 shows the top ranked SNPs or their pairs with the largest 10 weights β obtained by means of the standard Lasso and the proposed method for the heading date without vernalization.

4.6 The third dataset

The third dataset consists of 150 DH lines of barley obtained from the Steptoe x Morex cross. Values of the RMSR for the third dataset are shown in Table 8. It can be seen from the table that the proposed method provides outperforming results. Table 9 shows also reduced values of the RMSR for the case of taking into account the epistatic effect. Comparing Tables 8 and 9, we can conclude that the model taking into account the epistatic effect significantly improves the regressor accuracy when the model is constructed by using only 40 top ranked SNPs. Moreover, the standard Lasso method also shows better results when the epistatic effect is considered.

It should be noted that the results given in Tables 8 and 9 is obtained for a certain value of q, namely, for $q = 0.8$. However, it is interesting to analyze how the value q impact on numerical results by using the third dataset. Figs. 1-4 depict the difference D between RMSRs of the proposed and standard Lasso methods for the HD trait. The larger the values of D are, the better the corresponding proposed method. The positive values of D say that the proposed method outperforms the standard Lasso for the corresponding values of q. It can be seen from Figs. 1-4 that there is an optimal value of q for every condition of the model use such that D achieves its maximum at this q. For example, it follows from Fig. 1 that the best results by using only top ranked SNPs can be obtained by $q = 0.8$. If we use all SNPs and analyze the first examples, then the optimal value of q is 0.5 (see Fig. 2). The same conclusions can be inferred from pictures illustrating the methods taking into account the epistatic effect (see Figs. 3-4).

Table 10 shows the top ranked SNPs and their pairs with the largest 10 weights β obtained by means of the standard Lasso and the proposed method for the grain yield trait.

It is interesting to note that the use of t-statistics for computing weights β of SNPs by the same parameters for GY trait gives the following 10 top ranked SNPs:

$$\begin{array}{cccccc} 82 & 81 & 83 & 84 & 85 & 79 \\ 86 & 130 & 129 & 80\ . \end{array}$$

One can see that the most top ranked SNPs concentrated around the SNP with index 82. This is the obvious interaction of genes in a group of SNPs located at the same chromosome.

5 Conclusion

The results of numerical experiments and the logic underlying the proposed method have demonstrated that the proposed method outperforms the standard Lasso for many real datasets. Moreover, it takes into account the epistatic effect or the SNP-SNP interaction. It should be noted that the proposed method is very simple from a computation point of view. It does not require to develop a special software. The standard software (package “glmnet” in R) can be used for the method.

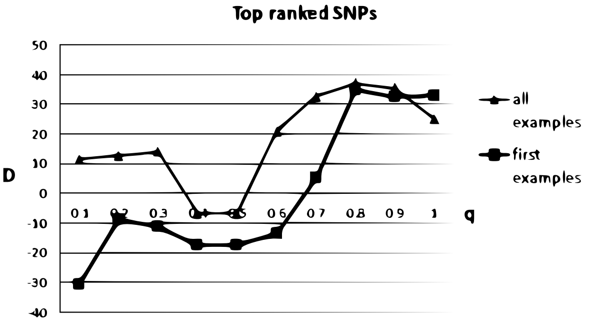


Figure 1: Difference between RMSRs of the standard and proposed Lasso methods for top SNPs.

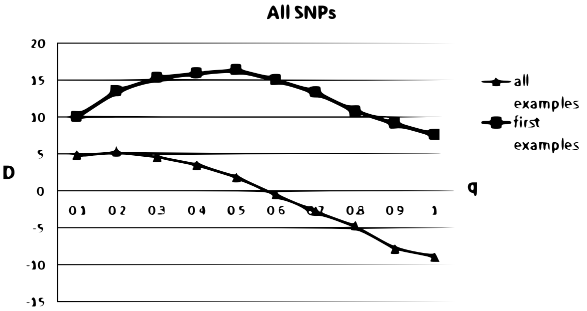


Figure 2: Difference between RMSRs of the standard and proposed Lasso methods for all SNPs.

It can be seen from the Bahadur representation that one of the crucial elements of the proposed method is a set of correlation coefficients between SNPs. It should be noted that they often use in GWAS as additional information. However, the correlation coefficients do not contain all probabilistic information about impacts of SNPs on values of a phenotype. The joint probabilities taking into account the correlation between SNPs can be viewed as a way for constructing association between SNPs and traits.

We have analyzed DH populations of barley. According to the DH method, only two types of genotypes occur for a pair of alleles, i.e., every x_{ij} takes only two values. At the same time, in diploid method, three genotypes occur, i.e., every x_{ij} takes three values. In this case the Bahadur representation cannot be applied, but the Sarmanov-Lancaster expansion [26, 39] can be used [1]. This is a direction for further research.

Of course, we have used a heuristic procedure by taking pairs of SNPs for computing R_{ki} by means of (3). We could consider joint probabilities of three and more SNPs. However, the increase of SNP numbers for computing the joint probabilities is impossible when the total number of SNPs is rather large. In this way, we can propose a multistep procedure when the large set of top ranked SNPs is consequently determined by computing the joint probabilities of SNP pairs at the first step, then by computing the joint probabilities of SNP triples but from the reduced set obtained at the previous step. This procedure can be continued. At that, we could use the ridge regression in order to avoid a situation when a very small number of SNPs are obtained at some step. However, this is a direction for further research. The above modification may be very useful when the number of lines or individuals is small.

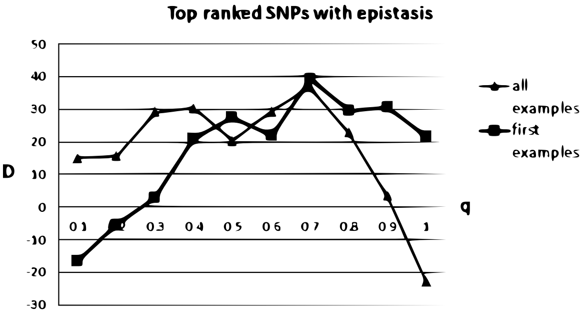


Figure 3: Difference between RMSRs of the standard and proposed Lasso methods for top ranking SNPs with epistasis.

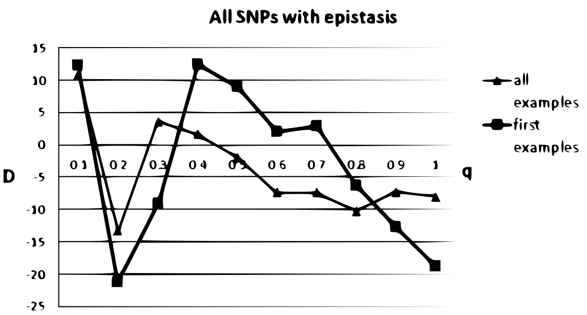


Figure 4: Difference between RMSRs of the standard and proposed Lasso methods for all SNPs with epistasis.

[1] A rather simple presentation of the Sarmanov-Lancaster expansion and its usage can be found in the paper I. Goodman and D.H. Johnson, Multivariate dependence and the Sarmanov-Lancaster expansion, 2005, http://www-ece.rice.edu/~igoodman/papers/goodman-johnson05.pdf

Table 1: RMSRs for the standard and proposed Lasso for OWBD.

	Top ranked SNPs				All SNPs			
	All lines		First 10 lines		All lines		First 10 lines	
Trait	S.L.	P.L.	S.L.	P.L.	S.L.	P.L.	S.L.	P.L.
SL	1.760	0.583	1.596	0.515	3.280	4.429	2.594	3.845
GN	75.62	71.62	61.16	57.96	140.6	139.3	103.8	110.0
FS	77.86	79.59	50.39	40.78	162.0	158.9	75.68	74.60
HGW	0.134	0.121	0.094	0.085	0.209	0.186	0.147	0.137
PH	24.92	24.92	16.23	16.23	237.5	237.5	147.5	147.5
SN	17.96	16.53	8.847	8.285	27.71	27.92	12.73	12.66
HD	31.08	31.08	29.12	29.12	130.3	130.3	82.50	82.50

Table 2: RMSRs for the standard and proposed Lasso for OWBD by different numbers of top ranked SNPs.

	All lines		First 10 lines	
SNP numbers	S.L.	P.L.	S.L.	P.L.
20	1.412	1.116	1.396	1.177
40	1.392	0.550	1.392	0.443
60	1.393	0.680	1.393	0.611
80	1.393	1.548	1.393	1.569

Table 3: RMSRs for the standard and proposed Lasso for OWBD with epistasis.

	Top ranked SNPs				All SNPs			
	All lines		First 10 lines		All lines		First 10 lines	
Trait	S.L.	P.L.	S.L.	P.L.	S.L.	P.L.	S.L.	P.L.
SL	0.815	0.724	0.697	0.652	2.866	3.638	2.622	2.630
GN	51.67	52.97	45.25	42.23	85.15	86.13	77.27	77.43
FS	44.99	70.42	33.19	27.53	72.00	102.1	51.17	50.86
HGW	0.077	0.056	0.069	0.052	0.156	0.107	0.118	0.098
PH	46.08	33.69	43.13	39.97	254.2	247.0	219.8	249.1
SN	12.21	10.42	6.842	5.274	24.26	23.84	13.34	10.71
HD	31.08	29.05	29.12	25.79	130.3	133.4	82.50	78.47

Table 4: Top ranked SNPs and their weights for the standard and proposed Lasso for OWBD HD with epistasis.

S.L.			P.L.		
SNP	chromosome	β	SNP	chromosome	β
997 × 1279	6 × 6	3.421	997 × 1279	6 × 6	3.401
138	1	3.314	903 × 325	5 × 2	3.193
896	5	3.176	1101	6	−2.764
1101 × 1152	6 × 6	2.750	138	1	2.661
734	4	−2.683	896	5	2.634
670 × 273	4 × 2	−2.542	1101 × 1152	6 × 6	2.583
324 × 903	2 × 5	2.128	725	4	−2.493
1096 × 976	6 × 6	−1.877	670 × 273	4 × 2	−2.012
997 × 526	6 × 3	1.826	734	4	−1.629
903 × 325	5 × 2	1.706	1101 × 976	6 × 6	−1.447

Table 5: RMSRs for the standard and proposed Lasso for Dicktoo-Morex without epistasis.

	Top ranked SNPs				All SNPs			
	All lines		First 10 lines		All lines		First 10 lines	
Trait	S.L.	P.L.	S.L.	P.L.	S.L.	P.L.	S.L.	P.L.
unvernalized	43.74	44.68	18.13	17.87	63.57	60.77	26.57	27.43
vernalized	79.56	78.37	27.95	26.49	108.3	106.7	26.46	25.81

Table 6: RMSRs for the standard and proposed Lasso for Dicktoo-Morex with epistasis.

	Top ranked SNPs				All SNPs			
	All lines		First 10 lines		All lines		First 10 lines	
Trait	S.L.	P.L.	S.L.	P.L.	S.L.	P.L.	S.L.	P.L.
unvernalized	34.33	38.85	24.08	17.63	58.63	58.28	30.46	27.81
vernalized	38.41	62.26	17.28	17.11	124.0	119.8	33.32	32.14

Table 7: Top ranked SNPs and their weights for the standard and proposed Lasso for Dicktoo-Morex HD with epistasis.

S.L.			P.L.		
SNP	chromosome	β	SNP	chromosome	β
112	7	-7.860	112	7	-7.238
22	2	6.387	110	7	-5.242
110	7	-5.296	20	2	4.426
20	2	4.074	22	2	4.377
113	7	-2.26	113	7	-3.55
51	3	-1.684	21	2	2.645
59 × 84	4 × 5	-1.394	50	3	-2.398
84	5	-1.261	49 × 113	3 × 7	1.627
19	2	1.002	84	5	-1.174
49	3	-0.980	33 × 50	2 × 3	-1.037

Table 8: RMSRs for the standard and proposed Lasso for Steptoe-Morex without epistasis.

	Top ranked SNPs				All SNPs			
	All lines		First 10 lines		All lines		First 10 lines	
Trait	S.L.	P.L.	S.L.	P.L.	S.L.	P.L.	S.L.	P.L.
HD	79.05	41.95	78.68	43.73	46.62	51.37	57.49	46.66
GY	104.0	78.99	95.81	74.24	137.3	140.4	151.3	163.9

Table 9: RMSRs for the standard and proposed Lasso for Steptoe-Morex without epistasis.

	Top ranked SNPs				All SNPs			
	All lines		First 10 lines		All lines		First 10 lines	
Trait	S.L.	P.L.	S.L.	P.L.	S.L.	P.L.	S.L.	P.L.
HD	77.15	40.31	78.52	39.14	44.12	51.52	50.15	47.26
GY	87.48	23.00	78.63	18.61	156.5	161.8	192.8	171.1

Table 10: Top ranked SNPs and their weights for the standard and proposed Lasso for the Steptoe-Morex GY with epistasis.

S.L.			P.L.		
SNP	chromosome	β	SNP	chromosome	β
82	3	9.562	82	3	13.158
53	2	-7.067	53	2	-6.687
81	3	5.278	222 × 114	7 × 4	5.355
29	1	-4.62	29	1	-5.089
42 × 53	2 × 2	4.126	68	2	4.599
20	1	-3.693	20	1	-3.543
111	4	3.510	108	4	3.338
68	2	2.377	154 × 19	5 × 1	-3.156
72	2	2.348	154 × 45	5 × 2	3.028
203	7	-1.675	108 × 105	4 × 3	2.881

References

[1] W. Altidor, T.M. Khoshgoftaar, J. Van Hulse, and A. Napolitano (2011) Ensemble feature ranking methods for data intensive computing applications. In B. Furht and A. Escalante, editors, *Handbook of Data Intensive Computing*, pages 349–376. Springer, New York.

[2] R.R. Bahadur (1961) A representation of the joint distribution of response to n dichotomous items. In H. Solomon, editor, *Studies in Item Analysis and Prediction*, pages 158–168. Stanford University Press, Palo Alto, CA.

[3] A.L. Beam, A. Motsinger-Reif, and J. Doyle (2014) Bayesian neural networks for detecting epistasis in genetic association studies. *BMC Bioinformatics*, 15(368):1–12.

[4] P.J. Bickel and E. Levina (2004) Some theory for fisher's linear discriminant function, 'naive bayes', and some alternatives when there are many more variables than observations. *Bernoulli*, 10(6):989–1010.

[5] J. Bocianowski (2014) Estimation of epistasis in doubled haploid barley populations considering interactions between all possible marker pairs. *Euphytica*, 196(1):105–115.

[6] P. Buhlmann and S. van de Geer (2011) *Statistics for High-Dimensional Data: Methods, Theory and Applications*. Springer Series in Statistics. Springer, Berlin Heidelberg.

[7] L. Chen, G. Yu, C.D. Langefeld, D.J. Miller, R.T. Guy, J. Raghuram, X. Yuan, D.M. Herrington, and Y. Wang (2011) Comparative analysis of methods for detecting interacting loci. *BMC Genomics*, 12:344:1–23.

[8] Y. Chutimanitsakun, R.W. Nipper, A. Cuesta-Marcos, L. Cistue, A. Corey, T. Filichkina, E.A. Johnson, and P.M. Hayes (2011) Construction and application for qtl analysis of a restriction site associated dna (rad) linkage map in barley. *BMC Genomics*, 12:4:1–13.

[9] L. Cistue, A. Cuesta-Marcos, S. Chao, B. Echavarri, Y. Chutimanitsakun, A. Corey, T. Filichkina, N. Garcia-Marino, I. Romagosa, and P.M. Hayes (2011) Comparative mapping of the oregon wolfe barley using doubled haploid lines derived from female and male gametes. *Theoretical and applied genetics*, 122(7):1399–1410.

[10] T.J. Close, P.R. Bhat, S. Lonardi, Y. Wu, N. Rostoks, L. Ramsay, A. Druka, N. Stein, J.T. Svensson, S. Wanamaker, S. Bozdag, M.L. Roose, M.J. Moscou, S. Chao, R.K. Varshney, P. Szucs, K. Sato, P.M. Hayes, D.E. Matthews, A. Kleinhofs, G.J. Muehlbauer, J. DeYoung, D.F. Marshall, K. Madishetty, R.D. Fenton, P. Condamine, A. Graner, and R. Waugh (2009) Development and implementation of high-throughput SNP genotyping in barley. *BMC Genomics*, 10:582:1–13.

[11] H.J. Cordell (2002) Epistasis: what it means, what it doesn't mean, and statistical methods to detect it in humans. *Human Molecular Genetics*, 11(20):2463–2468.

[12] G. de los Campos, D. Gianola, and D.B. Allison (2010) Predicting genetic predisposition in humans: the promise of whole-genome markers. *Nature Reviews Genetics*, 11(12):880–886.

[13] Z. Feng, X. Yang, S. Subedi, and P.D. McNicholas (2012) The LASSO and sparse least squares regression methods for SNP selection in predicting quantitative traits. *IEEE/ACM Trans. Comput. Biol. Bioinformatics*, 9(2):629–636.

[14] J.H. Friedman, T. Hastie, and R. Tibshirani (2010) Regularization paths for generalized linear models via coordinate descent. *Journal of Statistical Software*, 33(1):1–22.

[15] M.E. Goddard, N.R. Wray, K. Verbyla, and P.M. Visscher (2009) Estimating effects and making predictions from genome-wide marker data. *Statistical Science*, 24(4):517–529.

[16] X. Gu, G. Yin, and J.J. Lee (2013) Bayesian two-step Lasso strategy for biomarker selection in personalized medicine development for time-to-event endpoints. *Contemporary Clinical Trials*, 36(2):642 – 650.

[17] I. Guyon, J. Weston, S. Barnhill, and V. Vapnik (2002) Gene selection for cancer classification using support vector machines. *Machine Learning*, 46:389–422.

[18] B. Hayes (2013) Overview of statistical methods for genome-wide association studies (GWAS). *Methods in Molecular Biology*, 1019:149–169.

[19] P. Hayes, F.Q. Chen, A. Corey, A. Pan, T.H.H. Chen, E. Baird, W. Powell, W. Thomas, R. Waugh, Z. Bedo, I. Karsai, T. Blake, and L. Oberthur (1997) The dicktoo x morex population. In PaulH. Li and TonyH.H. Chen, editors, *Plant Cold Hardiness*, pages 77–87. Springer US.

[20] P.M. Hayes, T. Blake, T.H.H. Chen, S. Tragoonrung, F. Chen and.A. Pan, and B. Liu (1993) Quantitative trait loci on barley (Hordeum vulgare L.) chromosome 7 associated with components of winterhardiness. *Genome*, 36(1):66–71.

[21] P.M. Hayes and O. Jyambo (1993) Summary of QTL effects in the steptoe x morex population. *Barley genetics newsletter*, 23:98–143.

[22] A. Huang, S. Xu, and X. Cai (2013) Empirical Bayesian LASSO-logistic regression for multiple binary trait locus mapping. *BMC Genetics*, 14(5):1–14.

[23] O. Kohannim, D.P. Hibar, J.L. Stein, N. Jahanshad, Xue Hua, P. Rajagopalan, A.W. Toga, C.R. Jack Jr., M.W. Weiner, G.I. de Zubicaray, K.L. McMahon, N.K. Hansell, N.G. Martin, M.J. Wright, P.M. Thompson, and The Alzheimer's Disease Neuroimaging Initiative (2012) Discovery and replication of gene influences on brain structure using LASSO regression. *Frontiers in Neuroscience*, 6:1–13.

[24] R. Kohavi and G.H. John (1997) Wrappers for feature subset selection. *Artificial Intelligence*, 97(1-2):273–324.

[25] T.N. Lal, O. Chapelle, J. Weston, and A. Elisseeff (2006) Embedded methods. In *Feature extraction*, volume 207 of *Studies in Fuzziness and Soft Computing*, pages 137–165. Springer, Berlin Heidelberg.

[26] H.O. Lancaster (1958) The structure of bivariate distributions. *The Annals of Mathematical Statistics*, 29(3):719–736.

[27] E.S. Lander and D. Botstein (1989) Mapping mendelian factors underlying quantitative traits using RFLP linkage maps. *Genetics*, 121(1):185–199.

[28] I.-H. Lee, G.H. Lushington, and M. Visvanathan (2011) A filter-based feature selection approach for identifying potential biomarkers for lung cancer. *Journal of Clinical Bioinformatics*, 1(11):1–8.

[29] S.-H. Lee and C.-H. Jun (2011) Discriminant analysis of binary data following multivariate Bernoulli distribution. *Expert Systems with Applications*, 38(6):7795–7802.

[30] J. Li, B. Horstman, and Y. Chen (2011) Detecting epistatic effects in association studies at a genomic level based on an ensemble approach. *Bioinformatics*, 27(13):i222–i229.

[31] Z. Li and M.J. Sillanpaa (2012) Overview of LASSO-related penalized regression methods for quantitative trait mapping and genomic selection. *Theoretical and Applied Genetics*, 125(3):419–435.

[32] B.G. Lindsay, G.Y. Yi, and J. Sun (2011) Issues and strategies in the selection of composite likelihoods. *Statistica Sinica*, 21(1):71–105.

[33] J. Liu, J. Huang, S. Ma, and K. Wang (2013) Incorporating group correlations in genome-wide association studies using smoothed group LASSO. *Biostatistics*, 14(2):205–219.

[34] B.E. Madsen and S.R. Browning (2009) A groupwise association test for rare mutations using a weighted sum statistic. *PLoS genetics*, 5(2):e1000384.

[35] C.M. Mutshinda and M.J. Sillanpaa (2010) Extended bayesian LASSO for multiple quantitative trait loci mapping and unobserved phenotype prediction. *Genetics*, 186(3):1067–1075.

[36] J.O Ogutu, T. Schulz-Streeck, and H.-P. Piepho (2012) Genomic selection using regularized linear regression models: ridge regression, lasso, elastic net and their extensions. *BMC Proceedings*, 6(2):1–10.

[37] A. Pan, P.M. Hayes, F. Chen, T.H.H. Chen, T. Blake, S. Wright, I. Karsai, and Z. Bedo (1994) Genetic analysis of the components of winterhardiness in barley (Hordeum vulgare L.). *Theoretical and Applied Genetics*, 89(7-8):900–910.

[38] M.Y. Park and T. Hastie (2006) Regularization path algorithms for detecting gene interactions. Technical Report 2006-13, Department of Statistics, Stanford University.

[39] O.V. Sarmanov (1958) The maximum correlation coefficient (symmetric case). *Dokl. Akad. Nauk SSSR*, 120:715–718.

[40] S. Subedi, Z. Feng, R. Deardon, and F.S. Schenkel (2013) SNP selection for predicting a quantitative trait. *Journal of Applied Statistics*, 40(3):600–613.

[41] R. Tibshirani (1996) Regression shrinkage and selection via the Lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, 58(1):267–288.

[42] G. Tutz and J. Ulbricht (2009) Penalized regression with correlation-based penalty. *Statistics and Computing*, 19(3):239–253.

[43] M.G Usai, A. Carta, and S. Casu (2012) Alternative strategies for selecting subsets of predicting SNPs by LASSO-LARS procedure. *BMC Proceedings*, 6(2):1–9.

[44] X. Wan, C. Yang, Q. Yang, H. Xue, X. Fan, N.L.S. Tang, and W. Yu (2010) BOOST: A fast approach to detecting gene-gene interactions in genome-wide case-control studies. *The American Journal of Human Genetics*, 87(3):325–340.

[45] Y. Wang, G. Liu, M. Feng, and L. Wong (2011) An empirical comparison of several recent epistatic interaction detection methods. *Bioinformatics*, 27(21):2936–2943.

[46] H. Warren, J.-P. Casas, A. Hingorani, F. Dudbridge, and J. Whittaker (2014) Genetic prediction of quantitative lipid traits: Comparing shrinkage models to gene scores. *Genetic Epidemiology*, 38(1):72–83.

[47] N.R. Wray, Jian Yang, B.J. Hayes, A.L. Price, M.E. Goddard, and P.M. Visscher (2013) Pitfalls of predicting complex traits from SNPs. *Nature Reviews Genetics*, 14:507–515.

[48] J. Wu, B. Devlin, S. Ringquist, M. Trucco, and K. Roeder (2010) Screen and clean: a tool for identifying interactions in genome-wide association studies. *Genetic Epidemiol.*, 34(3):275–285.

[49] J. Xia, S. Visweswaran, and R.E. Neapolitan (2012) Mining epistatic interactions from high-dimensional data sets. In D.E. Holmes and L.C. Jain, editors, *Data Mining: Foundations and Intelligent Paradigms*, pages 187–209. Springer, Berlin Heidelberg.

[50] P. Yang, J. Ho, A. Zomaya, and B. Zhou (2010) A genetic ensemble approach for gene-gene interaction identification. *BMC Bioinformatics*, 11(1):524.

[51] C. Yao, D.M. Spurlock, L.E. Armentano, C.D. Page Jr., M.J. VandeHaar, D.M. Bickhart, and K.A. Weigel (2013) Random forests approach for identifying additive and epistatic single nucleotide polymorphisms associated with residual feed intake in dairy cattle. *Journal of Dairy Science*, 96(10):6716–6729.

[52] N. Yi, B.S. Yandell, G.A. Churchill, D.B. Allison, E.J. Eisen, and D. Pomp (2005) Bayesian model selection for genome-wide epistatic quantitative trait loci analysis. *Genetics*, 170(3):1333–1344.

[53] X. Zhang, S. Huang, Z. Zhang, and W. Wang (2012) Chapter 10: Mining genome-wide genetic markers. *PLoS Computational Biology*, 8(12):e1002828.

[54] Y. Zhang, B. Jiang, J. Zhu, and J.S. Liu (2011) Bayesian models for detecting epistatic interactions from genetic data. *Annals of Human Genetics*, 75(1):183–193.

[55] H. Zhou, D.H. Alexander, M.E. Sehl, J.S. Sinsheimer, and K. Lange (2011) Penalized regression for genome-wide association screening of sequence data. In *Pacific Symposium on Biocomputing*, pages 106–117. World Scientific Publishing.

[56] H. Zou (2006) The adaptive Lasso and its oracle properties. *Journal of the American Statistical Association*, 101(476):1418–1429.

[57] H. Zou and T. Hastie (2005) Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 67(2):301–320.

PSO with Crossover Operator Applied to Feature Selection Problem in Classification

Houassi Hichem, Mehdaoui Rafik and Maarouk Toufik Mesaaoud
Univ Khenchela, Fac. ST, ICOSI Lab., BP 1252 El Houria, 40004 Khenchela, Algeria
E-mail: Houassi_h@yahoo.fr, mehdaoui.rafik@yahoo.fr, toumaarouk@yahoo.fr

Keywords: Feature selection, particle swarm optimization (PSO), relevant feature vector (RFV), genetic algorithms (GA), crossover operator (CO).

Received: September 23, 2016

In recent years, there are a large number of features in datasets used in classification, which include relevant, irrelevant, and redundant features. However, irrelevant and redundant features decrease the computational time and reduce the classification performance. Feature selection is a preprocessing technique that which to choose a sub-set of relevant features to achieve a similar or even better classification performance than using all features. This paper presents two new hybrid algorithms for a feature selection called particle swarm optimization with crossover operator (denoted as PSOCO1 and PSOCO2); the algorithms are based on the integration of a particle swarm optimization (PSO) and a crossover operator (CO) of the genetic algorithms. A new relevant features vector (RFV) is introduced and used by our algorithms for execute a crossover operator between the RFV and other features vectors. To demonstrate the effectiveness of these algorithms, we compared them with standard PSO [14], PSO4-2 [8] and HGAPSO [28] on twelve benchmark datasets. The results show that the two proposed algorithms significantly reduce the number of selected features and achieve similar or even better classification accuracy in almost all cases.

Povzetek: Predstavljena sta dva nova algoritma za izbiro dobrih atributov v strojnem učenju.

1 Introduction

Since the technological advancement, data acquisition becomes easy and gigantic databases are collected daily. Therefore, the number of features in the current data analysis problems can now reach hundreds of thousands or more. In this situation, the feature selection is an important step in building a model of classification in large datasets because it reduces the number of features by removing irrelevant, noisy and redundant features. Feature selection has now been widely applied in many domains, such as bioinformatics [1], astronomy [2] and energy consumption [3]. Feature selection is used to search a subset of relevant features which can reduce the search space, decrease the classification performance and increase the computational cost of classification algorithms. Feature selection is a difficult task because of the exponentially increasing of the search space size (the number of available features in the data sets) [4]. Therefore, an exhaustive search is almost impossible [5]. Greedy algorithms have been used to solve exhaustively the feature selection problem, such as sequential forward selection (SFS) [6] and sequential backward selection (SBS) [7]. However, these approaches suffer from the high computational cost and stagnation in local optima [8]. Therefore, an efficient and global search feature selection algorithm is needed.

Different evolutionary computation algorithms (ECA) are recently proposed for solving the feature selection problem such as ant colony optimization (ACO) [9-11], genetic algorithm (GA) [12,13], particle swarm optimization (PSO) [14-16] and simulated annealing (SA) [17]. The particle swarm optimization is one of a relatively recent metaheuristics based population. It has proven its simplicity, efficiency and fast convergence [18, 19], but suffer from the premature convergence of a swarm and large number of parameters. Therefore, it is needed to develop a feature selection approach using PSO to increase simultaneously the feature selected number, decrease the classification accuracy and also to increase the algorithm's parameters number.

1.1 Goals

This paper aims to propose new improved PSO algorithms which can significantly decrease the size of selected feature subsets and improve the classification accuracy or at least keep it as is in original PSO. For this, we will propose two new improved PSO-based feature selection algorithms (denoted as PSOCO1 and PSOCO2) which uses two different strategies for particle's position updates. In order to achieve this goal, we propose two mechanisms for updating particle's position using a new introduced vector called "relevant features vector (RFV)", The RFV is a binary vector that contains 0 and 1, in which, 1 means that the attribute is selected and 0 is the opposite. The RFV is introduced as a perfect position in the search space since it contains all the features where their deletion decreases the classification accuracy. This vector is used for a crossing with the particle's positions which increases the probability of selecting the relevant features and removing the irrelevant features at each particle's update positions in the algorithm.

Our algorithms are developed and compared with other PSO-based feature selection mechanisms on many datasets with different numbers of features and instances.

1.2 Organization

The rest of this paper is organized as follows: Section 2 presents a brief background on standard particle swarm optimization (PSO), and reviews recent studies about PSO-based feature selection approaches. Section 3 describes in detail the two proposed modified PSO based feature selection approaches with new positions updating strategies. Section 4 describes experimental design and results on 12 benchmark data sets with discussions. Finally, section 5 provides conclusions.

2 Background and related work

This section provides a background about standard PSO [14] and related work on proposed feature selection using modified PSO algorithms in the literature.

2.1 Particle Swarm Optimization for feature selection

Particle swarm optimization (PSO) [15, 16] is a population-based optimization technique, was introduced by Eberhart and Kennedy [14] that which inspired from the nature social behavior, dynamic movements and communications of animals such as birds and fish in the search for food. In PSO, each particle in the swarm can represent a candidate solution of the problem. Each particle has a position, velocity and moving in the search space for searching the optimal solution, the positions and velocities of particles are represented respectively by a vectors $x_i = (x_{i1},x_{i2},...,x_{id})$ and $v_i = (v_{i1},v_{i2},...,v_{id})$, i = 1 to N, where d is the dimensionality of the search space and N is the swarm size. During searching, each particle of a swarm updates its position based on its best position (the personal best pbest) and the best position obtained by its neighborhood (the global best gbest). In the standard version of PSO [14] the swarm is initialized with a population of random positions and searches for the best solution in several iterations. In each iteration, the particle's velocity and position are updates according to the following equations:

$$x_{id}^{t+1} = x_{id}^{t} + v_{id}^{t+1} \quad (1)$$

$$v_{id}^{t+1} = w \times v_{id}^{t} + c_1 \times r1_i \times (p_{id} - x_{id}^{t}) + c_2 \times r2_i \times (p_{gd} - x_{id}^{t}) \quad (2)$$

Where t represents the t^th iteration in the evolutionary algorithm, d represents the d^th dimension in the search space, w is the inertia weight, c_1 and c_2 are two positive constants called cognitive and social parameters respectively. $r1_i$ and $r2_i$ are independent random values uniformly generated from (0, 1), p_{id} is the pbest of i^th particle and p_{gd} is the gbest of the swarm.

PSO was originally proposed for solving continuous problems. However, many problems, such as feature selection are discrete or binary problems. The binary PSO (BPSO) is applied for discrete problems. In BPSO, the position of each particle is represented in binary values which are 0 or 1. The particle's velocity in BPSO indicates that a particle might change its state to 1. A sigmoid function $s(v_{id})$ is introduced to transform v_{id} to the range of (0,1). BPSO updates the position of each particle according to the following formula:

$$x_{id} = \begin{cases} 1, if\ rand(\ \ ) < s(v_{id}) \\ 0, ot\square erwise \end{cases} \quad (3)$$

Where

$$s(v_{id}) = \frac{1}{1+e^{-v_{id}}} \quad (4)$$

rand () is a randomly generated number from (0, 1).

2.2 Recent studies about PSO for feature selection

More recently modified particle swarm optimization algorithms are proposed to feature selection problem, we classify these algorithms based on modifications of the standard PSO as follow:

(1) Modified initialization mechanisms

Xue et al. [8] developed three initialization strategies and three pbest and gbest updating mechanisms in PSO. Their experiments confirm that these new mechanisms increase the classification accuracy and reduce both the number of features and the computational time.

(2) Modified Gbest and Pbest updating mechanisms

In standard PSO, gbest is updated only when a better solution is found, but this mechanism is easy falling into local optimal solutions. Yang et al. [20] develop a new Gbest updating strategy in PSO for feature selection which that aims to avoid the particles converging at local optima. In this proposed strategy, when Gbest is unchanged after three iterations, a new Gbest will be created from pbests of relevant particles. Experiments show that the new algorithm outperforms both standard BPSO and GA-based feature selection in terms of classification accuracy. Chuang et al. [21] also proposed a Gbest updating mechanism, where the Gbest will be reset to zero vector if it unchanged after several iterations. This algorithm is compared in terms of the classification performance with the proposition of Yang et al. [20] using the cancer related human gene expression datasets. Experimental results show that this method could achieve higher classification accuracy in most cases.

(3) Multi Swarm

Liu et al. [22] and Fdhila et al. [23] introduce multi-swarm PSO (MSPSO) algorithms. In the proposition of [22] the algorithm is used for both the feature selection problem and the SVM parameters optimization. The experimental results show that their proposed method is faster and achieves better performance than grid search, standard BPSO and GA. However, the multi-swarm PSO algorithms are more expensive than the other methods in terms of both memory and computational time because of the large population size.

(4) Other modifications

Beside the above strategies, some researchers have considered other types of modifications to increase the PSO performance. Chuang et al. [19] propose a new algorithm which is called catfish binary particle swarm optimization (catfishBPSO), in which the new particles so-called catfish that are introduced for helping PSO to avoid premature convergence. The catfish particles replace particles with worst fitness in the swarm when gbest has not improved for a number of iterations. Experimental results show that catfishBPSO achieves better classification accuracy than genetic algorithms, linear forward selection (LFS) and greedy stepwise backward selection (GSBS). Wang et al. [24] proposed a triggered memory scheme for the memory-based PSO in dynamic environments which aims to use a memory scheme to restore useful information about previously found positions by the swarm. Experiments show that the proposed scheme can achieve better performance than standard PSO. Unler & Murat [15] developed a modified discrete PSO which use an adaptive feature selection procedure, where the features are selected according to their contribution to the subset of features already selected and also to the likelihood calculated by BPSO. This approach is faster and achieves better classification performance compared with the tabu search and scatter search algorithms using several datasets. Quantum-inspired PSO (QiPSO) [25] is a newly developed probabilistic evolutionary algorithm based on the classical particle swarm optimization algorithm and some concepts and theories of quantum computing, in order to improve the search capability and avoid premature convergence. Hamed et al. [26] propose a dynamic quantum-inspired PSO algorithm (DQiPSO) for feature selection and parameter optimization in neural networks for classification. In DQiPSO the particle is divided into two parts: In the first part, it uses the quantum information embedded in PSO for feature probability calculation, and in the second part, it uses the standard PSO with a real value to optimize the parameters in the neural network. The DQiPSO is compared with two methods: quantum-inspired PSO (QiPSO) and standard PSO algorithm (PSO). Experiments indicate that the DQiPSO is faster and achieves better classification performance than QiPSO and PSO in all test cases.

2.3 Hybridized PSO based feature selection approaches

Several evolutionary algorithms are developed for the feature selection problem such as PSO and genetic algorithm (GA). However, both PSO and GA suffer from shortcomings: The main shortcoming of PSO is the premature convergence of a swarm, and generally, GAs find the global optimum solution but suffer from a slow convergence rate. Mohemmed et al. [27] propose a hybrid algorithm (PSOAdaBoost) that incorporates PSO with an AdaBoost framework for face detection. The PSOAdaBoost is compared with AdaBoost (with exhaustive feature selection) and experiments indicate that a PSOAdaBoost algorithm has a better performance in terms of much less training time and better classification accuracy. In [28] a new hybrid approach is proposed, which is based on the integration of the GA and PSO, this hybridization is obtained through integrating the standard velocity and update rules of PSO with selection, crossover, and mutation from the GA. In this algorithm, each next generation of population; the new population is produced through enhancement, crossover, and mutation on the half top of the best performing particles generated in the current generation.

3 Proposed approaches

Given the simplicity andthe effectiveness ofPSO, we propose new approaches inspired fromthe principle of the standard PSO and we use the crossover operator in the genetic algorithmin order to well exploit the search space. Ourgoal is to propose two new modified PSO algorithms with a minimal number of parameters based onparticle swarm and crossover operator. The main idea of our algorithms is to integrate the crossover operator of GA into the PSO algorithm. The difference between our proposed approach and PSO is that the crossover operator is used to improve the standard PSO generating new solution for each particle using random walk. In this way, our algorithm can explore the search space by the crossover of the GA algorithm and exploit the population information with PSO. Our algorithms named PSOCO1 and PSOCO2 (Particle Swarm Optimization with Crossover Operator). These algorithms are similar, but they use two different updating mechanisms. Meanwhile, we use a new vector called relevant features vector (RFV) that contains relevant features in the dataset, the algorithm1 shows how to determine the RFV.

3.1 Relevant features vector (RFV)

Definition of relevant feature

The relevant feature is afeature whereitsdeletion from the feature set decreases theclassification rate ofthe data set instances.In other words, a relevantfeature"A"is a featurewherethe classificationrate ofdata setinstances usingall featuresisbetter thanthe classificationrate of the same data set instances usingall featuresexceptthe attribute "A".

Definition of relevant feature vector

The relevant features vector (RFV) is a vector contains all relevant features in the data set. The following algorithm describes the relevant features vector determination.

Algorithm 1 Pseudo-code of the relevant features vector determination

```
Input:
Vector of all features set A = ($a_1$, $a_2$, …, $a_d$)
Output:
Relevant features vector RFV = ($RFV_1$, $RFV_2$, …, $RVF_k$)
begin
        Evaluate the fitness “Fitness_A” using all features in
        A on the Data set;
  for (j=0 to feature number -1)do
```

```
        V = A- {$a_j$} /* $a_j$ is the feature number j */   Evaluate the
  fitness “Fitness_V” using all features
            In V on the Data set;
  if (Fitness_V < Fitness_A)
          RFV = RFV + { aj };
  endif
  end for
end.
```

3.2 New position’s update mechanisms

In this section, we will propose two new different particle’s position update mechanisms in PSO for feature selection with the goals of increasing the PSO parameters number, increasing the number of selected features and also the computational time of the feature selection algorithm. The new mechanisms are motivated by using of the crossover operator in GA. In our proposition, the inertia weight factor and two acceleration coefficients are removed and only length of crossover part is needed when modifying the particle’s position.

Position’s update mechanism 1

In each iteration, particle’s position is updated using the crossover operator in GA into three following vectors: RFV, Pbest and Gbest. First, the crossover operation is performed on particle’s current position withits Pbest. Second, crossoverof the particle’s current positionwith the Gbest. Third, crossoverof particle’s current positionwith the RFV. After creating new solutions (position vectors) by crossover, the algorithm proceeds to evaluate the new vectors (i.e. calculate the fitness of each new vector generated by crossover) to be able to update the values of Pbest by the vector with best classification accuracy. Algorithm 2 shows this first proposed update mechanism.

The particle’s position updated according to the following equations:

$X_i^k = MaxAcc\ ((X_i^k \copyright Pbest_i), (X_i^k \copyright Gbest), (X_i^k \copyright RFV))$ (5)

Where X_i^k is the position of a particle i at iteration k, Pbesti is the best position of the particle i up to the iteration k, Gbest is the best position of the swarm population up to the iteration k, RFV is the relevant features vector and © is the crossover operator. “MaxAcc” is a function which returns a vector with better classification accuracy.

Position’s update mechanism 2

In the second proposed update mechanism, The Pbest is updated by the new vector which generated by using the crossover of the current particle’s position and the RFV. Algorithm 3 shows this second proposed update mechanism.

The particle’s position updated according to the following equations:

$X_i^k = X_i^k \copyright RFV$ (6)

Where X_i^k is the position of particle i at iteration k, RFV is the relevant features vector and © is the crossover operator.

3.3 Crossover operator

This operator is inspired from the genetic algorithms that combines two vectors (parent vectors) to produce a new vectors (new shield vectors) may be better than both of the parent’s solutions if it takes the best characteristics from each of the parents. However, the original crossover operator does not always give good solutions because it randomly selects the parents, so we have replaced it by one specific crossover where the parents are not randomly selected: the first parent is always the current position vector and the second parent is the RFV vector in our first proposition, the RFV, the Pbest or the Gbest vectors are used as a second vector in our second proposition. The new crossover operator is inspired from the two points crossover.

Two Points Crossover

When performing crossover, two crossover points P1 and P2 are chosen in both parental vectors and the contents between these points are exchanged. In our proposition, the first point P1 is randomly chosen in the range [0, (d-µ)] as presents the equation (7), and the second point P2 is calculated using equation (8). Once the crossover is performed, new shields (new solutions) are created.

$P_1 = rand(0, (d- \mu))$ (7)

$P_2 = P_1 + \mu$ (8)

Where d is a dimension of a problem (a number of features in the dataset) and µ presents the length of crossover part (means the bits number of crossover operator).

The following algorithms describe the updates mechanisms used in the proposed feature selection algorithms present in the sub-section 3.4.

Algorithm 2 Pseudo-code of the updates mechanism 1

```
Inputs:
$X_i$= ($x_1$, $x_2$, …, $x_d$)   /* A current position vector of
a particle i*/
RFV = ($r_1$, $r_2$, …, $r_d$)   /* A relevant features vector */
Pbest = ($p_1$, $p_2$, …, $p_d$) /* A Pbest vector of a current
particle*/
Gbest = ($g_1$, $g_2$, …, $g_d$) /* A Gbest vector of a swarm*/
    Length of crossover part µ
Output:
New position vector $X_i$ = ($x_1$, $x_2$, …, $x_d$)
begin
Generate integer random number $P_1$ using the equation (7)
Generate integer number $P_2$ using the equation (8)
Vector VR= $X_i$; VP = $X_i$; VG = $X_i$;
for  j = $P_1$ to $P_2$ do
        VR[j]  = RFV[j]
        VP[j]  = Pbest[j]
        VG[j] = Gbest[j]
end for
Calculate fitness value of VR, VP and VG
Update the vector $X_i$ using equation (5).
end.
```

Algorithm 3 Pseudo-code of the updates mechanism 2

```
Inputs:
$X_i$= ($x_1$, $x_2$, …, $x_d$)   /* A current position vector of
            particle i */
RFV = ($r_1$, $r_2$, …, $r_d$)  /* A relevant features vector */
Length of crossover part “µ”
```

```
Output:
New position vector $X_i$= ($x_1$, $x_2$, …, $x_d$)
begin
Generate integer random number $P_1$ using the equation (7)
Generate integer number $P_2$ using the equation (8)
for j = $P_1$ To $P_2$ do
        $X_i$[j] = RFV[j]
end for
end.
```

3.4 PSOCO algorithm

Like any algorithm, the first step in our algorithm is to set some parameters required for its successful implementation. The advantages of our algorithm are its simplicity, a few requirement parameters to adjust and may guide to search for a small size feature subset with low classification performance. The main idea of this algorithm is that runs in two stages. In the first stage, the algorithm focuses on the determination of a relevant features vector (RFV) (new in our algorithms). In the second stage, the PSOCO algorithm starts searching for the best solution in the search space using a population of particles called swarm. The algorithm begins by initializing the swarm size and the positions of the population. Each particle's position is randomly initialized in the search space with a uniform distribution. Then it initializes the best find position by each particle (denoted by $Pbest_{i1}$, $Pbest_{i2}$,...,$Pbest_{id}$).Next, based on the fitness function, the algorithm calculates the quality of each particle to be able to take the best position find by the swarm(denotedbyGbest$_1$, Gbest$_2$,...,Gbest$_d$). Next, the algorithm begins the iterations for generating new solutions in order to improve the quality of the best solution found by the swarm. In each iteration, a new position is calculated for each particle as presented in algorithm 2 or algorithm 3.

Pseudo-code of PSOCO algorithm

We used two new particle's position update mechanisms presented in section 3.2 proposed new PSO-based feature selection algorithms called PSOCO1 and PSOCO2 by using respectively update mechanism1 and update mechanism2. The pseudo-code of PSOCO2 is similar to PSOCO1, with a small change in particle's position updates; in PSOCO1the particle's position is updated based on the algorithm2, but PSOCO2 uses the algorithm3 for updating the position of each particle. The pseudo-code of PSOCO1 is shown in Algorithm 4

Algorithm 4 Pseudo-code of the proposed algorithm (PSOCO1)

```
Inputs:
    The data set, population size, number of features, number
    of iterations, length of crossover part µ
Output:
    The selected feature subset (Subset of features that gives
    the maximum accuracy over the data set) and his
    classification accuracy.
begin
    Generate the relevant feature vector RFV using the
    algorithm 1;
Initialize all particles of a swarm randomly;
While Maximum iterations is not met do
        Evaluate the fitness of each particle on the Data set;
        /* the classification accuracy on the training set */
        for i=1 to population size do
            Update the Pbest of particle i;
            Update the Gbest;
            Update the position of particle i using algorithm 2
        end for
end while
Calculate the classification accuracy of the selected feature
        subset on the Data set;
end.
```

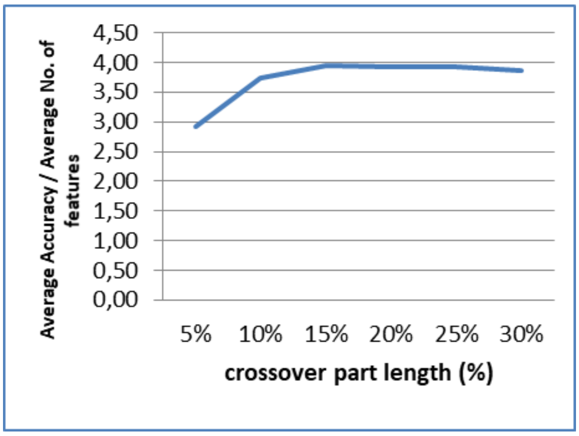


Figure 1: Effect of the crossover part length (parameter µ) on the ratio (Average N.F.S / Average Acc).

Dataset	# features	# classes	# instances
breast-cancer	10	2	699
bridges_v1	13	4	108
Zoo	17	7	101
Lymphographie	18	4	148
Flags	30	8	194
dermatology	33	6	366
Soybean	35	4	683
Audiology	69	24	226
LIBRAS	91	15	360
MUSK1	168	2	476
Arrhythmia	279	16	452
Isolet5	617	2	1559

Table 1: Datasets.

4 Design of experiments

4.1 Datasets and parameter settings

In our experiments, the twelve benchmark datasets (Table 1) are chosen from the UCI machine learning repositories [29], which have different numbers of features (from 10 to 617), classes and instances as the representative samples of the feature selection problem. For each dataset, all instances are used as training set and also as test set.

K-nearest neighbour (KNN) learning algorithm was used in the experiments, we use K=5 (5NN) [8]. Waikato environment of knowledge analysis Weka [30] is used to run the instances classification. Classification accuracy is evaluated by 5NN implemented in Java.

In order to examine the performance of the proposed algorithms PSOCO1 and PSOCO2 based on algorithm 2 and algorithm 3 respectively as the position's update mechanisms, the standard PSO [14], and two recent wrapper feature selection algorithms PSO4-2 [8] and HGAPSO [28], are used as benchmark techniques in the experiments. The same conditions were used in all experiments: A swarm size of 20 [19], an iteration number of 20 and all compared algorithms perform 20 independent runs on each dataset.

4.1.1 Determination of the crossover part length (µ)

The aim of this sub-section is to investigate the effect of the key parameter µ (length of crossover part) on the performance of PSOCO. It is difficult to determine what length (number of bits) should be selected to run the crossover operator. Selecting a large number of bits may complicate the performance in terms of PSOCO2's time running while selecting a small number of bits may deteriorate the classification performance. For that, various experiments were carried. We take PSOCO2 as selection feature algorithm and the 5-nearest neighbor (5NN) as learning algorithm to analyze the effects of µ on the performance of the PSOCO2. The PSOCO2 is analyzed on 20 independent runs. Table. 2 show the experimental results with varying values of the parameter µ. In table 2, "A.N.F.S" means the average number of features selected in the 20 independent runs of PSOCO2. "Acc" represents the classification accuracy using the selected features. The percentage in the first line in table 2: for example 10% means that the length of crossover part equals to 10% bits from the total number of features. Fig. 1 illustrates a graphical representation of the ratio between the average number of selected features and the average classification accuracies that is found by the PSOCO2 for all datasets (average of all Acc in Table.2).

In the plot of Fig. 1, the horizontal axis shows percentage of selected features for applying crossover operator

(Parameter µ), and the vertical axis shows the ratio between the average of A.N.F.S and the average of accuracies (last line in the Table. 2). From Fig. 1, it can be seen that this parameter has a significant effect on the number of selected feature and the classification accuracy using the selected features, it is clear that the value 15 of µ produces a better result. It can be also seen that the performance of PSOCO2 is stable once the value of µ exceeds 15.

Crossover part length (%)	**5%**		**10%**		**15%**		**20%**		**25%**		**30%**	
Data sets	**A.N.F.S**	**Acc**	**A.N.F.S**	**Acc**	**A.N.F.S**	**Acc**	**A.N.F.S**	**Acc**	**A.N.F.S**	**Acc**	**A.N.F.S**	**Acc**
iris	2,7	97,03	2,9	97,06	2,75	97,16	2,65	97,09	2,85	97,23	2,85	97,19
diabetes	5,75	83,52	5,75	83,52	5,7	83,53	5,4	83,45	5,65	83,5	5,75	83,52
labor	8,15	96,22	8,6	97,71	9,2	97,19	9,2	97,19	9,2	97,45	9,05	97,54
lymphographie	10,4	87,12	10,5	89,08	10,15	89,22	10,5	89,05	9,85	89,66	10,85	89,02
ionosphere	12,8	92,49	9,7	92,56	9,65	92,4	9,45	92,45	9,35	92,3	9,2	92,16
soybean	20,67	92,38	19,3	93,08	18	92,81	17,85	92,64	17,8	92,29	17,75	92,34
audiology	23,25	74,6	18,1	75,06	16,75	75,11	16,75	74,8	16,65	75,15	16,05	75,15
LIBRAS	28,6	88,52	20,7	88,62	19,8	88,66	19,4	88,3	18,9	88,58	18,8	88,08
MUSK1	40,6	99,05	29,8	99,39	23,7	99,41	22,9	99,3	22,8	99,3	23,7	99,39
Isolet5	158,5	99,15	119,78	99,09	115,9	98,91	118,4	98,99	119,9	98,99	122,9	98,95
Average	31,14	91,01	24,51	91,52	23,16	91,44	23,25	91,33	23,29	91,45	23,69	91,33

Table 2: Effect of the crossover part length on the performance of PSOCO2 using different dataset

Datasets	T. N. F	C. A. A. F	N. R. F	C. A. R. F	Algorithms	A. N. F	A. A	B. A
Breast-cancer	9	77,27	4	75,52	Standard PSO	7,6	79,86	80,06
					PSO4_2	7,4	79,86	80,06
					HGAPSO	6,9	79,56	80,06
					PSOCO1	4,3	78,56	79,72
					PSOCO2	6,55	79,52	80,06
Bridges_v1	12	71,02	4	75,7	Standard PSO	5,3	78,13	78,5
					PSO4_2	5,3	78,31	78,5
					HGAPSO	5,55	77,75	78,5
					PSOCO1	4,45	78,31	78,5
					PSOCO2	5,5	78,31	78,5
Zoo	17	95,04	3	65,34	Standard PSO	9,6	98,86	99
					PSO4_2	9,9	98,76	99
					HGAPSO	9,45	98,51	99
					PSOCO1	3,65	95,79	99
					PSOCO2	9,6	98,56	99
Lymphographie	18	87,16	9	85,13	Standard PSO	12,3	91,72	93,24
					PSO4_2	12,45	91,72	93,24
					HGAPSO	11,6	91,48	93,24
					PSOCO1	9,6	89,69	92,56
					PSOCO2	12,85	92,09	93,24
Flags	29	64.94	8	62.37	Standard PSO	14,85	76,46	78,35
					PSO4_2	14,5	76,64	78,35
					HGAPSO	14,4	75,79	77,83
					PSOCO1	9,15	73,96	75,77
					PSOCO2	14,1	77,5	80,41
Dermatology	34	98,36	16	95.62	Standard PSO	21,35	99,04	99,45
					PSO4_2	20,3	99,08	99,45
					HGAPSO	19,3	98,85	99,45
					PSOCO1	16,5	98,56	99,18
					PSOCO2	18,2	99,22	99,45
Soybean	35	92,82	18	91,5	Standard PSO	22,75	93,55	94,14
					PSO4_2	23,85	93,87	94,72
					HGAPSO	21,95	93,63	94,72
					PSOCO1	18,2	92,55	93,7
					PSOCO2	22,55	94,37	95,02
Audiology	69	72,12	15	71,68	Standard PSO	38,1	78,23	79,64
					PSO4_2	37,25	77,83	80,08
					HGAPSO	36,2	77,34	79,64
					PSOCO1	16,5	74,91	76,54
					PSOCO2	29,9	78,4	80,97
LIBRAS	90	86,38	17	79,72	Standard PSO	43,15	89,01	89,72
					PSO4_2	43,85	89,08	90
					HGAPSO	42,05	89,01	90,27
					PSOCO1	20,7	88,63	89,72
					PSOCO2	32,25	90,13	90,83
MUSK1	167	96,21	19	97,89	Standard PSO	80,55	99,25	99,57
					PSO4_2	62,05	99,25	99,78
					HGAPSO	78,5	99,1	99,78
					PSOCO1	23,25	99,38	99,57
					PSOCO2	49,65	99,63	100
Arrhythmia	279	91,63	41,05	89,59	Standard PSO	132,64	94,99	69,89
					PSO4_2	59,64	94,87	96,88
					HGAPSO	56,87	94,78	96,75
					PSOCO1	32,87	94,23	96,42
					PSOCO2	38,85	93,89	96,05
Isolet5	617	97,44	85,84	84,57	Standard PSO	301,25	96,38	98,32
					PSO4_2	254,35	98,65	98,98
					HGAPSO	248,63	98,77	98,98
					PSOCO1	92,59	97,98	98,54
					PSOCO2	112,87	98,89	98,64

Table 3: Experimental Results.

4.2 Experimental Results and Discussions

This section presents the results obtained from series of experiments conducted to evaluate the effectiveness of our algorithms. We experiment independently the PSOCO1 and PSOCO2 on twelve datasets (Table.1) and we compare its performance with the standard PSO [14], PSO4-2 [8] and HGAPSO [28]. Table 3 show the experimental results, we compare the classification performance of the two proposed algorithms with three other recent approaches in the literature on the twelve datasets. In Table 3, "T.N.F" means the total number of features in the dataset. "C.A.A.F" represents the classification accuracy using all features. "N.R.F" is the number of relevant features selected by algorithm 1. "C.A.R.F" shows the accuracy of classification using relevant features." A.N.F" represents the average number of selected features by each algorithm in 20 independent runs. "B.A" and "A.A" indicate respectively the best and the average classification accuracy obtained from the 20 runs.

4.2.1 Results of benchmark techniques

According to the previous results in Table 3, it can be seen that in all datasets, PSO4-2 [8] and HGAPSO [28] select a small number of features and achieve similar or higher classification accuracy than using all features in all cases. When comparing HGAPSO with standard PSO-based feature selection approach, it is clear in Table 3 that the HGAPSO approach reduces the number of features in 12 bases and decreases the average classification accuracy in 3 bases from the total of 12 cases. HGAPSO outperformed PSO4-2 in terms of selected feature number in 11 cases from a total of 12 cases. However, PSO4-2 finds better classification accuracy than HGAPSO in 9 cases, which means the reduction of the number of features, might decrease the classification performance.

This section presents the results obtained from series of experiments conducted to evaluate the effectiveness of our algorithms. We experiment independently the PSOCO1 and PSOCO2 on twelve datasets (Table.1) and we compare its performance with the standard PSO [14], PSO4-2 [8] and HGAPSO [28]. Table. 3 show the experimental results, we compare the classification performance of the two proposed algorithms with three other recent approaches in the literature on the twelve datasets. In Table. 3, "T.N.F" means the total number of features in the dataset. "C.A.A.F" represents the classification accuracy using all features. "N.R.F" is the number of relevant features selected by algorithm 1. "C.A.R.F" shows the accuracy of classification using relevant features." A.N.F" represents the average number of selected features by each algorithm in 20 independent runs. "B.A" and "A.A" indicate respectively the best and the average classification accuracy obtained from the 20 runs.

4.2.2 Results of PSOCO1

According to Table 3, on 9 of 12 cases, PSOCO1 used crossover between the current position and the Pbest, Gbest or RFV, achieved better classification performance than that of benchmark algorithms, and in all datasets tested, the number of selected features by PSOCO1 is significantly smaller than that of benchmark algorithms.

4.2.3 Results of PSOCO2

According to the results shown in Table 3, in most cases (11 of the 12 data sets), the selected feature subsets by our algorithm PSOCO2 contain about half of the available features. Using the selected feature, the ANN classifier can achieve about similar classification accuracy than using all features in almost all datasets. Moreover, in the cases of datasets with a large number of features (the three last datasets), the feature subsets evolved by PSOCO2 contains about seven-tenth of the available features. Comparing the results obtained by PSOCO2 with that of other benchmark algorithms, the average size of the feature subsets evolved by PSOCO2 is always smaller, and the reduction of the average size is more than 20% in all datasets (except the bridges_v1 dataset) and it is 70% in three last bases (datasets with a large number of features). The average classification accuracy achieved by the feature subsets resulted by PSOCO2 is better in almost all datasets.

4.2.4 Comparison between proposed methods and benchmark techniques

Comparing the two proposed methods with benchmark techniques, leads to the following observations, the number of features selected by our algorithms (PSOCO1 and PSOCO2) was similar or slightly inferior compared with the benchmark algorithms on the datasets with a relatively small number of features, but it was significantly inferior in the datasets with a large number of features. However, the classification accuracy achieved by our proposed algorithms was similar or slightly better than that achieved by the benchmark algorithms in almost all cases. Seen that the difference between our approaches and the benchmark techniques is the RFV, and then this demonstrates that RFV leads the swarm to select only the relevant features, which significantly decrease the selected feature subsets size found by our algorithms. In the updates equations of the original PSO equations (1) and (2), there are five parameters (w, c1, c2, r1, r2,), but in our algorithms there is only one parameter (Length of crossover part). Then, the updates mechanisms in our approaches motivated by both a new RFV vector and a small number of parameters can help PSOCO to take their advantages to obtain feature subsets with a smaller number of features and better or similar classification accuracy.

4.2.5 Discussions

The PSO algorithm is a simple and fast algorithm; however, it may lack the diversity of population between particles. Therefore, in this work, we add crossover operator between particles to the PSO during the process of new positions updating for efficiency exploring the search space. From the experimental results we can sum up the following:

- Our proposed approaches PSOCO1 and PSOCO2 can solve the feature selection problem effectively.
- The overall performance (the average classification accuracy) of our proposed approaches is equal or better than the original PSO and other compared optimization methods.
- The average size of the feature subsets selected by PSOCO2 is always smaller, and the reduction of the average size is more than 20% in all tested datasets and it is 70% in datasets with a large number of features.
- The algorithms PSOCO1 and PSOCO2 used few parameters compared with the original PSO.

The results suggest that the RFV vector guide the proposed PSOCO to search the solution around them as a perfect solution, which increases the probability of selecting the relevant features and removing the irrelevant features.

5 Conclusion and future works

The goal of this paper is to develop two new feature selection approaches based on PSO using a crossover operator from genetic algorithm and a proposed relevant features vector (RFV) to selecting a smaller number of features and achieving same or better classification accuracy. In the first algorithm, particle's position is updated using the crossover operator between current position on one hand and RFV, Pbest or Gbest on the other hand, but the second algorithm use only crossover operator between the current position and the RFV for updating the particle's position. The advantage of our approach is that it is simple and does not require too many parameters to initialize compared with the original PSO; the population size and the length of crossover part are the only essential parameters to initialize. In this study, we have conducted the experiments to compare the new algorithms with three other feature selection algorithms on 12 datasets of varying numbers of features and instances. The goal was successfully achieved by our new approaches: the size of selected features subset is reduced over the standard PSO scheme about 20% in the small datasets and 70% in the large datasets using PSOCO2. These results suggest that the proposed algorithms have great potential to reduce the dimensionality of the search space in order to maintain the classification accuracy.

For future works, Firstly, PSOCO will be applied to many other engineering optimization problems like 0–1 knapsack problem, job scheduling and transport engineering. Secondly, our two PSOCO approaches will be used with more classifiers like SVM and Artificial Neural Network (ANN) to verify and optimize their parameters. Finally, The PSOCO will be enhanced by changing the random particle's initial positions by employing a various Chaotic Maps techniques to diversify the initial population on the search space.

References

[1] Enny I S, Sri H, Agus H, Retantyo W, Mudjosemedi M (2015). Feature Selection of the Combination of Porous Trabecular with Anthropometric Features for Osteoporosis Screening. International Journal of Electrical and Computer Engineering (IJECE), vol. 5, no. 1, pp. 78–83.

[2] Zheng H, Zhang Y (2008). Feature selection for high dimensional data in astronomy. Advances of Space Research, vol. 41, pp.1960–1964.

[3] Wang Q, Liu F, Wang X (2014). Multi-objective optimization of machining parameters considering energy consumption. International Journal of Advanced Manufacturing Technology, vol. 71, pp. 1133–1142.

[4] Hlaing T (2012). Feature Selection and Fuzzy Decision Tree for Network Intrusion Detection. International Journal of Informatics and Communication Technology, vol. 1, no. 2, pp. 109–118.

[5] Kohavi R, John GH (1997). Wrappers for feature subset selection. Artificial Intelligence, vol. 97, no. 1--2, pp. 273–324.

[6] Colak S, Isik C (2003). Feature subset selection for blood pressure classification using orthogonal forward selection. In: Proceedings of IEEE 29th Annual Bioengineering Conference, pp. 122–123.

[7] Cotter SF, Kreutz-Delgado K., Rao BD (2001). Backward sequential elimination for sparse vector selection. Signal Processing, vol. 81, pp.1849–1864.

[8] Xue B, Zhang M, Browne WN (2014). Particle swarm optimization for feature selection in classification: Novel initialization and updating mechanisms. Applied soft computing, vol. 18, pp. 261–276.

[9] Kanan HR, Faez K (2008). An improved feature selection method based on ant colony optimization (ACO) evaluated on face recognition system. Applied Mathematics and Computation, vol. 205, pp. 716–725.

[10] Ani AA (2005). Ant colony optimization for feature subset selection. Proceedings of world academy of science, engineering and technology, vol. 4, pp. 35–38.

[11] Gao HH, Yang HH, Wang XY (2005). Ant colony optimization based network intrusion feature selection and detection. Proceedings of the 4th international conference on machine learning and cyberneting, vol. 6, pp. 3871–3875.

[12] Huang CL, Wang CJ (2006). A GA-based feature selection and parameters optimization for suppert vector machine. Expert systems with applications, vol. 31, pp. 231–240.

[13] Shuxin Z, Bin H (2013). Hybrid Feature Selection Based on Improved Genetic Algorithm.

TELKOMNIKA Indonesian Journal of Electrical Engineering, vol. 11, no. 4, pp. 1725–1730.

[14] Kennedy J, Eberhart R (1997). A discrete binary version of the particle swarm algorithm. In: IEEE International Conference on Systems, Man, and Cybernetics, vol. 5, pp. 4104–4108.

[15] Unler A, Murat A (2010). A discrete particle swarm optimization method for feature selection in binary classification problems. European Journal of Operational Research, vol 206, no. 3, pp. 528–539.

[16] Yang CS, Chuang LY, Ke CH, Yang CH (2008). Boolean binary particle swarm optimization for feature selection. In: IEEE Congress on Evolutionary Computation (CEC'08), pp. 2093–2098.

[17] Lin SW, Lee ZJ, Chen SC, Tseng TY (2008). Parameter determination of support vector machine and feature selection using simulated annealing approach. Applied soft computing, vol. 8, pp. 1505–1512.

[18] Gherboudj A, Chikhi S (2011). BPSO Algorithms for Knapsack Problem. In Computer and Information Science, vol. 162, pp. 217–227.

[19] Chuang LY, Tsai SW, Yang CH (2011). Improved binary particle swarm optimization using catfish effect for feature selection. Expert system with application, vol. 38, no. 10, pp. 12699–12707.

[20] Yang CS, Chuang LY, Ke CH (2008). Boolean binary particle swarm optimization for feature selection. In: IEEE Congress on Evolutionary Computation, pp. 2093–2098.

[21] Chuang LY, Chang HW, Tu CJ, Yang CH (2008). Improved binary PSO for feature selection using gene expression data. Computational Biology and Chemistry 32(29):29–38.

[22] Liu YN, Wang G, Chen HL, Dong H (2011). An Improved Particle Swarm Optimization for Feature Selection. Journal of Bionic Engineering, vol. 8, no. 2, pp.191–200.

[23] Fdhila R, Hamdani T, Alimi A (2011). Distributed MOPSO with a new population sub-division technique for the feature selection. In: IEEE 5th International Symposium on Computational Intelligence and Intelligence Informatics, pp. 81–86.

[24] Hongfeng W, Dingwei W, Shengxiang Y (2007). Triggered Memory-Based Swarm Optimization in Dynamic Environments. In: Workshops on Applications of Evolutionary, pp. 637–646.

[25] Sun J, Feng B, Xu W (2004). Particle Swarm Optimization with Particles Having Quantum Behavior. In: Congress on Evolutionary Computation, Portland, USA, pp. 325–331.

[26] Hamed HNA, Kasabov NK, Shamsuddin (2011). SM Quantum inspired particle swarm optimization for feature selection and parameter optimization in evolving spiking neural networks for classification tasks. Evolutionary algorithms, vol. 1, pp. 133–148.

[27] Mohemmed A, Zhang M, Johnston M (2009). Particle swarm optimization based AdaBoost for face detection. In: IEEE Congress on Evolutionary Computation, pp. 2494–2501.

[28] Ghamissi P, Benediktsson JA (2015). Feature selection based on hybridization of genetic algorithm and particle swarm optimization. IEEE Geoscience and remote sensing letters, vol. 12, no. 2, pp. 309–313.

[29] Frank A, Asuncion A (2016). UCI Machine Learning Repository University of California, School of Information and Computer Science, Irvine, CA, 2016. http://archive.ics.uci.edu/ml.

[30] Hall M, Frank E, Holmes G, Pfahringer B, Reutemann P, Witten IH (2008). The WEKA Data Mining Software: An Update. ACM SIGKDD Explorations Newsletter, vol. 11, pp.10–18.

A Multi-Agent based Approach for Simulating the Impact of Human Behaviours on Air Pollution

Sabri Ghazi
Laboratoire de gestion électronique du document, Computer Science department, University Badji Mokhtar
PO-Box 12, 23000, Annaba, Algeria
E-mail: sabri.ghazi@univ-annaba.dz

Julie Dugdale
University Grenoble Alps. LIG. France
Email: julie.dugdale@imag.fr

Tarek Khadir
Laboratoire de gestion électronique du document, Computer Science department, University Badji Mokhtar
PO-Box 12, 23000, Annaba, Algeria
E-mail: khadir@labged.net

Keywords: air pollution, multi-agent simulation, agent-based simulation, air quality, N-persons prisoners' dilemma, game theory

Received: April 20, 2016

This paper presents a Multi-Agent System (MAS) approach for designing an air pollution simulator. The aim is to simulate the concentration of air pollutants emitted from sources (e.g. factories) and to investigate the emergence of cooperation between the emission source managers and the impact this has on air quality. The emission sources are controlled by agents. The agents try to achieve their goals (i.e. increase production, which has the side effect of raising air pollution) and also cooperate with others agents by altering their emission rate according to the air quality. The agents play an adapted version of the evolutionary N-Person Prisoners' Dilemma game in a non-deterministic environment; they have two decisions: decrease or increase the emission. The rewards/penalties are influenced by the pollutant concentration which is, in turn, determined using climatic parameters. In order to give predictions about the concentration of pollutants: Particulates Matter (PM10), Sulphur Oxide and Dioxide (SO_x), Nitrogen Oxides (NO_x) and Ozone: (O_3), a two stage prediction method is used, a GPD (Gaussian Plume Dispersion) model and an ANN (Artificial Neural Network) prediction model. The prediction is calculated using the dispersal information and real data about climatic parameters (wind speed, humidity, temperature and rainfall). Every agent cooperates with its neighbours that emit the same pollutant, and it learns how to adapt its strategy to gain more reward. When the pollution level exceeds the maximum allowed level, agents are penalised according to their participation. The system has been tested using real data from the region of Annaba (North-East Algeria). It helped to investigate how the regulations enhance the cooperation and may help controlling the air quality. The designed system helps the environmental agencies to assess their air pollution controlling policies.

Povzetek: V prispevku je predstavljen večagentni sistem za simulacijo onesnaženja zraka.

1 Introduction

The question about how humans should moderate their exploitation of environmental resources has occupied researchers for decades [1]. Promoting social and economic growth without affecting the environmental equilibrium is important for maintaining sustainable development. This paper addresses the relation between human behaviours and their impact on air quality in socio-environmental systems. Air pollution is a major concern in many cities in the world, especially in developing countries. It has a direct influence on our health and quality of life [2]. The degradation in air quality should be estimated before the establishment or the expansion of urban or industrial activities. Air pollution simulation and decision support tools can help decision-makers to establish policies for environmental management and to predict the impact of their decisions on the environment and ecosystem. Many modelling approaches have been proposed to study air pollution. Most of them, ([3], [4], [5]) to cite a few, are mainly focused on the physical and chemical aspects of air pollution; the concentration and dispersal of pollutant. These models do not take into consideration human-decision factors. Air pollution is by nature distributed and includes the interaction of individuals involved in the exploitation of a dynamic ecological resource which is the air. The anthropogenic activities (road traffic, industrial and agricultural activities) are among the major sources of air pollution. All of these activities are controlled by humans; therefore, including the human decision factors in the modelling of air pollution is essential.

MAS (Multi-Agent System) based models are an appropriate method for modelling socio-environmental issues [6]. They allow us to model the behaviours of human actors sharing the exploitation of environmental resources. [7] presents a review of recent MAS models used to investigate socio-environmental problems. The models are classified according to: the decision making mechanism, the use or not of real data, the objective of the simulation, and the space and time representation. [8], [9] used a MAS approach to investigate the air pollution emission resulting from road activities; they used a traffic flow simulation and linked it to emission calculation. [10] used the same approach to study the effect of transport regulation on air pollution emission. [11] present a MAS designed for monitoring air quality in Athens, Greece. The MAS is a set of agents that control a network of sensors installed in an urban region. They verify and collect the data measured by sensors. [12] present a MAS to find the dispersion of air pollution in urban region. The pollution sources (polluters) are represented by homogeneous agents that emit pollution in their respective areas. Each agent pollutes with an emission rate. As the simulation runs, clusters are formed with different values of pollution concentration. At the end, a single cluster is formed, thus, the dispersion of pollution is estimated.

The managers of emission sources share the exploitation of the air by emitting pollutants. We aim to simulate their different personalities (e.g. eco-friendly, selfish) and investigate the relationship between the emergence of cooperation and its impact on air quality. The main questions addressed in this paper are: How do emission source controllers cooperate, are they able to achieve their goals while preserving a reasonable air quality? What regulatory rules should be adopted to enhance the cooperation and sustain the air quality? To investigate these questions we have designed a MAS simulation tool that helps to investigate the emergence of cooperation and its effects on air quality. The proposed simulator models the population of emission source controllers as a network of agents playing an Evolutionary NPPD (N-Person Prisoners' Dilemma) game. Evolutionary NPPD has been widely used for studying the emergence of cooperative behaviour among a population of selfish agents, how agents exhibit altruistic behaviours and under which condition cooperation will be sustained.

NPPD is a mathematical model, which models the conflict between players sharing the use of a common resource. Initially, it was formulated for two players where each one has to take two possible actions (defect or cooperate), and then receive a payoff according to their joint actions. A version for N-Persons has been proposed [13] where the payoff is calculated according to the number of agents choosing to cooperate; the payoff function is given in (1)

$$u(ncp) = \begin{cases} \frac{b*ncp}{N} - c & if\ s = 0 \\ \frac{b*ncp}{N} & if\ s = 1 \end{cases} \quad (1)$$

With $b > c > 0\ and\ \ c > \frac{b}{N}$, s is the action taken by the agent s ϵ{0,1} (where 0 means cooperate and 1 means defect), ncp is the number of players who chose to cooperate, N is the size of the player population and b is the defection temptation, the constant c is used to ensure that the cooperation reward is less than the defection reward.

[14], [15], studied the emergence of cooperation in a NPPD. The authors used different agent personalities and neighbourhoods in order to investigate their impact on the evolution of the game outcome. The experiment used different agent types with different initial co-operators ratio; this showed that for the case where all agents are Pavlovians (repeating actions that give them more satisfaction), the aggregate outcome of the game can be predicted for any number of agents and any payoff function. The choice of the agent's neighbours also has a big influence on the game equilibrium. [16] investigated the effect of social welfare preference on the emergence of cooperation among agents placed on a BA [17] network. The authors proposed a model where some of the agents also take into consideration social welfare and not only their payoff received from the game. Agents do not only care about their own payoff, but also the payoff of their neighbours.

[18] describes the use of a NPPD game to investigate the cooperation in a socio geographic community. The use of NPDD for environmental modelling has proved to be suitable since the exploitation of a shared ecological resource can be formulated as a tragedy of the common [19]. Each actor tends to maximise its profits by exploiting a shared ecological resource. Thus, a tragedy of the common arises. [20] uses a PD model to review Porters' hypothesis, which studies the relationship between productivity and eco-friendly technologies. The work models how strict environmental regulations can enhance innovation for a less polluting technology. Firms have two actions which are: to invest in a new less polluting process or to continue using the old one and be penalised according to the governmental regulations. [21] used a version of NPPD to investigate the cooperation in international environmental negotiation to reduce CO2 emissions. [22] presents an evolutionary game theory approach to study the influence of the ecological dynamic and payoff structures over the emergence of cooperative behaviour between landowners. The landowners are modelled as selfish agents aiming to maximize their profit by managing the number of deer on their lands. The main novelty of our approach is the inclusion of human decisions as a key element for simulating the air pollution evolution. We model the managers of the emission sources of pollutants as autonomous agents. These agents aim to maximise their own profit and we investigate this effect on air quality. The designed system helps investigating the efficiency of the regulatory rules used by air pollution controlling

agencies for maintaining the air quality. This is very important because it helps the environmental agencies to assess their air pollution controlling policies.

The paper is organised as follows: The methodology is presented in section (2) that describes a MAS approach for designing an air pollution agent based simulator. Subsection (2.1) presents the representation of space and time. Subsection (2.2) describes the dispersion model and the Artificial Neural Network (ANN) prediction model.

Many conceptualizations have been proposed to represent a socio-environmental system [22]. Generally, a socio-environmental simulation system can be represented as an interconnection of three components (or subsystems); each one is represented by a set of variables (attributes) forming its state at time *t*. The ecological component models the biotic (living) and abiotic (non-living) parts. The economic component represents the economic view point and groups the economic variables. The social component represents the human social networks such as decision-makers, firms, government agencies and consumers. A change in the state variable of each component affects other systems' state variables. For example, the increase in demand for a certain kind of fish, leads fishermen to intensify their exploitation; this in turn results in changes to the biodiversity. We present a generic formalization of a socio-environmental system. A coupled social and environmental system can be expressed as a set of economic, social and ecologic state variables. The state of the system at time step *t* can be formulated as (2).

$$ES_t =< Ec_t, Sc_t, Envc_t > (2)$$

Where Ec, Sc and Envc represent, the sets of economic, social and environmental state variables, respectively:

$$Ec_t =< Ec_{1,t}, \dots, Ec_{l,t} >, Sc_t =< Sc_{1,t}, \dots, Sc_{m,t} >$$
$$Envc_t =< Envc_{1,t}, \dots, Envc_{n,t} > (3)$$

In our case, the environment state variables at time step *t* are: $Envc_t =< c_{0,t}, .. c_{n,t}, WS_t, Hu_t, T_t, RF_t >$ (4)

$c_{i,t}$ is the concentration of the pollutant *i*, *WS*: wind speed, *T*: temperature, *Hu*: humidity and *RF* represents the rainfall, at time *t*. Assuming that the source of pollution at time *t* is modelled as:

$$S_t =< er_t, tc, X, Y, Z > \quad (5)$$

The source produces the pollutant *tc* with the rate *er* at the geo-position (*X,Y,Z*). Sources are controlled by agents. Every agent has to make a decision on which action to choose among all possible actions according to the state of the environment *ES* and its internal state at *t*. Let *A* be the set of actions $A = \{a_1, a_2, \dots, a_z\}$, the result of an action is the change in the emission rate of the pollutant from the controlled source. We can define this as a function that takes the agent's action and as a result gives the new emission rate (6).

$$F: A \rightarrow \mathbb{R} \quad (6).$$

Let π_t be the action vector done by *N* agents at time *t*: $\pi_t =< A_{0t}, .. A_{nt} >$ (7)

Let *Q* be the set of possible air quality index values: $Q = \{very\_bad, bad, average, good, very\_good\}$, each of these indexes has its numerical equivalent in terms of pollutant concentration, as shown in table 1.

The agent decision-making mechanism is given in (2.3). A test scenario is presented in section (3). Results are detailed and discussed in section (4). The paper ends with conclusions and the possible further directions of our work.

2 Model approach and architecture

SOx µg	***NOx µg***	***O₃ µg***	***PM10 µg***	***Indices***	***Category***
0 – 30	0-45	0-45	0-20	1	Very Good
30-60	45-80	45-80	20-40	2	Good
60-125	80-200	80-150	40-100	3	Average
125-250	200-400	150-270	100-200	4	Bad
>250	>400	>270	>200	5	Very Bad

Table 1: Air pollution quality.

The air quality can be modelled as a graph with T as transition function:

$$T(\pi_t, ES_t, current\_q) \longrightarrow new\_q,$$
$$current\_q \ and \ new\_q \in Q \ (8)$$

T takes as arguments the state of the system ES_t and the set of actions done by N agents and accordingly it moves the system from the current state (current_q) to a new state (new_q). Under some conditions current_q may be equal to new_q, which means that actions of the agents do not change the air quality under some climatic conditions.

Our simulation approach can be schematised as shown in figure 1. Agents' actions affect the emission rate of the sources. Then the dispersion algorithm is used to compute the dispersion, the aggregated value of pollutant concentration is used with climatic parameters to forecast the next 2 hours air pollution concentration and air quality. According to these forecasts, agents are rewarded or penalised. Agents then adapt their strategies to earn more reward and reduce penalties.

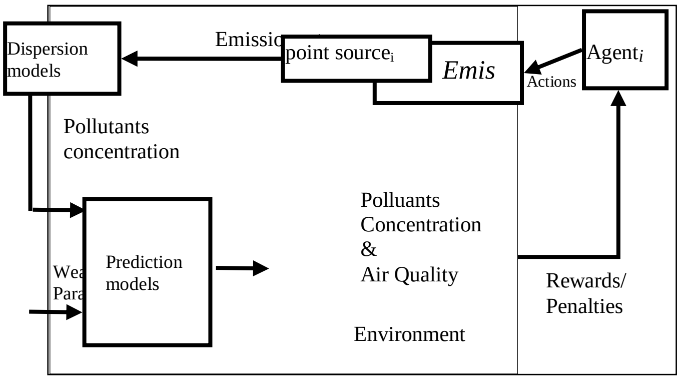


Figure 1: The simulation process, using the dispersion model and the prediction model.

2.1 The spatial and temporal scale of the simulation model

The simulation uses a discrete representation of time where each simulation step represents by default 6 hours of real time. The simulation's duration is defined in the interface and depends on the objective of the simulation (short or long term prediction).

Our model is based on the hypothesis that the action of the emission controllers (reducing or increasing emissions) has an impact within k time-steps. k is a parameter whose value is provided by the user according to the scenario and available data.

Since the simulation step k can be changed, we can represent a long term simulation horizon by giving k a higher value. So for example we can represent 1 step as 24 hours meaning that industrial polluters can take several days to adjust their production volumes. Setting k even higher, such as 2 weeks or 1 month would require data, which is not available, to see the evolution of the AQ.

The environment is modelled as a set of 3D boxes, each one represents one km^3. It can be represented as: $BX = \{bx_0, \ldots, bx_m\}$, every box is localised in the geo-position point gp(x,y,z) and has attributes representing the concentration of air pollutants ($cp_0,\ldots,cp_v$) and air quality, These attributes are used when agents are penalized according to the pollution level in the box where they are situated. In this case the position of the emission source in a box is relevant. Sources located in the same box are considered to be neighbours.

2.2 Dispersion and prediction models

The dispersion model helps to measure how the pollutant will spread in the air. It is calculated according to the distance from the point source, the wind speed and direction. We used a GPD (Gaussian Plum Dispersion model), which is frequently used in atmospheric dispersion [24]. The dispersion model is run in a steady way, which means that no parameter (wind speed, emission rate and wind direction) is changed during the simulation step. This provides a series of snapshots of the situation at each step. These snapshots are then fed into the ANN model to obtain a prediction about the concentration. Since we cannot combine the two models in a continuous way our solution of taking a series of snapshots and feeding it to the ANN mimics a continuous process. The GPD simulates the dispersal from a point source emission according to the emission rate (9).

$$C(x,y,z,H) = \frac{er_{i,t} * D}{2\pi U_t\ \sigma_y \sigma_z} * e^{-\frac{y^2}{2*\sigma_y^2}} * \left[(e^{-(\frac{(z-H)^2}{2\sigma_z^2})}) + (e^{-(\frac{(z+H)^2}{2\sigma_z^2})}) \right] \quad (9)$$

This means that the concentration of the pollutant at point (*x,y,z*) is calculated according to :

$er_{i,t}$: the emission rate in kilograms per hour of the source *i* in time step *t*.

U_i: the wind speed in metres per second at time step *t*, $\sigma_y \sigma_z$: the standard deviation of the concentration distributions in a crosswind in a vertical direction, these two parameters are chosen according to the stability class 'C' in the Guifford-Pasquill scale [25], and *H* is the height of the source from the ground. The decay term *D* is given in [26] and computed according to (10).

$$D = \begin{cases} e^{\left(1/(R*\frac{x}{U})\right)}, if\ R > 0 \\ 1,\ if\ R = 0 \end{cases}, \quad (10)$$

Where x is the downwind distance, u is the wind speed and *R* is the decay coefficient. The values for *R* are adopted from [27] for NO_x (0.45 h^{-1}) and SO_x (0.31 h^{-1}), PM10 is not considered (*R=0 and D=1*).

For simplification and due to lack of wind direction data, we assume that the wind direction does not change during the simulation step. The resulting pollution level from each source is aggregated and the average of each box is computed. Then the dispersion value of the pollutant is passed to an ANN prediction model as described in [28]. The ANN prediction models are designed to give a forecast of the five air pollutants and the air quality. This includes an uncertainty aspect caused by the weather conditions [29]. The ANN predictor uses the aggregated air pollution concentration value given by the dispersion model of each source and the four climatic parameters: wind speed, humidity, temperature and rain fall. These parameters greatly influence the pollutant concentration in the air [30].

O_3 is a secondary pollutant, which means that it is not emitted by sources, but results from the photochemical interaction between SO_x, CO_x, and organic components. Therefore, we used SO_x and CO_x dispersion information to predict the O_3 concentration.

For each pollutant a RBF (Radial Basis Function) network is designed and trained. The RBF is composed of three layers, the first layer is connected to the input of the network and its output is connected to the hidden layer, the neurones in the hidden layer have the RBF as the activation function. The outputs of the hidden layer are linearly combined to obtain the output of the network. Using a training data set, the objective is to find the optimal combination between the number of neurons in the hidden layer and the weight of each input. By increasing the number of the neurons in the hidden layer the algorithm [31] gives the optimal topology of the network. This is why many topologies are tested and only the best of them are taken. During the training step, each network receives as input a vector of the climatic condition parameters and the concentration of the pollutant at time *t*. Each network generates the desired output that is the value of pollutant concentration at time *t+PredictionHorizon*. The forecast given by each network is passed as input to predict the air quality index using a MLP (Multi-Layered Perceptron). The MLP network is trained using the local air quality standards as shown in table 1. Air quality predictions for the different pollutants are obtained on a *t+12* hours basis and give the most probable air quality category. The MLP model receives the predicted values of the five pollutants, CO_x, NO_x, O_3, PM10, and SO_x, and predicts the index values for air quality ranging from 1 to 5, with 1 being very good and 5 being very bad. To train the MLP network we used a Levenberg-Marquardt back-propagation algorithm. The MLP network final topology, obtained after several trials, is: 5 neurons in the first hidden layer, 10 in the second and finally a linear neuron for the output layer.

The accuracy of the ANN models are given in table 2, and is calculated using one year's worth of data according to RMSE (Rooted Mean Squared Error) formulated in (11):

$$RMSE = \sqrt{\frac{\sum_{i=1}^{lng}(P_i - R_i)^2}{lng}} \qquad (11).$$

Where lng is the length of the vectors, P and R are the predicted and measured values, respectively. The performances are computed using a validation data set that was not used in the training of the ANN models.

Model	Topology [# of input neuron -# of hidden neuron]	Validation error (RMSE)
PM10	[10-320]	16.1945 µg/m ^{3}
SOx	[10-90]	3.1618 µg/m^3
NOx	[10- 105]	9.7277 µg/m^3
CO$_x$	[10-45]	0.1220 µg/m^3
O3	[10-180]	39.8238 µg/m^3

Table 2: Validation error of the ANN prediction models using the validation data set.

2.3 Decision-making mechanism

Based on its internal state and the state of the environment, an agent has to choose an action to perform among all possible actions in order to reach its goals. This process is called decision-making. [32] presents a review of methods used for modelling decision-making in a coupled environmental and social system. Our system supports two cooperation strategies (centralized and evolutionary game) each one defines a decision-making mechanism. The centralized strategy (CS) is based on defining a central agent that represents the air pollution controlling agency. The central agent takes decisions according to the current air pollution level. The second strategy is based on an evolutionary game, where agents are rewarded and penalized according to the pollution levels; they make decisions according to their rewards. In our system, the cooperation strategy is defined within the simulation parameters.

2.3.1 Centralized Strategy (CS)

The task of maintaining the air quality is assigned to an agent, which represents the air pollution control agency. It uses the GPD and ANN models to predict the air quality and pollutant levels, and according to the predictions it sends a reduce emission message to the emission agents. Then it will check the air quality. It will continue doing this until the air quality is improved to reach the air quality index goal. The central agent has absolute authority and its orders are executed by the emission source controllers. Agents communicate their emission rate at each simulation step to the central agent. This strategy is based on the communication between agents. We assume that agents are rational and have an environmental-responsible personality; this means they favour air quality improvement over their own interests and communicate their exact emission rate to the central agent.

2.3.2 Evolutionary Game Cooperating Strategy

In the EG strategy, every agent has its own goals (earning more rewards and keeping its emission rate high) and shares a global goal of maintaining air quality with other agents. The appreciation function defined as: $app: Q \rightarrow R,$ allows comparing the air quality state at each step of the simulation. A global goal GG can be defined as (12). This means finding a set of actions π_t to be performed by agents at time *t*, which allows the system to move to a new state of air quality q_{t+1} that is better than the current state.

$$GG_t = \{T(\pi_t, q_t, q_{t+1}), app(q_{t+1}) > app(q_t)\} \quad (12)$$

An agent participates with other agents in the NPPD game, its own goal is to maximise its reward earned from the game. We adopted the approach of [33], where agents keep traces of their L previous steps (actions, rewards and its neighbours' rewards). To update the probabilities to increase or decrease the emission, we used [18] method. At each time step t the agent computes its weighted payoff, according to (13), and tries to maximise it (as its utility function) by taking it into consideration when computing its probability to increase or decrease its emission rate, respectively according to (14) and (15).

$$WP_i(t) = \sum_{i=1}^{L} w_i * M_i(t) \qquad (13)$$

Where: w_i is the weighting parameter where $\sum_{nbr=1}^{L-1} w_{nbr} = 1$ and $\forall\, i, j\ (i < j \rightarrow w_i > w_j)$, $M_i(t)$ is the payoff for the agent i for the time step t.

$$\begin{cases} Pc_i(t+1) = Pc_i(t) + (1 - Pc_i(t) * \alpha_i(t) & , if\ S_i = 0\ and\ WP_i(t) > 0 \\ Pc_i(t+1) = (1 - \alpha_i(t)\ \ ) * Pc_i(t), & if\ \ S_i = 0\ and\ WP_i(t) \le 0 \end{cases}$$
(14)

$$\begin{cases} Pd_i(t+1) = Pd_i(t) + (1 - Pd_i(t) * \alpha_i(t) & , if\ \ S_i = 1\ and\ WP_i(t) > 0 \\ Pd_i(t+1) = (1 - \alpha_i(t)\ \ ) * Pd_i(t), & if\ \ S_i = 1\ and\ WP_i(t) \le 0 \end{cases}$$
(15)

Where: Pc_i and Pd_i are respectively the probability to decrease (s=0) and increase (s=1) emissions for agent i, $\alpha_i(t)$ is the learning rate of agent i at time step t, s is the strategy played at time t. The learning rate is updated according to (17):

$$D_i = \sum_{j=1}^{L-1} \begin{cases} 0\ if\ X_{i,j} = X_{i,j+1} \\ 1\ if\ X_{i,j} \neq X_{i,j+1} \end{cases} \qquad (16)$$

$$\begin{cases} \alpha_i(t+1) = \alpha_i(t) + 0.015 & if\ D_i = 0 \\ \alpha_i(t+1) = \alpha_i(t) + 0.010 & if\ D_i > L - 1 \\ \alpha_i(t+1) = \alpha_i(t) - 0.010 & if\ D_i \le L - 1 \end{cases} \qquad (17)$$

Where D_i is the i-th agent actions homogeneity indicator, at time step t, $X_{i,j}$ is j-th action of the agent i. D_i is used to compare the last L actions of the agent. This is used to keep the agent from changing its actions. Agents are influenced by their neighbours, at each time; the average reward of the neighbours is calculated according to (18).

$$nP_i(t) = \left(\sum_{j=1}^{numberOfneighbours_i} M_j(t)\right) / numberOfneighbours_i \quad (18)$$

Where $M_j(t)$ is the payoff of the neighbour j and $numberOfneighbours_i$ is the number of neighbours for the i-th agent. We keep a trace of the nP of the L previous simulation steps and we compute their average in avgnP. The agent then uses the probabilities Pc, Pd and the average reward of its neighbours to choose an action according to (19):

$$\begin{cases} if\ S_i(t) = 0, S_i(t+1) = \begin{cases} 1\ , if\ WP_i < avgNP_i\ and\ Pd_i(t+1) > Pc_i(t+1), \\ 0, else \end{cases} \\ if\ S_i(t) = 1, S_i(t+1) = \begin{cases} 0\ , if\ WP_i < avgNP_i\ and\ Pc_i(t+1) > Pd_i(t+1), \\ 1, else \end{cases} \end{cases}$$

(19)

At each simulation step, every agent gets a reward or penalty according to its actions and according to the pollution level. We have adopted the payoff curve (1) with b=2 and c=-0.5, but in the general case these parameters can be defined by the user. When the pollution level is higher than the maximum allowed value, the participation of the agent to the current level of the pollution $\sigma_i(t)$ is computed according to (20).

$$\sigma_i(t) = \frac{ER_i(t)}{PL_e(t) - PL_{max}}, PL_e(t) > PL_{max} \qquad (20)$$

Where, $ER_i(t)$ is the emission rate of the i-th agent at time t, $PL_e(t)$ is the pollution level of the pollutant e at time t and PL_{max} is the maximum allowed value for the pollutant level according to the regulation and local standards. The penalty for agent i at time step t is calculated according to (21):

$$EcoFactor_i(t) = EcoFactor_i(t-1) + (1 - \frac{1}{e^{\sigma_i(t)}}) \ (21)$$

Two penalising strategies were used; the first uses (21) and is a cumulative penalty. This means that the penalties from each step are kept and the agent is penalised as long as it continues to increase its emission. The second penalising method is not cumulative, and agents are penalised just according to the current simulation step.

The reward of agent i, at the current time step t is computed according to (22), we compute the number of agents who choose to decrease their emission denoted ncp, after that we compute u use as defined in equation (1).

$$M_i(t) = \begin{cases} u(ncp) & if\ s_i = 0 \\ u(ncp) - EcoFactor_i(t) & if\ s_i = 1 \end{cases} (22)$$

3 Simulation scenarios using data from the region of Annaba

Annaba is a very industrialized region specialising in steel industries. The steel complex of Hadjar is located 12 kilometres south of the city of Annaba. The air pollution spreads over a radius of 6 km. According to [34], the complex annually releases into the atmosphere: 36890 tons of particles, 845 t of NO_X, 30895 t of CO_x, 2260 t of SO_x and 3093 t of NO_x. The petrochemical station (ASMIDAL) produces fertilizers and pesticide products that have a big influence on air quality. 5 industrial zones, that contain hundreds of factories, are very close to the urban area and have a large impact on air pollution. The seaport is located in the centre of the city and attracts a lot of heavy transport, which also leads to deterioration in the air quality.

The local pollution agency network provided hourly data for a two-year period from 01/01/2003 to 31/12/2004. The concentrations of air pollutants that have been continuously monitored are: Ozone (O_3), Particulate Matter (PM10), Nitrogen Oxides (NO_x), and Sulphur Oxides (SO_x). The dataset also includes four meteorological parameters: Wind Speed (WS), Temperature (T) and relative Humidity (H). Daily rainfall measurements (RF) were also provided by the water management agency. The 2003 dataset was used for training the ANN and the 2004 dataset was used for validation; this helped us to assess the performance of the model. The pollutant concentration measurements are in microgram/m^3 and they have been normalised using equation (23).

$$V'_p = \frac{V_p}{(\max(V_p) - \min(V_p))} \qquad (23)$$

Where *Vp* is a parameter vector, *min* and *max* are functions that return the minimum and maximum values of the vector. Negative values, resulting from faulty measurements, were replaced using the mean of the previous and next values. It is impossible to discard faulty values since gaps in the time series will result in a data shift that affects the ANN training process leading to poor generalisation properties. Similarly, faulty (blank) measures for pollutants and weather parameters were replaced by an average of the *v-q* and *u+q* previous and future values respectively, with *u* being the faulty sample and q the number of values to take into consideration. This ensures the continuity and consistency of the time series and allows efficient training of the ANN predictors. Table 3 presents the statistical properties of the available data for different pollutants and weather parameters, for some parameters data are not available (N/A).

We defined a simulation scenario for the Annaba region using the parameters in table 4. The goal levels for pollutants concentration were fixed according to the air quality index goal. For this scenario we aimed to reach a very good air quality level (Goal air quality index=1). The initial values (at t=0) for pollutant concentration and climatic parameters were fixed according to the dataset.

Parameter	2003 mean	2004 mean	2003 STD	2004 STD	Max value
PM10 µg/m^3	51.70	27.76	51.66	26.38	508
NOx µg/m^3	14.50	N/A	25.01	N/A	435.0
SO_x µg/m^3	7.60	N/A	14.78	N/A	190.0
CO µg/m^3	1.31	N/A	0.52	N/A	12.2
O_3 µg/m^3	N/A	42.27	N/A	64.58	688.0
Wind Speed µg/m^3	2.65	2.12	1.78	1.27	9.6
Humidity (%)	63.52	71.92	16.50	14.33	93.0
Temperature (°C)	18.96	16.82	7.76	6.30	42.1
Rainfall (mm)	N/A	2.96	N/A	9.27	73.9

Table 3: Statistical properties of the used dataset.

For the EG strategies we fixed the initial proportion of cooperating agents (agents choosing to decrease emissions) to *0.5*, this means that *50%* of the agents decrease their emission at *t=0*. The value of this parameter was chosen following the work of [14] and [15]. The proportion will change during the simulation according to the game outcome. The prediction was for the next 2 hours, the same as the simulation step. Each source emits according to its emission rate which cannot be higher than the maximum level defined in the simulation scenario. The position of sources was randomly generated and many sources are located in the same box.

Parameter Name	**Value**
Polluting activities and Policy parameters	
Number of PM10 sources	100
Number of SO_x sources	100
Number of NO_x sources	100
Number of CO sources	100
Max emission rate	2000 (gram/hour).
Goal PM10 level	20 µ gram/m^3
Goal SO_x level	30 µ gram/m^3
Goal NO_x level	45 µ gram/m^3
Goal O_3 level	45 µ gram/m^3
Number of memory steps (*L*)	4 steps
Initial proportion of cooperating agents	0.5
Environment parameters	
Number of boxes	20
Temperature at t=0	12.7 (°C)
Humidity at t=0	71.0 %
Wind Speed t=0	2.4 m/s
PM10 at t=0	13.0 µ gram/m^3
SO_x at t=0	17.0 µ gram/m^3
NO_x at t=0	2.0 µ gram /m^3
CO at t=0	0.5 µ gram /m^3
O_3 at t=0	29.0 µgram /m^3
Air Quality at t=0	2 (Good)
Total simulation time	4900 hours
K Simulation step	1 step = 2 hours
Prediction horizon	Next 2 hours

Table 4: Parameter values of the simulation scenario.

4 Results and discussion

We have built a simulator using the approach described above. We used the JADE agent framework [35] and ANN models from Encog [36]. We have defined 5 strategies: EG-CP (Evolutionary Game with Cumulative Penalty), EG-NCP (Evolutionary Game with No Cumulative Penalties), EG-NP (Evolutionary Game with No Penalty), CS (Centralized Strategy) and NC (No-Cooperation). The last one is used for comparison purposes. Using the parameters shown in table 4, we chose a strategy and ran the simulation 16 times. We then changed the strategy and ran the simulation again 16 times; since we have 5 strategies we obtain 80 simulations. The most explicative results are presented. For the CS and NC cases the simulator showed similar results for each run. For the EG strategies there were small differences between runs, especially concerning the proportion of cooperating agents. These changes are due to the random values used in the initialisation of some variables (neighbours rewards, first chosen action, weights, *k* last actions and rewards). The comparison is done according to the air quality index. Results are expressed in terms of the number of occurrences of air quality index as illustrated in figure 2, for example the number of times the air quality index equals 1 (very good). Figure 3, shows the evolution of the air quality index over time. The CS gives the best performance. With the CS the air quality index moves rapidly from bad to average and then to good and finally stabilises at very-good (which is the goal fixed in the simulation scenario). The EG-CP moves the index from bad to average, when the equilibrium is reached it stabilises in good and never reaches a very-good index. The EG-NCP strategy moves the air quality from bad to average and never improves. When penalties are not used (EG-NP) the air quality stabilises at bad. When cooperation is not used (NC), agents act selfishly and do not care about the pollution, therefore, the air quality oscillates between bad and very-bad. As the agents reach their maximum emission rate we can observe an oscillation, which is caused by the climatic conditions. The only thing that affects the pollutant concentration is the climatic conditions (the emission rate is constant); these have a big influence and are captured with the ANN model.

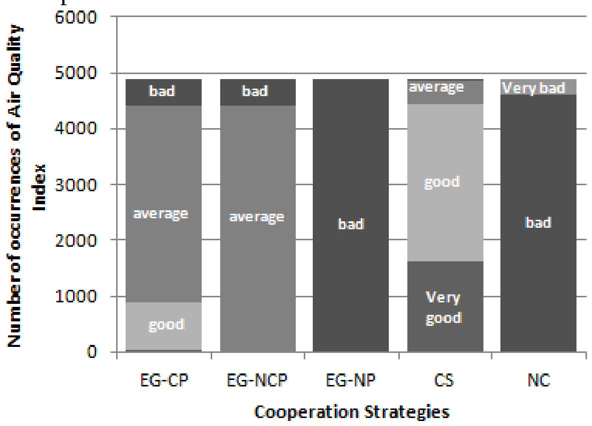


Figure 2: Air quality index using 5 different cooperation strategies.

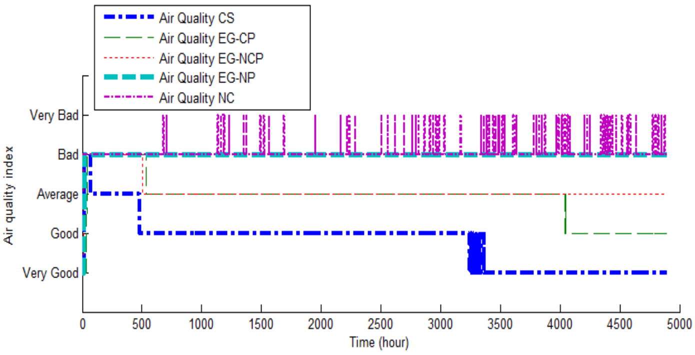


Figure 3: Air Quality index for 4900 hours.

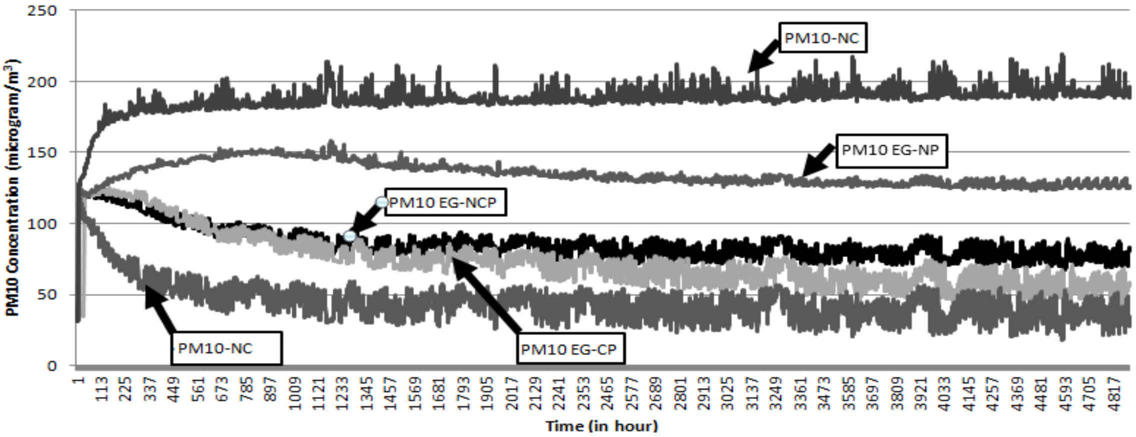


Figure 4: Concentration of PM10 for the four tested cooperation strategies compared with the no-cooperation strategy.

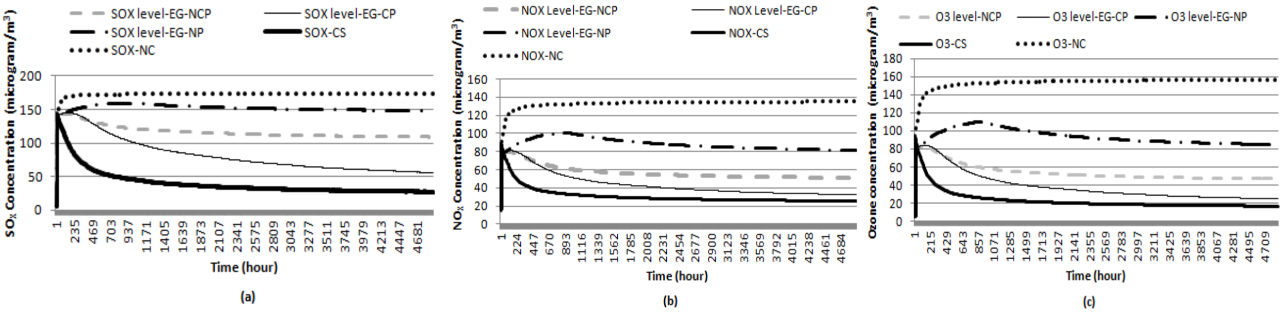


Figure 5 : Concentration of SO_X (a), NO_x (b) and O_3 (c) using the four strategies and the no-cooperation scenario.

Figure 4 shows the evolution of the PM10 concentration during the simulation time. The PM10 concentration shows many peaks compared with the other pollutants under the same climatic conditions. This is due to the dry nature of the weather in the Annaba area, with wildfires, and sandstorms coming from the great Sahara desert. These events have a big effect on the PM10 concentration but not on the other pollutants. The CS strategy takes less time to control the pollution level and keep it below the goal level defined in the simulation parameters. All of the EG strategies take longer, keeping it close to the goal level, but without ever reaching it. The penalising regulations have a big effect on the PM10 level. As illustrated, the EG-CP (cumulative penalising method) controls the pollution better than the non-cumulative one, and both methods perform better than the no-penalising strategy. The no-cooperation is presented in order to show the impact of cooperation on the PM10 level. Figure 5 shows the evolution of the SO_x, NO_x and Ozone concentrations during the simulation time using four different cooperation strategies. The CS strategy gives the best performance since the pollution concentration rapidly decreases. The EG strategies show the same performance as for PM10 and the pollution level is widely influenced by the selected penalising method. The CP strategy appears to be the best one followed by the NCP. The pollution slowly decreases, but not enough to reach the goal level if penalisation is not used.

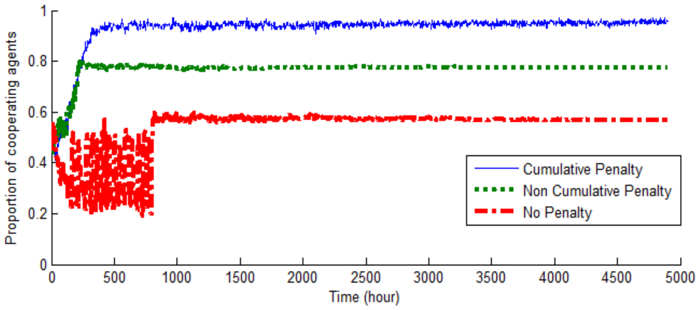


Figure 6: Proportions of cooperating agents for EG-CP, EG-NCP and EG-NP.

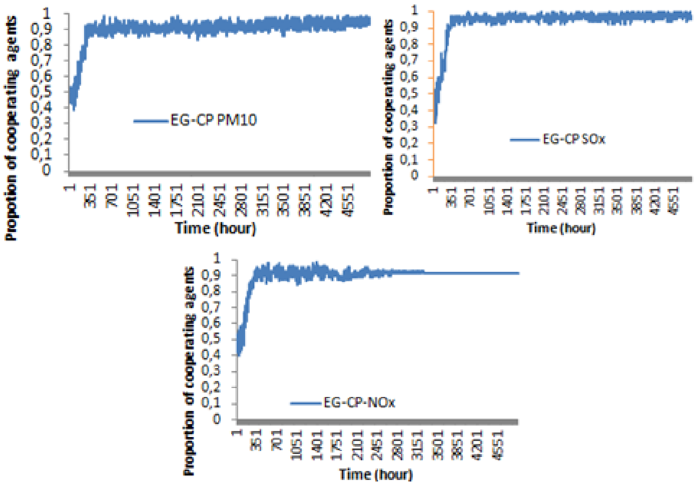


Figure 7: The proportion of cooperating agents according to the emitted pollutant for the EG-CP strategy.

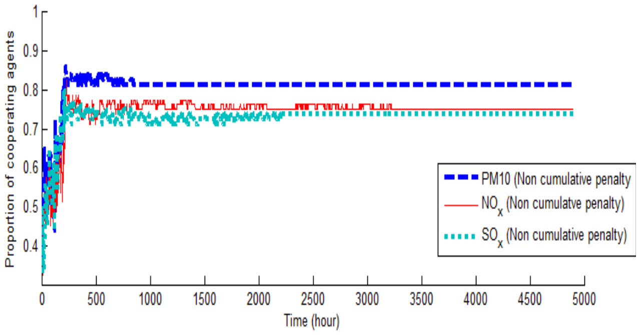


Figure 8: The proportion of cooperating agents for the EG-NCP strategy.

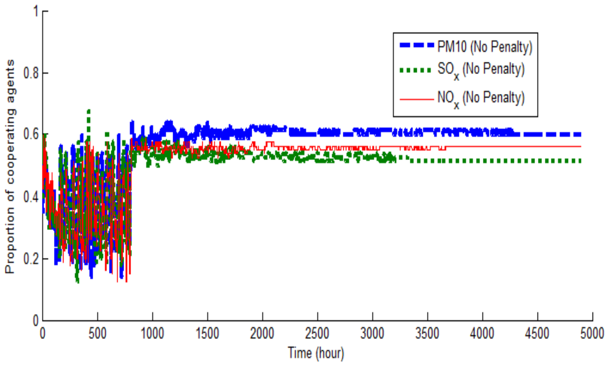


Figure 9: The proportions of cooperating agents, for the 4 groups of agents, when penalisation is not used**.**

The NC strategy gives the worst levels; when all agents are emitting pollution using their maximum emission rate, the pollutant level reaches alarming values and peak periods occur.

Figure 6 shows how the penalising method affects the proportion of cooperating agents. For the case of EG-CP the game equilibrium is reached at time step 387 and stabilises when the proportion of cooperating agents is between 0.93 and 0.95. The EG-NCP strategy stabilises early at time step 196 and oscillates between a cooperation ratio of 0.73 and 0.80, after which the equilibrium is fixed at 0.77. The EG-NP is the slowest; the equilibrium is reached at time step 808 with a cooperation ratio of 0.57. This happens because the agents are not penalised since the strategy does not include penalising methods. Figures 7, 8 and 9 show, respectively, the proportion of cooperating agents according to the pollutant for the three penalising strategies EG-CP, EG-NCP and EG-NP. The PM10 agents gives the highest cooperation ratio. This is because many peaks occur with this pollutant and the others pollutants cooperating ratio (NO$_x$ and SO$_x$) are influenced by the O$_3$ concentration. The more the pollutant exceeds the allowed level, the greater the proportion of co-operators. The equilibriums are disturbed by the pollution level, because, when the pollution has not yet reached the goal level, agents agents start being penalised, and thus they tend to cooperate more.

5 Conclusions

Anthropogenic activities are among the main causes of pollution and environmental problems. These activities have to be included in the simulation models. Modelling the interaction between social and ecological components is a very important aspect. A MAS approach allows us to model the social network of human-beings sharing the exploitation of common environmental resources. Manipulating the behaviour at an individual and group level helps to gain more knowledge about the impact of human decision-making on pollution and makes the simulation more realistic. Studies treating air pollution are usually concerned with the physical aspects (concentration and dispersion of pollutant), and do not include human-decision factors on the emission sources.. In our approach, we model the decision-making activity of the air pollution emission source managers. This helps to investigate the conditions and regulations that may enhance and maintain the air quality.

We used a two stage air pollution modelling method: a GPD dispersion model and an ANN forecasting model. The ANN predictor uses climatic parameters and dispersal information provided by the GPD model to make predictions. This helped to introduce the effect of uncertainty caused by the weather and made the simulation more realistic. Five cooperation strategies were tested. The centralized cooperation strategy (CS) showed the best performance, surpassing the reward/penalty strategies. However, the CS strategy needs an effective communication network between emission sources controllers and the regulation agency. Also, we assume that emission controllers communicate exactly their emission rate, which is not always the case. The reward/penalty strategies seem to be more realistic; penalising the polluting agents according to their participation during peak periods has a big influence on their behaviours. As shown in the simulation results, it helps reducing the pollution level and affects the evolution of the pollutant. Thus, air pollution regulations have a big impact on pushing the emission source controllers to take their polluting activity seriously; this is especially important during the peak periods where climatic conditions cause the pollutants to stagnate.

To summarise our study helps to: (1) Model and introduce human decision-making concerning emission sources and the process of simulating air pollution evolution. (2) Evaluate the possible cooperation between the actors concerned in managing the air quality. (3) Have a prediction about the efficiency of regulation rules for preserving the air quality. (4) Investigate the impact on air quality of the decision to expand or establish a new emission points.

Our work aims to provide a decision-making tool to the air pollution control agencies that will help them evaluate the regulations and policies concerning air pollution control. The current version of the system deals only with point emission sources. In future versions we aim to include line and area sources. Line sources model the road activities, whereas area sources model the waste management and agricultural sources of pollution. If data becomes available in future it could be interesting to experiment with different time representations. Fortunately the multi-agent system approach allows us to easily change to a different scale of time representation in the same simulation. We can envisage using one time representation for decisions and another for monitoring. The first can help us to see long term impacts (e.g. investing in less polluting activities), and the second can help to see the short term changes.

The simulator may also be enhanced by including topographic aspects of regions since this has a big influence on the dispersion of air pollutants. In addition, including more agent personalities and exploring other cooperation strategies are also among our future plans. Our system is designed in a generic way and it could be adapted for other types of pollution such as water pollution. This could be done by changing the current dispersion and the prediction models to a water pollution dispersion model.

6 References

[1] Folke, Carl. Resilience: The emergence of a perspective for social–ecological systems analyses. Global environmental change, 16.3, 2006. pp. 253-267.

[2] WHO (World Health Organisation), Ecosystems and Human Well-being: Health Synthesis . WHO Library Cataloguing-in-Publication Data, 2005.

[3] Kryza, M., Netzel, P., Drzeniecka-Osiadacz, A., Werner, M., & Dore, A. J. (2014). Forecasting the Ozone Concentrations with WRF and Artificial Neural Network Based System. In Air Pollution Modeling and its Application XXII Springer Netherlands, 2014, pp. 605-609.

[4] Holmes, NS and Morawska, L., A Review of Dispersion Modelling and its application to the dispersion of particles: An overview of different dispersion models available. Atmospheric Environment, 40(30), 2006. pp. 5902-5928.

[5] Lushi, Enkeleida, and John M. Stockie. An inverse Gaussian plume approach for estimating atmospheric pollutant emissions from multiple point sources. Atmospheric Environment, 44.8, 2010. pp. 1097-1107.

[6] Aulinas Montse, Turon Clàudia and Sànchez-Marrè Miquel. Agents as a Decision Support Tool in Environmental Processes:The State of the Art. Whitestein Series in Software Agent Technologies and Autonomic Computing. 2009. pp. 5–35.

[7] Ghazi, Sabri, Khadir, Tarek, Julie Dugdale. Multi-Agent Based Simulation of Environmental Pollution Issues: A Review. Highlights of Practical Applications of Heterogeneous Multi-Agent Systems. The PAAMS Collection. Springer International Publishing, 2014. pp. 13-21.

[8] Hülsmann, F., Gerike, R., Kickhöfer, B., Nagel, K., & Luz, R. . Towards a multi-agent based modeling approach for air pollutants in urban regions. In Proceedings of the Conference on “Luftqualität an Straßen, 2011. pp. 144-166.

[9] Kickhöfer, B., Hülsmann, F., Gerike, R., & Nagel, K. Rising car user costs: comparing aggregated and geo-spatial impacts on travel demand and air pollutant emissions. Smart Transport Networks: Decision Making, Sustainability and Market structure, NECTAR Series on Transportation and Communications Networks Research, 2012, pp.180-207.

[10] Kickhöfer, B., & Nagel, K. Towards high-resolution first-best air pollution tolls. Networks and Spatial Economics, 2013. pp. 1-24.

[11] Papaleonidas Andonis, Iliadis Lazaros. Hybrid and Reinforcement Multi Agent Technology for Real Time Air Pollution Monitoring. Artificial Intelligence Applications and Innovations IFIP Advances in Information and Communication Technology Volume 381, 2012, pp. 274-284.

[12] Ahat M., Ben Amor S., Bui M., Lamure M., Marie-Françoise Courel. Pollution Modeling and Simulation with Multi-Agent and Pretopology. Complex Sciences, Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering Volume 4, 2009, pp.225-231.

[13] Boyd, Robert, Richerson, Peter J. The evolution of reciprocity in sizable groups. Journal of Theoretical Biology, vol. 132, no 3, 1988, pp. 337-356.

[14] Szilagyi Miklos N., An Investigation of N-person Prisoners’ Dilemmas. Complex Systems, 14, 2003. pp. 155–174.

[15] Szilagyi Miklos N., Somogyi Iren, The Shape of Payoff Functions in N-person Games, Journal of Game Theory. 1(3), 2012,10-14 doi: 10.5923/j.jgt.20120103.02.

[16] Xianyu, Bo, and Ping Chen. Does Social Welfare Preference Always Promote Cooperation on Barabási and Albert Networks ?. Computational Economics, 37.3, 2011, pp. 249-266.

[17] Barabási, Albert-László, and Réka Albert. Emergence of scaling in random networks. science 286.5439, 1999: 509-512.

[18] Conrad Power, A Spatial Agent-Based Model of N-Person Prisoner's Dilemma Cooperation in a Socio-Geographic Community, Journal of Artificial Societies and Social Simulation vol. 12, no. 18, 2009, http://jasss.soc.surrey.ac.uk/12/1/8.html.

[19] Hardin, Garrett. The tragedy of the commons. Science, 162, 1968. pp. 1243-1248.

[20] Osang, Thomas, and Arundhati Nandy. Environmental regulation of polluting firms: Porter’s hypothesis revisited. Brazilian Journal of Business Economics. 3.3, 2003. pp. 128-148.

[21] Lange Andreas, , Vogt Carsten, , Cooperation in international environmental negotiations due to a preference for equity, Journal of Public Economics, Volume 87, Issues 9–10, September 2003, pp. 2049–2067.

[22] Touza, J., Drechsler, M., Smart, J. C., & Termansen, M. Emergence of cooperative behaviours in the management of mobile ecological resources. Environmental Modelling & Software, 45, 2013. pp. 52-63.

[23] De Vries, HJM Bert. Environmental Modelling. In : Principles of Environmental Sciences. Springer Netherlands, 2009. pp. 345-373.

[24] Stockie, John M. The mathematics of atmospheric dispersion modeling., Siam Review, 53.2, 2011. pp. 349-372.

[25] Hanna, Steven R., Gary A. Briggs, and Rayford P. Hosker Jr. Handbook on atmospheric diffusion. No. DOE/TIC-11223. National Oceanic and Atmospheric Administration, Oak Ridge, TN (USA). Atmospheric Turbulence and Diffusion Lab., 1982.

[26] User's guide for the industrial source complex (ISC3) dispersion models volume II -description of model algorithms, U.S. Environmental Protection Agency, 1995, http://www.epa.gov/scram001/userg/regmod/isc3v2.pdf (visited on 11/22/2014).

[27] Behera, Sailesh N. , Sharma, Mukesh. Degradation of SO2, NO2 and NH3 leading to formation of secondary inorganic aerosols: An environmental chamber study. Atmospheric environment, vol. 45, no 24, 2011 pp. 4015-4024.

[28] Ghazi, Sabri,, Khadir Med Tarek. Combination of artificial neural network models for air quality predictions for the region of Annaba, Algeria. International Journal of Environmental Studies, 69.1, 2012.pp. 79-89.

[29] Pelliccioni, Armando et Tirabassi, Tiziano. Air dispersion model and neural network: A new perspective for integrated models in the simulation of complex situations. Environmental modelling & software, vol. 21, no 4, (2006). pp. 539-546.

[30] Khedairia, S., Mohamed Khadir T.. Impact of clustered meteorological parameters on air pollutants concentrations in the region of Annaba, Algeria, Atmospheric Research 113, 2012. pp. 89-101.

[31] Chen, S., Cowan, C.F., and Grant, P. M., Orthogonal Least Squares Learning Algorithm for Radial Basis Function Networks, IEEE Transactions on Neural Networks, 2(2), 1991, 302-309.

[32] Li An, Modeling human decisions in coupled human and natural systems: Review of agent-based models, Ecological Modelling 229, 2012, 25– 36.

[33] Zhao-Han Sheng., Hou, Y. Z., Wang, X. L., & Du, J. G. ,The evolution of cooperation with memory, learning and dynamic preferential selection in spatial prisoner's dilemma game. In Journal of Physics: Conference Series, IOP Publishing, Vol. 96, No. 1, 2008, p. 012107.

[34] Tadjine, A., Djebar, H., & Courtois, A. Toxicité des poussières rejetées par le complexe sidérurgique d'Annaba sur quelques paramètres hématologiques du lapin Europeus. Environnement, Risques & Santé, 7(3), 2008, 209-215.

[35] Bellifemine, Fabio, POGGI, Agostino, et RIMASSA, Giovanni. JADE–A FIPA-compliant agent framework. In : Proceedings of PAAM. 1999. p. 33.

[36] Heaton, J. ,Encog: Library of Interchangeable Machine Learning Models for Java and C#. Journal of Machine Learning Research, 16, 2015., 1243-1247.

Fast Artificial Bee Colony for Clustering

Abba Suganda Girsang
Computer Science Department, BINUS Graduate Program - Master of Computer Science
Bina Nusantara University, Jakarta, Indonesia, 11480
E-mail: agirsang@binus.edu

Yohan Muliono and Fanny Fanny
Computer Science Department, School of Computer Science, Bina Nusantara University
Jakarta, Indonesia, 11480
E-mail: ymuliono@binus.edu, fanny.sa@binus.edu

Keywords: artificial bee colony, clustering, fast ABC, redundant process

Received: February 14, 2017

Artificial Bee Colony (ABC) is one of good heuristic intelligent algorithm to solve optimization problem including clustering. Generally, the heuristic algorithm will take the high computation time to solve optimization problem. Likewise, ABC also consumes too much time to solve clustering problem. This paper intends solving clustering problem using ABC with focusing reduction computation time called FABCC. This idea proposes detecting the pattern of redundant process then compacting it to effective process to diminish the computation process. There are five data sets to be used to prove the performance of FABCC. The results shows that FABCC is effective to prune the duration process up to 46.58 %.

Povzetek: Predstavljena je izboljšava v algoritmu Artificial Bee Colony za gručenje, ki dosega na merjenih domenah skoraj 50% pohitritev.

1 Introduction

Clustering is the unsupervised classification of patterns (observations, data items, or feature vectors) into groups (clusters). The clustering problem has been addressed in many contexts in many disciplines [1]. Yang and Kamel state as a machine learning perspective that clustering is unsupervised learning because no category labels denoting appropriate partition of the objects are used [2]. Generally, there are two types of data to be clustered; metric as a numerical data and non-metric as not a numerical data [3].

Clustering problem has a broad appeal as one of the steps in exploratory data analysis. Jain et al describe some important applications of clustering algorithms such as image segmentation, object recognition, and information retrieval [1].

Recently, population-based optimization based on behaviour of animal swarm often used as a solution to find an optimization problem such as travelling salesman problem [4]–[6] and clustering [7]–[11]. Artificial bee colony (ABC) algorithm is one of the most recently introduced swarm-based algorithms. ABC simulates the intelligent foraging behaviour of a honey bee swarm [8] to get the optimal solution. ABC has been proved as a good algorithm for solving clustering [8]. Their proposed method can cluster perfectly accurate for Cancer-Int, Iris and wine dataset. This algorithm also shows the good performance comparing with ten algorithms (PSO, BayesNet, MlpANN, RBF, KStar, Bagging, MultiBoost, NBTree, Ridor and VFI). However, compared to other evolutionary algorithms, ABC has a challenging problem. For example, the convergence speed of ABC is slower than the other representative of population-based algorithm [12]. Also, like the general evolutionary algorithm, ABC has many repetition computations before converging solution. However, some of researchers commonly depend on their algorithm to find a best solution only, nevertheless they forget about time needed to fully operate their algorithm. In that case, this research is conducted to focus on the time and try not to stray excess average best solution. Some researchers have attempt solving this problem. Girsang et al proposed BCOPR that is inspired bee colony optimization to gain the fast process to solve travelling salesman problem [13]. Lu et al also used bee swarm for fast clustering [14].This research conducts a fast algorithm using ABC algorithm to solve the cluster problem. The reason why this research focuses on artificial bee colony besides of recent success of clustering data in ABC as stated in [4] [7] [8], because ABC also have many slots to be modified as a fast algorithm. Karaboga used greedy algorithm to be applied in ABC [8] and Zhang [7] used Deb’s Algorithm [15] instead of greedy because Zhang believe that deb’s algorithm is much more simple.

The remainder of this paper is organized as follows: Section 2 gives a brief introduction to the clustering problem and artificial bee colony. Section 3 provides a detailed description of the FABCC algorithm, while the performance evaluation of the proposed algorithm is presented in Section 4. Finally, conclusions are offered in Section 5.

2 Related work

2.1 *K*-means Clustering

Clustering algorithm generally is classified into two big parts, first one known as hierarchical clustering and second one is partition clustering [16]. Hierarchical clustering group data objects with a sequence of partitions. Hierarchical procedures divided into two segments which are agglomerative and divisive. Where agglomerative approach begins with each pattern in distinct cluster (single cluster) and then will be merged later. Divisive pattern is vice versa, begin with a single cluster and will be divided later [1]. Partitional clustering algorithm obtains a single partition of data instead of a clustering structure. This technique usually produce clusters by optimizing a criterion function defined either locally or globally. This research will be using partition, because we already know the total of the clusters. And we start with the number of clusters without reducing or adding it.

Among such a varies clustering formulations that are based on minimizing a formal objective function, the most widely used and studied is k-means clustering [17]. Clustering based on k-means is closely related to a number of other clustering and location problems. These include the Euclidean k-medians in which the objective is to minimize the distance to the nearest centre of the centroid. The aim of the K-means algorithm is to divide M points in N dimensions into K clusters so that the within-cluster sum of squares is minimized. It is not practical to require that the solution has minimal sum of squares against all partitions, except when M, N are small and K = 2. We seek instead "local" optima, solutions such that no movement of a point from one cluster to another will reduce the within-cluster sum of squares[18]. This research will do a similar algorithm to k-means, modified the algorithm a little by adding greedy algorithm to swap the route of the clusters.

2.2 Artificial Bee Colony

Artificial Bee Colony (ABC) Algorithm is one of the swarm intelligence, whereas its copy the mechanism of honey bee swarm's intelligence to find the food source [19]. Originally, ABC optimization was proposed for solving numerical problems [20]. Therefore, the first studies aimed to evaluate the performance of ABC on the widely used set of numerical benchmark test functions and to compare it with that of well-known evolutionary algorithms such as Genetic Algorithm, Particle Swarm Optimization and Ant Colony Optimization. [19] There are 3 types of bee. The first one is employed bees which search for a food source. The food source value depends on many factors, such as its proximity to the nest, richness or concentration of energy, and the ease of extracting this energy [21]. Employed Bee will do a waggle dance later. The more the food source, the longer the waggle dance will last. The waggle dance represent the fitness value. Second is onlooker bees which wait employed bee to do waggle dance and choose randomly according to how much fitness value of the employed bees. The last one is scout bees which looking for the food source without pattern. The position of the food source represents a solution that can be made by the bees. And the amount of the nectar represents a better solution that can be found by the bees. Whereas in [19] Karaboga has proved that ABC can be used to optimize multivariable functions and ABC outperforms the other swarm intelligence algorithm such as Genetic Algorithm, Particle Swarm Algorithm and Particle Swarm Inspired Evolutionary Algorithm (PS-EA) [19]. The main steps of Artificial Bee Colony algorithm are:

The main steps of artificial bee colony algorithm are:

Step 1: Initialize the population of solutions randomly and evaluate them.

Step 2: Produce new solutions for each employed bees, evaluate them, and apply the greedy solutions for them and the greedy selection process.

Step 3: Calculate the probabilities of current sources (employed bees) with which they are preferred by the onlookers.

Step 4: Assign onlooker bees to employed bees according to probabilities.

Step 5: Produce new solutions for each onlooker bees, evaluate them, and apply the greedy selection process.

Step 6: Stop the exploitation process of the sources abandoned by bees and send the scouts in the search area for discovering new food sources, randomly.

Step 7: Memorize the best food source found so far.

Step 8: If the termination condition is still not met, repeat the algorithm process from **Step 2**, otherwise stop the algorithm.

As the development of Karaboga's artificial bee algorithm, Zhang contributes some additional step in Karaboga's algorithm, so it can be used for solve clustering problem. Zhang appends control parameter to his algorithm and he also have some different step with Karaboga's algorithm. Zhang adds the control parameter to scout phase which called upper bound that uses as the limit of scout number and in his algorithm, the scout phase will be different with employed bees phase. In Karaboga's algorithm, the scout only finds once the random food source and act like employed bees as well. However in Zhang's algorithm, the scout will act as scout that always find the new food source randomly as long as it is scout. The scout will only change into employed bees if the limit of scout number is reached, where the worst bee will still be scout, and the others will be employed bees.

The steps of Zhang's artificial bee colony algorithm for clustering are:

Step 1: Initialize the population of solutions and its control parameter. Order the first half of colony consists of the employed bees and the second half includes onlooker bees. Generate random position for each employed bees and evaluate it. Set scout number to zero.

Step 2: If the number of scouts is more than its upper bound, order the first half of colony, make the bees with worst solution quality as scouts and others as employed bees. Update the scout number.

Step 3: Produce new solutions for each employed bees, evaluate them, and apply the Deb's selection process. If the limit for abandonment is reached, the employed bee forgets its memory and become a scout. The scout number is adding by

Step 4: Send each scout into the search area for discovering new food sources randomly. When a new food is found, evaluate it, and apply the Deb's selection process.

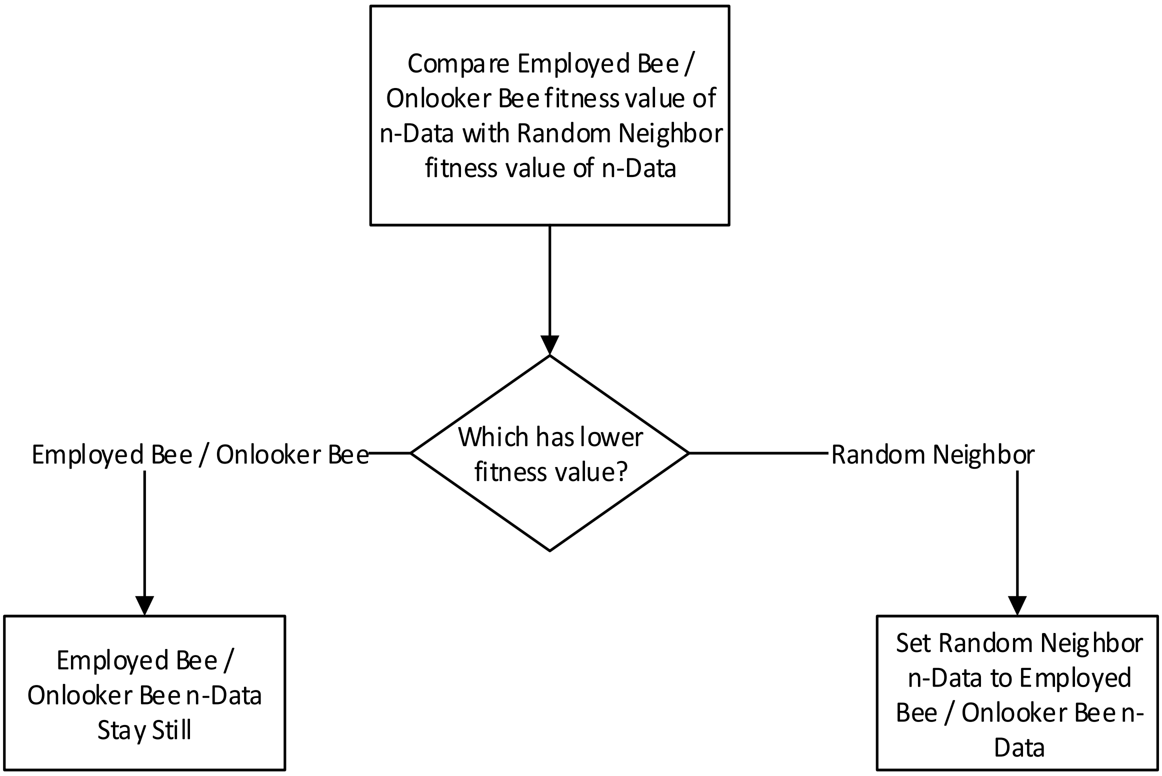

Compare Employed Bee / Onlooker Bee fitness value of n-Data with Random Neighbor fitness value of n-Data
Which has lower fitness value?
Employed Bee / Onlooker Bee
Random Neighbor
Employed Bee / Onlooker Bee n-Data Stay Still
Set Random Neighbor n-Data to Employed Bee / Onlooker Bee n-Data

Figure 1: Greedy Selection Process.

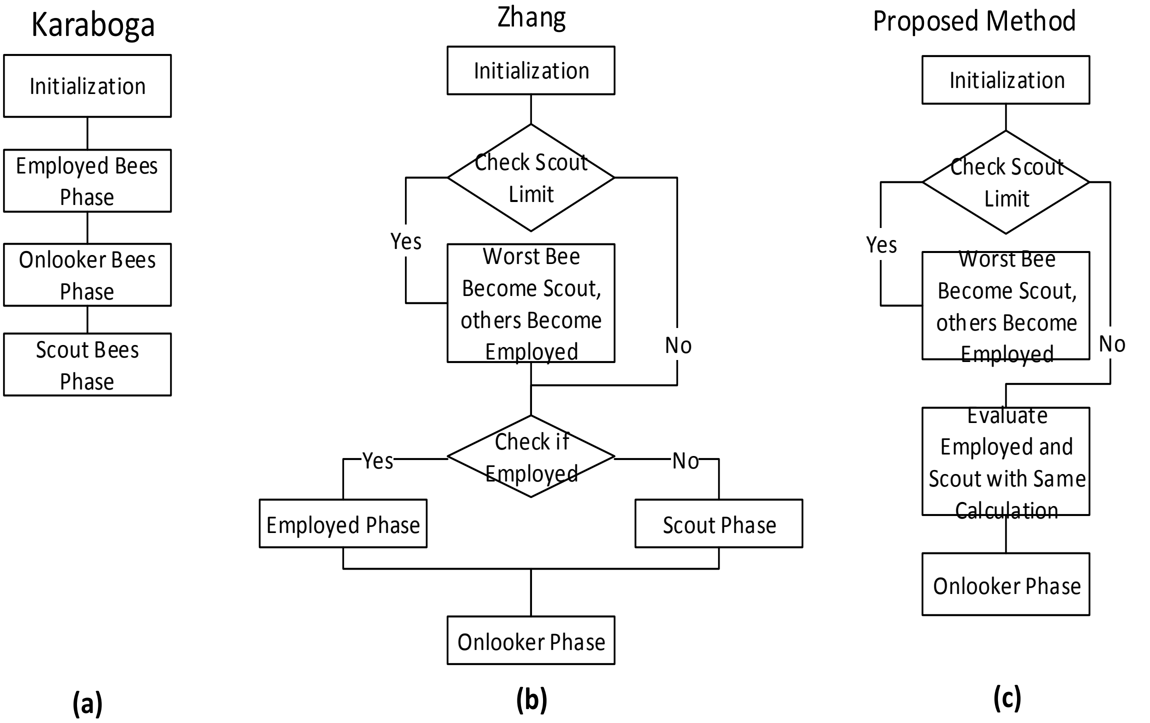


Figure 2: Comparison of ABC algorithm between (a) Karaboga's ABC (b) Zhang's ABC for clustering and (c) proposed method.

Step 5: Calculate the probability value of the current food sources with which employed bees are preferred by the onlookers.

Step 6: Produce new solutions for each onlooker bees, evaluate them, and apply the Deb's selection process to update the corresponding employed bee's memory or the current food sources.

Step 7: For each employed bee and scout, if its memorized position is better than the previous achieved best position, then the best position is replaced by it. If the termination condition is still not met, repeat the algorithm process from **Step 2**, otherwise stop the algorithm.

3 Proposed method

3.1 The concept

This section firstly is described the combination of Karaboga [8] and Zhang [7] algorithm. Most of steps uses the Zhang's algorithm, except selection process uses Karaboga's algorithm with greedy algorithm.

The greedy algorithm is the key to find the best solution. So, for the next experiment, the centroid or food source calculation is reconstructed, after the greedy algorithm has been done, whereas the employed bees compared its food source to another bee's food source as shown on Fig.1.

Besides of the greedy algorithm, the proposed metod, FABCC, also combines Zhang's algorithm and Karaboga's algorithm in scout phase. Unlike Karaboga's algorithm, bee on FABCC mimics Zhang's algorithm that the sequence process of bee is employed bee, scout bee, and then onlooker bee as shown in Fig. 2. However FABCC adopts Karaboga's algorithm calculation to determine fitness value.

3.2 ABCC and FABCC

From the literature studies in Section 2, it mentions there are three bees in this algorithm, consisting of employed

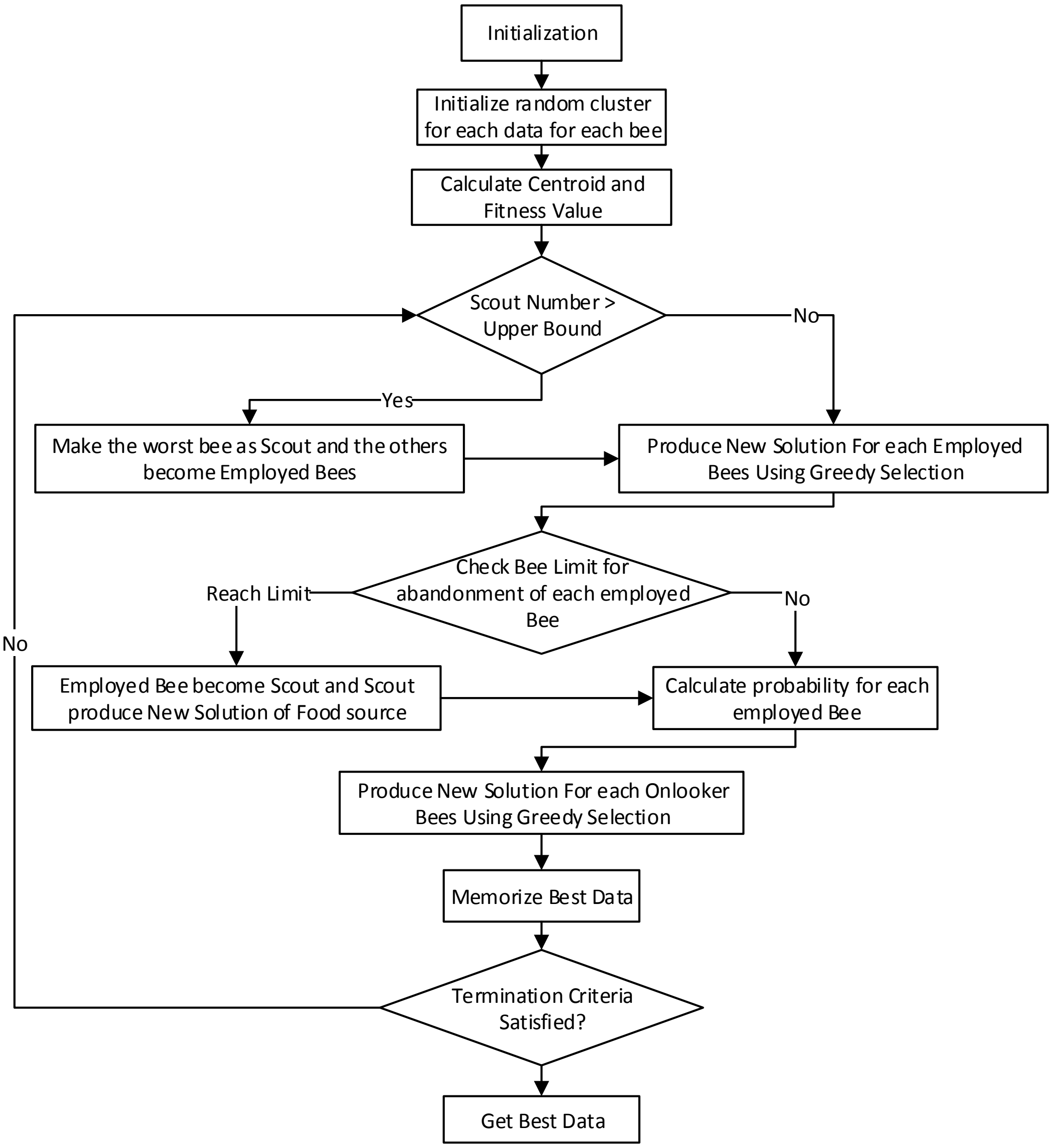

Initialization
Initialize random cluster for each data for each bee
Calculate Centroid and Fitness Value
Scout Number > Upper Bound
No
Yes
Make the worst bee as Scout and the others become Employed Bees
Produce New Solution For each Employed Bees Using Greedy Selection
Check Bee Limit for abandonment of each employed Bee
Reach Limit
No
Employed Bee become Scout and Scout produce New Solution of Food source
Calculate probability for each employed Bee
Produce New Solution For each Onlooker Bees Using Greedy Selection
Memorize Best Data
Termination Criteria Satisfied?
No
Get Best Data

Figure 3: ABC for clustering algorithm flowchart.

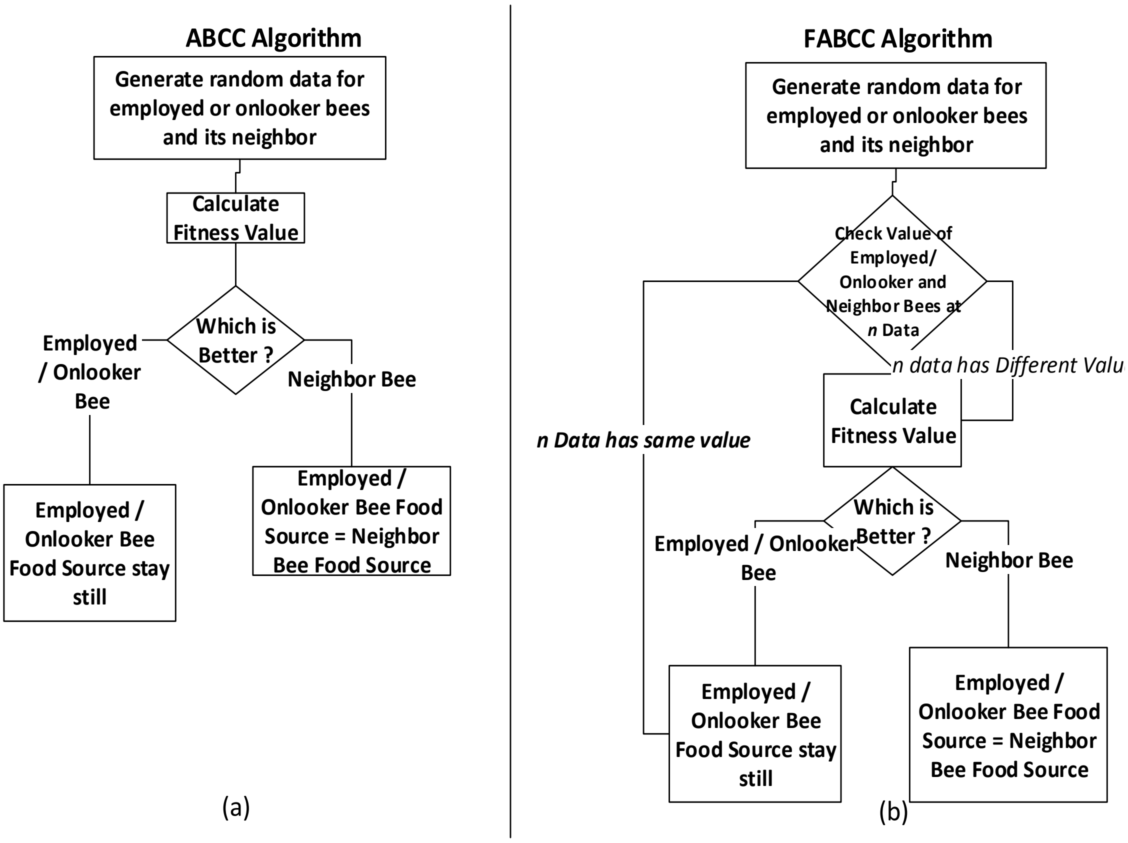

ABCC Algorithm
Generate random data for employed or onlooker bees and its neighbor
Calculate Fitness Value
Which is Better ?
Employed / Onlooker Bee
Neighbor Bee
Employed / Onlooker Bee Food Source stay still
Employed / Onlooker Bee Food Source = Neighbor Bee Food Source
(a)
FABCC Algorithm
Generate random data for employed or onlooker bees and its neighbor
Check Value of Employed/ Onlooker and Neighbor Bees at n Data
n data has Different Valu
n Data has same value
Calculate Fitness Value
Which is Better ?
Employed / Onlooker Bee
Neighbor Bee
Employed / Onlooker Bee Food Source stay still
Employed / Onlooker Bee Food Source = Neighbor Bee Food Source
(b)

Figure 4: Comparison of employed bee / onlooker bee phase between (a) ABCC phase and (b) FABCC phase.

bee, onlooker bee, and scout bee. Which are each of them has their own fitness value. The process of searching the best fitness value consisting of eight steps and the flowchart is shown in Fig. 3.

Step 1: Initialize the population of solutions and its control parameter. Order the first half of colony consists of the employed bees and the second half includes onlooker bees. Generate random position for each employed bees and evaluate it. Set scout number to zero.

Step 2: If the number of scouts is more than its upper bound, order the first half of colony, make the bees with worst solution quality as scouts and others as employed bees. Update the scout number.

Step 3: Produce new solutions for each employed bees, evaluate them, and apply the greedy selection process.

Step 4: If the limit for abandonment is reached, the employed bee forgets its memory and become a scout for discover a search space and get the new food source randomly, and scout act like employed bees. The scout number is adding by one.

Step 5: Calculate the probability value of the current food sources with which employed bees are preferred by the onlookers.

Step 6: Produce new solutions for each onlooker bees, evaluate them, and apply the greedy selection process.

Step 7: Compare employed bees and onlooker bees which have same food source and save the best quality for each food source.

Step 8: Memorize the best food source quality overall. If the termination condition is still not met, repeat the algorithm process from **Step 2**, otherwise stop the algorithm.

In the experiment using ABCC Algorithm for big data that needs many iteration, ABCC takes too much computation time. This section explained a proposed method for Fast Artificial Bee algorithm to reduce the computation time. In Artificial Bee Colony algorithm, employed bees and onlooker evaluation phase. The comparison of employed or onlooker bee and its random neighbor is compared from their fitness value regardless the compared data is similar or not (Fig. 4.a). However in FABCC algorithm, one step is added for check the compared data. If the compared data of employed or onlooker bee and its neighbor is similar, the calculation of fitness value is skipped and the bee's data still same with the current data (Fig. 4.b).

4 Experimental results

The parameters used in fast ABC for clustering (FABCC) are shown in Table 1. The description of those parameters is as follows.

a. The number of bee is 20 which is grouped into 3 types of bee.

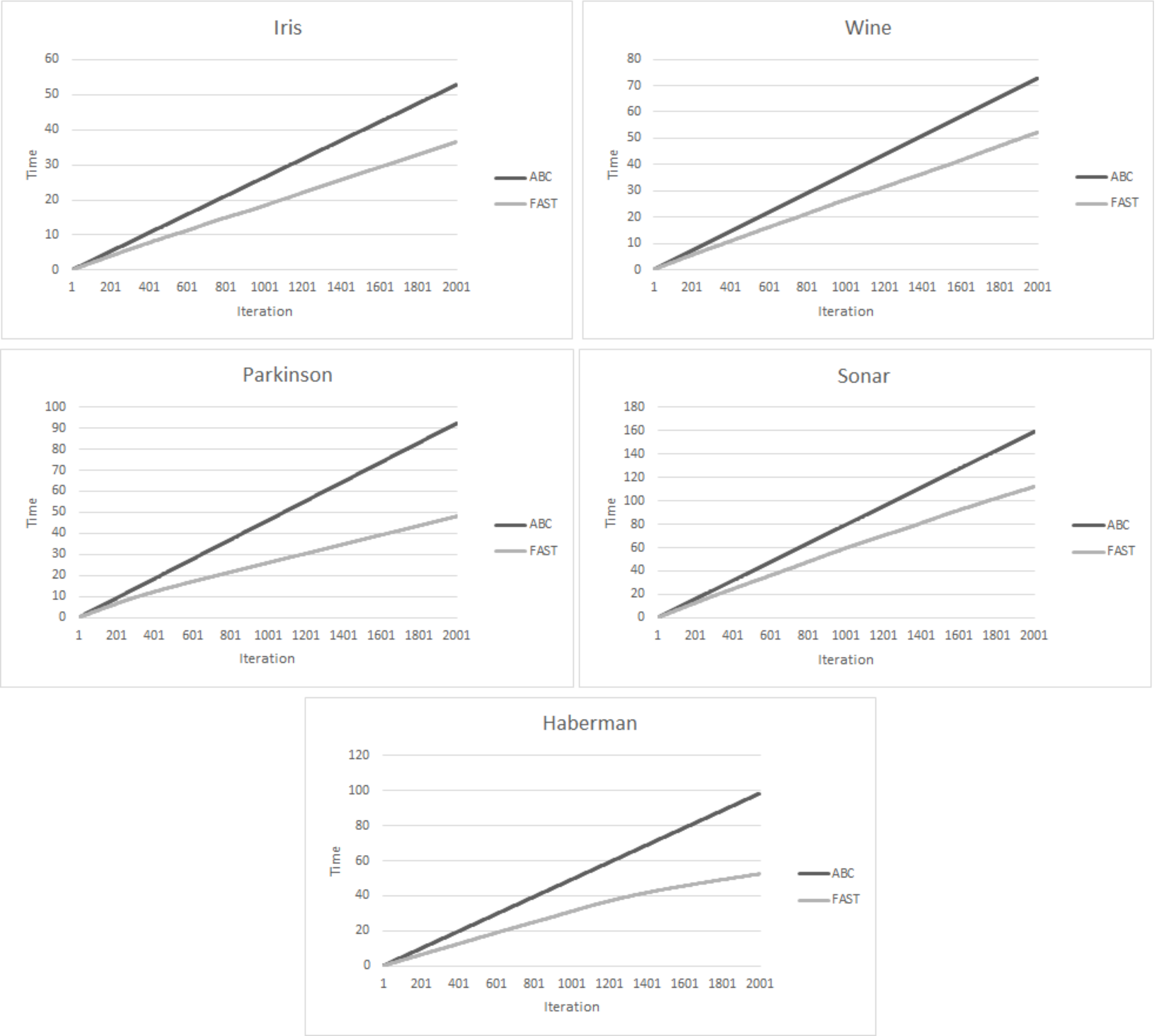


Figure 5: Duration time process of ABCC and FABCC.

b. The number of initial employed bee and the number of onlooker are 10.
c. The chance of searching better condition for employed bees means that if employed bees failed to gain better solution after 100 times consecutively, it will leave its pattern and become a scout bee to search another pattern.
d. The maximum cycle of number selected is 2000

Parameter	Value
Total Bees	20
Employed Bees	10
Scout	Up to 5
Onlooker Bees	10
Limit for Abandonment	100
Maximum Cycle Number	2000

Table 1: Parameter used for experiment.

This research uses several data sets to be evaluated as shown in Table 2. This research focuses in two aspects, quality and processing time. The quality of the program can be evalueted from the result of fitness value.

Fig. 5 shows the different computation time of original algorithm for clustering (ABCC) and modified algorithm for clustering (FABCC) for five data sets. In every data sets, the figures show that the differences of ABCC and FABCC starting in 300th iteration. It means there is no siginifance 1-200 iterations From 1st iteration computation time.

Data Set	Number of Patterns	Number of Clusters	Number of Attributes
Iris	150	3	4
Wine	178	3	13
Haberman	306	2	3
Connectionist Bench (Sonar)	208	2	60
Parkinson	195	2	22

Table 2: Data sets.

It can be divergent because in beginning the bees still generate some various pattern to be learned. After through some iterations, the preferred pattern will be created. Bees learn from the previous pattern.The pattern which has same with previous patterns in many times is identified as the repition process. Therefore, it can be pointed as the pruned pattern to prevent the redundant computation. For more detail, Fig. 6 shows that the computation time in FABCC tends to decrease in each iteration after several

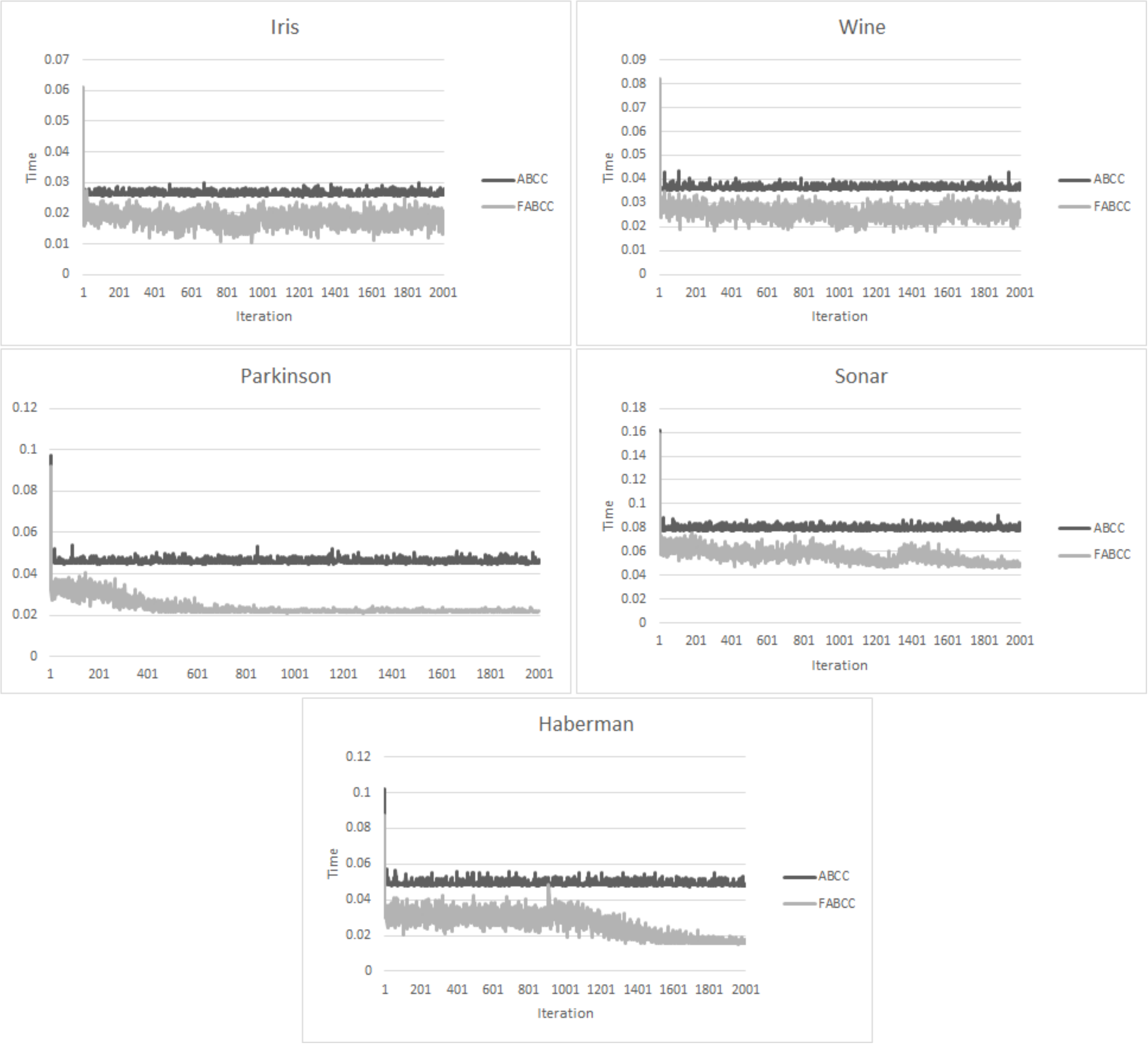


Figure 6: Computation time per iteration for ABCC and FABCC.

iteration while computation time in ABCC constant since the first iteration. In FABCC, the computation time could decrease because the bee in FABCC will learn the pattern of the data. When the algorithm generates patterns, bee will truncate some calculation. Therfore the computation time decrease after many iterations.

The results of parametes used in Table 1 are shown in Fig.7 and Table 3. Fig. 7 shows the convergent ABCC and FABCC. They start convergent after about 800 iteration. Table 3 shows the results of proposed algortithm, FABCC which is compared original algorithm, ABCC algorithm. The results are categorized two parts, computation time and quality (fitness value). This evaluations are run 30 times for each parts. Each of parts consists some test statistics such as mean, min, max, and standard deviation (SD). Min is considered as best solution while max is considered as the worst solution. All standard deviation (SD) of the results are too small comparing mean (less than 1 %). The small deviation indicates is almost same to the expected value. This study also conducted the Wilcoxon signed-rank test. The wilcoxon test is used to analyze the results of paired observations from two data (in this case results ABCC and FABCC) are different or not. The bound significant (α) is used less than 0.01. If *Asymp. Sig* $< \alpha$, it indicated that these two related samples (FABCC and ABCC) are different significantly. Table 3 also shows that the difference of ABCC's and FABCC's fitness value are not significant. In best case, FABCC can achive as the rescults of ABCC in all of data sets. The FABCC is only a little outperform ABCC in mean, and worst value. The differences are only is less than 1 %. However the computation time in FABCC can be decrease significant comparing the ABCC. They can be different about 30-50 %. This means FABCC can be applied to reduce computation time as the problem of ABC as one of the heuristic algorithm. Table 3 also shows that the difference of ABCC's and FABCC's fitness value are not significant. In best case, FABCC can achive as the rescults of ABCC in all of data sets. The FABCC is only a little outperform ABCC in mean, and worst value. The differences are only is less than 1 %. However the computation time in FABCC can be decrease significant comparing the ABCC. They can be different about 30-50%. This means FABCC can be applied to reduce computation time as the problem of ABC as one of the heuristic algorithm.

Data Set	ABCC (Computation Time)				FABCC (Computation Time)				Asymp. Sig. (2-tailed) ABCC-FABCC
	Min (best)	Max (worst)	Mean	SD	Min (best)	Max (worst)	Mean	SD	
Iris	53.76	54.01	53.94	0.001	37.08	38.21	37.62	0.001	0.002
Wine	74.57	75.23	74.80	0.002	52.18	56.17	54.56	0.002	0.003
Haberman	100.76	101.2	100.91	0.002	46.42	59.22	53.91	0.012	0.005
Sonar	159.99	160.35	160.11	0.001	111.78	126.83	118.15	0.011	0.009
Parkinson	93.76	95.64	94.05	0.002	48.54	63.95	56.05	0.009	0.007
	ABCC (Fitness Value)				FABCC (Fitness Value)				
Iris	78.94	78.94	78.94	0.000	78.94	79.11	79.02	0.001	0.001
Wine	2370700	2370700	2370700	0.000	2370700	2370700	2370700	0.000	0.000
Haberman	30507	30507	30507	0.000	30507	30524	30513.4	0.004	0.002
Sonar	280.53	280.61	280.57	0.001	280.53	280.71	280.62	0.002	0.000
Parkinson	1343400	1343400	1343400	0.000	1343400	1343981	1343710	0.008	0.001

Table 3: Results of experiment.

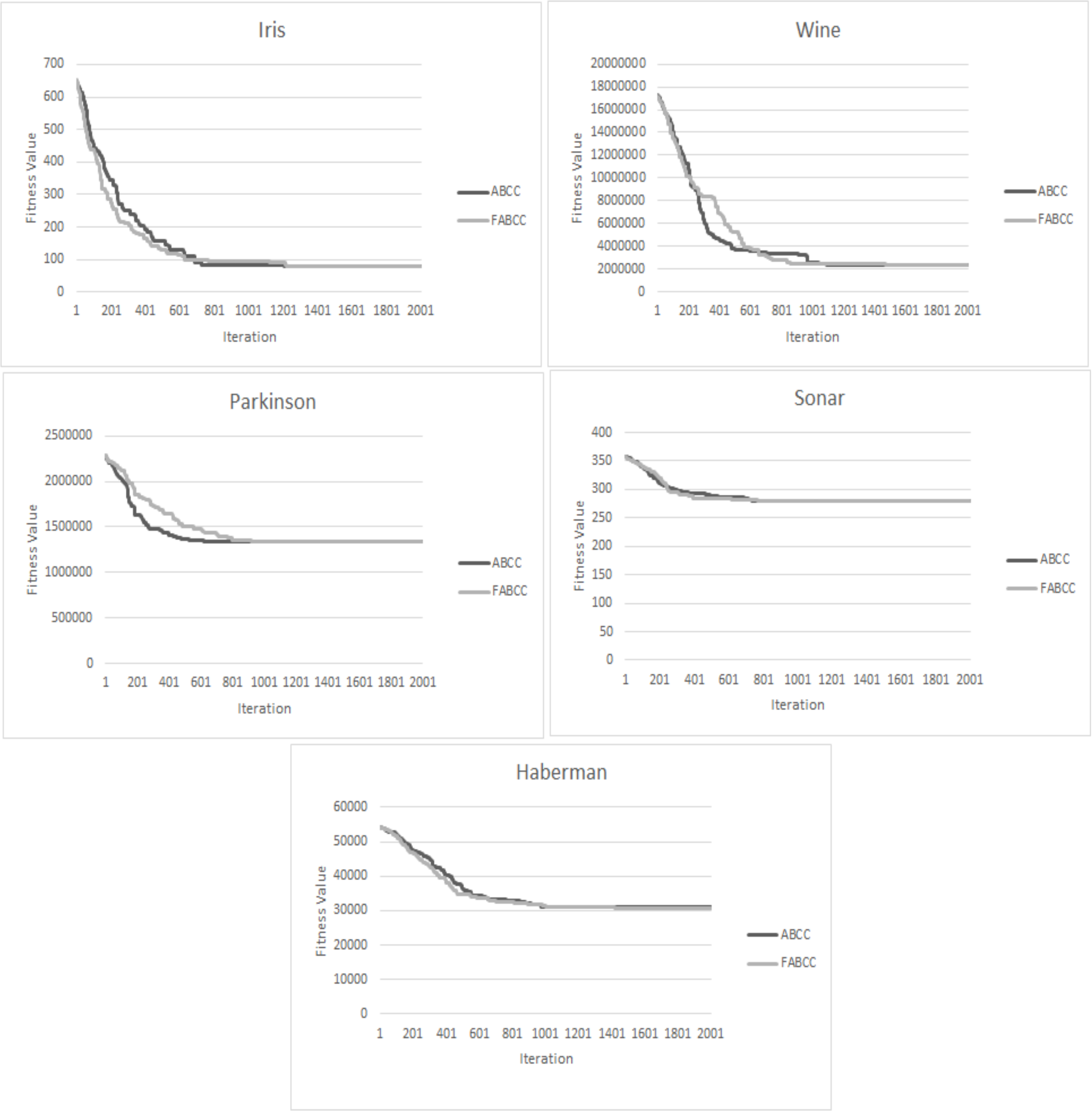


Figure 7: Fitness value of ABCC and FABCC.

5 Conclusion

This research uses bee colony algorithm by combining Zhangs and Karaboga algorithm. The choice of ABC algorithm sequence is based on Zhang and fitness value calculation is based on the original Karaboga. This proposed method, FABCC, also concludes that some redundant process occurs on bee colony algorithm for clustering. The redundant process identifies as a pattern that is able to compress. The result shows FABCC is effective to reduce computation time. It can be proved by conducts five datasets Iris, Wine, Haberman, Sonar, and Parkinson. The results shows that it can reduce 30-50% of computation time, while the fitness value only reduce less than 1%.

This study focuses on only small five datasets for clustering. It can be extended using the other big datasets. It might be have the different characteristic. However in our exploration, the redundant process always occurs in most of metaheuristic algorithm. In next research, researchers can put their effort to remake calculation of fitness value, in calculating in fitness value there are many iterations and redundant calculation to be observed to prune the redundant pattern.

6 References

[1] a. K. Jain, M. N. Murty, and P. J. Flynn, “Data clustering: a review,” *ACM Comput. Surv.*, vol. 31, no. 3, pp. 264–323, 1999.

[2] Y. Yang and M. Kamel, “Clustering ensemble using swarm intelligence,” *Swarm Intell. Symp. 2003. SIS ’03. Proc. 2003 IEEE*, pp. 65–71, 2003.

[3] T. Zhang, R. Ramakrishnan, and M. Livny, “BIRCH: An Efficient Data Clustering Method for Very Large Databases,” *Proc. 1996 ACM SIGMOD Int. Conf. Manag. Data*, vol. 1, pp. 103–114, 1996.

[4] B. Akay and D. Karaboga, “A modified Artificial Bee Colony algorithm for real-parameter optimization,” *Inf. Sci. (Ny).*, vol. 192, pp. 120–142, 2012.

[5] A. Ouaarab, B. Ahiod, and X.-S. Yang, “Discrete cuckoo search algorithm for the travelling salesman problem,” *Neural Comput. Appl.*, vol. 24, no. 7–8, pp. 1659–1669, 2013.

[6] S.-M. Chen and C.-Y. Chien, “Solving the traveling salesman problem based on the genetic simulated annealing ant colony system with particle swarm optimization techniques,” *Expert Syst. Appl.*, vol. 38, no. 12, pp. 14439–14450, 2011.

[7] C. Zhang, D. Ouyang, and J. Ning, “An artificial bee colony approach for clustering,” *Expert Syst. Appl.*, vol. 37, no. 7, pp. 4761–4767, 2010.

[8] D. Karaboga and C. Ozturk, “A novel clustering approach: Artificial Bee Colony (ABC) algorithm,” *Appl. Soft Comput.*, vol. 11, no. 1, pp. 652–657, 2011.

[9] S. Goel, A. Sharma, and P. Bedi, “Cuckoo Search Clustering Algorithm: A novel strategy of biomimicry,” *2011 World Congr. Inf. Commun. Technol.*, pp. 916–921, 2011.

[10] S. Rana, S. Jasola, and R. Kumar, “A review on particle swarm optimization algorithms and their applications to data clustering,” *Artif. Intell. Rev.*, vol. 35, no. 3, pp. 211–222, 2011.

[11] C.-L. Huang, W.-C. Huang, H.-Y. Chang, Y.-C. Yeh, and C.-Y. Tsai, “Hybridization strategies for continuous ant colony optimization and particle swarm optimization applied to data clustering,” *Appl. Soft Comput.*, vol. 13, no. 9, pp. 3864–3872, 2013.

[12] R. Storn and K. Price, “Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces,” *J. Glob. Optim.*, pp. 341–359, 1997.

[13] A. S. Girsang, C.-W. Tsai, and C.-S. Yang, “A Fast Bee Colony Optimization for Traveling Salesman Problem,” *2012 Third Int. Conf. Innov. Bio-Inspired Comput. Appl.*, vol. 1, no. c, pp. 7–12, 2012.

[14] Y. Lu, S. Lu, F. Fotouhi, Y. Deng, and S. Brown, “FGKA: A fast genetic k-means clustering algorithm,” *Proc. 2004 ACM …*, pp. 1–2, 2004.

[15] K. D. David E. Goldberg, “A comparative analysis of selection schemes used in genetic algorithms.”

[16] J. Han and M. Kamber, “Data Mining: Concepts and Techniques,” *Ann. Phys. (N. Y).*, vol. 54, p. 770, 2006.

[17] T. Kanungo, D. M. Mount, N. S. Netanyahu, C. D. Piatko, R. Silverman, and a. Y. Wu, “An efficient k-means clustering algorithm: analysis and implementation,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, no. 7, pp. 881–892, 2002.

[18] J. a Hartigan and M. a Wong, “Algorithm AS 136: A k-means clustering algorithm,” *Appl. Stat.*, vol. 28, no. 1, pp. 100–108, 1979.

[19] D. Karaboga and B. Basturk, “A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm,” *J. Glob. Optim.*, vol. 39, no. 3, pp. 459–471, 2007.

[20] D. Karaboga, B. Gorkemli, C. Ozturk, and N. Karaboga, “A comprehensive survey: artificial bee colony (ABC) algorithm and applications,” *Artif. Intell. Rev.*, vol. 42, no. 1, pp. 21–57, 2014.

[21] D. Karaboga and B. Basturk, “On the performance of artificial bee colony (ABC) algorithm,” *Appl. Soft Comput.*, vol. 8, no. 1, pp. 687–697, 2008.

A Pairing Free Secure Identity-based Aggregate Signature Scheme under Random Oracle

Eman Abouelkheir
Electrical Department, Faculty of Engineering, Kafr Elsheikh University, Egypt
E-mail: emanabouelkhair@eng.kfs.edu.eg

Jolanda G. Tromp
Department of Computer Science, State University of New York Oswego, USA
E-mail: jolanda.tromp@oswego.edu

Keywords: information security, aggregate signature, without parings, security proof, random oracle

Received: November 15, 2016

The signature aggregation is efficient for the communication links as the time complexity is independent of n different users. The bilinear pairing requires super-singular elliptic curve groups that have a spacious range of elements. Also, the point multiplication over elliptic curve is less computational cost than the pairings, therefore, the pairing-based schemes expose more computational complexity than schemes that without pairings. This paper introduces a new efficient and secure pairing free signature scheme based on the idea of aggregation. Also, the proposed scheme without pairings offers lower computational cost than other schemes from pairings as it saves 68.69% from computations.

Povzetek: Ta prispevek se ukvarja s kriptografskimi algoritmi, konkretno s shemo digitalnih podpisov. Opisana je izboljšava obstoječega algoritma, ki dosega pohitritev za dve tretjini, hkrati ostaja varna.

1 Introduction

Cryptography has two primitive issues; confidentiality and authentication. Digital signature achieves the authentication issue. Also, for efficient communication links, schemes should provide low time complexity. Moreover, low time complexity is useful for battery and bandwidth saving of the channel in networks [1].

There are many cryptographic algorithms provide privacy, such as signature schemes, authentication schemes, and encryption schemes [2]. For providing privacy and anonymity to a user, these schemes have to be properly combined. Schemes and methods such as group signature schemes, blind signatures, aggregate signatures, zero-knowledge proof methods, homomorphic encryption schemes offer several useful privacy-enhancing properties, e.g. identity hiding, binding information, data confidentiality, unlinkability, intracebility, etc. Recently, many applications and services require privacy protection over communication systems. The current secure communication systems support authentication, data integrity, and non-repudiation. But, the communication systems users and providers can need different security properties that are out of basic security properties. These advanced properties are usually connected with user privacy. The following text summarizes the advanced security properties and requirements.

• Privacy/Anonymity - privacy protection is ensured for every user in the system who follows the rules. Users can communicate anonymously. Their identities can be revealed only in special cases, e.g. when a user breaks a rule, authority order, police order, emergency events etc. Two types of privacy protection can be distinguished: a basic anonymity to protect a user identity against passive attacks and a full anonymity to protect also against active attacks [3]. Signatures needed when an attacker gets access to all old messages. When the unlinkability property ensured, then the attacker is not able to connect certain signatures together.

- Responsibility/revocation - every user, has to be revealed and revoked using the certain key when breaking the rules of a system. The revocation assures that the revoked user has no rights in whole systems afterward. The revocation helps protect the system against repeated misusing. In some applications, the traceability of malicious users' messages is demanded.
- Efficient and secure key management - key exchange, revocation, and establishment in systems have to be efficient computationally/memory low cost and secure. In privacy-preserving solutions, key management has to keep user privacy.
- Efficient and secure execution of cryptographic protocols - the phases of a cryptographic protocol should be as efficient as possible to minimize the negative influence of a system, especially, if the restricted devices have been deployed.
- Exculpability - neither revocation or key manager, can be able to generate a valid signature behalf another user who hold trace keys. The user cannot be accused that makes signature which he does not make. This property is mainly needed in group signature schemes,

i.e exculpability that is ensured in [4] by Boneh, Boyen and Shacham.

The aggregate signature idea arises from those different signers aggregated into a concise aggregate signature on different documents[5]. Using the aggregate signatures instead of using n signature by n different users in many application such as key distribution in PKI reducing the communication overhead and offer efficient computational cost.

Another example, in router securing system, each router need to sign its part of a route in the communication link then transmits all the signatures to the following router. Without the aggregation concept, transmitting the different signatures exposes high communication overhead[6]. The aggregate signature could be used instead of individual signs for this goal. Recently, there are two signature schemes are proposed. The first one [5] provides flexible aggregation based on pairings. The second [7] provides only sequential aggregation using certified trapdoor permutations. For the schemes in [5,7], the authors proposed aggregates signature schemes which size is independent of n users. Specifying individual signers by some public information needed for public verification. Aggregate signature schemes that specify the signers with their public information may be similar as the traditional signatures and both are not efficient. Thus, specifying the signers with their identities is more useful than specifying them by their public keys.

Cryptography from pairings has many prime properties. It is supposed that Pairing-based cryptography with smaller parameters can present a desired security level as the general elliptic curve cryptography. Suppose that there is an elliptic curve E has elements defined over F_q. But the pairing-based cryptography is working with the functions and elements defined over F_q^k, where k is some random and chosen to be secure. Either the elliptic curve hard problem (ECDLP) defined over $E(F_q)$ or the discrete logarithm problem (DLP) defined over F_q^{k*} are basic problems that the cryptography from pairings security depends on [8]. Because of the previous clarification, this paper introduces a new pairing-free signature aggregation scheme based on the general elliptic curve cryptosystem depends on signers identities.

The idea behind the identity-based cryptography (IBC) [9], to use some information belongs to a signer (such as an email ID) as a user public key rather than using public-key and certificate management. Therefore, the IBC system requires Private Key Generator (PKG) that is called a trusted third party, that generates the private keys for all identities based on its master key and the signer identity. Boneh and Franklin [10] and Cocks [11] propose many identity-based encryption schemes. Also, old schemes in [9,12,13]; recent schemes and analyses include [14,15,16,17].

The concept of signature aggregation allows different signers to sign different messages. This leads to efficient communication and fewer computations. In any aggregate signature scheme n different signatures are considered as one single signature. The aggregation approach can be used instead of using public key certificates to satisfy efficient communication and computations. Aggregate signatures have many applications such as mutual authentication between vehicles in VANETs and in wireless sensor network.

The goal of my paper to introduce a new secure pairing-free aggregate signature scheme. The proposed scheme security is proven in the random oracle and assuming a hard Diffie-Hellman problem. Also, the proposed scheme saves the time complexity by 68.69% . The new aggregate signature and its analysis is the modified version of the scheme in [18].

The rest of the paper is organized as follow : section two presents the digital signature approach versus the water marking. Also, section three describes preliminaries. Then section four introduces the generic model for the proposed scheme. Moreover, section five presents the security requirements of any aggregate signatures based on user's identities. In section six and seven, the proposed scheme is presented with the formal security proof under random oracle respectively. In section eight, the results and discussion are introduced. The proposed scheme is compared with other in literature in section nine. Section ten concludes the proposed work. Finally, the future scope introduced in section eleven.

2 Digital signing versus watermarking

A digital signing is an approach of cryptography used for securing the communication links. The goal of the signature to verify the end to end communication system users.

The digital signing operation is similar to the handwritten signing operation and exactly as a paper signature. It used to verify the identity of a user using its digital certificate. This paper is concerned with the digital signature approach.

The goal of watermarking to hide the information into a digital signal that provides a copyright protection in a digital format[19]. Many watermarking schemes have been proposed. In 2012, Nilanjan Dey, Poulami Das, Sheila Sinha Chaudhuri, and Achintya Das, [20] used Alattar's method efficiently for watermark insertion and extraction for an EEG signal. In 2013, K. P. Arijit, D. Nilanjan, S. Sourav, D. Achintya, and S. Ch. Sheli [21] proposed a new technique for reversible watermarking is used for the color image .In 2014, Nilanjan Dey, Goutam Dey, Sheli Sinha Chaudhuri, and Sayn Chakraborty [22] proposed two novel blind-watermarking mechanisms are; 1- session key based blind-watermarking mechanism and 2- self-recovery based blind-watermarking mechanism, into the Electromyogram (EMG) signal. In 2015, Nilanjan Dey, Monalisa Dey , Sainik Kumar Mahanta ,and Achintya Das [23] proposed a technique is to prevent any modifications in a transmitted biomedical ECG signal. In 2016, Y. B. Amar, I. Trabelsi, D. Nilanjan and S.

Bouhlel [24] proposed watermarking scheme used for copyright protection purposes.

3 Preliminary

3.1 Bilinear pairing

Suppose G_1 is a cyclic group has an order q, q is prime. This group is generated by the point P over an elliptic curve E and defined over the prime field F_q. Let $\hat{e}$ be a pairing where $\hat{e}: G \times G \to G_T$. For any P, Q, R (points over an elliptic curve E) and $c, d \in F_q$, c, d are integers. The pairings satisfy the following properties:

- linearity: $\hat{e}(cP, dQ) = \hat{e}(P.Q)^{cd}$.
- NonDegenerate: $\hat{e}(P.P) \neq 1$.
- Easy to compute : $\hat{e}(P.Q)$ it must be easy and efficient computed.

3.2 Elliptic Curve Cryptography(ECC)

ECC is an public key cryptography approach based on the mathematics of elliptic curves. ECC is faster than RSA and uses smaller keys, but still, provides the same level of security .

Suppose $E_q(a,b)$ are the set of points over the elliptic curve E that defined over the prime field F_q, E defined by $y^2 \bmod q = (x^3 + ax + b) \bmod q$ and a, b must satisfy the equation $\Delta = (4a^3 + 27b^2) \bmod q \neq 0$. The cyclic group $G_q = \{(x,y): x, y \in F_q \quad \}$ $(x,y) \in E / F_q$, G_q is an elliptic curve additive group. The group identity element in G_q is O ; the infinity point; The scalar multiplication on G_q defined as $k.P = P + .. + ... + P + P$. For some integer $n > 0$, a point P of order n satisfy $n{\cdot}P = O$. ECC was proposed in 1985, by Miller [25] and Koblitz [26]. When comparing ECC with other public key cryptosystems, it was found that ECC-based public key cryptosystem has many advantages such as low computation cost, smaller key size, low storage space cost etc. It is known that the discrete logarithm problem based on ECC (ECDLP) of any elliptic curve element that has a public point known base point, is harder than the discrete logarithm problem (DLP) over the finite field F_q

Security is not the only cryptography objective goal but also, there are many factors as the problems associated with key management and protection, hash functions , defective use of random generators, and the incompact private key software. The ECC implementation issues are [27]:

- Used in Diffie Hellman cryptosystem and also, digital signing approach.
- There are many available standardized elliptic curves approved by NIST for the multiple security requirements.
- Elliptic curves cryptosystems enable comprehensive information on algorithms.

The Benefits of elliptic curve based cryptosystems over RSA cryptosystem:

- The elliptic curve based cryptosystem key takes significantly less memory for the same security level. Table I indicates the key size for RSA and ECC for the same security level.
- Smaller key size in ECC leads to faster digital signature generation and therefore saving resources.

In the other hand, ECC has disadvantages versus RSA crypto system. It is complicated in mathematical backgrounds.

NIST guidelines for key size of ECC , RSA, and AES			
ECC	RSA	Ratio	AES
163	1024	1:6	
256	3072	1:12	128
384	7680	1:20	192
512	15360	1:30	256

Table 1: Security level of various key sizes in ECC and RSA.

3.3 Computational problems

Here, a briefly review of some mathematical problems:

Definition 3.3.1. Suppose g be a group generator of the group G where $g \in G$. The CDH related to g is how to compute g^{cd} given by (g^c, g^d), $c, d \in Z_q^*$.

Definition.3.3.2. (Computational Diffie-Hellman (CDH) Problem) over an elliptic curve. Given P point over an elliptic curve and $c, d \in Z_q^*$ then for known $(P, cP, dP) \in G_p$, it is hard to compute of cdP over the group G_q

Definition.3.3.3. (Computational Diffie-Hellman (CDH) Assumption). Let A be an adversary able to break the CDH problem with a trivial probability, if given the tuple $(P, cP, dP) \in G_p$ of CDH problem where $c, d \in Z_q^*$, then A could solve the CDH with the trivial advantage $Adv_{A,G_p}^{CDH} = \Pr[A(P, cP, dP) = cdP : c, d \in Z_q^*]$.

4 Aggregate signature model

An identity-based aggregate signature scheme model has composed six algorithms:

- **Setup phase**: with input k; the security parameter; the public key generator (PKG) generates the master mpk and private keys msk and the system parameters $params$. Finally, the PKG publishes $params$, mpk and keeps msk secret.
- **Key Extract**: PKG runs this algorithm using the signer identity ID_i ;delivered by the signer U_i, $param$ and msk as an input. The output is the signer private key d_i and the PKG sends the signer private key d_i via secure channel to the user U_i.

- **Sign**: this algorithm takes the user identity ID_i, his private key d_i , message m_i and $param$ as input to create a valid signature σ_i on m_i by the signer U_i .
- **Aggregate**: this algorithm takes $\{\sigma_i\}_{i=1}^{n}$ as an input, any third party can generate the signature aggregation σ_{agg} for all the messages with their identities $\{m_i, ID_i\}_{i=1}^{n}$.
- **Signature Verification**: with input σ_{agg} the user U_i performs two checking operation first; whether σ_i by ID_i is a valid signature on m_i and outputs "Valid" if true otherwise , "Invalid". Second; with input ID_i and $\{m_i, ID_i\}_{i=1}^{n}$ checks the validity of the aggregate signature σ_{agg} on σ_i and outputs "Valid" if true otherwise , "Invalid".

5 Security algorithm

5.1 Unforgeability

The proposed scheme security model follows the scheme proposed by [18] with slight variations. The security model follows a game with three phases: setup, training and forgery phase. Two attacks in this security model are considered; adaptive chosen message and identity attacks. Thus, the scheme is secure under those attacks against any forgery. if the adversary A has not a significant advantage in any probabilistic time algorithm in this game :

- **Setup:** by executing this algorithm the challenger C obtains the parameters $param$ and the msk and deliver the $param$ to the adversary A .
- **Training:** A query the following oracle after the setup algorithm:
  - Extract oracle: With ID_i A makes a query and C obtains the private key d_i with ID_i and deliver it to A
  - Signing oracle: A queries the signing oracle with ID_i , m_i then generates a valid signature σ_i on m_i .
- **Forgery:** A generates an aggregate signature σ_{agg} on $\{m_i\}_{i=1}^{n}$ for $\{ID_i\}_{i=1}^{n}$ with input $\{\sigma_i\}_{i=1}^{n}$ in which at least target identity $ID_T \in \{ID_i\}_{i=1}^{n}$. The adversary A forge the signature if there is a valid σ_{agg} for a pair (ID_T, m_T) with the advantage:
  $Adv_A\{\Pr[A(Verify(\sigma_{agg}) = Valid)\}$

6 The proposed scheme

6.1 Setup

- In this phase, the PKG selects three additive groups G_1, G_2, G_3 of order q (prime number) where $q > 2^k$, k is the security parameter. Then the PKG selects two pairs of integers (a, b) satisfying $(4a^3 + 27b^2) \bmod q \neq 0$. Also, the PKG selects a generator point P of G_1 on the elliptic curve E defined by $y = (x^3 + ax + b) \bmod q$ over the finite field F_q^* and chooses a the following hash functions $H_o : \{1,0\}^* \times G_1 \times G_2 \times G_3 \to F_q^*$, $H_1 : \{1,0\}^* \times G_1 \times G_2 \to F_q^*$, and $H_2 : \{1,0\}^* \to F_q^*$
- Then, the PKG randomly picks up $s \in F_q^*$, s is msk and calculates the mpk $P_{pub} = s.P$. The PKG keeps msk secrete and the $params = (G_1, G_2, G_3 n, q, P, P_{pub}, H_o, H_1, H_2)$ public.

6.2 Key extraction

This algorithm follows the following steps:

1. Picks up randomly $x_i \in F_q^*$ and calculates $X_i = x_i.P$
2. Computes $d_i = (x_i + s.q_i) \bmod q$, for the all users $q_i = H_o(ID_i \,||\, X_i)$, $i = 1..n$.
3. The PKG sends the corresponding secrete key $< d_i >$ and the public key $< X_i, q_i >$ to the users through a secrete channel

6.3 Signing

With input (X_i, d_i, ID_i) :

1. Selects a random number $r_i \leftarrow_R F_q^*$ and calculates: $W_i = x_i P$
2. Computes $: h_{1i} = H_1(W_i, X_i, m_i, ID_i)$, and $h_{2i} = H_2(h_{1i}, W_i, X_i, m_i, ID_i)$
3. Computes: $v_i = (r_i h_{1i} + d_i h_{2i}) \bmod q$, $Z_i = v_i.P$.

The signature of ID_i on message m_i is $\sigma_i =< Z_i, X_i, h_{1i}, h_{2i} >$

6.4 Aggregate.

On input $(\{\sigma_i, ID_i\}_{i=1}^{n})$ a set of signatures $\sigma_i =< Z_i, X_i, h_{2i}, h_{1i} >$, with the identity ID_i , $i = 1..n$, $\sigma_i =< Z_i, X_i, h_{2i}, h_{1i} >$ are the signatures of the messages m_i : $Z_{agg} = \sum_{i=1}^{n} Z_i$, $Z_i = v_i.P$, $i = 1..n$

The aggregate signature will be $\sigma_{agg} = \{< Z_i, X_i, h_{1i}, h_{2i} >\}_{i=1}^{n}, Z_{agg}$

6.5 Signature verification.

With the input from $\sigma_{agg} = \{< Z_i, X_i, h_{1i}, h_{2i} >\}_{i=1}^{n}, Z_{agg}$ any user can verify this signature. The verification process as follow:

- Computes $W_i' = h_{1i}^{-1}[Z_i - h_{2i}(X_i + q_i P_{pub})]$ to recover W_i
- Checks if the following equations holds: $h_{1i} \overset{?}{=} H_1(m_i, ID_i, X_i, W_i)$ and $h_{2i} \overset{?}{=} H_2(m_i, ID_i, X_i, W_i, h_{1i})$

6.6 Proof of correctness

$$W_i' = h_{1i}^{-1}[Z_i - h_{2i}(X_i + q_i P_{pub})]$$

$$= \sum_{i=1}^{n} h_{1i}^{-1}[v_i P - h_{2i}(X_i + q_i P_{pub})]$$

$$= \sum_{i=1}^{n} h_{1i}^{-1}[(r_i h_{1i} + d_i h_{2i})P - h_{2i}(X_i + q_i P_{pub})]$$

$$= \sum_{i=1}^{n} h_{1i}^{-1}[(r_i h_{1i} + (x_i + sq_i)h_{2i})P - h_{2i}(X_i + q_i P_{pub})]$$

$$= \sum_{i=1}^{n} h_{1i}^{-1}[(r_i h_{1i} P + h_{2i}(X_i + q_i P_{pub}) - h_{2i}(X_i + q_i P_{pub})] = W_i$$

7 The proposed scheme security proof

The security proof demonstrates that ECDLP could be solved without significant probability ε^o . Also, An adversary A may forge this scheme without significant probability ε^o against chosen message and identity attacks

7.1 Theorem1.

The signature scheme is secure against chosen message and identity attacks if there is an adversary A with a polynomially bounded (t, ε') query for q_{H_o}, q_{H_1}, q_{H_2}, q_S and q_E who can forge the proposed scheme with a non-negligible advantage ε', C may forge the signature with a non-significant advantage:

$$\varepsilon_o = \frac{1}{9} \cdot \frac{10.(q_{sign} + 1)(q_{sign} + q_{H_2} + q_{H_1}).(1 - \frac{q_{Extract}}{q_{H_0}})}{2^{k+1}} \cdot \frac{1}{q_{H_1}} \varepsilon$$

(1)

Proof:

a) Setup

The challenger C selects a group G_1 with a generator point P. Then, C randomly selects $a \in Z_q^*$ and calculate $P_{pub'} = a.P$. C obtains the following four hash oracles: H_o, H_1, H_2 then deliver the public $param = (G_1, G_2, H_o, H_1, H_2, a.P)$ to A

A asks C for different queries as follow:

b) H_o query

- ○ Firstly, C delivers the system parameters to A, then C with input ID, X selects q randomly and returns it to A.
- ○ In another case, A might know the public component X that corresponds to an identity ID. When A makes a query for ID, there are two cases:
  - In the case of $: ID \in \{ID\}_{i=1}^{n}$, the challenger C suits $ID = ID_i$, computes $X = x.P$, x is anonymous, C wants to solve the ECDLP for x , as it is part of ECDLP. After this, C stores $<\perp, q, ID>$ in $H_o list$.
  - If $i \neq 1$, C selects $x, q \in Z_q^*$ randomly, sets $X = x.P$, delivers $< q, X >$ to the signer such that $q = H_o(ID || X)$ and stores $< x, q, ID >$

c) Extract query

When A queries for the private key of ID , C does the following

- ○ C checks the $H_o list$ to verify whether or not there is an entry for ID. If $H_o list$ does not contain an entry for ID, return $\perp$
- ○ Otherwise, if the entry corresponding to ID in $H_o list$ is of the form $< ID, X, x, q >$ and returns $< x, X, *, * >$, if $ID \notin \{ID\}_{i=1}^{n}$ then C recovers the tuple $< X, x, q, ID >$ from $H_o list$ and returns $< X, q, ID >$ and compute $d = x + aq$ then returns d to A .

d) H_1 query

When (m, ID, W) , is submitted to H_1 queries

for the first time C returns checks of $H_1 list$ whether the tuples (m, ID, W) in $H_1 list$ C returns h_1, otherwise C chooses a new random $h_1 \leftarrow_R F_q^*$ includes $< h_1, m, ID, W >$ to the $H_1 list$ then C returns h_1

e) H2 queries

When $< h_1, m, ID, W >$, is submitted to H_2 queries

the first time C returns checks of $H_2 list$ whether the tuples $< h_1, m, ID, W >$ in $H_2 list$,C returns h_2, otherwise C chooses a new random $h_2 \leftarrow_R F_q^*$ includes $< h_2, h_1, m, ID, W >$ to the $H_2 list$ then C returns h_2

f) Sign Oracle

For each new query (m, ID_i) , C proceeds as follows:

- If $ID_i \neq ID_1$, C signs a message m as follows:
- If the public key of ID_i has been replaced:

1) Obtains $< X_i, q_i >$ by calling H_o query oracle on ID
2) Selects $r \leftarrow_R F_q^*$ randomly, calculates: $W = r.P$.

3) Computes: $h_1 = H_1(m_i, W, ID_i, X_i)$ by calling H_1 query on input $< m, W, ID_i, X_i >$, and $h_2 = H_1(m_i, W, ID_i, X_i, h_1)$ by calling H_2query on input $< m_i, W, ID_i, X_i, h_1 >$, and

Obtains the secrete key d from the extract query and computes: $v = (rh_1 + dh_2) \bmod q$, $Z = v.P$.

The signature of ID on message m is $\sigma =< Z, X, h_1, h_2 >$. Otherwise, C signs m in the usual manner by using x_i (obtained from the H_o query) and d_i (obtained from extract query)

- If $ID_i = ID_l$, C does the following:

1) Generates a random $h_1, h_2, v \in F_q^*$
2) Sets $v_i = v, h_{1i} = h_1, h_{2i} = h_2$
3) Computes: $Z_i = v_i P$, and $W_i = h_{1i}^{-1}[Z_i - h_{2i}(X_l + q_l P_{Pub'})]$
4) Updates the lists H_1list and H_2list respectively with the following tuples $< h_{1i}, W_i, ID_i, m_i >$ and $< h_{2i}, W_i, ID_i, m_i, h_{1i} >$. Generate a different $h_1, h_2, v \in F_q^*$ then repeat steps 3 and 4 if any entry in the list H_1list or H_2list is similar as the tuples generated.
5) C returns the signature $< Z_i, X_i, h_{1i}, h_{2i} >$ on m_i by ID_i .

Note the generated signature is valid due to:

$$h_{2i}X_l + h_{1i}W_i + h_{1i}q_l P_{Pub'Pub'}$$
$$= h_{2i}X_l + h_{1i}[h_{1i}^{-1}[Z_i - h_{2i}(X_l + q_l P_{Pub'})] + h_{1i}q_l P$$
$$= h_{2i}X_l + Z_i - h_{2i}X_l - h_{2i}q_l P_{Pub'} + h_{1i}q_l P$$
$$= Z_i = v_i P$$

This shows that $< Z_i, X_i, h_{1i}, h_{2i} >$ will able to be a valid signature to the adversary A.

7.2 Forgery phase

7.2.1 Lemma 1

After the adversary A generate $< Z_1..Z_n, X_1..X_n, Z_{agg} >$ on the message $\{m_i\}_{i=1}^n$ by user identities $\{ID_i\}_{i=1}^n$. A can generate a valid $< Z_1..Z_n, X_1..X_n, Z_{agg} >$ with probability ξ' if there exists ID_l where $l \in \{1,..,n\}$. The algorithm could be flunk in the following places :

- For the extract oracle if the adversary queries for the ID_l then the algorithm flunks. If q_E is the maximum extract queries number made by the adversary. The probability of non-querying for the extract phase is:

$$P[q_{Extract}(ID_i) \neq ID_l] = 1 - \frac{q_{Extract}}{q_{H_o}^*} \qquad (2)$$

where $q_{H_o}^*$ is the queries maximum number .

A may success if $ID_l \notin \{ID_i\}_{i=1}^n$ or if the adversary A make a query for the signing oracle on m_i with user identity ID_l. This happen if:

$$\Pr[ID_i = ID_l,\ \forall i = 1,..,n \quad \text{and}$$
$$D_i \neq ID_l \forall l = 1,\ldots,n, l \neq i] = \frac{n}{2.q_{H_o}^*} \qquad (3)$$

From the previous probabilities, A can break the scheme under adaptive chosen message and identity attack with the advantage:

$$\varepsilon' = \varepsilon.(1 - \frac{q_E}{q_{H_o}^*})\frac{n}{2.q_{H_o}^*} \qquad (4)$$

The adversary A may generate a valid aggregate signature without signer secrete key with the probability

$$\varepsilon = \frac{10(q_{sign} + 1)(q_{sign} + q_{H_2} + q_{H_1})}{2^k} \qquad (5).$$

7.2.2 Lemma 2

A made queries for Extract , $H_o$query , H_1query , H_2query , Sign query as the previous queries. A may generate a valid aggregate signature with probability $\varepsilon'' \geq \frac{1}{9}$ for n users. C computes $W's$ as same as the previous, and then generate a valid signature $< Z_i, X_i, Z_{agg} >$. Using two valid signatures C does the following:

$$Z_{agg} = \sum_{i=1}^{n} v_i.P = (\sum_{i=1}^{n} r_i h_{1i} + \sum_{i=1}^{n} d_i h_{2i}).P$$

$$Z'_{agg} = (\sum_{i=1}^{n} r_i h_{1i} + \sum_{i=1}^{n} d_i h'_{2i}).P$$

$$Z_{agg} - Z'_{agg} = \sum_{i=1}^{n} d_i(h_{2i} - h'_{2i}).P$$

$$Z_{agg} - Z'_{agg} - \sum_{i=1, i\neq l}^{n} d_i(h_{2i} - h'_{2i}).P = d_l(h_{2l} - h'_{2l}).P$$

Thus C knows all the private keys multiplied the point P over the elliptic curve by $d_l(h_{2l} - h'_{2l}).P$. Also, C knows $d_l.P$ by multiplying the final equation by $(h_{2l} - h'_{2l})^{-1}$, but C cannot get d_l unless solving the ECDLP and it is hard under the assumption (ECDLP). C might solve ECDLP with probability:

$$\varepsilon_o = \frac{10(q_{sign} + 1)(q_{sign} + q_{H_2} + q_{H_1})}{2^k}.(1 - \frac{q_{Extract}}{q_{H_o}^*}).\frac{n}{2q_{H_o}^*}.\frac{1}{9}$$

(6)

$$= \frac{1}{9}.\frac{10(q_{sign} + 1)(q_{sign} + q_{H_2} + q_{H_1})(1 - \frac{q_{Extract}}{q_{H_o}^*}).n}{2^{k+1}}.\frac{1}{q_{H_o}^*}$$

(7)

By this, the proposed identity based aggregate signature over is secure against any forgery with a non-significant probability ε_o. Under this assumption C might solve the ECDLP

8 Results and discussion

When analyzing time complexity of the proposed scheme, it is found that it consumes only two point multiplication over elliptic curve in an individual signing process. Through the verification process, the proposed scheme consumes two point multiplication, one modular inverse operation and two point addition over the elliptic curve. All the computations are relative to the modular multiplication process. The proposed scheme consumes 127.84 T_{ML} in one individual complete signing and verification process

9 Comparative study

This section shows the comparative study between the proposed signature scheme without pairing with the scheme with pairings in [28] in the case of individual signing. The computations are all relative to the modular multiplication. Table II indicates the definitions for the cryptographic operations.

Notation	Description
T_{ML}	The time complexity needed to execute the modular multiplication
T_{EM}	The time complexity needed to execute elliptic curve scalar point multiplication, $1T_{EM} \approx 29T_{ML}$
T_{BP}	The time complexity needed to execute the pairings operation, $1T_{BP} \approx 3T_{EM} \approx 87T_{ML}$
T_{PX}	The time complexity needed to execute pairing-based exponentiation, $1T_{PX} \approx \frac{1}{2}T_{BP} \approx 43.5T_{ML}$
T_{EA}	The time complexity needed to execute the point addition over elliptic curve, $1T_{EA} \approx 0.12T_{ML}$
T_{IN}	The time complexity needed to execute the modular inversion operation, $1T_{IN} \approx 11.6T_{ML}$

Table 2: Definition of different cryptographic operations.

The scheme in [28] uses the identity-based signature from pairings. Craig and Zulfikar scheme consumes 406.24 T_{ML} in an individual signing operation while, the proposed pairing free scheme consumes 127.84 T_{ML} in an individual operation and therefore the proposed scheme shows lower time complexity than in [28], as it saves 68.69% from the computations as in table III.

10 Conclusion

This paper introduces a new aggregate signature scheme without pairings. It saves 68.69% of computational cost than another scheme in [28] in pairings. The security proof of the proposed scheme shows that it is secure in random oracle model. The aggregate signature schemes are very useful when needing authentication in vehicular ad hoc network and e-commerce applications.

11 Future scope

The idea of the aggregate signature used in securing the communication networks such as vehicular area network VANETs and Mobile area networks MANETs. Also , aggregate signature used in the e-commerce applications. The proposed scheme should be used in VANETs to provide aggregate authentication with low computational cost.

	Signature				**Verification**				Total (in T_{ML})
	T_{EM}	T_{BP}	T_{IN}	T_{EA}	T_{EM}	T_{BP}	T_{IN}	T_{EA}	
Craig, and Zulfikar	4	-	-	-	1	3	-	2	406.24 T_{ML}
IDB-ASC	2	-	-	-	2	-	1	2	127.84 T_{ML}

Table 3: Comparison of computational cost.

12 References

[1] K. C. Barr and K. Asanovic. Energy Aware Lossless Data Compression. In Proceeding of Mobisya 2005

[2] A. Prace. Privacy Preserving Cryptographic Protocols For Secure Heterogeneous Networks. *Thesis of Doctoral, 2014*

[3] X. Boyen and B. Waters. Full Domain Subgroup Hiding and Constant Size Group Signatures In Public Key Cryptography. *Lecture notes in computer science, vol. 4450, pp.1-15, 2007*

[4] D. Boneh, X. Boyen and H. Shacham. Short Group Signatures. In Advances in Cryptology- CRYPTO 2004, Springer , pp. 227-242.

[5] D. Boneh, C. Gentry, B. Lynn and H. Shacham. Aggregate and verifiably encrypted signatures from bilinear maps. *In Proceeding of Eurocrypt 2003, vol. 2656 of LNCS, pp. 416–432.*

[6] S. Kent, C. Lynn and K. Seo. Secure border gateway protocol (secure-bgp). *IEEE Journal Selected Areas in Comm., pp.582–592, 2000.*

[7] A. Lysyanskaya, S. Micali, L. Reyzin and Shacham H. Sequential aggregate signatures from trapdoor permutations. *In Proceeding of Eurocrypt 2004, vol. 9999 of LNCS, pp. 74–90.*

[8] Z. Cao and L. Liu. On the Disadvantages of Pairing-based Cryptography. International Journal of Network Security, vol.17, no.4, pp.454-462, July 2015.

[9] A. Shamir. Identity-based Cryptosystems and Signature Schemes. *In Proceeding of Crypto 1984, vol. 196 of LNCS, pp 47–53.*

[10] D. Boneh and M. Franklin. Identity-based Encryption from the Weil Pairing. *SIAM Journal of Computing, vol.32, no.3, pp.586–615, 2003*

[11] C. Cocks. An Identity Based Encryption Scheme Based on Quadratic Residues. *In Proc. of IMA Int. Conf., vol. 2260 of LNCS, pp.360–363, 2001*

[12] Fiat A. and Shamir A., "How to prove yourself: Practical solutions to identification and signature problems", In Proceeding of Crypto 1986, vol. 263 of LNCS, pp. 186–194.

[13] L. C. Guillou and J. J. Quisquater. A Paradoxical Identity-Based Signature Scheme Resulting from Zero-Knowledge. *In Proceeding of Crypto 1988, vol. 403 of LNCS, pp 216–231.*

[14] J. C. Cha and J. H. Cheon. An Identity-Based Signature From Gap Diffie-Hellman Groups. *In Proceeding of PKC 2003, vol. 2567 of LNCS, pp.18–30 .*

[15] X. Boyen. Multipurpose Identity-Based Signcryption (A Swiss Army Knife For Identity-based Cryptography). *In Proceeding of Crypto 2003, vol. 2729 of LNCS, pp 383–399.*

[16] Libert B. and Quisquater J.-J., "*Identity based undeniable signatures*,". In Proc. of CT-RSA 2004, pp. 112–125.

[17] M. Bellare, CH. Namprempre and G. Neven. Security Proofs for Identity-Based Identification And Signature Schemes. *In Proceeding Proc. of Eurocrypt 2004, vol.3027 of LNCS, pp.268*-286.

[18] S. Sharmila, D. Selvi, S. S. Vivek, J. Shriram and C. P. Rangan. Identity Based Partial Aggregate Signature Scheme Without Pairing. *IACR Cryptology eprint Archive , 2010.*

[19] M. S. Murty, D. Veeraiah and A. S. Rao. Digital Signature and Watermark Methods For Image Authentication using Cryptography Analysis. *Signal & Image Processing : An International Journal (SIPIJ) Vol.2, No.2, June 2011.*

[20] N. Dey, P. Das, S. S. Chaudhuri , and A. Das. A Session Based Watermarking technique Within the NROI of Retinal Fundus Images for Authencation Using DWT Spread Spectrum and Harris Corner Detection. *In the Proceeding of the Fifth International Conference on Security of Information and Networks, 2012.*

[21] N. Dey, A. K. Pal, S. Samanta, A. Das, and S. S. Chaudhuri. Optimisation of Scaling Factors in Electrocardiogram Signal Watermarking using Cuckoo Search. *International Journal of Bio-Inspired Computation vol.5, no. 5, pp.315-326, 2013.*

[22] N. Dey, G. Dey, S. Chakraborty, and S. S. Chaudhuri. Feature Analysis of Blind Watermarked Electromyogram Signal in Wireless Telemonitoring. *In the series Annals of Information Systems vol. 16 pp. 205-229, 2014.*

[23] N. Dey, M. Dey, S. K. Mahata and D. Ach. Tamper Detection of Electrocardiographic Signal using Watermarked Bio-hash Code in Wireless Cardiology. *Inteligence International Journal of Signal and Imaging Systems Engineering , Vol.8, no.1-2 , 2015.*

[24] Amar, Y. B., I. Trabelsi, N. Dey, and S. Bouhlel. Euclidean Distance Distortion Based Robust And Blind Mesh Watermarking. *International Journal of Interactive Multimedia and Artificial , vo.4, No.2, pp.46-51,2016.*

[25] V. S. Miller. Use of elliptic curves in cryptography. In the Proceedings of the CRYPTO 1985, LNCS, Springer-Verlag vol.218,pp.417-426, 1985.

[26] N. Koblitz. Elliptic Curve Cryptosystem. Journal of Mathematics of Computation vol.48, pp. 203-209, 1987.

[27] K. Magons. Applications and Benefits of Elliptic Curve Cryptography. University of Latvia, Faculty of Computing, Rain¸a bulvaris 19, Riga, LV-1586, Latvia

[28] C. Gentry, and Z. Ramzan. Identity-Based Aggregate Signatures. In the Proceeding of 9th International Conference on Theory and Practice in Public-Key Cryptography, New York, NY, USA, pp.24-26, 2006.

Static and Incremental Overlapping Clustering Algorithms for Large Collections Processing in GPU

Lázaro Janier González-Soler, Airel Pérez-Suárez and Leonardo Chang
Advanced Technologies Application Center (CENATAV)
7ma A # 21406, Playa, CP: 12200, Havana, Cuba
E-mail: jsoler@ceneatav.co.cu and http://www.cenatav.co.cu/index.php/profile/profile/userprofile/jsoler
E-mail: asuarez@cenatav.co.cu and http://www.cenatav.co.cu/index.php/profile/profile/userprofile/asuarez
E-mail: lchang@cenatav.co.cu and http://www.cenatav.co.cu/index.php/profile/profile/userprofile/lchang

Keywords: data mining, clustering, overlapping clustering, GPU computing

Received: November 18, 2016

Pattern Recognition and Data Mining pose several problems in which, by their inherent nature, it is considered that an object can belong to more than one class; that is, clusters can overlap each other. OClustR and DClustR are overlapping clustering algorithms that have shown, in the task of documents clustering, the better tradeoff between quality of the clusters and efficiency, among the existing overlapping clustering algorithms. Despite the good achievements attained by both aforementioned algorithms, they are $O(n^2)$ so they could be less useful in applications dealing with a large number of documents. Moreover, although DClustR can efficiently process changes in an already clustered collection, the amount of memory it uses could make it not suitable for applications dealing with very large document collections. In this paper, two GPU-based parallel algorithms, named CUDA-OClus and CUDA-DClus, are proposed in order to enhance the efficiency of OClustR and DClustR, respectively, in problems dealing with a very large number of documents. The experimental evaluation conducted over several standard document collections showed the correctness of both CUDA-OClus and CUDA-DClus, and also their better performance in terms of efficiency and memory consumption.

Povzetek: OClustR in DClustR sta prekrivna algoritma za gručenje, ki dosegata dobre rezultate, vendar je njuna kompleksnost kvadratnega reda velikosti. V tem prispevku sta predstavljena dva paralelna algoritma, ki temeljita na GPU: CUDA-OClus in CUDA-DClus. V eksperimentih sta pokazala zmožnost dela z velikimi količinami podatkov.

1 Introduction

Clustering is a technique of Machine Learning and Data Mining that has been widely used in several contexts [1]. This technique aims to structure a data set in clusters or classes such that objects belonging to the same class are more similar than objects belonging to different classes [2].

There are several problems that, by their inherent nature, consider that objects could belong to more than one class [3, 4, 5]; that is, clusters can overlap each other. Most of the clustering algorithms developed so far do not consider that clusters could share elements; however, the desire of adequately target those applications dealing with this problem, have recently favored the development of *overlapping clustering algorithms*; *i.e.*, algorithms that allow objects to belong to more than one cluster. An overlapping clustering algorithm that has shown, in the task of documents clustering, the better tradeoff between quality of the clusters and efficiency, among the existing overlapping clustering algorithms, is OClustR [6]. Despite the good achievements attained by OClustR in the task of documents clustering, it has two main limitations:

1. It has a computational complexity of $O(n^2)$, so it could be less useful in applications dealing with a large amount of documents.
2. It assumes that the entire collection is available before clustering. Thus, when this collection changes it needs to rebuild the clusters starting from scratch; that is, OClustR does not use the previously built clustering for updating the clusters after changes.

In order to overcome the second limitation, the DClustR algorithm was proposed by Pérez-Suárez *et al.* in [7]. DClustR introduced a strategy for efficiently updating the clustering after multiple additions and/or deletions from the collection, making it suitable for handling overlapping clustering in applications where the collection changes frequently, specially for those applications handling multiple changes at the same time. Nevertheless, DClustR still suffers from the first limitation; that is, like OClustR, it is $O(n^2)$. This implies that when the collection grows a lot, the time that DClustR uses for processing the changes could make it less useful in real applications. Moreover, when the collection grows, the memory space used by

DClustR for storing the data it needs will also grow, making DClustR a high memory consumer and consequently, making it not suitable for applications dealing with large collections. Motivated by the above mentioned facts, in this work we extend both OClustR and DClustR for efficiently processing very large document collections.

A technique that has been widely used in recent years in order to speed-up computing tasks is parallel computing and specifically, GPU computing. A GPU is a device that was initially designed for processing algorithms belonging to the graphical world, but due to its low cost, its high level of parallelism and its optimized floating-point operations, it has been used in many real applications dealing with a large amount of data.

The main contribution of this paper is the proposal of two GPU-based parallel algorithms, namely *CUDA-OClus* and *CUDA-DClus*, which enhance the efficiency of OClustR and DClustR, respectively, in problems dealing with a very large number of documents, like for instance news analysis, information organization and profiles identification, among others.

Preliminary results of this paper were published in [8]. The main differences of this paper with respect to the conference paper presented in [8] are the following: (1) we introduce a new GPU-based algorithm, named CUDA-DClus, which is a parallel version of the DClustR algorithm, that is able to efficiently process changes in an already clustered collection and to efficiently process large collections of documents, and (2) we introduce a strategy for incrementally building and updating the connected components presented in a graph, allowing CUDA-DClus to minimize the memory needed for processing the whole collection. It is important to highlight that in CUDA-DClus we only analyze the additions of objects to the collection, because this is the case in which it could be difficult to apply DClustR in real applications dealing with large collections, since this is the case that makes the collection grow.

The remainder of this paper is organized as follows: in Section 2, a brief description of both the OClustR and DClustR algorithms are presented. In Section 3, the CUDA-OClus and CUDA-DClus parallel clustering algorithms are proposed. An experimental evaluation, showing the performance of both proposed algorithms on several document collections, is presented in Section 4. Finally, the conclusions as well as some ideas about future directions are presented in Section 5.

2 OClustR and DClustR algorithms

In this section, both the OClustR [6] and DClustR [7] algorithms are described. Since DClustR is the extension of OClustR for efficiently processing collections that can change due to additions, deletions and modifications, the OClustR is first introduced and then, the strategy used by DClustR for updating the clustering after changes is presented. All the definitions and examples presented in this section were taken from [6, 7].

2.1 The OClustR algorithm

In order to build a set of overlapping clusters from a collection of objects, OClustR employs a strategy comprised of three stages. In the first stage, the collection of objects is represented by OClustR as a *weighted thresholded similarity graph*. Afterwards, in the second stage, an initial set of clusters is built through a cover of the graph representing the collection, using a special kind of subgraph. Finally, in the third stage the final set of overlapping clusters is obtained by improving the initial set of clusters. Following, each stage is briefly described.

Let $O = \{o_1, o_2, \dots, o_n\}$ be a collection of objects, $\beta \in [0, 1]$ a similarity threshold, and S:$O \times O \to \Re$ a symmetric similarity function. A *weighted thresholded similarity graph*, denoted as $\widetilde{G}_\beta = \langle V, \widetilde{E}_\beta, S\rangle$, is an undirected and weighted graph such that $V = O$ and there is an edge $(v, u) \in \widetilde{E}_\beta$ iff $S(v, u) \geq \beta$; each edge $(v, u) \in \widetilde{E}_\beta, v \neq u$ is labeled with the value of $S(v, u)$. As it can be inferred, in the first stage OClustR must compute the similarity between each pair of objects; thus, the computational complexity of this stage is $O(n^2)$.

Once $\widetilde{G}_\beta$ is built, in the second stage OClustR builds an initial set of clusters through a covering of $\widetilde{G}_\beta$, using *weighted star-shaped sub-graphs*.

Let $\widetilde{G}_\beta = \langle V, \widetilde{E}_\beta, S\rangle$ be a weighted thresholded similarity graph. A *weighted star-shaped sub-graph* (*ws-graph*) in $\widetilde{G}_\beta$, denoted by $G^\star = \langle V^\star, E^\star, S\rangle$, is a sub-graph of $\widetilde{G}_\beta$, having a vertex $c \in V^\star$, called the *center* of $G^\star$, such that there is an edge between c and all the other vertices in $V^\star \setminus \{c\}$; these vertices are called *satellites*. All vertices in $\widetilde{G}_\beta$ having no adjacent vertices (i.e., isolated vertices) are considered *degenerated* ws-graphs.

For building a covering of $\widetilde{G}_\beta$ using ws-graphs, OClustR must build a set $W = \{G_1^\star, G_2^\star, \dots, G_k^\star\}$ of ws-graphs of $\widetilde{G}_\beta$, such that $V = \bigcup_{i=1}^k V_i^\star$, being $V_i^\star, \forall i = 1 \dots k$, the set of vertices of the ws-graph $G_i^\star$. For solving this problem, OClustR searches for a list $C = \{c_1, c_2, \dots, c_k\}$, such that $c_i \in C$ is the center of $G_i^\star \in W, \forall i = 1..k$. In the following, we will say that a vertex v is *covered* if it belongs to C or if it is adjacent to a vertex that belongs to C. For pruning the search space and for establishing a criterion in order to select the vertices that should be included in C, the concept of *relevance* of a vertex is introduced.

The *relevance* of a vertex v, denoted as $v.relevance$, is defined as the average between the *relative density* and the *relative compactness* of a vertex v, denoted as $v.densityR$ and $v.compactnessR$, respectively, which are defined as follows:

$$v.densityR = \frac{|\{u \in v.Adj / \, |v.Adj| \geq |u.Adj|\}|}{|v.Adj|},$$

$$v.compactnessR = \frac{|\{u \in v.Adj / AIS(G_v^\star) \geq AIS(G_u^\star)\}|}{|v.Adj|},$$

where $v.Adj$ and $u.Adj$ are the set of adjacent vertices of v and u, respectively; $G_v^\star$ and $G_u^\star$ are the ws-graphs determined by vertices v and u, and $AIS(G_v^\star)$ and $AIS(G_u^\star)$

are the *approximated intra-cluster similarity* of $G^\star_v$ and $G^\star_u$, respectively. The *approximated intra-cluster similarity* of a ws-graph $G^\star$ is defined as the average weight of the edges existing in $G^\star$ between its center and its satellites.

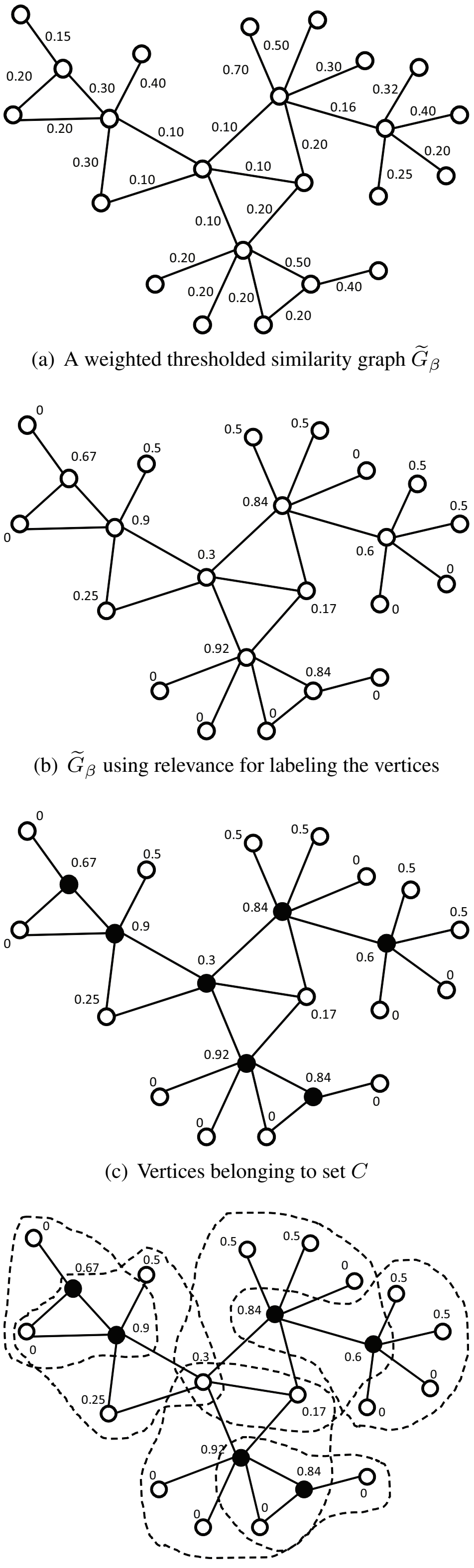
Based on the above definitions, the strategy that OClustR uses in order to build the list C is composed of three steps. First, a *candidate list* L containing the vertices having relevance greater than zero is created; isolated vertices are directly included in C. Then, L is sorted in decreasing order of their relevance and each vertex $v \in L$ is visited. If v is not covered yet or it has at least one adjacent vertex that is not covered yet, then v is added to C. Each selected vertex, together with its adjacent vertices, constitutes a cluster in the initial set of clusters. The second stage of OClustR also has a computational complexity of $O(n^2)$. Figure 1 shows through an example, the steps performed by OClustR in the second stage for building the initial set of clusters.

Finally, in the third stage, the final clusters are obtained though a process which aims to improve the initial clusters. With this aim, OClustR processes C in order to remove the vertices forming a *non-useful* ws-graph. A vertex v forms a non-useful ws-graph if: a) there is at least another vertex $u \in C$ such that the ws-graph u determines includes v as a satellite, and b) the ws-graph determined by v shares more vertices with other existing ws-graphs than those it only contains. For removing non useful vertices, OClustR uses three steps. First, the vertices in C are sorted in descending order according to their number of adjacent vertices. After that, each vertex $v \in C$ is visited in order to remove those non-useful ws-graphs determined by vertices in $(v.Adj \cap C)$. If a ws-graph $G^\star_u$, with $u \in (v.Adj \cap C)$, is non-useful, u is removed from C and the satellites it only covers are "virtually linked" to v by adding them to a list named $v.Linked$; in this way, those vertices virtually linked to v will also belong to the ws-graph v determines. Once all vertices in $(v.Adj \cap C)$ are analyzed, v together with the vertices in $v.Adj$ and $v.Linked$ constitute a final cluster. This third stage also has a computational complexity of $O(n^2)$. Figure 2 shows through an example, how the final clusters are obtained from the initial clusters showed in Figure 1(d).

2.2 Updating the clusters after changes: the DClustR algorithm

Let $\widetilde{G}_\beta = \langle V, \widetilde{E}_\beta, S\rangle$ be the weighted thresholded similarity graph that represents an already clustered collection O. Let $C = \{c_1, c_2, \ldots, c_k\}$ be the set of vertices representing the current covering of $\widetilde{G}_\beta$ and consequently, the current clustering. When some vertices are added to and/or removed from O (i.e., from $\widetilde{G}_\beta$), there could happen the following two situations:

1) Some vertices become uncovered. This situation occurs when at least one of the added vertices is uncovered or when those vertices of C covering a specific vertex were deleted from $\widetilde{G}_\beta$.



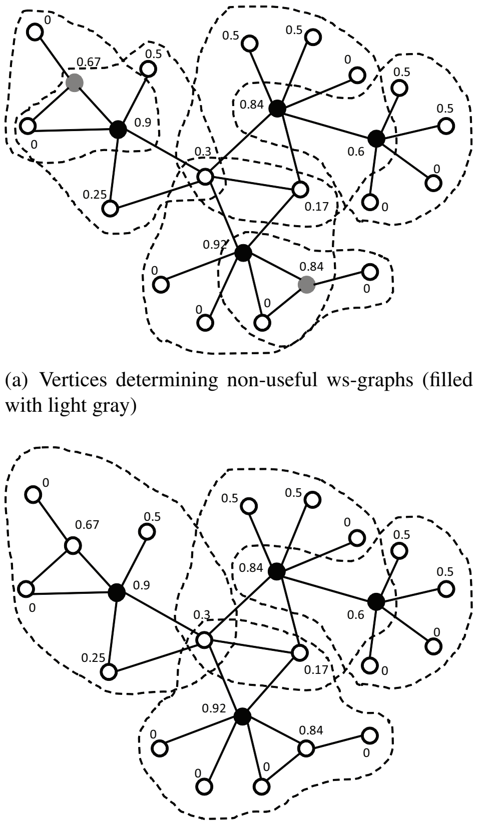
(a) A weighted thresholded similarity graph $\widetilde{G}_\beta$

(b) $\widetilde{G}_\beta$ using relevance for labeling the vertices

(c) Vertices belonging to set C

(d) Set of initial clusters

Figure 1: Illustration of how OClustR builds the initial set of clusters.



(a) Vertices determining non-useful ws-graphs (filled with light gray)

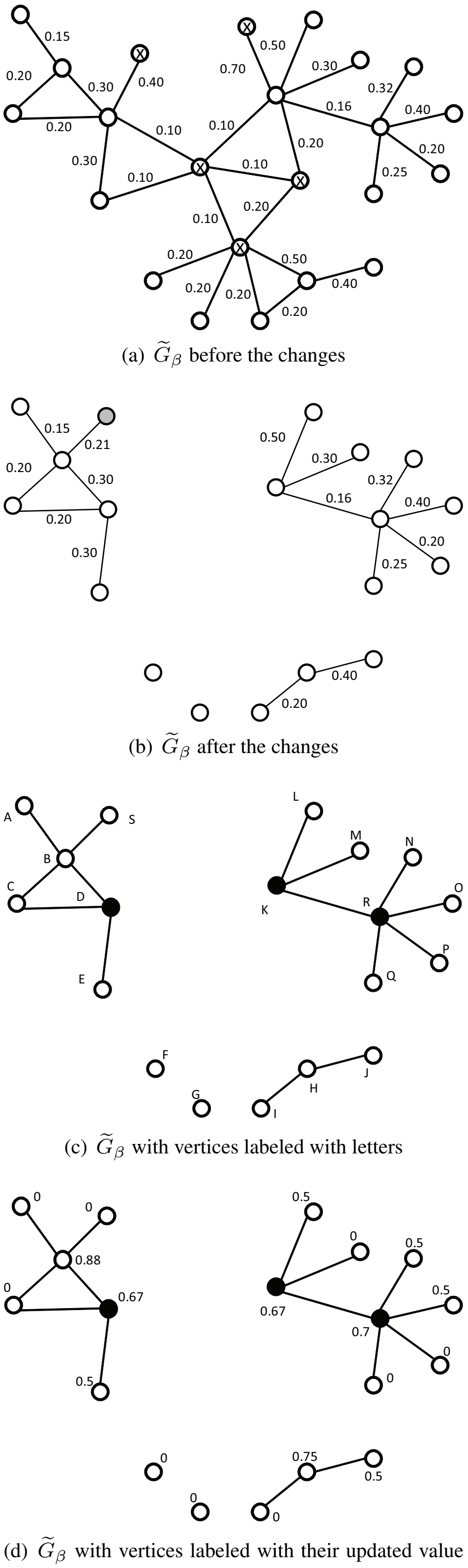
(b) Final set of overlapping clusters

Figure 2: Illustration of how the final clusters are obtained by OClustR in the third stage.

2) The relevance of some vertices changes and, as a consequence, at least one vertex $u \notin C$ appears such that u has relevance greater than at least one vertex in C that covers vertices in $u.Adj \cup \{u\}$. Vertices like u could determine ws-graphs with more satellites and less overlapping with other ws-graphs than other ws-graphs currently belonging to the covering of $\widetilde{G}_\beta$.

Figure 3, illustrates the above commented situations over the graph $\widetilde{G}_\beta$ of Figure 1(a). Figure. 3(a), shows the graph $\widetilde{G}_\beta$ before the changes; the vertices to be removed are marked with an "x". Figure 3(b), shows graph $\widetilde{G}_\beta$ after the changes; vertices filled with light gray represent the added vertices. Figures 3(c) and 3(d), show the updated graph $\widetilde{G}_\beta$ with vertices labeled with letters and with their updated value of relevance, respectively; vertices filled with black correspond with those vertices currently belonging to C. As it can be seen from Figures 3(c) and 3(d), vertices S, F, G, I, H and J became uncovered after the changes, while vertex B, which does not belong to C, has a relevance greater than vertex D, which already belongs to C.

Taking into account the above mentioned situations, in order to update the clustering after changes DClustR first detects which are the connected components of $\widetilde{G}_\beta$ that were affected by changes and then it iteratively updates the covering of these components and consequently, their clus-



(a) $\widetilde{G}_\beta$ before the changes

(b) $\widetilde{G}_\beta$ after the changes

(c) $\widetilde{G}_\beta$ with vertices labeled with letters

(d) $\widetilde{G}_\beta$ with vertices labeled with their updated value of relevance

Figure 3: Illustration of how some changes in the collection affect the current covering of the graph $\widetilde{G}_\beta$ of Figure 2(b).

tering.

The connected components that are affected by changes are those that contain vertices that were added or vertices that were adjacent to vertices that were deleted from $\widetilde{G}_{\beta}$. Since DClustR has control over these vertices it can build these components through a depth first search, starting from any of these vertices. Let $G' = \langle V', E', S \rangle$ be a connected component affected by changes, whose covering must be updated. Let $C' \subseteq C$ be the set of vertices of G' which determine ws-graphs (i.e., clusters) covering G'. DClustR follows the same principles of OClustR; that is, it first builds or completes the covering of G' in order to build an initial set of clusters (stage 1) and then, it improves these clusters in order to build the final set of clusters of G' (stage 2). In fact, DClustR uses the same steps that OClustR for the above two mentioned stages, but unlike OClustR, DClustR modifies the way in which the candidate list L, used in stage 1, is built.

In order to build candidate list L, DClustR first recomputes the relevance value of all vertices in G' and it empties the list $c.Linked$, for all vertices $c \in C'$; this last action is supported by the fact that, after changes, there could be ws-graphs that were considered as non useful, which could be no longer so. Let $V_+ \subseteq (V' \setminus C')$ be the set of vertices of G' with relevance greater than zero, which do not belong to C'. For building the candidate list L, both C' and V_+ are processed.

For processing V_+, DClustR visits each vertex $v \in V_+$ and it verifies a) if v is uncovered, or b) if at least one adjacent vertex of v is uncovered, or c) if there is at least one vertex $u \in v.Adj$, such that there is no other vertex in C' covering u whose relevance is greater than or equal to the relevance of v. If any of these three conditions is fulfilled, v is added to L. Additionally, if the last condition is fulfilled, all those vertices like u are marked as "activated" in order to use them when C' is being processed. The computational complexity of the processing of V_+ is $O(n^2)$.

For processing C', DClustR visits the adjacent vertices of each vertex $v \in C'$. Any vertex $u \in v.Adj$ having greater relevance than v is added to L; in these cases, v is additionally marked as "weak". Once all the adjacent vertex of v were visited, if v was marked as "weak" or at least one of its adjacent vertices were previously marked as "active", v is removed from C' since it could be substituted by a more relevant vertex. However, if v has a relevance greater than zero, it is still considered as a candidate and consequently, it is added to L. The computational complexity of the processing of C' is $O(n^2)$.

Figure 4, shows the updated set of overlapping clusters obtained by DClustR when it processes the graph in Figure 3(d); vertices filled with black represent the vertices determining ws-graphs that cover each connected component of $\widetilde{G}_{\beta}$.

Like OClustR, the computational complexity of DClustR is $O(n^2)$.

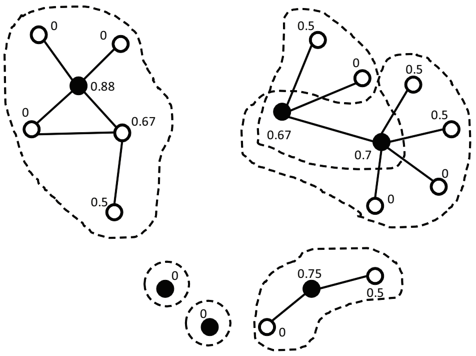


Figure 4: Updated set of overlapping clusters obtained by DClustR.

3 Proposed parallel algorithms

As it was mentioned in Section 1, despite the good achievements attained by OClustR and DClustR in the task of documents clustering, these algorithms are $O(n^2)$ so they could be less useful in applications dealing with a very large number of documents. Motivated by this fact, in this section two massively parallel implementations in CUDA of OClustR and DClustR are proposed in order to enhance the efficiency of OClustR and DClustR in the above mentioned problems. These parallel algorithms, namely *CUDA-OClus* and *CUDA-DClus*, take advantage of the benefits of GPUs, like for instance, the high bandwidth communication between CPU and GPU, and the GPU memory hierarchy.

Although in their original articles both OClustR and DClustR were proposed as general purpose clustering algorithms, the parallel extensions proposed in this work are specifically designed for processing documents. This application context is the same in which both OClustR and DClustR were evaluated and it is also a context in which very large collections are commonly processed. In the context of document processing, both CUDA-OClus and CUDA-DClus use the cosine measure [9] for computing the similarity between two documents; this measure is the function that has been used the most for this purpose [10]. The cosine measure between two documents d_i and d_j is defined as:

$$cos(d_i, d_j) = \frac{\sum_{k=1}^{m} d_i(k) * d_j(k)}{\|d_i\| \cdot \|d_j\|}, \qquad (1)$$

where $d_i(k)$ and $d_j(k)$ are the weights of the k term in the description of the documents d_i and d_j, respectively; $\|d_i\|$ and $\|d_j\|$ are the norms of documents d_i and d_j, respectively.

In experiments conducted over several document collections, it was verified that the first stage of OClustR, the construction of the similarity graph, consumes the 99% of the processing time of the algorithm. The remaining 1% is mainly dominated by the computation of the relevance of the vertices. Based on this fact, the above two mentioned steps are the ones that will be implemented in CUDA by CUDA-OClus; remaining steps are high memory consuming tasks that are more favored with a CPU implementation. Analogously, in these experiments it was also verified that the most time consuming steps of DClustR are the updating of the graph after changes and the recomputing of the relevance, so these steps will be implemented in CUDA by CUDA-DClus. In this case, it could be noticed also that the detection of the connected components affected by changes is a high memory consuming task performed by DClustR, so it is also important to address this problem in CUDA-DClus.

Finally, it is also important to mention that since we are dealing with the problem of processing very large document collections, CUDA-DClus only tackles additions, which are the changes that could increase the size of the collection. Implementing deletions is irrelevant for overcoming problems related with large document collections.

Following, the CUDA-OClus algorithm is first introduced and then, the CUDA-DClust algorithm is presented.

3.1 CUDA-OClus algorithm

Let $D = \{d_1, d_2 \ldots, d_n\}$ be a collection of documents described by a set of terms. Let $T = \{t_1, t_2, \ldots, t_m\}$ be the list containing all the different terms that describe at least one document in D. CUDA-OClus represents a document $d_i \in D$ by two parallel vectors, denoted by T_{d_i} and W_{d_i}. The first one contains the position that the terms describing d_i have in T, and the second one contains the weights that those terms have in the description of d_i.

For building $\widetilde{G}_\beta = \langle V, \widetilde{E}_\beta, S\rangle$, OClustR demands S to be a symmetric similarity measure, so the similarity between any two documents (i.e., vertices in $\widetilde{G}_\beta$) needs to be computed only once. Based on this fact and considering the inherent order the documents have inside a collection D (i.e., vertices in V), for building the edges relatives to a vertex $v \in V$ it is only necessary to compute the similarity between v and each vertex following v in V. Let Suc_v be the list of vertices that follow a vertex v in V. To speed up the construction of $\widetilde{G}_\beta$, for each vertex $v \in V$, CUDA-OClus will compute in parallel the similarity between v and the vertices in Suc_v.

Considering the definition of the cosine measure, it can be seen from Expression (1) that its numerator is a sum of independent products which could be computed all at once. On the other hand, taking into account that the norm of a document can be computed while the document is being read, the denominator of Expression (1) can be also resolved with no extra time. Based on these facts, CUDA-OClus also parallelizes the computation of the similarity between a pair of vertices, in order to speed up even more the construction of $\widetilde{G}_\beta$.

In order to carry out the previous idea, CUDA-OClus builds a *grid* comprised of k square blocks, each block having a shared memory square matrix (SMM); where $k = \frac{n}{t} + 1$ and t is the dimension of both the blocks and the matrices. A *grid* is a logic representation of a matrix of threads in the GPU. The use of SMM and its low latency will allow CUDA-OClus to not constantly access the CPU memory, speeding up the calculus of the similarity between two vertices. CUDA-OClus assigns to t the maximum value allowed by the architecture of the GPU for the dimension of a SMM.

When CUDA-OClus is going to compute the similarity between a vertex v and the vertices in Suc_v, it first builds a vector P_v of size m. This vector has zero in all its entries excepting in those expressed by the positions stored in T_v; these last entries contain their respective weights stored in W_v. Once P_v has been built, the list Suc_v is partitioned into k sublists. Each one of these sublists is assigned to a block constituting the grid and the SMM associated with that block is emptied; i.e., all its cells are set to zero. When a sublist $Q = \{v_1, v_2, \ldots, v_p\}$ is assigned to a block inside a grid, each vertex in Q is assigned to a column of the block. In this context, to assign a vertex v_i to a column means that each row of the column points to a term describing v_i; in this way, the jth row points to the jth term describing v_i. Figure 5 shows an example of how the list Suc_v is divided by CUDA-OClus into k sublists and how these sublists are assigned to the blocks constituting the grid. The example on Figure 5 shows how the first vertex of the sublist assigned to “block 0” is assigned to the first column of that block; the other assignments could be deduced from this example.

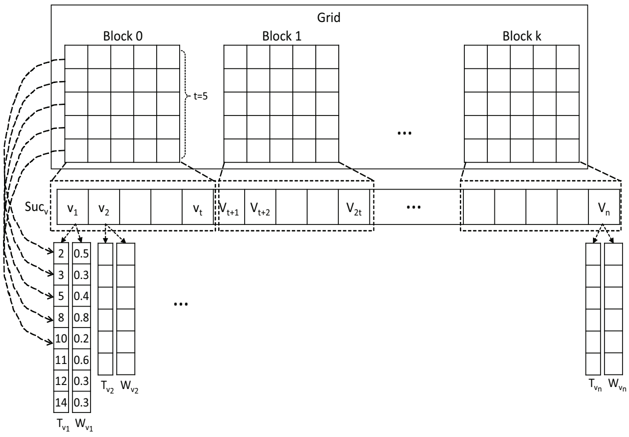


Figure 5: Illustration of how CUDA-OClus divides Suc_v and assigns each resulting sublist to the blocks.

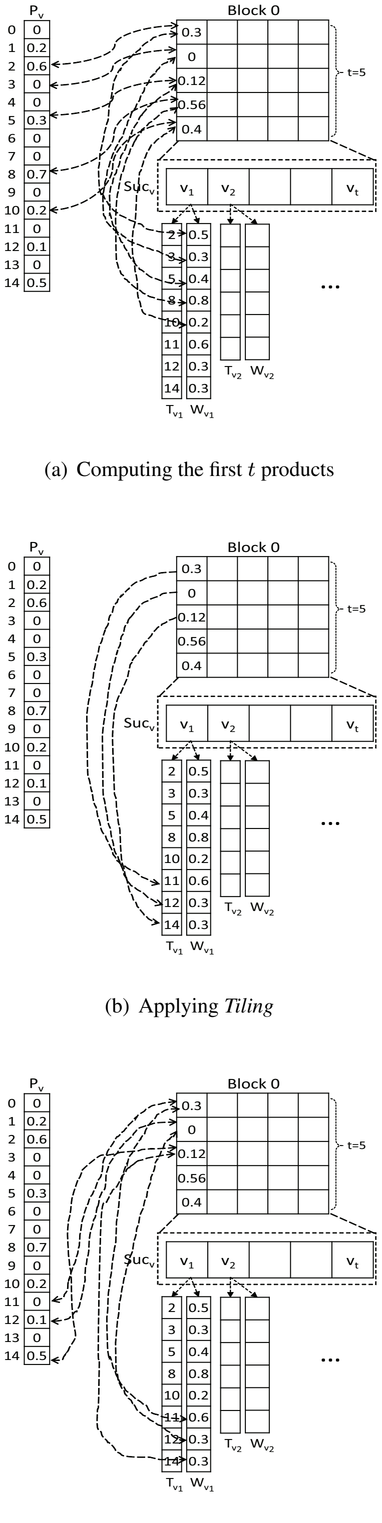
Each row inside a column of a block has a thread that performs a set of operations. In our case, the threads associated with the ith column will compute the similarity between v and its assigned vertex v_i. With this aim, the thread associated with each row inside the ith column will compute the product between the weight that the term pointed by that row has in the description of v_i, and the weight this same term has in the description of vertex v. It is important to note that although the sum in the numerator of Expression (1) runs over all the terms in T, the products that will be different from zero are only those between terms shared by both documents; this is the reason we only use the terms of v_i and multiply their weights by the weights that these terms have in v; remaining terms in v are useless.

Given that the jth row of the column to which vertex v_i has been assigned, points to the jth term of T_{v_i}, the weight this term has in v_i is stored at the jth position of W_{v_i} and the weight this same term has in v is stored at P_v, in the entry referred by the value stored at the jth position of T_{v_i}. The result of the product between the above mentioned weights is added to the value the jth row already has in the SMM. If the description of a vertex v_i assigned to a column of a block exceeds the length of the column (i.e., t) a *tiling* is applied at this block. Tiling [11] is a technique that consists on dividing a data set into a number of small subsets, such that each subset fits into the block; i.e., the SMM. Thus, when the rows of a column point at the next t terms, the products between the weights these terms have in the description of v_i and v are computed and accumulated into the values these rows have in the SMM. This technique is applied until all the terms describing the vertices assigned to the columns have been processed. Figure 6 shows how the similarity between the vertex v_1 assigned to the first column of "Block 0" and v is computed. In this example, it has been assumed that there are 15 terms describing the documents of the collection, the size of the block is $k = 5$, and $T_v = \{1, 2, 5, 8, 10, 12, 14\}$, $W_v = \{0.2, 0.6, 0.3, 0.7, 0.2, 0.1, 0.5\}$, $T_{v_1} = \{2, 3, 5, 8, 10, 11, 12, 14\}$ and $W_{v_1} = \{0.5, 0.3, 0.4, 0.8, 0.2, 0.6, 0.3, 0.3\}$.

As it can be seen from Figure 6(a), each thread of the t rows of the first column computes the product between the weight of the term it points at, and the weight this same term has in P_v (i.e., the description of v). As it was mentioned before, the computed products are stored in the SMM of that block. Note from Figure 6(a) that the product computed by the second row is zero since vertex v does not contain the term pointed out by this row; i.e., term having index 3rd in T. Figure 6(b) shows how when Tiling is applied, the remaining terms describing v_1 are pointed by the rows of the first column. Figure 6(c) shows how the products between the remaining terms of v_1 and v are performed. Finally, Figure 7 shows which are the values stored in the first column of the SMM of "Block 0", once all the products have been computed.

Once all the terms describing the vertices assigned to a block have been processed, a *reduction* is applied over each column of the block. Reduction [12] is an operation that computes in parallel the sum of all the values of a column of the SMM and then, it stores this sum in the first row of the column. Figure 8 shows the final sum obtained for the first column of "Block 0".

The sum obtained on the column to with vertex v_i has been assigned corresponds with the numerator of the cosine measure between v and v_i. This sum is then divided by the product of the norms of v and v_i, which have been previously computed; the result of this division (i.e., the similarity between v and v_i) is copied to the CPU. Using this result CUDA-OClus decides if it should create or not an edge between v and v_i, during this step CUDA-OClus also updates the value of $AIS(v)$ and $AIS(v_i)$.



(a) Computing the first t products

(b) Applying *Tiling*

(c) Computing remaining products

Figure 6: Illustration of how CUDA-OClus computes the similarity between a vertex v and the vertices in Suc_v.

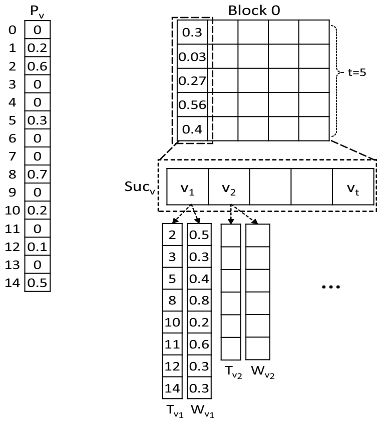


Figure 7: Final results stored in the SMM after processing all terms of v_1.

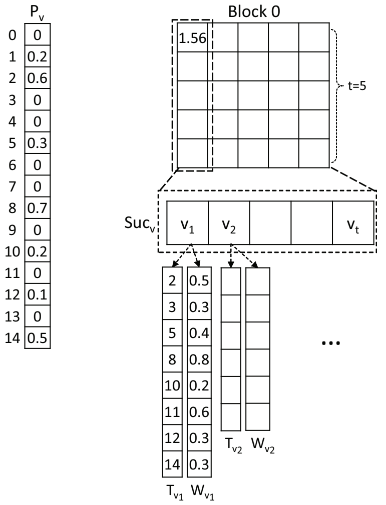


Figure 8: Result of applying Reduction on the first column of "Block 0".

The pseudocode of cosine similarity function is shown in Algorithm 1.

Once the thresholded similarity graph $\widetilde{G}_\beta$ has been built, CUDA-OClus speeds up the computation of the other time-consuming step: the computation of the relevance of the vertices. In order to do that, CUDA-OClus computes in parallel the relevance of all the vertices of $\widetilde{G}_\beta$. Moreover, for each vertex v, CUDA-OClus computes in parallel the contribution each adjacent vertex of v has over the relevance of v, speeding up even more the computation of the relevance of v. In order to accomplish this idea, the list of vertices of $\widetilde{G}_\beta$ is partitioned into k sublists and each sublist is assigned to a block inside a grid. However, in this case, when a vertex v_i of a sublist is assigned to a column of a block, each row in that column will point to an adjacent vertex of v_i; e.g., the jth row points at the jth adjacent vertex of v_i. Different from building graph $\widetilde{G}_\beta$, now the threads associated with a column will compute the relevance of its assigned vertex. With this aim, the thread on each row of that column will compute the contributions the vertex pointed by that row has over the relevance of the vertex assigned to the column.

Algorithm 1: CUDA implementation of the cosine similarity function.

```
Input: $Suc_v$ the list of vertices that follow a vertex $v$, $P_v$ weights of
       vertex $v$, $W_v$ weights associated to $v$, $T_v$ position of terms that
       represent to $v$, $Norm_v$ is the norm of $v$
Output: $similarity$: cosine similarity values between $v$ and $Suc_v$
__shared__ float $SMM[R][C]$ ;   // R = C because block
  are squared
int $tid = threadIdx.x + blockDim.x * blockIdx.x$;
if ($tid < |Suc_v|$) then
    $u = Suc_v[tid]$;
    int $tid_y = threadIdx.y$;
    float $sum = 0$;
    while ($tid_y < |W_u|$) do
        /* Accumulating the multiplication between
           weights of $v$ and $u$                          */
        $sum$ += $W_u[tid_y] * P_v[T_u[tid_y]]$;
        $tid_y$ += $R$ ;       // Applying tiling technique
    $SMM[threadIdx.y][threadIdx.x] = sum$;
/* Waiting that whole threats compute
   multiplications between weights of $v$ and
   $u \in Suc_v$                                           */
__syncthreads();
/* Applying reduction technique to calculate
   $\sum_{k=1}^{m} d_i(k) * d_j(k)$                        */
int $i = R/2$;
while ($i\ != 0$) do
    if ($threadIdx.y < i$) then
        $SMM[threadIdx.y][threadIdx.x]$ +=
          $SMM[threadIdx.y + i][threadIdx.x]$;
    __syncthreads();
    $i = i/2$;
if ($threadIdx.y == 0$ && $tid < |Suc_v|$) then
    $similarity[tid] = 0$;
    if ($Norm_v > 0$ && $Norm_u > 0$) then
        /* Dividing between the multiplication of
           norms of $v$ and $u$                            */
        $similarity[tid] =$
          $SMM[0][threadIdx.x]/(Norm_v * Norm_u)$
```

Let v be a vertex assigned to a column and u one of its adjacent vertices. Vertex u contributes $\frac{1}{|v.Adj|}$ to the relevance of v if $|v.Adj| \geq |u.Adj|$; otherwise, its contribution is zero. This case represents the contribution u has to the relevance of v through the relative density of v. On the other hand, u contributes $\frac{1}{|v.Adj|}$ to the relevance of v if $AIS(v) \geq AIS(u)$; otherwise, its contribution is zero. This other case represents the contribution u has to the relevance of v through the relative compactness of v. The total contribution provided by a vertex is added to the value the row already has in the SMM; similar to the case of building graph $\widetilde{G}_\beta$, the SMM of each block is initially emptied. If v has more than t adjacent vertex, a Tiling is applied. Once all the adjacent vertices of v has been processed, a Reduction is applied in order to compute the relevance of v. Obtained values are then copied to the CPU.

The pseudocode of cosine similarity function is shown in Algorithm 2.

As it was mentioned before, the remaining steps of OClustR were not implemented in CUDA because they are

Algorithm 2: CUDA implementation of the relevance function.

```
Input: $\widetilde{G}_\beta$ weights threshold similarity graph,$AIS(\widetilde{G}_\beta)$ is the
    approximated intra-cluster similarity of $\widetilde{G}_\beta$
Output: $relevance$: relevance values of vertices
$\_\_shared\_\_ \; float \; SMM[R][C]$ ;  // R = C because block
  are squared
$int \; tid = threadIdx.x + blockDim.x * blockIdx.x$;
if ($tid < |V|$) then
    $int \; tid_y = threadIdx.y$;
    $float \; sum = 0$;
    while ($tid_y < |Adj[tid]|$) do
        /* Checking if the density and compactness
           conditions are met                          */
        if ($|Adj[tid_y]| \leq |Adj[tid]|$) then
            $sum+ = 1$;
        if ($|AIS[tid_y]| \leq |AIS[tid]|$) then
            $sum+ = 1$;
        $tid_y$ += $R$ ;       // Applying tiling technique
    $SMM[threadIdx.y][threadIdx.x] = sum$;
/* Waiting that whole threats check density and
   compactness conditions                              */
$\_\_syncthreads()$;
/* Applying reduction technique to calculate
   relevance                                           */
$int \; i = R/2$;
while ($i \; ! = \; 0$) do
    if ($threadIdx.y < i$) then
        $SMM[threadIdx.y][threadIdx.x]$ +=
          $SMM[threadIdx.y + i][threadIdx.x]$;
    $\_\_syncthreads()$;
    $i = i/2$;
if ($threadIdx.y == 0$ && $tid < |V|$) then
    $relevance[tid] = 0$;
    if ($|Adj[tid] > 0|$) then
        /* Dividing between the number of
           adjacents of the current vertex             */
        $relevance[tid] =$
          $SMM[threadIdx.y][threadIdx.x]/(2 * |Adj[tid]|)$;
```

more favored with a CPU implementation since they are high memory consumption tasks.

3.2 CUDA-DClus algorithm

In order to update an already clustered collection when changes take effect, in our case additions, DClustR first detects, in the graph $\widetilde{G}_\beta$ representing the collection, which are the connected components that were affected by changes and then, it updates the cover of those components and consequently, the overall clustering of the collection.

As it was stated in Section 2.2, the connected components affected by additions are those containing at least one added vertex. Thus, each time vertices are added to $\widetilde{G}_\beta$, in addition to computing the similarity between these vertices and those already belonging to $\widetilde{G}_\beta$ in order to create the respective edges, DClustR also needs to build from scratch each affected connected component in order to update their covers. In order to reduce the amount of information DClustR needs to store in memory, CUDA-DClus proposes to represent the graph $\widetilde{G}_\beta$ using an array of *partial connected components*, named Arr_{PCC}, and two parallel arrays. The first of these parallel arrays, named V, contains the vertices in the order in which they were added to $\widetilde{G}_\beta$. The second array, named PC_V, contains the index of the partial connected component to which each vertex belongs. This new representation allows CUDA-DClus to not need to rebuild the affected components each time the collection changes, but keeping the affected components updated each time vertices are added to the graph $\widetilde{G}_\beta$, with no extra cost.

Let $\widetilde{G}_\beta = \langle V, \widetilde{E}_\beta, S\rangle$ be the thresholded similarity graph representing the collection of documents. A *partial connected component* (PCC) in $\widetilde{G}_\beta$ is a connected subgraph induced by a subset of vertices of $\widetilde{G}_\beta$. A partial connected component is represented using two arrays: one array containing the indexes the vertices belonging to that component have in $\widetilde{G}_\beta$, and the other array containing the adjacency list of the aforementioned vertices.

The array of partial connected components representing $\widetilde{G}_\beta$ is built once while $\widetilde{G}_\beta$ is being constructed. The strategy used by CUDA-DClus for this purpose is as follows. In the first step, CUDA-DClus adds a vertex in V for each document of the collection and then, PC_V is emptied (i.e., it is filled with -1), meaning that the vertices do not belong to any PCC yet. In the second step, CUDA-DClus processes each vertex $v_i \in V$. If v_i does not belong yet to a PCC, CUDA-DClus creates a new PCC and it puts v_i in this component; when a vertex v is added to a PCC, the index this PCC has in the array Arr_{PCC} is stored in the array PC_V, at the entry referred to by the index v has in array V; this is the way CUDA-DClus uses to indicate that now v belongs to a PCC. Following, CUDA-DClus computes the similarity between v_i and the vertices in Suc_{v_i}, using the strategy proposed by CUDA-OClus. Once these similarities have been computed, CUDA-DClus visits each vertex $u \in Suc_{v_i}$. If $S(v_i, u) \geq \beta$ and u does not belong to any PCC yet, then u is inserted in the PCC to which v_i belongs and the adjacency lists of both vertices u and v_i are modified in order to indicate they are similar to each other; otherwise, if u already belongs to a PCC only the adjacency lists of both vertices are modified. In this last case, if the partial connected components to which both v_i and u belong are not the same, we will say that these partial connected components are *linked*.

As an example, let $\widetilde{G}_\beta$ be initially empty and $D = \{d_1, d_2, \ldots, d_9\}$ be the set of documents that will be added to the collection. For the sake of simplicity, we will assume that CUDA-DClus already added the documents in $\widetilde{G}_\beta$ as vertices and that the similarities existing between each pair of documents are those showed in Table 1. Taking into account the above mentioned information, Figures 9, 10, 11 and 12 exemplify how CUDA-DClus builds the array of partial connected components representing graph $\widetilde{G}_\beta$, for $\beta = 0.3$.

As it can be seen from Figure 9, firstly CUDA-DClus processes vertex v_1 in order to build the first PCC. As the result of the above process vertices v_3 and v_5 are added to the first PCC, which is now constituted by vertices v_1, v_3 and v_5. The second PCC is built when vertex v_2 is processed, see Figure 10; this component is finally constituted by vertices v_2, v_4 and v_7. Afterwards, as it can be seen

Vert./Vert.	v_1	v_2	v_3	v_4	v_5	v_6	v_7	v_8	v_9
v_1	-	0	0.4	0	0.5	0	0	0	0
v_2	0	-	0	0.4	0	0	0.5	0	0
v_3	0.4	0	-	0.7	0.6	0.3	0	0	0
v_4	0	0	0.7	-	0	0	0	0	0
v_5	0.5	0	0.6	0	-	0	0	0	0
v_6	0	0	0.3	0	0	-	0	0	0
v_7	0	0	0	0	0	0	-	0	0
v_8	0	0	0	0	0	0	0	-	0.5
v_9	0	0	0	0	0	0	0	0.5	-

Table 1: Similarities existing between each pair of vertices of the example.

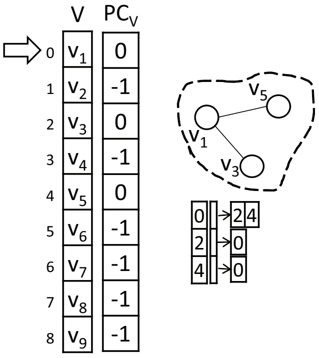


Figure 9: Processing vertex v_1.

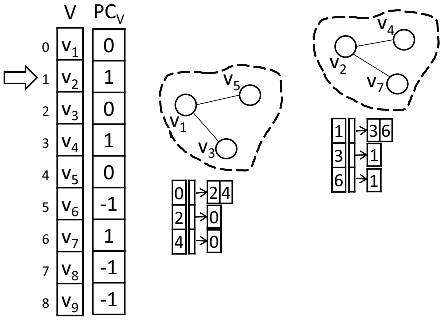


Figure 10: Processing vertex v_2.

from Figure 11, when vertex v_3 is being processed, CUDA-DClus updates the first PCC by adding vertex v_6 and updating the adjacency list of vertices v_3 and v_5; CUDA-DClus also updates the second PCC by modifying the adjacency list of vertex v_4, which is similar to vertex v_3. In this example, these two partial connected components were joined by a dash line in order to illustrate the fact that they are linked since vertices v_3, belonging to the first PCC, and v_4, belonging to the second PCC, are similar. Finally, the third PCC is created when CUDA-DClus processes vertex v_8,

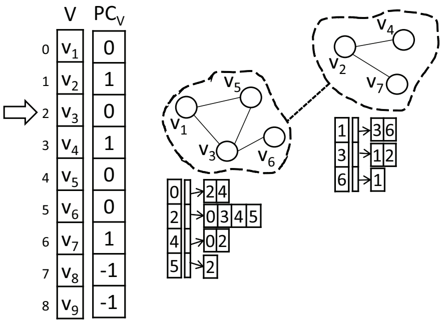


Figure 11: Processing vertex v_3.

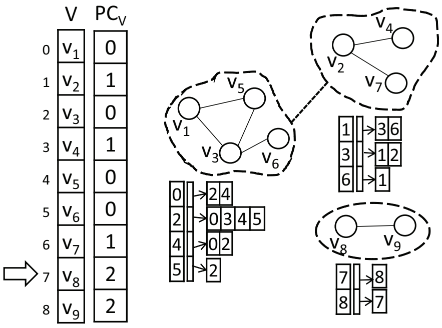


Figure 12: Processing vertex v_8.

as it can be seen in Figure 12. The processing of vertices v_4, v_5, v_6, v_7 and v_9 does not affect the partial connected components built so far, therefore, it was not included in the example.

We would like to emphasize two facts about the above commented process. The first fact is that, since this is the first time the array Arr_{PCC} representing $\widetilde{G}_\beta$ is built, all these components are already in system memory. The second fact is that if we put a PPC $P_i \in Arr_{PCC}$ into a set Q_{P_i} and then, iteratively we add to Q_{P_i} all the linked PCC of each PCC belonging to Q_{P_i}, the resulting set is a connected component. Proof is straightforward by construction. Hereinafter, we will say that Q_{P_i} is the connected component induced by PCC P_i.

Once the array Arr_{PCC} representing $\widetilde{G}_\beta$ was built, CUDA-DClus processes each of its partial connected components in order to build the clustering. For processing a PCC $P_i \in Arr_{PCC}$ that has not been processed in a previous iteration, CUDA-DClus first builds Q_{P_i} and then, CUDA-DClus recomputes the relevance of the vertices belonging to this component using the strategy proposed by CUDA-OClus. Once the relevance of the vertices have been recomputed, CUDA-DClus follows the same steps used by DClustR for updating the covering and consequently, the clustering of Q_{P_i}. Remaining steps of DClustR were not implemented in CUDA because they are more favored with a CPU implementation. Once the clustering has been updated, CUDA-DClus stores the existing partial connected components in the hard drive, releasing in this way the system memory.

Once $\widetilde{G}_\beta$ changes due to the additions of documents to the collection, CUDA-DClus updates the array Arr_{PCC} representing $\widetilde{G}_\beta$ and then, it updates the current clustering. In order to update the array Arr_{PCC}, CUDA-DClus adds for each incoming document, a vertex in $\widetilde{G}_\beta$ and then, CUDA-DClus sets to -1 the entries that these vertices occupy in PC_V, in order to express that they do not belong to any PCC yet. Let $M = \{v_1, v_2, \ldots, v_k\}$ be the set of added vertices. Afterwards, for processing a vertex $v_i \in M$, CUDA-DClus slightly modifies the strategy it uses for creating the partial connected components. Now, rather than computing the similarity of v_i only with the vertices that came after v_i in V (i.e., Suc_{v_i}), CUDA-DClus also computes the similarity of v_i with respect to the vertices that belong to $\widetilde{G}_\beta$ before the changes; that is, the similarities are now computed between v_i and each vertex in $Suc_{v_i} \cup (V \setminus M)$. Remaining steps are the same.

Let $D_1 = \{d_{10}, d_{11}, \ldots, d_{15}\}$ be the set of documents that were added to the collection represented by graph $\widetilde{G}_\beta$, whose array of partial connected components was built in Figure 9, and let $v_{10}, v_{11}, \ldots, v_{15}$ be the vertices that were consequently added in $\widetilde{G}_\beta$ by CUDA-DClus. For the sake of simplicity, in the example it is assumed that none of the vertices belonging to $\widetilde{G}_\beta$ before the changes is similar to the added vertices, with the only exception of v_2 whose similarity with v_{10} is 0.5. Table 2 shows the similarities between each pair of the added vertices. Figures 13, 14, 15 and 16 show, assuming $\beta = 0.3$, how CUDA-DClus updates the array of partial connected components representing $\widetilde{G}_\beta$ after the above mentioned additions. In these figures, vertices filled with light gray are those that were added to the collection.

	v_{10}	v_{11}	v_{12}	v_{13}	v_{14}	v_{15}
v_{10}	-	0.4	0.3	0.6	0	0
v_{11}	0.4	-	0	0.4	0	0
v_{12}	0.3	0	-	0.4	0	0
v_{13}	0.6	0.4	0.4	-	0	0
v_{14}	0	0	0	0	-	0.5
v_{15}	0	0	0	0	0.5	-

Table 2: Similarities existing between each pair of added vertices.

As it can be seen in Figure 13, firstly, CUDA-DClus processes vertex v_{10} and, as a result of this processing, another PCC is created for containing vertices v_{10}, v_{11}, v_{12} and v_{13}. This new PCC was joined with the PCC determined by vertex v_2, through a dash line, in order to reflect the fact that they are linked since vertices v_2 and v_{10} are similar. Furthermore, as it can be seen in Figures 14 and 15, this fourth PCC is updated when vertices v_{11} and v_{12} are processed, in order to reflect the fact that they are similar to vertex v_{13}. Finally, a fifth PCC is created when vertex v_{14} is processed,

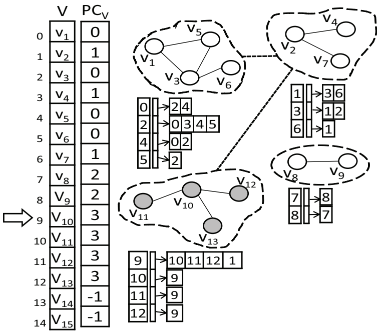


Figure 13: Processing vertex v_{10}.

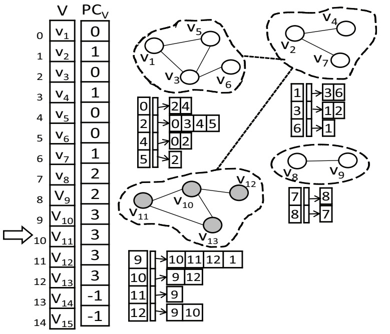


Figure 14: Processing vertex v_{11}.

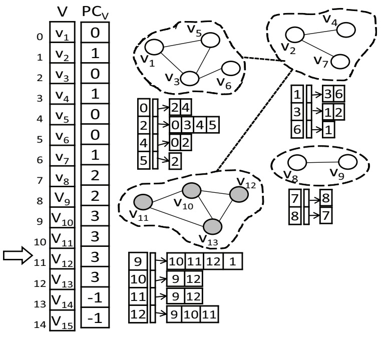


Figure 15: Processing vertex v_{12}.

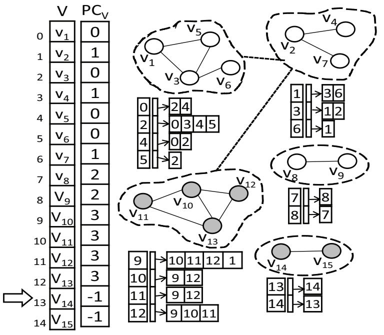


Figure 16: Processing vertex v_{14}.

see Figure 16; this PCC contains vertices v_{14} and v_{15}.

Once the array Arr_{PCC} has been updated, CUDA-DClus processes each new PCC following the same strategy commented above, in order to update the current clustering. It is important to highlight that, different from when Arr_{PCC} was created, this time the partial connected components loaded into the system memory are those belonging to the connected components determined by each new created PCC; the other partial connected components remain in the hard drive. Although in the worst scenario an incoming document can be similar to all existing documents in the collection, generally similarity graphs are very sparse so it is expected that the new representation proposed by CUDA-DClus as well as the strategy it uses for updating the array of partial connected components, help CUDA-DClus to save system memory.

4 Experimental results

In this section, the results of several experiments done in order to show the performance of the CUDA-OClus and CUDA-DClus algorithms are presented. The experiments were conducted over eight document collections and were focused on: (1) assessing the correctness of the proposed parallel algorithms wrt. their original non parallel versions, (2) evaluating the improvement achieved by the proposed algorithms with respect to the original OClustR and DClustR algorithms, and (3) evaluating the memory both CUDA-DClus and DClustR consume when they are processing the same collection. All the algorithms were implemented in C++; the codes of OClustR and DClustR algorithms were obtained from their authors. For implementing CUDA-OClus and CUDA-DClus the CUDA Toolkit 5.5 was used. All the experiments were performed on a PC with Core i7-4770 processor at 3.40 GHz, 8GB RAM, having a PCI express NVIDA GeForce GT 635, with 1 GB DRAM.

The document collections used in our experiments were built from two benchmark text collections commonly used in documents clustering: Reuters-v2 and TDT2. The Reuters-v2 can be obtained from http://kdd.ics.uci.edu, while TDT2 benchmark can be obtained from *http://www.nist.gov/speech/tests/tdt.html*. From these benchmarks, eight document collections were built. The characteristics of these collections are shown in Table 3. As it can be seen from Table 3, these collections are heterogeneous in terms of their size, dimension and the average size of the documents they contain.

Coll.	#Docs.	#Terms	Terms/Docs.
Reu-10K	10000	33370	27
Reu-20K	20000	48493	41
Reu-30K	30000	59413	50
Reu-40K	40000	70348	58
Reu-50K	50000	74720	64
Reu-60K	60000	81632	69
Reu-70K	70000	91490	76
Tdt-65K	65945	114828	210

Table 3: Overview of the collections used in our experiments.

In our experiments, documents were represented using the Vector Space Model (VSM) [13]. The index terms of the documents represent the lemmas of the words occurring at least once in the collection; these lemmas were extracted from the documents using Tree-tagger[1]. Stop words such as: articles, prepositions and adverbs were removed. The index terms of each document were statistically weighted using their term frequency. Finally, the cosine measure was used to compute the similarity between two documents [9].

4.1 Correctness evaluation

As it was mentioned before, the first experiment was focused on assessing the correctness of the proposed algorithms. With this aim, we will compare the clusterings built by CUDA-OClus and CUDA-DClus with respect to those built by the original OClustR and DClustR algorithms, under the same conditions. For evaluating CUDA-OClus we selected the Reu-10K, Reu-20K, Reu-30K, Reu-40K and Reu-50K collections; whilst for evaluating CUDA-DClus we selected Reu-10K, Reu-20K and Reu-30K collections. These collections were selected due to they resemble the collections over which both OClustR and DClustR were evaluated in [6] and [7], respectively.

In order to evaluate CUDA-OClus, we executed OClustR and CUDA-OClus over the Reu-10K, Reu-20K, Reu-30K, Reu-40K and Reu-50K collections, using $\beta = 0.25$ and 0.35. We used these threshold values as these values obtained the best results in several collections as reported in the original OClustR [6] and DClustR [7] articles. Then, we took the clustering results obtained by OClustR as *ground truth* and we evaluateds the clustering results obtained by CUDA-OClus in terms of their accuracy, using the FBcu-

[1] http://www.ims.uni-stuttgart.de/projekte/corplex/TreeTagger

bed [14] and the Normalized Mutual Information (NMI) [15] external evaluation measures.

FBcubed is one of the external evaluation measures most used for evaluating overlapping clustering algorithms and unlike of other external evaluation metrics, it meets with four fundamental constrains proposed in [14] (cluster homogeneity, cluster completeness, rag bag and cluster size vs quantity). On the other hand, NMI is a measure of similarity borrowed from information theory, which has proved to be reliable [15]. Both metrics take values in $[0, 1]$, where 1 means identical results and 0 completely different results. In order to take into account the inherent data order dependency of CUDA-OClus, we executed CUDA-OClus twenty more times over the above mentioned collections, for each parameter value, varying the order of their documents. Table 4 shows the average FBcubed and NMI values attained by CUDA-OClus for each selected collection, using $\beta = 0.25$ and 0.35.

FBCubed					
Threshold	Reu-10K	Reu-20K	Reu-30K	Reu-40K	Reu-50K
β=0.25	0.999	0.999	1.000	0.998	1.000
β=0.35	0.999	1.000	1.000	1.000	0.999
NMI					
Threshold	Reu-10K	Reu-20K	Reu-30K	Reu-40K	Reu-50K
β=0.25	0.997	0.999	1.000	0.999	1.000
β=0.35	0.998	1.000	1.000	1.000	0.999

Table 4: Average FBcubed and NMI values attained by CUDA-OClus for each selected collection.

As it can be seen from Table 4, the average FBcubed and NMI values attained by CUDA-OClus are very close to 1, meaning that the clusters CUDA-OClus builds are almost identical to those built by OClustR. The differences between the clusterings are caused by the inherent data order dependency of the algorithms and also because of the different floating point arithmetic used by CUDA.

In order to asses the validity of CUDA-DClus, in the second part of the first experiment, we will compare the clustering results built by CUDA-DClus with respect to those obtained by DClustR. With this aim, we obtain a ground truth by executing DClustR over the Reu-30K collection, also using $\beta = 0.25$ and $\beta = 0.35$, and then, we process Reu-20K and Reu-10K collections, in this order, as if they were additions of documents to the collection. That is, we are going to add the documents contained in Reu-20K to the current collection (i.e., Reu-30K) and update the clustering using DClustR and after that, we are goind to add Reu-10K to the collection resulting from previous additions (i.e., Reu-30K union Reu-20K) and update the clustering again. We repeated the above mentioned execution under the same parameter configuration but using CUDA-DClus instead of DClustR and afterwards. Then, we take the results obtained by DClustR as ground truth and we evaluate each of the three clustering results obtained by CUDA-DClus in terms of their accuracy, using the FBcubed and NMI external evaluation measures. Like in the first part of this experiment, we executed CUDA-DClus twenty times under the above mentioned experimental configuration, each time varying the order of the documents inside the collections. Table 5 shows the average FBcubed and NMI values attained by CUDA-DClus for each selected collection, using $\beta = 0.25$ and 0.35.

FBCubed			
Threshold	Reu-30K	Reu-30K+Reu-20K	Reu-30K+Reu-20K+Reu-10K
$\beta = 0.25$	0.999	0.995	0.998
$\beta = 0.35$	0.995	0.996	0.991
NMI			
Threshold	Reu-30K	Reu-30K+Reu-20K	Reu-30K+Reu-20K+Reu-10K
$\beta = 0.25$	0.998	0.994	0.999
$\beta = 0.35$	0.997	0.998	0.995

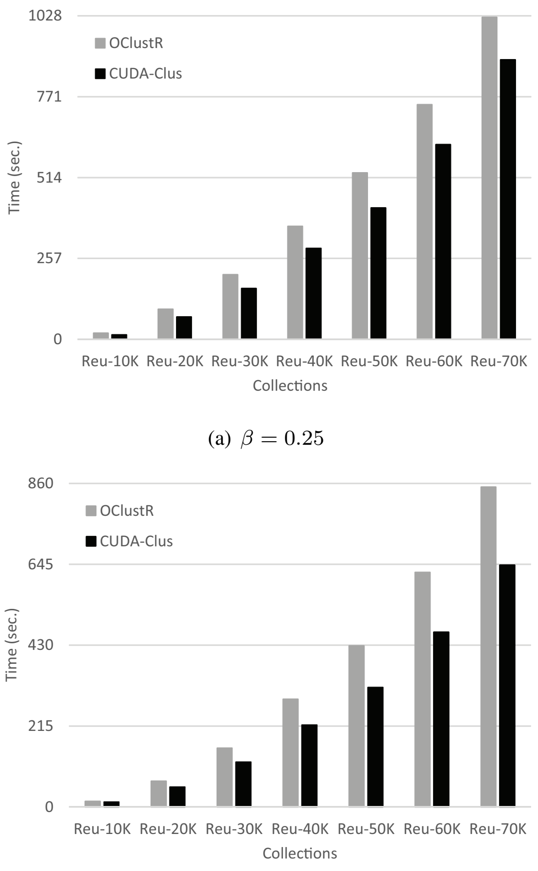
Table 5: Average FBcubed and NMI values attained by CUDA-DClus for each selected collection.

As it can be seen from Table 5, the average FBcubed and NMI values attained by CUDA-DClus are very close to 1, meaning that the clusters it builds are almost identical to those built by DClustR. From this first experiment, we can conclude that the speed-up attained by CUDA-OClus and CUDA-DClus does not degrade their accuracy wrt. the original non parallel versions.

4.2 Execution time evaluation

In the second experiment, we evaluate the time improvement achieved by CUDA-OClus and CUDA-DClus with respect to OClusR and DClustR, respectively. With this aim, we execute both OClustR and CUDA-OClus over Reu-10K, Reu-20K, Reu-30K, Reu-40K, Reu-50K, Reu-60K and Reu-70K, using $\beta = 0.25$ and 0.35 and we measured the time they spent. Like in the previous experiment, in order to take into account the data order dependency of both algorithms, we repeated the above mentioned executions twenty times, for each collection and each parameter configuration, but varying the order of the documents of the collections. Figure 17 shows the average time both OClustR and CUDA-OClus spent for clustering each selected collection, for each parameter configuration.

As it can be seen from Figure 17, CUDA-OClus is faster than OClustR over each selected dataset and for both values of β; for $\beta = 0.25$ and $\beta = 0.35$, CUDA-OClus is respectively 1.26x and 1.29x faster than OClustR. It is important to note from Figure 17 that as the size of the processed collection grows, the difference in the time spent for each algorithm also grows; this behavior shows how well CUDA-OClus scale when the size of the collection grows. We would like to highlight the fact that the specifications of the computer used in the experiments provided advantage to CPU-based algorithms over GPU-based algorithms, since a Core i7-4770 processor at 3.40 GHz with 8GB RAM is superior to a PCI express NVIDA GeForce GT 635, with 1 GB DRAM, which only has two streaming processors and a limited memory. Hence, taking into account the execution model of a GPU, in which the grid blocks are numerated and they distributed among all streaming multiprocessors, which execute simultaneously one



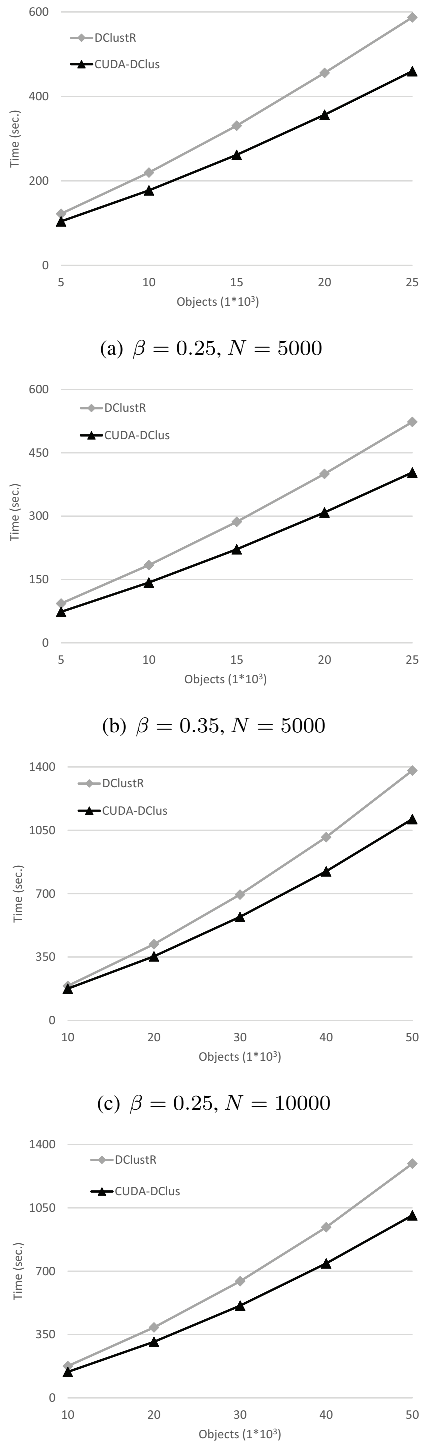
(a) $\beta = 0.25$

(b) $\beta = 0.35$

Figure 17: Time spent by OClustR and CUDA-OClus for clustering the selected experimental datasets, using $\beta = 0.25$ and 0.35.

task over a specific block, then we expect that if we use a powerful GPU with more streaming multiprocessor, the difference between the processing time achieved by parallel version and sequential version will be higher than the one showed in this experiments.

In order to compare both DClustR and CUDA-DClus, in the second part of the second experiment, we clustered the Reu-50K collection using both algorithms and then, we measured the time each algorithm spent for updating the current clustering each time N documents of Tdt-65K collection are incrementally added to the existing collection. In this experiment we also used $\beta = 0.25$ and 0.35, and we set $N = 5000$ and $N = 1000$ which are much greater values than those used to evaluate DClustR [7]. In order to take into account the data order dependency of both algorithms, the above mentioned executions were also repeated twenty times, for each collection and each parameter configuration, but varying the order of the documents of the collections. Figure 18 shows the average time both DClustR and CUDA-DClus spent for updating the current clustering, for each parameter configuration.



(a) $\beta = 0.25, N = 5000$

(b) $\beta = 0.35, N = 5000$

(c) $\beta = 0.25, N = 10000$

(d) $\beta = 0.35, N = 10000$

Figure 18: Time spent by DClustR and CUDA-DClus for updating the current clustering, using $\beta = 0.25$ and 0.35, for $N = 5000$ and 10000.

As it can be seen from Figure 18, CUDA-DClus has a better behavior than DClustR, for each parameter configuration, when multiple additions are processed over the selected dataset, showing an average speed up of 1.25x and 1.29x for $\beta = 0.25, N = 5000$ and $\beta = 0.35, N = 5000$ respectively. Moreover, it also showed an average speed up of 1.19x and 1.26x for $\beta = 0.25, N = 10000$ and

$\beta = 0.35, N = 10000$ respectively. As in the first part of this experiment, it can be seen also from Figure 18, that the behavior of CUDA-DClus, with respect to that of DClustR, becomes better as the size of the collection grows; in this way, we can say that CUDA-DClus also scales well as the size of the collection grows.

4.3 Memory use evaluation

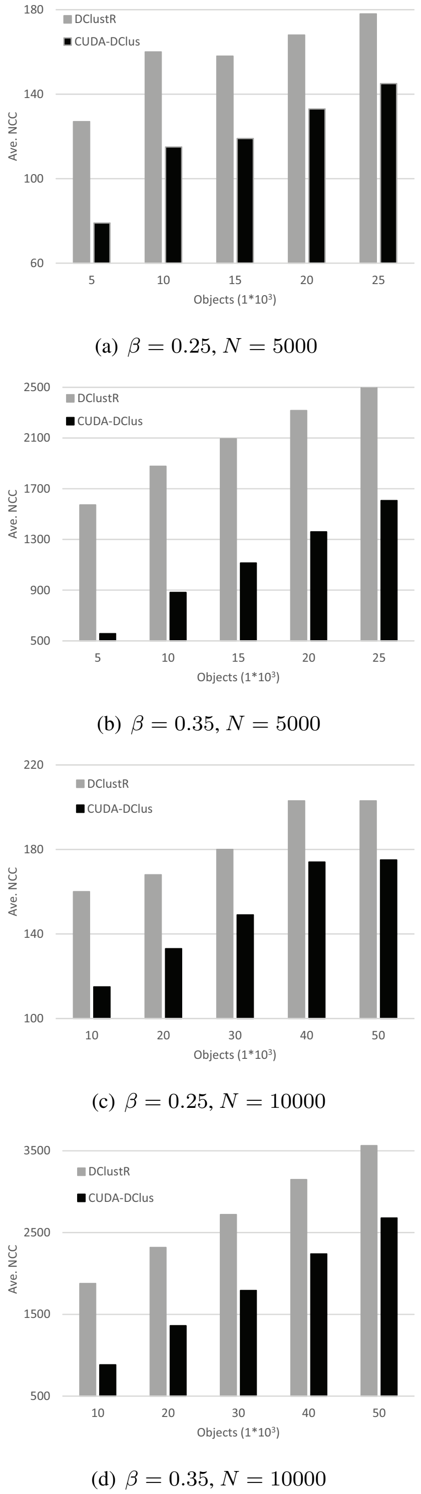
Although the spatial complexity of both algorithm is $O(|V| + \left|\widetilde{E}_\beta\right|)$, the strategy CUDA-DClus proposes for representing $\widetilde{G}_\beta$ should allow to reduce the amount of memory needed to update the clustering each time the collection changes. Thus, in the third experiment, we compare the amount of memory used by CUDA-DClus against that used by DClustR, when processing the changes performed in the second experiment. The amount of connected component loaded by both algorithms when they are updating the current clustering after changes, is directly proportional to the memory used. Based on this, Figure 19 shows the average number of connected components (i.e., Ave. NCC) each algorithm load into system memory, when processing the changes presented in Figure 18, for each parameter configuration.

As it can be seen from Figure 19, CUDA-DClus consumes less memory than DClustR, for each parameter configuration, thereby hence resulting the memory usage of CUDA-DClus is respectively 22.43% for $\beta = 0.25$ and 42.46% for $\beta = 0.35$ less than the one of DClustR. The above mentioned characteristic, plus the fact that CUDA-DClus is also faster than DClustR, makes CUDA-DClus suitable for applications processing large document collections.

We would like to highlight that in the worst scenario, if the clustering of all the connected components needs to be updated, all the partial connected components will be loaded to system memory and thus, our proposed CUDA-DClus and DClustR will have a similar behavior. Additionally, taking into account the results of experiments in sections 4.1 and 4.2, we can conclude that the strategy proposed for reducing the memory used by CUDA-DClus does not include any considerable cost in the overall processing time of CUDA-DClus or in its accuracy.

5 Conclusions

In this paper, we introduced two GPU-based parallel versions of the OClustR and DClustR clustering algorithms, namely CUDA-OClus and CUDA-DClus, specifically tailored for document clustering. CUDA-OClus proposes a strategy in order to speed up the most time consuming steps of OClustR. This strategy is reused by CUDA-DClus in order to speed up the most time consuming steps of DClustR. Moreover, CUDA-DClus proposes a new strategy for representing the graph $\widetilde{G}_\beta$ that DClustR uses for representing the collection of documents. This new representation allows CUDA-DClus to reduce the amount of memory it needs to use and also it helps CUDA-DClus to avoid rebuilding the affected components each time the collection changes but still keep them updated after each changes,



(a) $\beta = 0.25, N = 5000$

(b) $\beta = 0.35, N = 5000$

(c) $\beta = 0.25, N = 10000$

(d) $\beta = 0.35, N = 10000$

Figure 19: Average number of connected components DClustR and CUDA-DClus load into system memory when they are updating the current clustering, using $\beta = 0.25$ and 0.35, for $N = 5000$ and 10000.

with no extra cost.

The proposed parallel algorithms were compared against their original versions, over several standard document collections. The experiments were focused on: (a) assess the correctness of the proposed parallel algorithms, (b) evaluate the speed-up achieved by CUDA-OClus and CUDA-DClus with respect to OClustR and DClustR, respectively, and (c) evaluate the memory both CUDA-DClus and DClusR consumes when they are processing changes. From the experiments, it can be seen that both CUDA-OClus and CUDA-DClus are faster than OClustR and DClustR, respectively, and that the speed up these parallel versions attain do not degrade their accuracy. The experiments also showed that CUDA-DClus consumes less memory than DClustR, when both algorithms are processing changes over the same collection.

Based on the obtained results, we can conclude that both CUDA-OClus and CUDA-DClus enhance the efficiency of OClustR and DClustR, respectively, in problems dealing with a very large number of documents. These parallel algorithms could be useful in applications, like for instance news analysis, information organization and profiles identification, among others. We would like to mention, that even when the proposed parallel algorithms were specifically tailored for processing documents with the purpose of using the cosine measure, the strategy they propose can be easily extended to work with other similarity or distance measures like, for instance, euclidean and manhattan distances.

As future work, we are going to explore the use in CUDA-OClus and CUDA-DClus of other types of memories in GPU such as texture memory, which is a faster memory than the one both CUDA-OClus and CUDA-DClus are using now. Besides, we are going to evaluate both algorithms over more faster GPU cards, in order to have a better insight of the performance of both algorithms when the number of CUDA cores are increased.

References

[1] E. Bae, J. Bailey, G. Dong, A clustering comparison measure using density profiles and its application to the discovery of alternate clusterings, Data Mining and Knowledge Discovery 21 (3) (2010) 427–471.

[2] A. K. Jain, M. N. Murty, P. J. Flynn, Data clustering: a review, ACM computing surveys (CSUR) 31 (3) (1999) 264–323.

[3] S. Gregory, A fast algorithm to find overlapping communities in networks, in: Machine learning and knowledge discovery in databases, Springer, 2008, pp. 408–423.

[4] J. Aslam, K. Pelekhov, D. Rus, Static and dynamic information organization with star clusters, in: Proceedings of the seventh international conference on Information and knowledge management, ACM, 1998, pp. 208–217.

[5] A. Pons-Porrata, R. Berlanga-Llavori, J. Ruiz-Shulcloper, J. M. Pérez-Martínez, Jerartop: A new topic detection system, in: Progress in Pattern Recognition, Image Analysis and Applications, Springer, 2004, pp. 446–453.

[6] A. Pérez-Suárez, J. F. Martínez-Trinidad, J. A. Carrasco-Ochoa, J. E. Medina-Pagola, Oclustr: A new graph-based algorithm for overlapping clustering, Neurocomputing 121 (2013) 234–247.

[7] A. Pérez-Suárez, J. F. Martínez-Trinidad, J. A. Carrasco-Ochoa, J. E. Medina-Pagola, An algorithm based on density and compactness for dynamic overlapping clustering, Pattern Recognition 46 (11) (2013) 3040–3055.

[8] L. J. G. Soler, A. P. Suárez, L. Chang, Efficient overlapping document clustering using gpus and multicore systems, in: Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications - 19th Iberoamerican Congress, CIARP 2014, Puerto Vallarta, Mexico, November 2-5, 2014. Proceedings, 2014, pp. 264–271.

[9] M. W. Berry, M. Castellanos, Survey of text mining, Computing Reviews 45 (9) (2004) 548.

[10] R. Gil-García, A. Pons-Porrata, Dynamic hierarchical algorithms for document clustering, Pattern Recognition Letters 31 (6) (2010) 469–477.

[11] J. Sanders, E. Kandrot, CUDA by example: an introduction to general-purpose GPU programming, Addison-Wesley Professional, 2010.

[12] D. B. Kirk, W. H. Wen-mei, Programming massively parallel processors: a hands-on approach, Newnes, 2012.

[13] G. Salton, A. Wong, C.-S. Yang, A vector space model for automatic indexing, Communications of the ACM 18 (11) (1975) 613–620.

[14] E. Amigó, J. Gonzalo, J. Artiles, F. Verdejo, A comparison of extrinsic clustering evaluation metrics based on formal constraints, Information retrieval 12 (4) (2009) 461–486.

[15] A. Lancichinetti, S. Fortunato, J. Kertész, Detecting the overlapping and hierarchical community structure in complex networks, New Journal of Physics 11 (3) (2009) 033015.

Weighted Density Center Clustering Protocol for Wireless Sensor Networks

Amira Slimani and Mohammed Redjimi
Universite 20 Aout 1955- Faculte des sciences, Departement d'informatique- 21000 Skikda, Algeria
E-mail: amira201015@hotmail.fr, redjimimed@yahoo.fr

Djamel Slimani
LIS laboratory, Department of electronics communication, University of Setif 1, 19000 Setif, Algeria
E-mail: slimani.djamel@ymail.com

Keywords: WSNs, LEACH, clustering, hierarchical protocols

Received: December 29, 2016

Wireless sensor networks (WSNs) are often composed of a huge number of micro sensors. Low energy batteries and low processing capabilities characterize these tiny devices. The most critical challenge in these networks is the energy conservation. Hierarchical architecture with clustering of nodes is suitable for solution of many problems in the WSNs and has many benefits as energy efficiency, data aggregation and scalability. Low Energy Adaptive Clustering Hierarchy (LEACH) and its centralized version LEACH-C are the most popular hierarchical protocols. This paper proposes a cluster based routing protocol for WSNs, which is an improvement of LEACH-C. It is based on the weighted density center and energy criteria in the selection of the cluster heads (CH). The proposed protocol provides a significant increasing in network lifetime and more energy efficiency. The simulation results show that the proposed algorithm performs better than LEACH-C, Improved-LEACH and the Distance and Energy Aware LEACH (DE-LEACH) algorithms

Povzetek: Prispevek predlaga protokol usmerjanja v brezžičnih senzorskih omrežjih (WSN) kot izboljšanje protokola LEACH-C.

1 Introduction

With the recent advancements in wireless communication and miniaturization technology, micro sensors have become possible and popular techniques in military, health, security, commercial and industrial applications [1-4]. These tiny sensors, dispersed in huge numbers and autonomous manner, have a capacity to form self-configured networks [5-6]. In Wireless Sensor Network (WSN) applications. Useful collected data by sensors from environment are then sent to the base station. Therefore, the most challenge of these devices is the energy conservation since they operate with limited non-rechargeable batteries. In large scale WSNs, clustering technique is usually used to provide effectiveness in energy saving and topology stability [7-8]. In cluster-based protocols, nodes are regrouped in clusters in where a leader node is elected as a cluster head (CH). Each CH receives the data transmitted by the other node members of the cluster and in turn transmits them in aggregated form to the base station. So, the CH must be efficiently selected, as it is required to organize activities in cluster. These activities deal with data aggregation, monitoring and scheduling the cluster communication to optimize the energy of sensors and extend the network lifetime [3]. Several protocols have been proposed in literature [9-20] to optimize the operation of sensors and raise the network efficiency. Among these research works, the hierarchical protocols based on network division into clusters and election of a cluster leaders according to metrics are the mostly considered. The Low Energy Adaptive Clustering Hierarchy (LEACH) is the first and the most popular hierarchical protocol, which operates on rounds. In each round, two principal operations are performed. The first; called setup phase, concerns the creation of clusters and the election of their leaders. The second deals with the data transmission to the base station by the CHs [10]. The Low Energy Adaptive Clustering Hierarchy centralized (LEACH-C) is the centralized version of LEACH that operates in rounds as in LEACH and each round is divided into two phases: setup and steady state. It provides the energy conservation efficiency particularly in setup phase where it takes into account the energy level of sensors in selection of CHs, which is not considered in LEACH [11]. After these two popular protocols, many other approaches have been proposed and adopted. Some of them are based on LEACH [12-15]. The Mobile-LEACH (M-LEACH) [12] is a multi-hop version of LEACH, which uses a multi hop to send data to the base station. The Two Levels Low Energy Adaptive Clustering Hierarchy protocol (TL-LEACH) [13] uses two types of CHs: primary and secondary CH to ensure more reliability in the CH. The Improved-LEACH [14] routing communication protocol for wireless sensor network proposes a vice cluster head for each cluster during the communication process. The Distance and Energy Aware LEACH (DE-LEACH) [15] is an improved LEACH routing protocol for WSNs in where the network is divided into two parts according to the distance of sensors from the base station. In the other hand, several routing protocols based on fuzzy logic are proposed for the WSNs as Cluster Head Election mechanism using Fuzzy logic (CHEF) [16].This latter uses two fuzzy descriptors for CH election; the first is based on the residual energy and the second on the distance between nodes having the same radius r. This protocol operates on rounds. The Energy-aware Clustering Protocol using Fuzzy-logic (ECPF) [17] is another protocol based on fuzzy logic, which processes on three operations: initialization, processing and finalization. Other techniques are used for WSNs such as bio-inspired and heuristic-based algorithms. Among them are the Ant Colony Optimization (ACO), the Genetic Algorithms (GA) and the Particle Swarm Optimization (PSO). The ant colony optimization algorithms are among the most popular used techniques in the last few years. [18] Proposed an ACO, which uses a probabilistic approach to choose the best solutions. This approach is based on metrics and the most important one is the pheromone value, which is a variable that defines the quality of path. ’Ants’, in these protocols, are placed in each node transmitter. Many other protocols are proposed for WSNs and they have alternative objectives added to the energy as in [19] where its main goal is the network security. Scalable data coupled clustering for large-scale WSNs algorithm is proposed in [20] in order to improve and achieve high scalability of the network. To reduce the energy consumption and extend the lifetime of sensor network, many research works have proposed a multi-hop communication. However, in the multi-hop algorithms, the intermediate nodes routing cannot be avoided, which in turn may increase the energy consumption.

In order to overcome this problem, the approach proposed in this paper considers a transmission mode in where the message is directly transmitted from the cluster members to the cluster head (CH). The CH election is based on the energy criteria and weighted density center (WDC) of cluster nodes. The WDC idea reduces significantly the communication distances among nodes. Consequently, the energy conservation is enhanced.

This approach is inspired from LEACH-C protocol. However, when selection of CH is considered, the WDC algorithm chooses the nearest point to all nodes in the cluster and it is elected as cluster head. Then, the data transmission from the nodes of the cluster to the CH is carried out in a single hop and over a short distance. This mode of transmission avoids losses due to long node transmissions and reduces the number of paths and energy consumption in the WSN.

The rest of this paper is organized as follows: section 2 describes the network model. Section 3 deals with the energy model. Section 4 presents a description of the proposed algorithm. Sections 5 and 6 present respectively the simulation results and the conclusion.

2 The network model

The following properties are used to model a sensors region:

- The sensor nodes and the base station are assumed stationary once they are deployed in the environment.
- The base station is static, and its location is initially known.
- Wireless sensor network includes homogeneous sensor nodes.
- Initially, all sensor nodes have the same amount of energy.
- The base station is not limited in terms of energy, memory and computing power.
- The nodes are eligible to determine its current energy level and location information through GPS service.
- All the sensor nodes are immobile and having fixed node identification.
- Data aggregation is done at the CH node.

The following figure (Figure 1) shows the initial network vision:

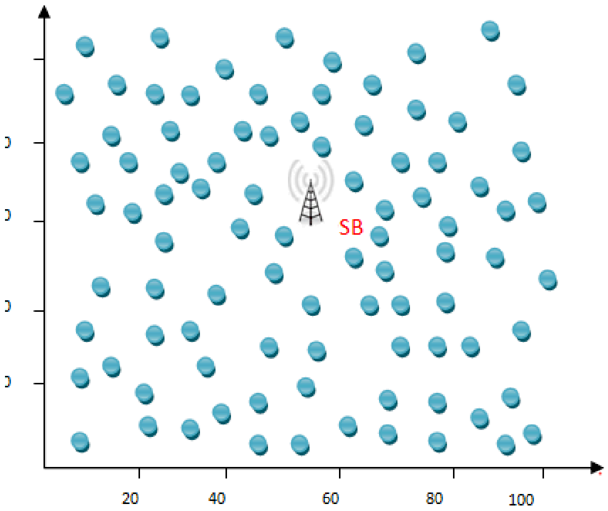


Figure 1: Network Model.

3 The energy model

The energy consumption model, which is used in the proposed protocol, is presented figure 2. It is the same radio model, which is used in LEACH-C [10]. The transmitter dissipates energy for a transmission of k-bits data, which is given by equation 1

$$E_{tx}(k,d) = \begin{cases} E_{elec} \times k + E_{amp} \times k \times d^2, & d < d_0 \\ E_{elec} \times k + E_{amp} \times k \times d^4, & d \geq d_0 \end{cases} \quad (1)$$

Where

$E_{tx}(k,d)$: Transmitter dissipated energy

E_{elec} : Electronic devices energy

E_{amp} : Power amplifier

d: Transmission distance

d_0: is the threshold distance that depends on the environment.

k: is the number of transmitted bits.

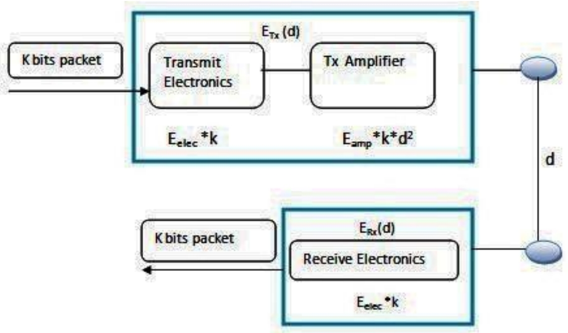


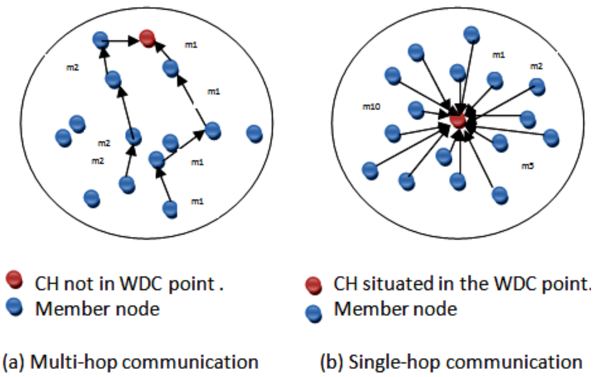
Figure 2: Radio energy model [10].

For receiving of k-bits data by each sensor node, the receiver-dissipated energy E_{rx} is given by equation 2

$$E_{rx}(k,d) = E_{elec} \times k \qquad (2)$$

4 The proposed WDC-LEACH-C clustering protocol

The proposed protocol called Weighted Density Center Clustering Protocol based on LEACH-C (WDC-LEACH-C) operates in several rounds; each round is divided into two phases. The first is the setup phase, which consists of the cluster formation, CHs election and the scheduling transmission of nodes in each cluster. The scheduling is created by using a Time Division Multiple Access (TDMA) protocol. The CHs election is based on metrics: initial energy, residual nodes energy and the distance's criteria. This important latter parameter is managed by the WDC of nodes in the same cluster. Therefore, the message is directly transmitted from the cluster member's to the cluster head (CH). The criteria of WDC is not considered in LEACH-C, Improved-LEACH and DE-LEACH. When the setup phase is completed, the second phase (steady state) begins by a data collecting, aggregation and transmission to the base station. WDC-LEACH-C follows a strategy of clustering in setup phase basing on the remaining energy of sensors, distances among nodes and a single hop communication. The proposed protocol minimizes the communication distance and the number of transmission routes and consequently reduces the energy consumption. Figure 3 shows a comparison between a multi-hop communication and a WDC single-hop communication in terms of paths.



(a) Multi-hop communication (b) Single-hop communication

Figure 3: Number of paths in multi-hop and WDC single-hop communication.

The proposed protocol is working as follows: Initially, after deployment of nodes in harsh environments, each sensor sends its information (local information and energy level) to the remote BS. Note that the periodicity of these initial packets is ignored in simulation time. After receiving this information, a division of network into equal regions is carried out to form balanced clusters in terms of nodes number. Once the clusters are formed, a CH should be chosen from each cluster. CH should be the node, which holds the highest energy and the closest to the cluster nodes. The selection of CH is done in the following manner:

- Computing the average separation distance $d_{avg}^{(i)}$ from any node i to all nodes of the same cluster using the equation (3)

$$d_{avg}^{(i)} = \frac{1}{n}\sum_{\substack{j=1 \\ i\neq j}}^{n} d_{ij} \qquad (3)$$

Where
$d_{avg}^{(i)}$: is the average distance
d_{ij}: is the distance from the node i to the node j
n : is the total number of cluster nodes.

- Choosing the node placed at the center of gravity of the cluster. We call it the weighted density center node (WDC) in equation (4) :

$$d_{WDC} = \min\left\{d_{avg}^{(i)}\right\}_{i=1}^{n} \qquad (4)$$

- Selection of nearest nodes to WDC to constitute group G: this can be done by performing the following test

```
{
Init G=0;
For (i=1…n)
If  $d_{avg}^{(i)} < d_{Threshold}$ then
    $G = G + 1;$
End
}
```

Where $d_{Threshold}$ is approached by the following formula that selects the perfect group G in each cluster

$$d_{Threshold} = \frac{d_{WDC} + \frac{1}{n}\sum_{i=1}^{n} d_{avg}^{(i)}}{2} \qquad (5)$$

- Among a group G, selection of the most energized node to be the cluster head (CH). This is done in the following way:
  1- Computation of the probability of the remaining energy for each node in a group G.

$$P_j = \frac{E_{residual}^{(j)}}{E_{initial}^{(j)}} \qquad j \in G \qquad (6)$$

  2- Choosing the node among G that has a maximum energy. This can be done by (7)

$$E_{max} = max\{P_j\}^{G}{}_{j=1} \qquad (7)$$

The CH is chosen to be the node j that has a maximum energy.

The figures 4 and 5 depict the WDC operations and the CH election, and the figure 6 shows the general Flowchart of clustering and CHs election.

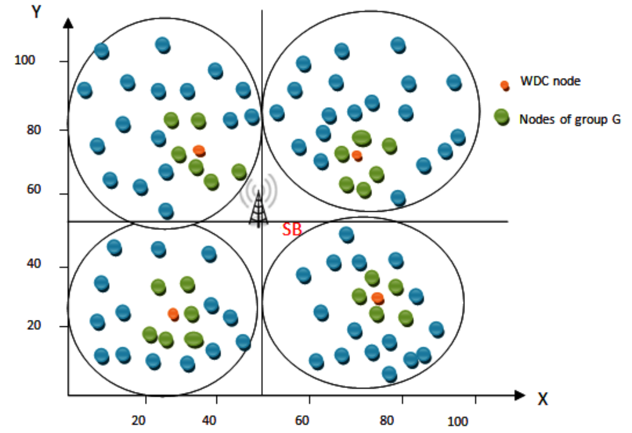


Figure 4: WDC calculation.

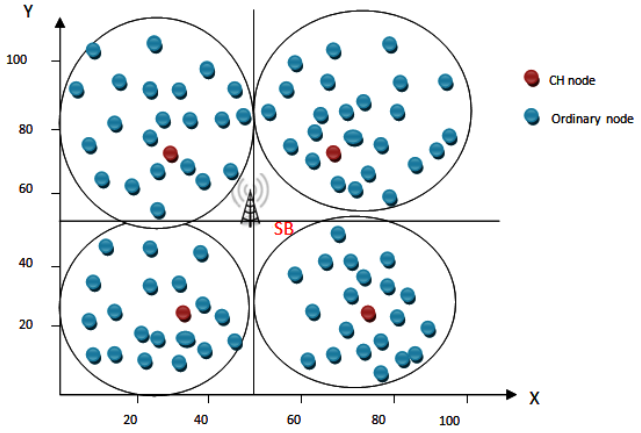


Figure 5: CH selection.

Parameters	Values
Simulation area	(100 m× 100 m)
Network size	100 sensors
BS Location	(50,50)
Initial Energy	2 joules
Minimum Energy	0.001 joule
Maximum duration of the simulation	3600 seconds
Data packet size	25 Bytes

Table 1: Simulation parameters.

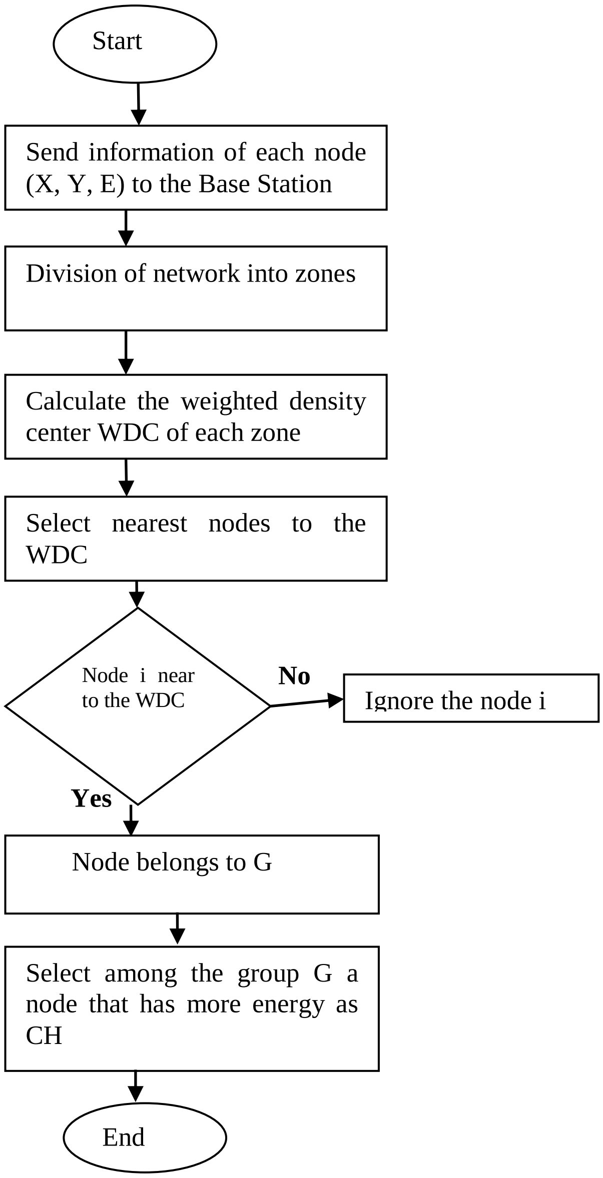

Start
Send information of each node (X, Y, E) to the Base Station
Division of network into zones
Calculate the weighted density center WDC of each zone
Select nearest nodes to the WDC
Node i near to the WDC
No
Ignore the node i
Yes
Node belongs to G
Select among the group G a node that has more energy as CH
End

Figure 6: Flowchart of the clusters formation and CHs election in WDC-LEACH-C.

5 Simulation and results

In this section, the simulation process is carried out on WSNs with 100 nodes where the following parameters (Table 1) are used:

In order to validate the performance of the proposed WDC-LEACH-C algorithm, a comparison with LEACH-C, Improved-LEACH (Impro-LEACH in figures) and DE-LEACH algorithms has been done. In this comparison, five important metrics are considered:

1- Network lifetime: the network lifetime can be defined by three ways

FND (First Node Died): is also called stability period, this is the time interval between the start of simulation and the death of the first node.

HLD (Half Node Died): is the time between the start of simulation and the time of death of the half of nodes.

LND (Last Node Died): is the time between the start of simulation and the time of death of the last node.

2- Number of alive nodes per round: this will measure the number of alive nodes in each round.

3- Consume energy: represents the energy consumption of nodes over simulation rounds. The main objective of routing protocols is to save the energy of sensors because if the sensor node runs out of energy then it will be dead and its life will be over too, so energy is the key factor needed to be considered in WSNs.

4- The quantity of data received by the BS is measured in Bytes.

5- Each data packets received by the BS corresponds to 25 Bytes. The periodicity of these packets is ignored in simulation time because our clustering algorithm does not include periodic transmissions to the base station as in LEACH –C algorithm.

Figure 7 represents the network lifetime in which FND, HND and LND are shown for the LEACH-C, Improved-LEACH, DE-LEACH and for the proposed protocol WDC-LEACH-C. Seen from this figure, death time of the first, half and last nodes in WDC-LEACH-C is latter than that in LEACH-C, Improved-LEACH, and DE-LEACH. Therefore, the WDC-LEACH-C provides a better long life for the network due to the minimization of the number and length of routes, which optimizes the energy consumption of sensors.

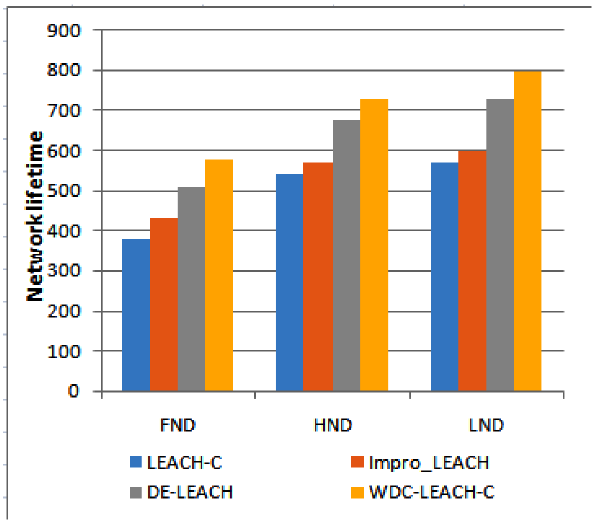


Figure 7: Network lifetime.

Figure 8 shows the number of alive nodes per round in network. As we can see from this figure; the vanishing of alive nodes is decreasing slowly in the proposed algorithm WDC-LEACH-C, compared to LEACH-C, Improved-LEACH and DE-LEACH algorithms.When all nodes run out of energy in LEACH-C, Improved-LEACH and DE-LEACH, nodes in WDC-LEACH-C can still run for several additional rounds. Consequently, the network stability and life time in the proposed algorithm is much better with respect to DE-LEACH, Impro-LEACH and LEACH-C algorithms.

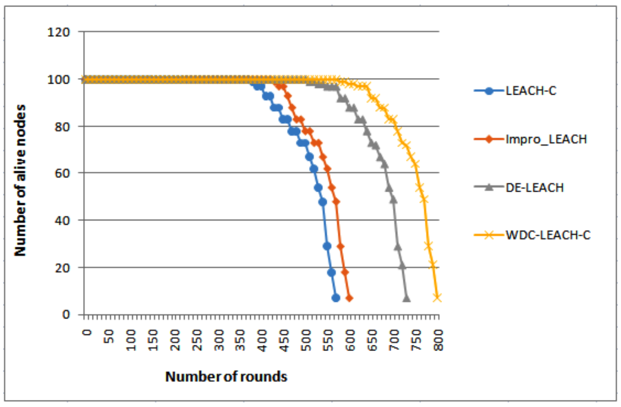


Figure 8: Number of alive nodes.

Figure 9 illustrates the energy consumption of nodes in network. Observed from this figure, we can obtain that our proposed protocol WDC-LEACH-C consumes less energy than LEACH-C, Improved-LEACH and DE-LEACH algorithms over the simulation rounds. This can be explained by the fact that the proposed algorithm provides better clustering and a good choice of CH. Since the CH is the WDC node, short transmission distances are used. Thus, network performance has been greatly improved.

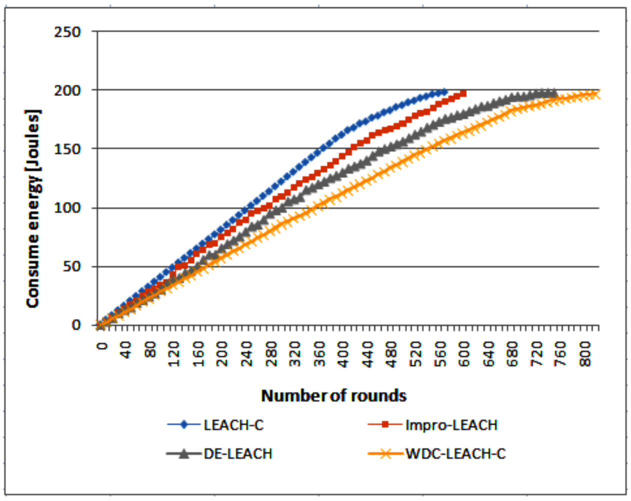


Figure 9: Energy consumption of nodes over simulation rounds.

Figure 10 shows the amount of data received by the base station. As shown in figure 10, the amount of data received by the base station is much greater in the proposed protocol compared to DE-LEACH, Impro-LEACH and LEACH-C.

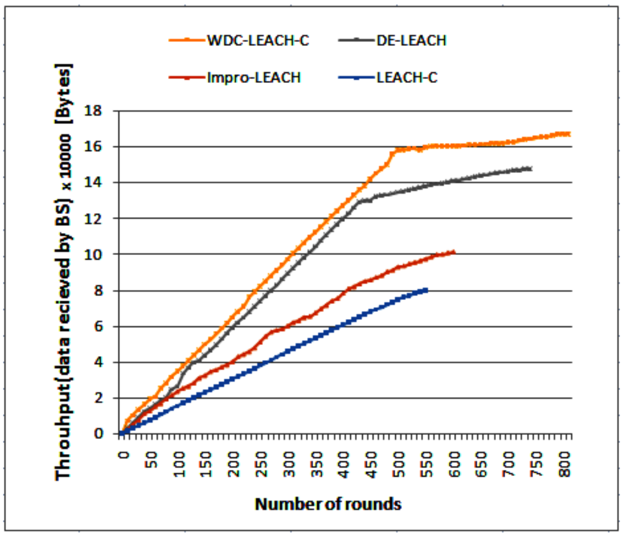


Figure 10: Quantity of data received by the BS.

Figure 11 shows that the total number of packets sent to the base station is higher in WDC-LEACH-C compared to DE-LEACH, Improved-LEACH and LEACH-C algorithms. These results include initial packets of energy and GPS position

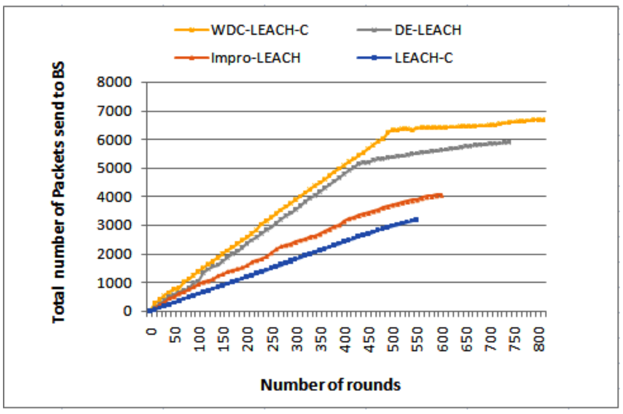


Figure 11: Number of packets received by the BS.

Table 2 summarizes the obtained results. As we can see from this table, the WDC-LEACH-C outperforms the other algorithms. This performance is in terms of energy consumption, prolonging network lifetime, data and received packets by the BS.

6 Conclusion

In this paper, WDC-LEACH-C a single hop communication protocol for homogenous WSNs is proposed. The developed algorithm is based on the weighted density center (WDC) of cluster nodes and on the residual energy in the selection of cluster head (CH). WDC-LEACH-C algorithm ensures better energy saving and a prolonging lifetime of the network compared to LEACH-C, Improved-LEACH and DE-LEACH algorithms.

7 References

[1] Akyildiz, I. F., Su, W., Sankarasubramaniam, Y., & Cayirci, E. (2002). Wireless sensor networks: a survey. *Computer networks*, *38*(4), 393-422.

[2] Baronti, P., Pillai, P., Chook, V. W., Chessa, S., Gotta, A., & Hu, Y. F. (2007). Wireless sensor networks: A survey on the state of the art and the 802.15. 4 and ZigBee standards. *Computer communications*, *30*(7), 1655-1695.

[3] Boyinbode, O., Le, H., & Takizawa, M. (2011). A survey on clustering algorithms for wireless sensor networks. *International Journal of Space-Based and Situated Computing*, *1*(2-3), 130-136.

[4] Aseri, T. C. (2014, March). Comparison of routing protocols in wireless sensor network using mobile sink-A survey. In *Engineering and Computational Sciences (RAECS), 2014 Recent Advances in* (pp. 1-4). IEEE.

[5] Afsar, M. M., & Tayarani-N, M. H. (2014). Clustering in sensor networks: A literature survey. *Journal of Network and Computer Applications*, *46*, 198-226.

[6] Jiang, C., Yuan, D., & Zhao, Y. (2009) Towards clustering algorithms in wireless sensor networks- a survey. In *Wireless Communications and Networking Conference, WCNC 2009. IEEE*, pp. 1-6, doi: 10.1109/WCNC.2009.4917996

[7] Kawadia, V., & Kumar, P. R. (2003, March). Power control and clustering in ad hoc networks. In *INFOCOM 2003. Twenty-Second Annual Joint Conference of the IEEE Computer and Communications. IEEE Societies* (Vol. 1, pp. 459-469). IEEE.

[8] Karenos, K., Kalogeraki, V., & Krishnamurthy, S. V. (2008). Cluster-based congestion control for sensor networks. *ACM Transactions on Sensor Networks (TOSN)*, *4*(1), 5.

[9] Ahmad, H., & Kohli, N. (2015, November). A survey on various static and mobile base stations and wireless charging of sensor nodes in WSNs. In *Communication Networks (ICCN), 2015 International Conference on* (pp. 16-22). IEEE.

[10] Heinzelman, W. R., Chandrakasan, A., &

Table 2. Simulation results.

Protocol	First node died in round	Last node died in round	Total energy consumption (Joules)	Total quantity of data sent to the BS (Bytes)	Total number of packets sent to BS
LEACH-C	382	569	376	79845	3193
Impro-LEACH	429	600	305	101434	4057
DE-LEACH	511	732	250	147599	5903
WDC-LEACH-C	581	801	217	167132	6680

Balakrishnan, H. (2000, January). Energy-efficient communication protocol for wireless microsensor networks. In *System sciences, 2000. Proceedings of the 33rd annual Hawaii international conference on* (pp. 10-pp). IEEE.

[11] Heinzelman, W. B., Chandrakasan, A. P., & Balakrishnan, H. (2002). An application-specific protocol architecture for wireless microsensor networks. *IEEE Transactions on wireless communications*, *1*(4), 660-670.

[12] Mhatre, V., & Rosenberg, C. (2004, June). Homogeneous vs heterogeneous clustered sensor networks: a comparative study. In *Communications, 2004 IEEE International Conference on* (Vol. 6, pp. 3646-3651). IEEE.

[13] Loscri, V., Morabito, G., & Marano, S. (2005, September). A two-level hierarchy for low-energy adaptive clustering hierarchy (TL-LEACH). In *Vehicular Technology Conference, 2005. VTC-2005-Fall. 2005 IEEE 62nd* (Vol. 3, pp. 1809-1813). IEEE.

[14] Zhao, F., Xu, Y., & Li, R. (2012). Improved LEACH routing communication protocol for a wireless sensor network. *International Journal of Distributed Sensor Networks*, *2012*.

[15] Kumar, S., Prateek, M., Ahuja, N.J., & Bhuchan, B. (2014, February). DE-LEACH Distance and Energy Aware LEACH. *International Journal of Computer Applications,* 88(9), 36-42.

[16] Runkler, T. A. (2008). Wasp swarm optimization of the c-means clustering model. *International Journal of Intelligent Systems*, *23*(3), 269-285.

[17] Pinto, P., Runkler, T. A., & Sousa, J. M. (2005). Wasp swarm optimization of logistic systems. In *Adaptive and Natural Computing Algorithms* (pp. 264-267). Springer, Vienna.

[18] Bonabeau, E., Dorigo, M., & Theraulaz, G. (1999). *Swarm intelligence: from natural to artificial systems* (No. 1). Oxford university Press.

[19] Engelbrecht, A. P. (2006). *Fundamentals of computational swarm intelligence*. John Wiley & Sons.

[20] Kim, J. M., Park, S. H., Han, Y. J., & Chung, T. M. (2008, February). CHEF: cluster head election mechanism using fuzzy logic in wireless sensor networks. In *Advanced communication technology, 2008. ICACT 2008. 10th international conference on* (Vol. 1, pp. 654-659). IEEE.

[21] Taheri, H., Neamatollahi, P., Younis, O. M., Naghibzadeh, S., & Yaghmaee, M. H. (2012). An energy-aware distributed clustering protocol in wireless sensor networks using fuzzy logic. *Ad Hoc Networks*, *10*(7), 1469-1481.

[22] Sharma, T., Kumar, B., Berry, K., Dhawan, A., Rathore, R. S., & Gupta, V. (2014, April). Ant based cluster head election algorithm in wireless sensor network to avoid redundancy. In *Communication Systems and Network Technologies (CSNT), 2014 Fourth International Conference on* (pp. 83-88). IEEE.

[23] Eschenauer, L., & Gligor, V. D. (2002, November). A key-management scheme for distributed sensor networks. In *Proceedings of the 9th ACM Conference on Computer and Communications Security* (pp. 41-47). ACM.

[24] Chidean, M. I., Morgado, E., del Arco, E., Ramiro-Bargueno, J., & Caamaño, A. J. (2015). Scalable data-coupled clustering for large scale WSN. *IEEE Transactions on Wireless Communications*, *14*(9), 4681-4694.

Persistent Homology and Machine Learning

Primož Škraba
Artificial Intelligence Laboatory[†], Jozef Stefan Institute
E-mail: primoz.skraba@ijs.si

Keywords: persistent homology, topological data analysis, overview

Received: March 27, 2018

In this position paper, we present a brief overview of the ways topological tools, in particular persistent homology, has been applied to machine learning and data analysis problems. We provide an introduction to the area, including an explanation as to how topology may capture higher order information. We also provide numerous references for the interested reader and conclude with some current directions of research.

Povzetek: V tem članku predstavljamo pregled topoloških orodij, predvsem vztrajno homologijo, ki je uporabna na področju strojnega učenja in za analizo podatkov. Začnemo z uvodom v področje in razložimo, kako topologija lahko zajame informacije višjega reda. Članek vsebuje tudi reference na pomembna dela za zainteresiranega bralca. Zaključimo s trenutnimi smernicami raziskav.

1 Introduction

Topology is the mathematical study of spaces via connectivity. The application of these techniques to data is aptly named topological data analysis (TDA). In this paper, we provide an overview of one such tool called persistent homology. Since these tools remain unfamiliar to most computer scientists, we provide a brief introduction before providing some insight as to why such tools are useful in a machine learning context. We provide pointers to various successful applications of these types of techniques to problems where machine learning has and continues to be used.

We begin with a generic TDA pipeline (Figure 1). The input is a set of samples, usually but not always embedded in some metric space. Based on the metric and/or additional functions (such as density), a multiscale representation of the underlying space of data is constructed. This goes beyond considering pairwise relations to include higher-order information. Persistent homology is then applied. This is a tool developed from algebraic topology, which summarizes the whole multiscale representation compactly in the form of a persistence diagram. This compact representation can then be applied to various applications.

The goal of this paper is to provide a brief overview and introduce the main components in the TDA pipeline.

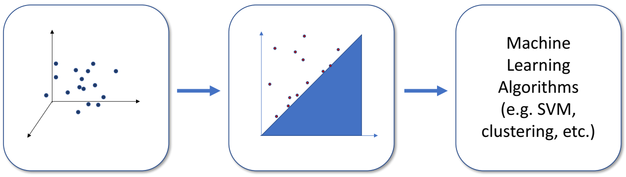


Figure 1: The TDA pipeline - taking in a points in in sime metric space along with potentially other information, the data is turned into a compact representation called a persistence diagram. This summary can then be input into machine learning algorithms rather than the raw point cloud.

2 Simplicial complexes

Representations of the underlying space are built up simple pieces glued together. There are many different approaches to this, however the simplest is perhaps the *simplicial complex*. A *simplex* is the convex combination of k points. A single point contains only itself, an edge is the convex combination of two points, three points make a triangle, four points a tetrahedron and so on (see Figure 2). More generally, a k-dimensional simplex is the convex combination of $(k+1)$ points. Just as an edge in a graph represents a pairwise relationship, triangles represent ternary relationships and higher dimensional simplices higher order relations. A graph is an example of a one-dimensional complex, as it represents all pairwise information - all higher order information is discarded. As we include higher dimensional simplices, we include more refined information yielding more accurate models. Note that these models need to not exist in an ambient space (i.e. may not be embedded), but rather represents connectivity information. The geometric realization of simplicial complexes has a long history of study in combinatorics but we do not address it here.

There are three main obstacles to this type of modeling. The first is lack of data. While it may be counterintuitive, in the age of big data we are often still faced with a lack of data. This is due to the non-uniformity and non-homogeneity of data. It may not make sense to consider 10-way relationships, if this data is only available for a small

[†] ARRS Project TopRep N1-0058

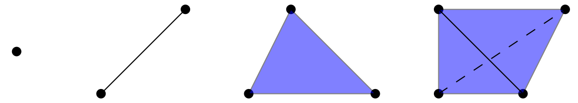


Figure 2: Simplicies come in different dimension. From left to right, a vertex is 0-dim, an edge is 1-dim, a triangle 2-dim and a tetrahedron is 3-dim.

subset of data. The second is computation. As we consider higher order relationships, there is often a combinatorial blow-up as one must consider all k-tuples, leading to preprocessing requirements which are simply not feasible. The final obstacle is interpretability. While we can understand a simplex locally, understanding the global structure becomes increasingly challenging.

This is the starting point for the tools we discuss below. Much of the effort of machine learning on graphs is trying to understand the qualitative properties of an underyling graph. This is often done by computing statistical features on the graph: degree distributions, centrality measures, diameter, etc. To capture higher order structure, we require a different set of tools. First, we note that a collection of simplices fit together. Just as in a graph, edges can only meet at an edge, simplices can only be glued together along lower dimensional simplices, e.g. triangles meet along edges or at a vertex. This represents a constraint on how simple building blocks (e.g. simplices) can be glued together to form a space. While this does not seriously limit the resulting spaces which can be represented, it does give us additional structure.

The starting point for the introduction is to describe the gluing map, called the *boundary operator*. For each k-simplex it describes the boundary as a collection of $k-1$ simplices. For example, the boundary of an edge consists of its two end points, the boundary of a triangle consists of its three edges (Figure 3). This can be represented as a matrix with the columns representing k-simplices and the rows $k-1$ simplices, which we denote ∂_k. The k-dimensional *homology* can be defined as

$$H_k = \frac{\text{ker } \partial_k}{\text{im } \partial_{k+1}}$$

The kernel is simply the collection of k-simplices which form the nullspace of the matrix which correspond to *cycles* (note that this agrees with the notion of graph-theoretic cycles). We the disregard all such cycles which bound regions filled-in by higher dimensional simplices. What remains is the numner of k-dimensional holes in the space. Specifically, 0-dimensional homology corresponds to the number of connected components, 1-dimensional homology the number of holes and so forth. The k-th Betti number, β_k is the number of independent such features. This is analogous to the rank of a matrix describing the number of basis elements a vector space has. This yields a qualitative description of the space. For a more complete introduction to homology, we recommend the book by Munkres [24] or the more advanced book by Hatcher[18]. An alternative intorduction which also includes persistent homology (described in the following section) can be found in Edlesbrunner and Harer[13]. Our goal here is to point out the intuition behind simplicial complexes and one approach to describing them qualitatively. We do note that the algorithms and implementations are readily available [2, 19, 25, 23] and can often be interpreted through linear algebra.

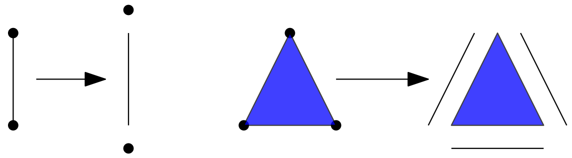


Figure 3: Simplicies are glued together in a specific way with each simplex is glued to lower dimensional simplices, called its boundary. Here we show an edge has 2 verticies as its boundary and a triangle has three edges as its boundary.

3 Persistent homology

One problem with homology and topological features in general is that they are unstable. Adding a point to a space changes the number of components and the correspoding Betti number. This would make it seems as though this technique were not suitable for the study of data. A key insight from [14, 39], is that we need not look at a single space but rather a sequence of spaces, called a *filtration*. This is an increasing sequence of nested spaces, which appears often when dealing with data.

$$\emptyset \subseteq X_0 \subseteq X_1 \subseteq \ldots \subseteq X_N$$

For example a weighted graph can be filtered by the edge weights. Perhaps the most ubiquitous example is a finite metric space, where the space is a complete graph and the weights are distances. This occurs whenever the notion of a “scale" appears, *Persistent homology* is the study of how qualitative features evolve over parameter choices. For example, the number of components is monotonically decreasing as we connect points which are increasingly far away. This is in fact precisely *single linkage clustering*. Higher dimensional features such as holes can appear and disappear at different scales.

The key insight is that the evolution of features over parameter choices can be encoded compactly in the form of a barcode or persistence diagram (Figure 4). We do not go into the algebraic reasons why this exists, rather we concentrate on its implications. An active research area has been to extend this to higher dimensional parameter spaces [6, 22, 34], but has remained a challenging area. We refer the reader to [13] for introductions to persistent homology and its variants. For the next section, rather than consider a persistence diagram rather than a barcode. Here each bar is mapped to a point with the starting point of the bar as the x-coordinate and the end point as the y-coordinate.

Consider a function on a simplicial complex, $f : K \to \mathbb{R}$ where we define the filtration as the *sublevel set* $f^{-1}(-\infty, \alpha]$. That is, we include all simplices with a lower function value. As we increase α, the set of simplicies with a lower function value only grows, hence we only add simplices. Therefore, we obtain an increasing sequence of topological spaces, i.e. a filtration. Define $X_\alpha := f^{-1}(-\infty, \alpha]$, then

$$X_{\alpha_1} \subseteq X_{\alpha_2} \subseteq \cdots \subseteq X_{\alpha_n} \qquad \alpha_1 \leq \alpha_2 \leq \cdots \leq \alpha_n$$

As another example in a metric space, we include all edges which represent a distance less than α. Consider a perturbed metric space, giving rise to a different function g. The following theorem establishes stability - that if the input (in this case, the function) does not change much, the output should not change much.

Theorem 1 ([11]). *Let K be two simplicial complexes with two continuous functions $f, g : X \to \mathbb{R}$. Then the persistence diagrams $\mathrm{Dgm}(f)$ and $\mathrm{Dgm}(g)$ for their sublevel set filtrations satisfy*

$$d_B(\mathrm{Dgm}(f), \mathrm{Dgm}(g))) \leq ||f - g||_\infty.$$

where $\mathrm{Dgm}(\cdot)$ represents the persistence diagram (i.e. a topological descriptor which is a set of points in $\mathbb{R}^2$) and $d_B(\cdot)$ represents bottleneck distance. This is the solution to the optimization which constructs a matching between the points in two diagrams which minimizes the maximum distance between matched points. While it is difficult to overstate the importance of this result, it does have some drawbacks. In particular the bound is in terms of the ∞-norm which in the presence of outliers can be very large. Recently this result has been specialized to Wasserstein stability, which is a much stronger result (albeit in a more limited setting).

Theorem 2 ([36]). *Let $f, g : K \to \mathrm{R}$ be two functions. Then $W_p(\mathrm{Dgm}(f), \mathrm{Dgm}(g)) \leq \|f - g\|_p$.*

Wasserstein distance is common in the machine learning and statistics literature as it is a natural distance between probability distributions. This recent result indicates that the distances between diagrams is indeed more generally stable and so suitable for applications. Stability has become an area of study in its own right and we now have a good understanding of the types of stabilty we can expect. The literature is too vast to list here so we limit ourselves to a few relevant pointers [3, 8].

4 Topological features

Here we describe some applications of persistence to machine learning problems. The key idea is to use persistence diagrams as feature vectors as input further machine learning algorithms, There are several obstacles to this. The most important is that the space of persistence diagrams is quite pathological. The first approach to move around this are *persistence landscapes* [4]. This lifts persistence diagrams into a Hilbert space which allows them to be fed into most standard machine learning algorithms. This has been followed up by rank functions [33], as well as several kernels [30], More recently, there has been work on learning optimal functions of persistence diagrams using deep learning [20].

There has also been significant work on the statistical properties of persistence diagrams and landscapes [16], including bootstrapping techniques [9].

These techniques have been applied to a number of application areas. Perhaps most extensive is in geometry processing. Combined with local features such as curvature or features based on heat kernels, different geoemtric structure can be extracted including symmetry [26], segmentation [35], and shape classification and retrival [7].

Another application area where persistence diagrams have been found to be informative are for biology, especially for protein docking [1] and modelling pathways in the brain [17]. The final application area we mention is material science. This is an area where machine learning has not yet been applied extensively. Partially due to the fact that the input is of a significantly different flavor than that which is typical in machine learning. For example, standard image processing techniques do not work well with scientific images such as electron microscope images. By using topological summaries, the relevant structure is well-captured [32, 21]. This area is still in the early stages with many more exciting developments expected.

We conclude this section by noting that persistence diagrams are not the only topological features which have been applied. Originally, the *Euler curve* was applied to fMRIs [38][1]. This feature has been extensively studied in the statistics literature, but is provably less informative than persistence diagrams - although it is far more computationally tractable. In addition to fMRI, it has been applied to various classification problems [31].

5 Other applications

In additon to providing a useful summary and features for machine learning algorithms, a second direction of interest is the map back to data. This inverse probelm is very difficult and can often be impossible in general. Nonetheless, the situation is often not as hopeless as it would seem. Some of the first work in this direction is re-interpreting single linkage clustering through the lens of persistence [10]. While it is well known that single linkage clusters are unstable, it is possible to use persistence to show that there exist stable parts of the clusters and a “soft" clustering algorithm can be developed to stabilize clusters, where each data point is assigned a probability that it is assigned to a given cluster. A current direction of research is to find similar stable representations in the data for higher dimensional structures (such as cycles).

[1] We note that this is where the term *topological inference* first used

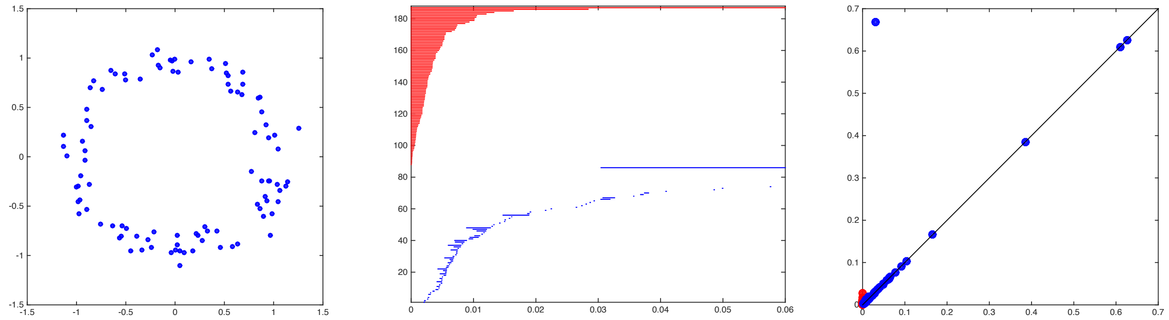


Figure 4: Persistence in a nutshell. Given input points (left), we compute a barcode (middle). which shows how long features live. The red shows the lifetimes of when components merge, while the blue bars show 1-dimensional holes. We can map each bar to a point by taking the start and end as the x and y coordinates respectively giving us the persistence diagram (right). Here we see that the big whole in the middle of the data set appears as a prominent feature (the blue dot far from the diagonal on the right).

A related problem is one of parameterization. That is, find intrinsic coordinates describing the data, extending successful techniques in dimensionality reduction, This includes linear methods such as PCA and MDS as well as non-linear methods such as ISOMAP and LLE. The first such work coordinizaed the space of textures using a Klein bottle as the underlying model [28] - a topological model found a few years prior [5]. This was however built by hand. The first class of general methods is first to map cicrular coordinates to data [12]. This is particularly useful when dealing with recurrence in time-varying systems. Recurrence (including periodicity) is naturally modeled by an angle, Combining persistence with least-squares optimization provides an automatic pipeline to finding such coordinates. This was applied to characterizing human motions such as different walks and other activities [37]. Further work has shown how to construct coordainte systems for higher dimensional structures based on the projective plane [27].

The final direction we consider is to encode topological constraints in machine learning algorithms. In [29] topological priors were used to aid in parameter selection. For example, the reconstruction of a racetrack should have one component and one hole (the main loop). Computing the persistence with respect to a reconstruction parameter (e.g. bandwith of a kernel) can allow us to choose a parameter value where the reconstruction has the desired topological “shape." The encoding of topological constraints is still in the very early stages but has the potential to provide a new type of regularization to machine learning techniques.

6 Discussion

Topological data analysis and applications of topology are still in their early stages. Various efforts to bridge the gap between algebraic topology and statistics (and probability) has made rapid progress over the last few years which has culminated in a dedicated R-package [15]. At the same time, increasingly efficient software exists for computing persistent homology exists, where now it is feasible to consider billions of points in low dimensions. This is increasingly bridging the gap between theory and practice.

The area has undergone rapid development over the last 10 years and is showing no signs of slowing down. In terms of theory, the primary question drinving the community is the notion of multi-dimensional or multi-parameter persistence, where the computational obstacles are much more daunting. Nonetheless, progress is being made. Success promises to further reduce the need and dependence on parameter tuning.

The combination of deep learning techniques with topological techniques promises to provide new areas of applications as well as potentially performance. These methods are primarily complementary allowing them to build on each other. In conclusion, while obstacles remain, the inclusion of topological techniques into the machine learning toolbox is rapidly making progress.

References

[1] Pankaj K Agarwal, Herbert Edelsbrunner, John Harer, and Yusu Wang. Extreme elevation on a 2-manifold. *Discrete &amp; Computational Geometry*, 36(4):553–572, 2006.

[2] U Bauer. Ripser. https://github.com/Ripser/ripser, 2016.

[3] Ulrich Bauer and Michael Lesnick. Induced matchings of barcodes and the algebraic stability of persistence. In *Proceedings of the thirtieth annual symposium on Computational geometry*, page 355. ACM, 2014.

[4] Peter Bubenik. Statistical topological data analysis using persistence landscapes. *The Journal of Machine Learning Research*, 16(1):77–102, 2015.

[5] Gunnar Carlsson, Tigran Ishkhanov, Vin De Silva, and Afra Zomorodian. On the local behavior of spaces of natural images. *International journal of computer vision*, 76(1):1–12, 2008.

[6] Gunnar Carlsson and Afra Zomorodian. The theory of multidimensional persistence. *Discrete &amp; Computational Geometry*, 42(1):71–93, 2009.

[7] Frédéric Chazal, David Cohen-Steiner, Leonidas J Guibas, Facundo Mémoli, and Steve Y Oudot. Gromov-hausdorff stable signatures for shapes using persistence. In *Computer Graphics Forum*, volume 28, pages 1393–1403. Wiley Online Library, 2009.

[8] Frédéric Chazal, Vin De Silva, Marc Glisse, and Steve Oudot. The structure and stability of persistence modules. *arXiv preprint arXiv:1207.3674*, 2012.

[9] Frédéric Chazal, Brittany Terese Fasy, Fabrizio Lecci, Alessandro Rinaldo, Aarti Singh, and Larry Wasserman. On the bootstrap for persistence diagrams and landscapes. *arXiv preprint arXiv:1311.0376*, 2013.

[10] Frédéric Chazal, Leonidas J Guibas, Steve Y Oudot, and Primoz Skraba. Persistence-based clustering in riemannian manifolds. *Journal of the ACM (JACM)*, 60(6):41, 2013.

[11] David Cohen-Steiner, Herbert Edelsbrunner, and John Harer. Stability of persistence diagrams. *Discrete &amp; Computational Geometry*, 37(1):103–120, 2007.

[12] Vin De Silva, Dmitriy Morozov, and Mikael Vejdemo-Johansson. Persistent cohomology and circular coordinates. *Discrete &amp; Computational Geometry*, 45(4):737–759, 2011.

[13] Herbert Edelsbrunner and John Harer. *Computational topology: an introduction*. American Mathematical Soc., 2010.

[14] Herbert Edelsbrunner, David Letscher, and Afra Zomorodian. Topological persistence and simplification. In *Foundations of Computer Science, 2000. Proceedings. 41st Annual Symposium on*, pages 454–463. IEEE, 2000.

[15] Brittany Terese Fasy, Jisu Kim, Fabrizio Lecci, and Clément Maria. Introduction to the r package tda. *arXiv preprint arXiv:1411.1830*, 2014.

[16] Brittany Terese Fasy, Fabrizio Lecci, Alessandro Rinaldo, Larry Wasserman, Sivaraman Balakrishnan, Aarti Singh, et al. Confidence sets for persistence diagrams. *The Annals of Statistics*, 42(6):2301–2339, 2014.

[17] Margot Fournier, Martina Scolamiero, Mehdi Gholam-Rezaee, Hélène Moser, Carina Ferrari, Philipp S Baumann, Vilinh Tran, Raoul Jenni, Luis Alameda, Karan Uppal, et al. M3. topological analyses of metabolomic data to identify markers of early psychosis and disease biotypes. *Schizophrenia Bulletin*, 43(suppl_1):S211–S212, 2017.

[18] Allen Hatcher. *Algebraic topology*. 2002.

[19] Gregory Henselman and Robert Ghrist. Matroid filtrations and computational persistent homology. *arXiv preprint arXiv:1606.00199*, 2016.

[20] Christoph Hofer, Roland Kwitt, Marc Niethammer, and Andreas Uhl. Deep learning with topological signatures. In *Advances in Neural Information Processing Systems*, pages 1633–1643, 2017.

[21] Yongjin Lee, Senja D Barthel, Paweł Dłotko, S Mohamad Moosavi, Kathryn Hess, and Berend Smit. Quantifying similarity of pore-geometry in nanoporous materials. *Nature Communications*, 8, 2017.

[22] Michael Lesnick. The theory of the interleaving distance on multidimensional persistence modules. *Foundations of Computational Mathematics*, 15(3):613–650, 2015.

[23] Dmitriy Morozov. Dionysus. *Software available at http://www. mrzv. org/software/dionysus*, 2012.

[24] James R Munkres. *Elements of algebraic topology*, volume 4586. Addison-Wesley Longman, 1984.

[25] Vidit Nanda. Perseus: the persistent homology software. *Software available at http://www. sas. upenn. edu/˜ vnanda/perseus*, 2012.

[26] Maks Ovsjanikov, Quentin Mérigot, Viorica Pătrăucean, and Leonidas Guibas. Shape matching via quotient spaces. In *Computer Graphics Forum*, volume 32, pages 1–11. Wiley Online Library, 2013.

[27] Jose A Perea. Multi-scale projective coordinates via persistent cohomology of sparse filtrations. *arXiv preprint arXiv:1612.02861*, 2016.

[28] Jose A Perea and Gunnar Carlsson. A klein-bottle-based dictionary for texture representation. *International journal of computer vision*, 107(1):75–97, 2014.

[29] Florian T Pokorny, Carl Henrik Ek, Hedvig Kjellström, and Danica Kragic. Topological constraints and kernel-based density estimation. *Advances in Neural Information Processing Systems*, 25, 2012.

[30] Jan Reininghaus, Stefan Huber, Ulrich Bauer, and Roland Kwitt. A stable multi-scale kernel for topological machine learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4741–4748, 2015.

[31] Eitan Richardson and Michael Werman. Efficient classification using the euler characteristic. *Pattern Recognition Letters*, 49:99–106, 2014.

[32] Vanessa Robins, Mohammad Saadatfar, Olaf Delgado-Friedrichs, and Adrian P Sheppard. Percolating length scales from topological persistence analysis of micro-ct images of porous materials. *Water Resources Research*, 52(1):315–329, 2016.

[33] Vanessa Robins and Katharine Turner. Principal component analysis of persistent homology rank functions with case studies of spatial point patterns, sphere packing and colloids. *Physica D: Nonlinear Phenomena*, 334:99–117, 2016.

[34] Martina Scolamiero, Wojciech Chachólski, Anders Lundman, Ryan Ramanujam, and Sebastian Öberg. Multidimensional persistence and noise. *Foundations of Computational Mathematics*, 17(6):1367–1406, 2017.

[35] Primoz Skraba, Maks Ovsjanikov, Frederic Chazal, and Leonidas Guibas. Persistence-based segmentation of deformable shapes. In *Computer Vision and Pattern Recognition Workshops (CVPRW), 2010 IEEE Computer Society Conference on*, pages 45–52. IEEE, 2010.

[36] Primoz Skraba and Katharine Turner. Wasserstein stability of persistence diagrams. submitted to the Symposium of Computational Geometry 2018.

[37] Mikael Vejdemo-Johansson, Florian T Pokorny, Primoz Skraba, and Danica Kragic. Cohomological learning of periodic motion. *Applicable Algebra in Engineering, Communication and Computing*, 26(1-2):5–26, 2015.

[38] Keith J Worsley, Sean Marrett, Peter Neelin, Alain C Vandal, Karl J Friston, Alan C Evans, et al. A unified statistical approach for determining significant signals in images of cerebral activation. *Human brain mapping*, 4(1):58–73, 1996.

[39] Afra Zomorodian and Gunnar Carlsson. Computing persistent homology. *Discrete &amp; Computational Geometry*, 33(2):249–274, 2005.

Research on Intelligent English Oral Training System in Mobile Network

Fen Zhu
Foreign Languages School of Luoyang Institute of Science and Technology, Luoyang, Henan, 471000, China
E-mail: fenzhu_lit@163.com

Technical Paper

Keywords: mobile network, Android system, spoken English, resonance peak, evaluation

Received: March 13, 2018

With the rapid development of mobile networks, mobile learning, as a new learning form, is gradually accepted by people. Based on the Android mobile platform, this paper designed a spoken English training system that could be applied to mobile network equipment from the aspects of speech recognition, pronunciation scoring and function setting. Based on the characteristics of the Android system, this paper selected the MEL cepstrum coefficient as the feature parameters to speech recognition, and introduced the dynamic time neat algorithm as the matching algorithm of the speech recognition pattern to make speech recognition more suitable for mobile Internet devices. Besides, the voice formant was used as a reference for oral scores and the scoring method based on single reference template was adopted. Finally, the spoken English training system was developed under the eclipse integration environment. The test results showed that the success rate of voice input was over 98%, and the accuracy rate of spoken voices of monophthong words, diphthong words and polysyllabic words was 97.15%, 94.96% and 93.62% respectively, suggesting that the system could accurately input and score English learners' spoken English, and assist English pronunciation.

Povzetek: Prispevek se ukvarja z mobilnim učenjem angleščine na sistemih z Androidom..

1 Introduction

With the deepening of economic globalization, communication between China and other counties has become increasingly frequent. Therefore English which is the most extensively applied language worldwide has gradually been an indispensable tool in daily life and work, and moreover many English training institutions and learning tools have emerged. But the traditional learning mode, i.e. face-to-face teaching mode in training institutions, usually cannot achieve a good result in spoken English, which contributes to the large difference of pronunciation between English and Chinese. People who grow up in Chinese environment will make the mistake of pronunciation unconsciously when learning oral English. Moreover English teachers who have correct pronunciation and are able to guide pronunciation are lack of in China. Time and environment for spoken English practice are also not enough.

With the rapid development of mobile information technology, mobile network terminals such as smartphone and panel personal computer have almost covered every aspect of our life. Smartphone based oral English training software is more convenient and practical compared to the traditional teaching mode and can effectively avoid the shortcomings of the traditional teaching mode. Mobile network device based mobile learning has been extensively studied. Wang et al. [1] found that computer corpus based teaching mode was more effective than the traditional teaching mode. Alamer et al. [2] designed and develop mobile Web technology and API based lightweight language learning management system. The system aimed to allow language students to view and download learning content on their phones and complete interactive tasks designed by teachers. Milutinovic [3] et al. proposed a mobile adaptive language learning model, whose main goal was to improve the mobile language learning process using adaptive technology. The proposed model was designed to take advantage of unique opportunities to transfer learning content in real learning situations. Taking Android smartphone as the application platform, this study aimed to build an intelligent spoken English training system that could be used on mobile network devices.

2 Mobile learning

Mobile learning [5] refers to the use of portable mobile communication equipment and technology so that learners can choose their preferred way to study any time and place. Compared with the time fixed English classroom learning mode, mobile learning has extensiveness, timeliness and interactivity features, giving learners a more relaxed and pleasant learning experience. In addition, the multimedia combination of audio, text, video, image and animation makes mobile learning more vivid. Mobile language learning enables learners to have more learning options, to make full use of fragmented time, and to be efficient and flexible.

3 Intelligent spoken English training system design

3.1 Speech recognition

3.1.1 Speech signal preprocessing

(1) Speech signal digitization

Speech signal can be analyzed and processed by computer through digital conversion. This paper uses the headset of Android phone as the input device of voice signal, and uses the Audio Record Wizard [6] of Android system to collect the underlying data. According to Nyquist frequency theorem, the sampling frequency of 7000 Hz is used to collect the speech signal.

(2) Pre-emphasis

In order to eliminate the influence of mouth and nose radiation, speech signals are usually pre-emphasized by a first-order high-pass filter [7]. Pre-emphasis refers to improving the resolution of the high-frequency part of speeches by emphasizing the high-frequency part of speeches based on the difference between signal properties and noise properties. Usually pre-emphasis is realized using first-order FIR high-pass digital filter [16]. The formula used by the filter is shown below.

$$H(x) = 1 - \varepsilon x^{-1}, \tag{1}$$

Where ε refers to the pre-emphasis coefficient and is set to 0.98 in this study. Set the speech signal at the nth time point to be $s(n)$, then the weighted signal is:

$$s_2(n) = s(n) - \varepsilon s(n-1), \tag{2}$$

Where $s_2(n)$ refers to the speech signal after pre-emphasis and $s(n-1)$ refers to the last filter output value.

(3) Windowing processing

In order to ensure continuous and complete voice signals in each frame, a window function is generally multiplied before processing each frame of speech [8]. This paper uses Hamming window function to window the signal, with the formula as follows:

$$w(n) = \begin{cases} 0.54 - 0.46\cos[2\pi n/(N-1)], & 0 \le n \le N-1 \\ 0, & n = othervalue \end{cases}. \tag{3}$$

(4) Endpoint detection

According to the characteristics of the Android platform, this paper uses the combination of short time average energy [9] and short-time zero-crossing rate to detect the endpoint. The short-term average energy is calculated as follows: set the short time average energy and frame length of the n-th frame of speech signal $s_n(h)$ to E_n and N respectively, then the calculation formula is as follows:

$$E_n = \sum_{m=0}^{N-1} x_n^2(h), 0 \le h \le N-1 \tag{4}$$

According to the size of the short-term energy, the learner's voice and noise can be distinguished, and high energy signal is the speech signal. However, this method is less stable under low SNR conditions. Therefore, it is necessary to use short-time zero-rate method. Set the speech signal to be $x_n(m)$, then the short-time zero-crossing rate is:

$$z_n = \frac{1}{2}\sum_{m=0}^{N-1} \left|\operatorname{sgn}[x_n(h)] - \operatorname{sgn}[x_n(h-1)]\right| \quad \operatorname{sgn}[x] = \begin{cases} 1 & (x \ge 0) \\ -1 & (x \le 0) \end{cases}, \tag{5}$$

Where $\operatorname{sgn}[\ ]$ refers to the sign function. According to the low frequency band of voiced sound energy and the high frequency band of voiceless sound energy, the zero-crossing rate of the speaker is stable relative to the ambient noise and the sound segment can be clearly identified.

3.1.2 Extraction of speech signal features

Feature extraction [17] was performed after the preprocessing to highlight the data features of pattern matching, improve recognition rate, compress information and reduce computation load and storage. The commonly used feature parameters include Mel-frequency cepstral coefficient (MFCC) which has strong recognition performance and anti-noise capacity, linear predictive coefficient which has small computer load but general efficacy and accent sensitivity parameter which has favorable performance in recognition the middle frequency band of signals.

In this system, Mel Frequency Cepstrum Coefficient (MFCC) [10] is used as the characteristic parameter of oral training. MEL scale and frequency have the following relationship:

$$f_{mel} = 2595 \ln(1 + f/700), \tag{6}$$

Where f refers to the actual frequency of the signal.

Fourier transform [11] is performed on each frame of speech signal after preprocessing to obtain the signal spectrum. Then, the spectrum square is cut off, Mel band-pass filter is applied for filtering, all of the filter outputs undergo logarithm calculation, and then discrete cosine transform is made on DCT to obtain MFCC, the process is shown in Figure 1.

$$C(n) = \sqrt{\frac{2}{N}} \log w(l) \cos\left\{\left(l - \frac{1}{2}\right)\frac{n\pi}{L}\right\}, (n = 1,2,\ldots,p)(l = 1,2,\ldots,L)' \tag{7}$$

Where L refers to the number of filters, $w(l)$ refers to the output of each triangle filter, N refers to the length of each frame, and p refers to the order of parameters.

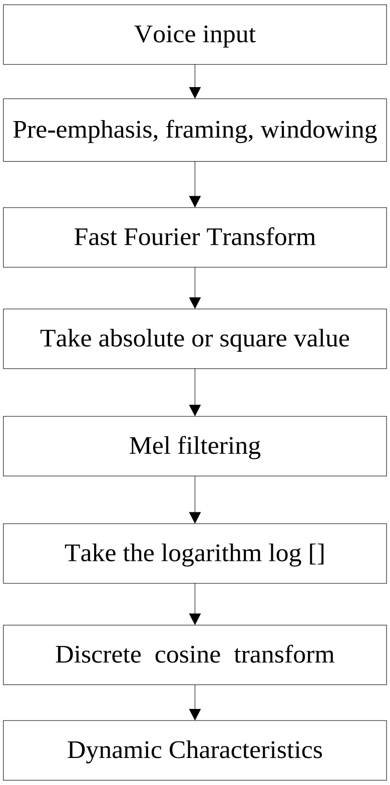

Voice input
Pre-emphasis, framing, windowing
Fast Fourier Transform
Take absolute or square value
Mel filtering
Take the logarithm log []
Discrete cosine transform
Dynamic Characteristics

Figure 1: MFCC feature extraction process.

3.2 Speech signal pattern matching

In this paper, Dynamic Time Warping (DTW) [12] is used to match the characteristics of speech signals. Firstly, set the eigenvector sequence of the standard template to be $B=\{B(1),B(2),\ldots,B(m),\ldots,B(M)\}$, where M is the total speech frame number, m is the time series label of the signal frame and $B(m)$ is the eigenvector of the m-th frame.

The eigenvector sequence of the speech test template is $T=\{T(1),T(2),\ldots,T(n),\ldots,T(Q)\}$, where Q is the number of frames, n is the sequence number of the speech in the template and $T(n)$ is the eigenvector of the nth frame. The similarity between the test template and the standard template is represented by vector distance, and the similarity decreases with the increase of vector distance. Euclidean distance [13] is used to represent the distance between $T(n)$ and $B(m)$, as follows:

$$d[T(n),B(m)]=\sum_{i=1}^{p}(t_i-b_i)^2 \quad , \tag{8}$$

Where t_i refers to the eigenvector of the i-th dimension of $T(n)$, b_i refers to the eigenvector of the i-th dimension of $B(m)$. The dynamic time warping is to map the time axis n of the speech test template to the time axis m of the standard template to obtain the minimum vector distance of the template, as follows:

$$D=\min_{w(n)}\sum_{n=1}^{N} d[T(n),b(m)] \tag{9}$$

Dynamic time warping generally requires finding a path which goes through each intersection with the distance measure sum of the intersection at the path minimized. Generally, constraint conditions are given:

Boundary condition:

$$w(1)=1, w(N)=M \tag{10}$$

Continuity condition:

$$w(n+1)-w(n)=\begin{cases}0,1,2 & w(n)\neq w(n-1)\\ 1,2 & w(n)=w(n-1)\end{cases} \tag{11}$$

With the above two conditions met and the frame distance accumulated sum the minimum, the optimal path $m=w(n)$ is sought as follows: starting from (1, 1), backstepping is repeated until (N, M) to find the optimal matching path. $D(N,M)$ refers to template distance of the matching path and the minimum matching distance is $D\min(N,M)$, which is taken as the measuring criterion for the similarity matching degree between templates.

3.3 Pronunciation scoring

Firstly, the average matching distance of frames is calculated:

$$\bar{d}=\frac{D(N.M)}{N} \quad , \tag{12}$$

Where $D(N,M)$ refers to the total matching distance of the test templates, N refers to the frame length of the test templates. When selecting the average frame matching distance, the effect of the speech length is eliminated. In terms of scoring, this paper proposes a scoring method based on the single reference template. The range of pronunciation score is 0~100, and the scoring method is as follows:

$$score=\frac{100}{1+e(d)^f}\,, \tag{13}$$

Where d refers to the average frame matching degree, and e and f are the scoring parameters obtained based on the experience of spoken English teachers and matching distance.

3.4 Scoring parameter selection

In this study, the formant was taken as a criterion to evaluate the learner's spoken language pronunciation, and the learners' spoken English pronunciation quality was judged by the similarity contrast between the pronunciation formant of the test model and the standard model. Formant refers to the areas where energies are concentrated in the speech spectrum and it reflects the physical characteristics of the resonant cavity. In the process of producing vowels and consonants in the oral cavity, the harmonic vibration frequency of the sound is regulated by the sound cavity, which is strengthened or attenuated irregularly, and the region with high degree of enhancement forms the resonance peak. In the spectrum of vowels, the first three resonant peaks play a key role in the quality of sound. The first two resonant peaks are particularly sensitive to the height of the tongue position. The higher the first resonance peak, the lower the tongue position, and the second and third formants also have a certain relationship with the tongue position, but the relationship between them is not particularly prominent. Therefore, the first resonance peak is chosen as the judging basis for the pronunciation quality. In this paper, the resonance peak is extracted using linear prediction method [14]. Regarding the sound channel as a resonant cavity, then the resonant peak is the resonant frequency of the wall.

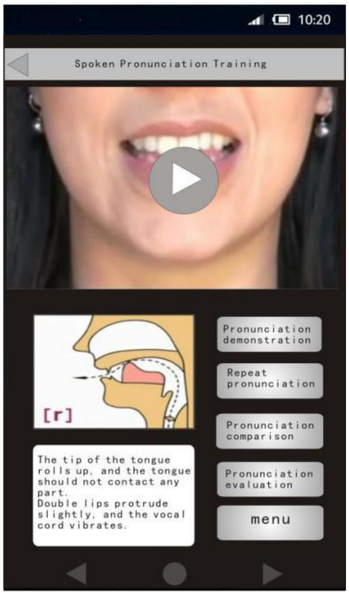
3.5 Function and interface design

The oral English training system based on the Android smartphone platform can provide effective feedback to learners' oral English pronunciation through animation, audio, video and image forms. The function design of the system is as follows: First of all, the system should have standard pronunciation audios and videos to guide learners, and introduce the key points of English pronunciation and tongue type in the form of pictures and texts. Before establishing the system, spoken phonetic materials such as phonetic symbols, words, and sentences need to be collected. Folders of pictures, videos and texts should be established separately for system access. We use AudioTrack for audio and video playback, specifically, class method for speech signal playback and Video View class method in Android SDK for video playback. Secondly, the system should be able to prompt the learner to read the words and phrases, record and play back the voice signals, create a cache folder, and record the recorded voice signals according to the MP3 format. AudioRecond class method is used to record voice signals, and the sampling frequency is set to 8000Hz, channel mono, 16-bit sampling bits.

Then, the system uses the speech recognition and related algorithms to score learners' spoken pronunciations and establish a spoken appraisal folder. The Shared Preferences component of the Android system is used to store the learner's spoken rating results. Finally, the system should have the function of comparing learner's spoken pronunciation with standard pronunciation, and use the Achart Engine to show the comparison chart of the changes of formant to the oral learners so as to make the learners find the problems more intuitively. Besides, the system should give advice on spoken pronunciation based on the relationship between tongue shape, mouth shape and signal formant.

Interface design: The main interface includes four oral training options of vowels, consonants, words and sentences and the learners can choose the items according to their own willingness. At the same time, the resonance peak comparison chart and historical scoring items are added on the main interface to facilitate learners to view. Help options and exit keys are also set. Training score interface elements include pronunciation demonstration, pronunciation following, pronunciation contrast, pronunciation evaluation, main menu, oral demonstration (animation, audio and video, pictures and other forms) and the corresponding text description.

The development of the system is mainly done in the Eclipse integration environment. Specific development and operating environment: PC operating system Windows7 (32bt); Development components: Java JDK 8.0, Eclipse [15] 4.5 (Mars), Android SDK 4.0; Hardware Environment: Glory Play 6X (RAM: 3GB, ROM: 32G, Android 6.0); Programming Language: Java. Figure 2 shows the interface effect.



The tip of the tongue rolls up, and the tongue should not contact any part.
Double lips protrude slightly, and the vocal cord vibrates.

Figure 2: Oral training system main interface and rating interface.

4 System test results

This study invites three experienced English teachers as score judges and 10 college students as the subjects of the oral English training system. The scoring is based on tongue type, mouth type, pronunciation completeness and clarity, 25 points for each item. The average of the scores given by the three teachers is taken as the final score.

4.1 Speech input test

First, the subjects' speech was recorded and the recognition rate of the speech input system was tested. According to the system instructions, the subjects read after the system of 20 monophthong words, 15 diphthong words and 15 polysyllabic words. The three teachers judged whether the speech input was successful and the results are shown in Table 1.

Word type	Monophthong	Diphthong	Polysyllabic
Total number	20	15	15
Accuracy	100%	100%	96%

Table 1: Speech input success rate of the system.

4.2 Scoring accuracy test

Based on the speech input, scores were given by the system and the three teachers respectively. Suppose the system score of the i-th word was x_i, and the score by the teachers was y_i, the similarity of the two scores was calculated according to the formula $\mu_i = 1 - \frac{|x_i - y_i|}{y_i}$, then, the scoring accuracy of n samples can be calculated based on $\bar{\mu} = \frac{(\mu_1 + \mu_2 + ... + \mu_n)}{n}$, as shown in Table 2.

Word type	Monophthong	Diphthong	Polysyllabic
Total number	20	15	15
Accuracy	97.15%	94.96%	93.62%

Table 2: System test accuracy results.

As shown in Table 2, the accuracy on monophthong word pronunciation reached 97.15%, and that on diphthong words and polysyllabic words reached 94.96% and 93.62% respectively. With the increase of vowels in the words, the pronunciation became complicated, which affected the scoring mode. But, the scoring accuracy reached above 90% on average.

5 Conclusion

With the rapid development of mobile network and the upgrading of mobile network equipment, the concept of mobile learning has been gradually integrated into our life. This paper focused on mobile learning and designed an oral English training system that could be used on Android smartphones. Firstly, we designed the speech signal preprocessing, feature extraction and signal pattern matching of system speech recognition. According to the characteristics of Android system, the dynamic time regulation algorithm with small amount of computation was introduced as the pattern matching algorithm of speech signals. Then, according to the pronunciation characteristics of spoken English, we selected the pronunciation resonance peak as the reference of system scoring, and determined the single reference template as the scoring method. Afterwards, the system functions were designed from the three aspects of pronunciation demonstration, pronunciation imitation and pronunciation evaluation. Finally, the system was tested, the results of which showed that the system had a high success rate in the recognition of the spoken word pronunciation and a high accuracy in spoken English scoring. In general, the system we designed initially met the needs of speech accuracy and scoring accuracy of mobile spoken English training, and provided some points that need attention in pronunciation, which is helpful for spoken English training.

6 References

[1] An L L, Wu Y N, Liu Z, Liu RS (2012). An Application of Mispronunciation Detecting Network for Computer Assisted Language Learning System. Journal of Electronics & Information Technology, 34(9), pp. 2085-2090.

[2] Alamer R A, Al-Otaibi H M, Al-Khalifa H S (2015). L3MS: A Lightweight Language Learning Management System Using Mobile Web Technologies, 2015 IEEE 15th International Conference on Advanced Learning Technologies (ICALT), IEEE, Hualien, Taiwan, pp. 326-327.

[3] Milutinovic M, Bojovic Z, Labus A, Bogdanovic B, Despotovic-Zrakic M (2016). Ontology-based generated learning objects for mobile language learning. Computer Science & Information Systems, pp. 4-4.

[4] Troussas C, Virvou M, Alepis E (2014). Multifactorial user models for personalized mobile-assisted language learning. Frontiers in Artificial Intelligence & Applications, 262, pp. 275-282.

[5] Sharples M, Arnedillosánchez I, Milrad M, Vavoula G (2014). Mobile Learning. R Keith Sawyer, pp. 501-521.

[6] Hu Y, Azim T, Neamtiu I (2015). Versatile yet lightweight record-and-replay for Android. ACM Sigplan International Conference on Object-Oriented Programming, Systems, Languages, and Applications, ACM, Newyork, USA, pp. 349-366.

[7] Deepa D, Shanmugam A (2011). Enhancement of noisy speech signal based on variance and modified gain function with PDE preprocessing technique for digital hearing aid. Journal of Scientific & Industrial Research, 70(5), pp. 332-337.

[8] Takagi T, Seiyama N, Miyasaka E (2015). A method for pitch extraction of speech signals using autocorrelation functions through multiple window lengths. Electronics & Communications in Japan, 83(2), pp. 67-79.

[9] Sahoo T R, Patra S (2014). Silence Removal and Endpoint Detection of Speech Signal for Text Independent Speaker Identification. International Journal of Image Graphics & Signal Processing, 6(6), pp. 27-35.

[10] Valentini-Botinhao C, Yamagishi J, King S (2012). Mel cepstral coefficient modification based on the Glimpse Proportion measure for improving the intelligibility of HMM-generated synthetic speech in noise. Proc. Interspeech., 631-634.

[11] Rathore P S, Boyat A, Joshi B K (2013). Speech signal analysis using Fourier-Bessel Expansion and Hilbert Transform Separation Algorithm. IEEE International Conference on Signal Processing, Computing and Control, IEEE, Solan, India, pp. 1-4.

[12] Dhingra S, Nijhawan G, Pandit P (2013). Isolated speech recognition using MFCC and DTW. International Journal of Advanced Research in Electrical Electronics & Instrumentation Engineering, 2(8), pp. 4085-4092.

[13] Lang F Y, Li X G (2012). Multi-Sensors Information Fusion Based on Momentis Method and Euclid Distance. Advanced Materials Research, 383-390(383-390), pp. 5447-5452.

[14] Yusnita M A, Paulraj M P, Yaacob S, Bakar SA, Saidatul A (2011). Malaysian English accents identification using LPC and formant analysis. IEEE International Conference on Control System, Computing and Engineering, IEEE, Penang, Malaysia, pp. 472-476.

[15] Wang L, Groves P, Ziebart M (2013). Urban Positioning on a Smartphone: Real-time Shadow Matching Using GNSS and 3D City Models. Proceedings of the 26th International Technical Meeting of The Satellite Division of the Institute of Navigation, Nashville Convention Center, pp. 1606-1619.

[16] Thakral S, Goswami D, Sharma R, Prasanna CK, Joshi AM (2016). Design and implementation of a high speed digital FIR filter using unfolding. IEEE, Power India International Conference, pp. 1-4.

[17] Han Z, Wang J (2016). Dynamic feature extraction for speech signal based on MUSIC. Control and Decision Conference, pp. 3770-3773.

Probability Matrix Decomposition Based Collaborative Filtering Recommendation Algorithm

Yili Tan and Huijuan Zhao
Department of Statistics, College of Science, North China University of Science and Technology, Hebei, 063210, China
E-mail: tangyili_tyl@163.com

Yourong Wang
Department of Basic, Tangshan College, Hebei, 063000, China

Min Qiu
Public Mathematics Department, Hubei University of Automotive Technology, Hubei, 442002, China

Technical Paper

Keywords: probability matrix decomposition, collaborative filtering, recommendation algorithm

Received: April 8, 2018

With the development of the society, the increased amount of information has extensively appeared on the Internet. It includes almost all the content we need. But information overload makes people unable to correctly find the information they need. Collaborative filtering recommendation algorithm can recommend items for users according to their demands. But traditional recommendation algorithm which has defects such as data sparsity needs to be improved. In this study, the collaborative filtering recommendation algorithm was analyzed, an improved collaborative filtering recommendation algorithm based on the probability matrix decomposition was put forward, and the feasibility of the algorithm was verified. Moreover the traditional algorithms including user based collaborative filtering algorithm, item based collaborative filtering algorithm, singular value decomposition based collaborative filtering algorithm and basic matrix based collaborative filtering algorithm were tested. The test results demonstrated that the proposed algorithm had a higher accuracy compared to the traditional algorithms, and its mean absolute error and root-mean-square error were significantly smaller than those of the traditional algorithms. Therefore it can be applied in the daily life.

Povzetek: V sestavku je predstavljena dekompozicija verjetnostne matrike s priporočilnim algoritmom na osnovi skupinskega filtriranja.

1 Introduction

He increased amount of information which appeared due to the development of Internet technology increases the difficulty of finding the target information. Therefore, many recommendation algorithms were proposed. Such recommendation algorithms can filter information according to the personal preference; hence they have been universally applied in fields such as web browsing, film recommendation and e-commerce [1]. Li [2] analyzed the sales records in the current tea leaves sales system by combining Hadoop distributed system with the traditional collaborative filtering algorithm to obtain the recommendation rules which could satisfy the preference of customer and help users find the tea leaves they needed.

Yu et al. [3] proposed the weighed cloud model attributes based service cluster algorithm and calculated the user score similarity using the weighed Pearson correlation coefficient method of service cluster algorithm and the user service selection index weight. They found that the algorithm could accurately calculate service recommendation credibility, satisfying the demands of users on service credibility, and enhance the success rate of the user service selection. The collaborative filtering algorithm has high degree of individualization and automation, but it exhibits a few problems such as sparsity and system extensibility. Therefore, in this study a probability matrix decomposition based collaborative filtering algorithm was put forward to correct up the defects of the traditional collaborative filtering algorithm and performed simulation experiments. The experimental results suggested that the mean absolute error (MAE), the root-mean-square error (RMSE) and the accuracy of the algorithm could reach the expected levels. This work provides a reference for the application of probability matrix decomposition based collaborative filtering recommendation algorithm in the searching of Internet information.

2 Collaborative filtering recommendation algorithm

2.1 Collaborative filtering algorithms based on different elements

2.1.1 Collaborative filtering algorithm based on users

User based collaborative filtering algorithm focuses on users. It recommends using user-item score matrix. It firstly searches for users which are similar to the target users and then recommends the selection of the searched users to the target users. The algorithm has two functions, i.e. one for calculating the similarity between adjacent users to establish matrix and one for recommending the target users using algorithm evaluation method.

2.1.2 Collaborative filtering algorithm based on items

Item based collaborative filtering algorithm can provide recommendations to users based on evaluation data after establishing user-item evaluation data model. In details, it calculates the similarity between different items to determine the preference of target users and then recommends similar items to target users. The algorithm has functions for calculating the similarity between items, establishing similarity matrix and recommending target users by scoring similar items using algorithm evaluation method.

2.2 Collaborative recommendation algorithm based on probability matrix decomposition

The probability matrix decomposition can reflect the information of users and items to low-dimensional characteristic space in the aspect of probability and then analyze the concerns of uses about items using the linear combination of low-dimensional vectors [4].

Item score matrix could be expressed as $F_{i\times j}$; a matrix $M_{a\times i}$ whose mean value was 0 and variance was α_M^2 and a random number matrix $N_{a\times j}$ whose mean value and variance were 0 and α_N^2 respectively were produced by MATLAB [5], in which a refers to the dimension of decomposition, $M_{a\times i}$ refers to a-dimensional characteristic square matrix of users, and $N_{a\times j}$ refers to the a-dimensional characteristic square matrix of item. Vector M_m and N_n were the corresponding potential characteristic vectors. In general, $F_{i\times j} \neq {M^T}_{a\times i} \times N_{a\times j}$. The matrix ${M^T}_{a\times i} \times N_{a\times j} \to F_{i\times j}$ was obtained through the learning of machine training.

Suppose the mean value of the error between actual score F_{mn} and predicted score $\hat{F}_{mn}$ as 0 and the variable of F_{mn} and $\hat{F}_{mn}$ as Gaussian distribution of α_F^2, then the probability distribution is $q(F_{mn} - M_m^T N_n | 0, \alpha_F^2)$. $q(F_{mn} | M_m^T N_n, \alpha_F^2)$ was obtained through translation. Then the condition of the score matrix F was:

$$q(F|M,N,\alpha_F^2) = \prod_{m=1}^{i}\prod_{n=1}^{j}\left[K(F_{mn}|M_m^T N_n, \alpha_F^2)\right] I_{mn}$$

, (1)

Where I_{mn} stands for indicator function, $I_{mn}=1$ means user m has scored item n, and $I_{mn}=0$ means user m has not scored item n.

As M and N could not include each other, the mean value of M and N was 0, and α_M^2 and α_N^2 had Gaussian distribution, then

$$Q(M|\alpha_M^2) = \prod_{m=1}^{i} K(M_m|0, \alpha_M^2 I) \quad , \qquad (2)$$

$$Q(N|\alpha_N^2) = \prod_{n=1}^{i} K(N_n|0, \alpha_N^2 I) \quad , \qquad (3)$$

Where Q stands for probability.

The joint probability distribution of M and N can be obtained from equation (1), (2) and (3).

$$q(M,N|F,\alpha_F^2,\alpha_M^2,\alpha_N^2) = \prod_{m=1}^{i}\prod_{n=1}^{j}\left[K(F_{mn}|M_m^T N, \alpha_F^2)\right] I_{mn} \times \prod_{m=1}^{i} K(M_m|0,\alpha_M^2 I) \times \prod_{n=1}^{j} K(N_n|0,\alpha_N^2 I)$$

(4)

The logarithm of the probability distribution of M and N was calculated:

$$\ln q(M,N|F,\alpha_F^2,\alpha_M^2,\alpha_N^2) = -\frac{1}{2\alpha_F^2}\sum_{m=1}^{i}\sum_{n=1}^{j} I_{mn}(F_{mn} - M_m^T N_n)^2 - \frac{1}{2\alpha_M^2}\sum_{m=1}^{i} M_m^T N_n - \frac{1}{2\alpha_N^2}\sum_{n=1}^{j} u_n^T N_n \, .$$

(5)

The maximum solution of equation (5) was replaced with the minimum solution of error function containing normalization parameters [6]:

$$L_{\min} = \frac{1}{2}\sum_{m=1}^{i}\sum_{n=1}^{j} I_{mn}(F_{mn} - M_m^T N_n)^2 + \frac{\beta M}{2}\left(\sum_{m=1}^{i}\|M_m\|^2 + \frac{\beta N}{2}\sum_{n=1}^{j}\|N_n\|^2\right),$$

(6)

Where $\beta_M = \frac{\alpha_F^2}{\alpha_M^2}$ and $\beta_N = \frac{\alpha_F^2}{\alpha_N^2}$. As $\alpha_M^2 = \alpha_N^2$, then target function was:

$$L_{\min} = \frac{1}{2}\sum_{m=1}^{i}\sum_{n=1}^{j} I_{mn}(F_{mn} - M_m^T N_n)^2 + \frac{\beta}{2}\left(\sum_{m=1}^{i}\|M_m\|^2 + \sum_{n=1}^{j}\|N_n\|^2\right) \qquad (7)$$

The relationship between regularization parameter β and $\alpha_F^2, \alpha_M^2, \alpha_N^2$ can be obtained from the equation (7).

The algorithm calculated function using stochastic gradient descent method [7]. It could obtain the decline direction of numerical values using derivatives and then calculate variables constantly on this direction until the minimal point was obtained.

The solution of the point suggested that the updating formulas of M_m, N_n were transformed to the following formulas in each iteration:

$$l = F_{mn} - M_m^T N_n \, , \qquad (8)$$

$$M_m \leftarrow M_m + \theta \times (l \times N_n - \beta \times M_m) \, , \qquad (9)$$

$$N_n \leftarrow N_n + \theta \times (l \times M_m - \beta \times N_n) \, , \qquad (10)$$

Where θ stands for the learning rate of the stochastic gradient descent.

3 Experiment

3.1 Experimental data

A 100k data set originated from the movies provided by GroupLens project team from University of Minnesota were used in the experiment, denoted as data set A.

Data set A included 100,000 scores for 1,682 movie items given by 943 users. Each user scored 20 movie items at least. The score was an integer between 0 and 5. The more the user liked the movie the higher was the score. The sparseness of the data set A suggested the percentage of the movie items which were not scored by the users, i.e. 1-100,000/(943×1682) ≈ 0.937.

The 100,000 scores in data set A were randomly divided into two disjoint sets, the training and the testing set. The training set which included 80% of the data was expressed as S1, while the testing set which included 20% of the data was expressed as S2. The data set A was divided 10 times to perform cross validation on the algorithm.

To enhance the recommendation efficiency of the algorithm, batch processing module was added. The 100000 scores were divided into 10 batches. 10000 scores were processed every time. This way, the computational quantity of the system and the convergence instability of the model produced in calculation could be reduced.

The collaborative filtering recommendation algorithm based on probability matrix decomposition performed as follows.

```
Input: training set and testing set
Output: Predicted score and square root error
Data such as regularization parameter were set
The number of movies and users were set.
If the iteration epoch < max epoch, then the 100000 scores were divided into 10 groups, 10,000 in each group, for separate processing.
If the patch processing was lower than 10, then the loss function q was calculated, and then matrix calculation was performed.
End
The predicted scores in the testing set were revised to positive integers through rounding off, and then square root error was calculated.
End
```

3.2 Scoring criteria

3.2.1 MAE

The MAE measure included the calculation of the absolute and average values of the difference between a predicted score and a real score [8]; hence it could be used for detecting the average difference between a predicted score and a real score. The smaller the value of MAE was, the more accurate the algorithm was.

$$MAE=\frac{1}{c}\sum e\in M\ and\ k\in N\left|d_{ek}-x_{ek}\right|,\quad(11)$$

where d_{ek} stands for the predicted score of the user e on item k, x_{ek} stands for the real score of the user e on the item k, set M and N stand for the sets of users and items in the testing set, and c stands for the number of d_{ek} or x_{ek} .

3.2.2 Root-mean-square error

Root-mean-square error refers to the average value of quadratic sum of the error between the two scores. The smaller the root-mean-square error was, the more accurate the prediction was [9].

$$RMSE=\sqrt{\frac{1}{c}\sum e\in M\ and\ k\in N(d_{ek}-x_{ek})^2},\quad(12)$$

where d_{ek} stands for the predicted score of the user e on item k, x_{ek} stands for the real score of user the e on item k, set M and N stand for the sets of users and items in the testing set, and c stands for the number of d_{ek} or x_{ek} .

3.2.2 Accuracy

Accuracy could be expressed as:

$$Accuracy=\frac{|X|}{|R|},\quad(13)$$

Where $X=\{d_{ek}|d_{ek}=x_{ek}\}$, i.e. set X was the set of the predicted scores which were equal to the real scores in the testing set, $d_{ek}\in D$ (D was the set of the predicted scores), and $x_{ek}\in R$ (R was the set of the real scores).

Both, the corrected probability of an item and the prediction accuracy, could be recommended to users.

3.3 Design of experiment

The specific content of the experiment was as follows.

To analyze the application performance of probability matrix decomposition based collaborative filtering algorithm in the experimental aspect, the movie evaluation mentioned in the preceding text was taken as the data set, and the user based collaborative filtering algorithm, the item based collaborative filtering algorithm and the probability matrix decomposition based collaborative filtering algorithm were compared. To better analyze the application performance of the probability matrix decomposition based collaborative filtering algorithm, the other two algorithms, i.e. the basic matrix based collaborative filtering algorithm and the singular value decomposition based collaborative filtering algorithm, were also tested. The parameter setting of the algorithms is shown in Table 1.

The user based collaborative filtering algorithm and the item based collaborative filtering algorithm were tested six times. The algorithm itself corresponds to the six characteristic factor numbers (dimension k) of the probability matrix decomposition collaborative filtering algorithm and the singular value decomposition based collaborative filtering algorithm. The basic matrix

Algorithm	User based collaborative filtering algorithm	Item based collaborative filtering algorithm	Probability matrix decomposition collaborative filtering algorithm	Basic matrix collaborative filtering algorithm	Singular value decomposition based collaborative filtering algorithm
Neighbourhood or model	Neighbourhood	Neighbourhood	Probability matrix decomposition collaborative filtering model	Basic matrix collaborative filtering model	Singular value decomposition based collaborative filtering model
Characteristic factor number (dimension k)	\	\	[10,60]	[10,60]	6
Learning rate	\	\	0.02	0.02	\

Table 1: The parameter setting of the five algorithms.

collaborative filtering algorithm and the singular value decomposition based collaborative filtering algorithm used in the performance comparison were developed by referring to the relevant literature and revised according to the data which needed to be detected. The characteristic factor number of singular value decomposition based collaborative filtering algorithm was fixed, 6. The setting of characteristic factor number of the basic matrix collaborative filtering algorithm was the same as the probability matrix decomposition collaborative filtering algorithm, [10, 60], and the unit stepping was set to10.

4 Experimental results and analysis

4.1 MAE and RMSE

It could be noted from the Figure 1 and 2 that the predictive recommendation performance of the user based collaborative filtering algorithm was the poorest, and the prediction performance of the singular value decomposition based collaborative filtering algorithm was not affected by characteristic factor number, but was poorer than that of the user based collaborative filtering algorithm. The performance of the user based collaborative filtering algorithm was far worse than that of the probability matrix decomposition collaborative filtering algorithm and the basic matrix based collaborative filtering algorithm. The performance of the probability matrix decomposition collaborative filtering algorithm and the basic matrix based collaborative filtering algorithm was similar, but the probability matrix decomposition collaborative filtering algorithm was still superior. The reason why there was a significant difference between the performance of the user based collaborative filtering algorithm and the item based collaborative filtering algorithm is the score given by a scorer was probably affected by the view of other scorers who had the same interests. The reason why the performance of the singular value decomposition based collaborative filtering algorithm in the prediction and recommendation was significantly poorer than that of the basic matrix collaborative filtering algorithm and the probability matrix decomposition collaborative filtering algorithm was the fact that the singular value decomposition based collaborative filtering algorithm was actually an improved version of the item based collaborative filtering algorithm and therefore had similar shortcomings as the original algorithm, i.e., the matrix obtained after the preprocessing had data distortion compared to the original matrix, which could have affected the accuracy and the similarity of the score prediction. But the performance of the singular value decomposition based collaborative filtering algorithm was better than of the item based collaborative filtering algorithm, indicating the improved accuracy of the singular value decomposition based collaborative filtering algorithm.

It could be noted from the Figure 1 that the values of the root-mean-square error (RMSE) corresponding to the basic matrix based collaborative filtering algorithm and the probability matrix decomposition collaborative filtering algorithm gradually decreased with the increase of the characteristic factor number; the larger the characteristic factor number, the smaller the decrease amplitude. When the characteristic factor number was 50, the value of RMSE was the minimum, and the prediction accuracy was the highest; when the characteristic factor number was between 10 and 20, the decrease amplitude of RMSE of the basic matrix based collaborative filtering algorithm and the probability matrix decomposition collaborative filtering algorithm was large, around 1.14% and 0.700% respectively. It was found that the values of the RMSE of the two algorithms were lowly sensitive to the characteristic factor number, especially of the probability matrix decomposition collaborative filtering algorithm. When the characteristic factor number was larger than 40, the fluctuation of the RMSE was quite small.

Similar to Figure 1, the MAE corresponding to the basic matrix based collaborative filtering algorithm and the probability matrix decomposition based collaborative filtering algorithm also decreased with the increase of the characteristic factor number and reached the minimum values, 0.675 and 0.666 respectively, when the characteristic factor number was 60. Moreover it was noted that when MAE was taken as the evaluation index, the curves of the basic matrix collaborative filtering algorithm and the probability matrix decomposition based collaborative filtering algorithm nearly coincided, and the prediction performance was also close. Moreover, since the RMSE is more sensitive to the measurement error, the probability matrix decomposition based collaborative filtering algorithm had an advantage over the basic matrix collaborative filtering algorithm due to the addition of the regularization term.

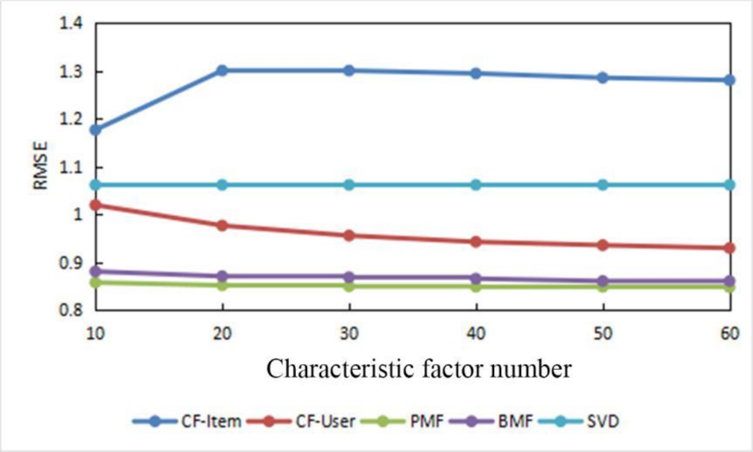


Figure 1: Variation of the RMSE of the five algorithms with the increase of the characteristic factor number.

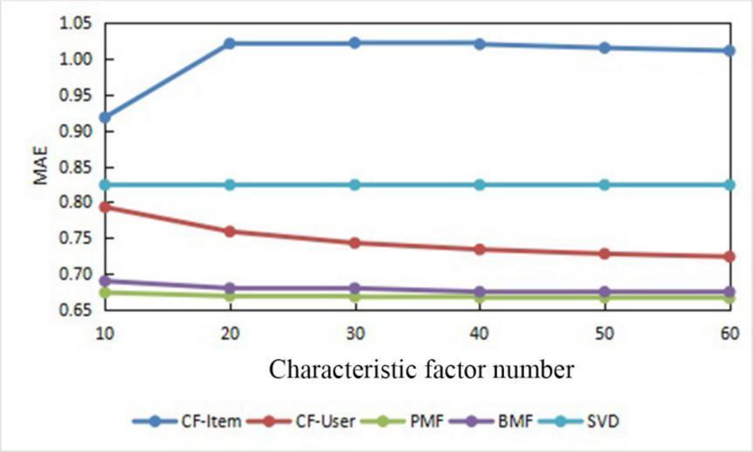


Figure 2: Variation of the MAE of the five algorithms with the increase of the characteristic factor number.

4.2 Accuracy

It was found from the comparison of the RMSE and the MAE between the five algorithms that the prediction performance of the CF-User, the item based collaborative filtering algorithm and the singular value decomposition based collaborative filtering algorithm was significantly different from the basic matrix based collaborative filtering algorithm and the probability matrix decomposition based collaborative filtering algorithm. Therefore, only the accuracy of the basic matrix based collaborative filtering algorithm and the probability matrix decomposition based collaborative filtering algorithm were considered. The results are shown in Table 2.

It could be noted from Figure 3 that the tendency of the accuracy of the basic matrix collaborative filtering algorithm and the probability matrix decomposition based collaborative filtering algorithm was opposite to the tendencies of the MAE and the RMSE. When the characteristic factor number was small, the accuracy was low; the accuracy increased first and then tended to be stable with the increase of the characteristic factor number

Characteristic factor number	PMF	BMF
10	0.40321	0.34666
20	0.41321	0.36022
30	0.41403	0.36142
40	0.41300	0.36134
50	0.41299	0.36132
60	0.41298	0.36132

Figure 3: The accuracy of the basic matrix based collaborative filtering algorithm and the probability matrix decomposition based collaborative filtering algorithm under different characteristic factor numbers.

and nearly had no fluctuation when the characteristic factor number was larger than 30. It was because the effective information increased with the increase of the characteristic factor number. The accuracy of the probability matrix decomposition based collaborative filtering algorithm was much higher than that of the basic matrix based collaborative filtering algorithm. Therefore the proposed algorithm could improve the searching speed and preciseness.

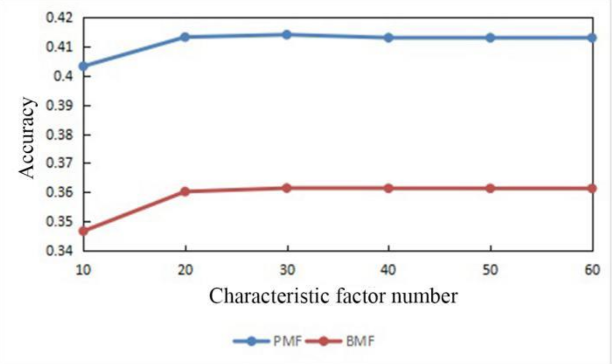


Figure 4: Comparison of the accuracy of the probability matrix decomposition based collaborative filtering algorithm (PMF) and the basic matrix based collaborative filtering algorithm (BMF).

The recommendation system which can filter diversified data is an effective filtering approach [10]. It can recommend individual information to users according to users' requirements. Therefore it can be convenient for information collection and has been extensively applied on the Internet. Wei et al. [11] put forward project category similarity and interestingness measure based collaborative filtering recommendation algorithm which could recommend information to users through calculating project categories and interestingness and had high prediction preciseness. In a study of Chen et al. [12], a mixed recommendation system was put forward to recommend users with learning projects searching. In the test, the algorithm effectively collected information, suggesting a favorable performance.

5 Conclusion

In conclusion, the probability matrix decomposition based collaborative recommendation algorithm was put forward in this study, and then it was developed for data searching recommendation. Afterwards the MAE, the RMSE and the accuracy of the algorithm were tested. Moreover the MAE values, the RMSE values and the accuracy of the CF-User, the item based collaborative filtering algorithm, the singular value decomposition based collaborative filtering algorithm and the basic matrix based collaborative filtering algorithm were compared. The testing results suggested that the improved collaborative recommendation algorithm had the highest preciseness and accuracy, and the preciseness and the accuracy became the largest and stable when the characteristic factor number was more than 40. Therefore it could be applied in a computer searching system. This work provides a reference for the progress of the collaborative recommendation algorithm.

6 Acknowledgement

This study was supported by the Fund of Natural Science of Hebei (A2015209040).

7 References

[1] Wang W, Wu YH, Wu YY. (2016). A Multi-stage Heuristic Algorithm for Matching Problem in the Modified Miniload Automated Storage and Retrieval System of E-commerce. *Chinese Journal of Mechanical Engineering*, 29(3):1-8.

[2] Li L. (2017). Analytical Application of Hadoop-Based Collaborative Filtering Recommended Algorithm in Tea Sales System. *International Conference on Applications and Techniques in Cyber Security and Intelligence. Edizioni della Normale, Cham*, 434-441.

[3] Yu Z Y, Wang J D, Zhang H W, Niu K. (2016). Services recommended trust algorithm based on cloud model attributes weighted clustering. *Automatic Control & Computer Sciences*, 50(4):260-270.

[4] Zhang F, Yang J, Tai Y, Tang J. (2015). Double Nuclear Norm Based Matrix Decomposition For Occluded Image Recovery and Background Modeling. *IEEE Trans Image Process*, 24(6):1956-1966.

[5] Simpson R, Devenyi G A, Jezzard P, Hennessy TJ, Near J. (2017). Advanced processing and simulation of MRS data using the FID appliance (FID-A)-An open source, MATLAB-based toolkit. *Magnetic Resonance in Medicine*, 77(1):e56841.

[6] Ando T, Bai J. (2015). Selecting the Regularization Parameters in High-dimensional Panel Data Models: Consistency and Efficiency. *Econometric Reviews*, 151014164537003.

[7] Wu K, Sun Y, Huai Y, Jia SQ, Chen X, Jin YQ. (2015). Multi-perturbation stochastic parallel gradient descent method for wavefront correction. *Optics Express*, 23(3):2933-44.

[8] Franses P H. (2015). A note on the Mean Absolute Scaled Error. *International Journal of Forecasting*, 32(1):20-22.

[9] Nuutinen M, Virtanen T, Häkkinen J. (2016). Performance measure of image and video quality assessment algorithms: subjective root-mean-square error. *Journal of Electronic Imaging*, 25(2):023012.

[10] Knobelsdorff P M G V, Reiferscheid F, Straakholder T M, Wirtz MS. (2004). Wird der Algorithmus des European Resuscitation Council zur kardiopulmonalen Reanimation eingehalten?. *Intensivmedizin Und Notfallmedizin*, 41(1):22-28.

[11] Wei S, Ye N, Zhang S, Huang X, Zhu J. (2012). Item-Based Collaborative Filtering Recommendation Algorithm Combining Item Category with Interestingness Measure.

International Conference on Computer Science and Service System. IEEE Computer Society, 2038-2041.

[12] Chen W, Niu Z, Zhao X, Li Y. (2014). A hybrid recommendation algorithm adapted in e-learning environments. World Wide Web-internet & Web Information Systems, 17(2):271-284.

The Decision Model for the Optimal Configuration Management of Knowledge Employees in Enterprises

Chunjiang Yang
Yantai Nanshan University, Longkou, Shandong
E-mail: yangyujiang_123@sina.com

Yujiang Yang
School of Mining Engineering, University of Science and Technology Liaoning, Anshan, Liaoning, China

Xiuli Tan
Yantai Nanshan University, Longkou, Shandong, China

Technical Paper

Keywords: knowledge employee, bacterial foraging algorithm, employee dispatch

Received: April 3, 2018

With the development of knowledge economy, the role of knowledge employees in enterprises is becoming more and more important, and employees with rich knowledge represent the tower of strength in the development of enterprises. Knowledge employees with strong autonomy and creativity have stronger self-awareness in daily work. Thus, how to dispatch knowledge employees and assign them to proper positions, is of utter importance. To solve the problem of employee dispatch, bacterial foraging algorithm was proposed to optimize the configuration management of knowledge employees in enterprises in this study, a mathematical model based on the optimal configuration of knowledge employees was established, and a simulation experiment was carried out on the model. It was found that the algorithm could effectively optimize the configuration of knowledge employees. It is a feasible method for employee dispatch in enterprises.

Povzetek: Predstavljena je metoda za organiziranje zaposlenih v družbi znanja.

1 Introduction

Since the 21st century, information age has started, and knowledge economy has gained a rapid development. Producing products using cheap labour force has not been able to ensure the survival and development of enterprises; therefore, enterprises should improve its competitiveness by taking advantages of knowledge employees. Knowledge employees are the most active cells in enterprises, and the task assignment and scheduling for them directly determines the coordination between user demand, employee personality and enterprises benefits, the utilization of knowledge resources and the competitiveness and values of enterprises. Bogdanowicz et al. [13] considered that managing knowledge could bring challenges to the field of human resource development, especially when employees concerned more about their employment ability, and that an enterprise should pay attention to knowledge employees if placing great importance on knowledge. But knowledge employees are of high risks to leave enterprises because of their autonomy and creativity. How to optimize the configuration of knowledge employees is an urgent problem. A project may not be completed in time if decisions are made only considering the task assignment and scheduling of employees involved in that project regardless of the integrity of multiple projects. Through discussing the concept, connotation and composition of knowledge cooperative capability, Cao [1] concluded the influence factors and established an evaluation indicator system for the flood carrying capacity of knowledge employees in enterprises based on collaboration through introducing the concept of cooperation process, in order to improve the cooperative ability of knowledge employees and ensure the effective development of enterprises. Liu [2] introduced the loyalty of knowledge employees in state-owned coal enterprises, analysed the main reason for the decrease of loyalty in details, and finally concluded the strategies for improving loyalty, which was beneficial to the correct configuration of human resources and the improvement of enterprises. Based on the study of optimal configuration of knowledge employees, this study put forward working arrangement and dispatch for knowledge employees based on bacterial foraging algorithm to highlight their advantages and satisfy the demands of clients efficiently. Bacterial foraging algorithm was used to establish a mathematical model for the optimal configuration of knowledge employees, and then the feasibility of the algorithm was verified by a simulation experiment.

2 Configuration of knowledge employees in enterprises

2.1 Knowledge employee

The term "knowledge employee" refers to a person that works by applying symbols and concepts and using knowledge or information [3]. Knowledge employees as the carriers of knowledge play a vital role in the competition between enterprises, the innovation and utilization of knowledge and the reasonable configuration of resources [4]. Compared to non-knowledge employees, knowledge employees have the following characteristics.

(1) Employees have specialty in knowledge, good education background and high individual quality.

(2) They have strong wishes for realizing self-worth and high requirements and are interested in challenging works.

(3) With strong creativity and autonomy, they can independently fulfil most of creative tasks and produce new knowledge achievements based on their own knowledge [5].

(4) The working process is difficult to be monitored. The work form of knowledge employees with high randomness and subjectivity is different with that of employees in traditional workshops and offices.

(5) Their work results cannot be measured directly. The labour achievements created by knowledge employees exist in the form of inventions which cannot be directly measured by ordinary economic forms [6].

2.2 Configuration management of knowledge employees

The competition between enterprises and knowledge creation in enterprises are usually fulfilled by knowledge employees. The configuration management of knowledge employees refers to fulfil tasks using the shortest time.

3 The decision model for optimal configuration

3.1 Bacterial foraging algorithm

Bacterial foraging algorithm is a swarm intelligent optimization algorithm which achieves optimization through chemotaxis [7], reproduction and migration behaviours [8] according to the basic rules of the growth and evolution of bacteria [16]. Firstly, the individual evolutionary mechanism is formulated according to the reproduction rules of bacteria. Then the motion patterns of individuals in the algorithm are established according to the characteristics of bacterial foraging. Finally, the information share system was established according to the information exchange means of bacterial in colonies.

3.2 The flow of solving optimal configuration with bacterial foraging algorithm

X, Y and Z were set as the maximum execution counts of taxis, reproduction and migration operations respectively. x, y and z are the count values of the three operations. U stands for the maximum swimming distance, and w stands for the number of swimming steps.

(1) The parameters of the algorithm were initialized, and X, Y and Z were set to 1.

(2) Taxis operation was performed, x←x+1. The bacteria were cycled. Before reaching the maximum times, the bacteria were turned over and swam.

(3) After taxis operation, reproduction operation was performed if the reproduction number did not reach the maximum value Y. Then the second procedure was repeated. The next step was done if the value reached Y.

(4) Then the bacteria were processed by migration operation, and the number of bacteria was initialized randomly.

(5) If the maximum migration number did not reach Z after migration operation, then it returned to the second procedure; if it did, then the algorithm ended and the result was output.

The flow of the optimal configuration based on bacterial foraging method is shown in Figure 1.

3.3 Modelling

Figure 2 shows the flow of mathematical modelling.

3.3.1 Embodiment of abstract problems

Based on the consideration on actual situation, this study found that staff dispatch and allocation should satisfy three conditions, i.e., the maximum customer satisfaction, uniform workload of knowledge employees and minimum enterprises cost [9]. Therefore, the optimal configuration of knowledge employees was implemented based on the three conditions. The details of the three conditions are as follows.

(1) Customer satisfaction maximization refers to maximizing satisfaction of customers on the services provided by enterprises, which is beneficial to the long-term development of enterprises benefits. Client satisfaction can be expressed by the time spent on tasks. Maximum client satisfaction also means the shortest time spent on tasks.

(2) Uniform working load of knowledge employees refers to uniform allocation of working load according to the working ability and conditions of employees. To realize the uniform allocation of workload, the work time and average time difference of employees should be the minimum.

(3) Enterprise cost minimization refers to minimize resource cost in the process of production and services. The main purpose of enterprises is profits. Minimizing enterprise cost can efficiently gain profits.

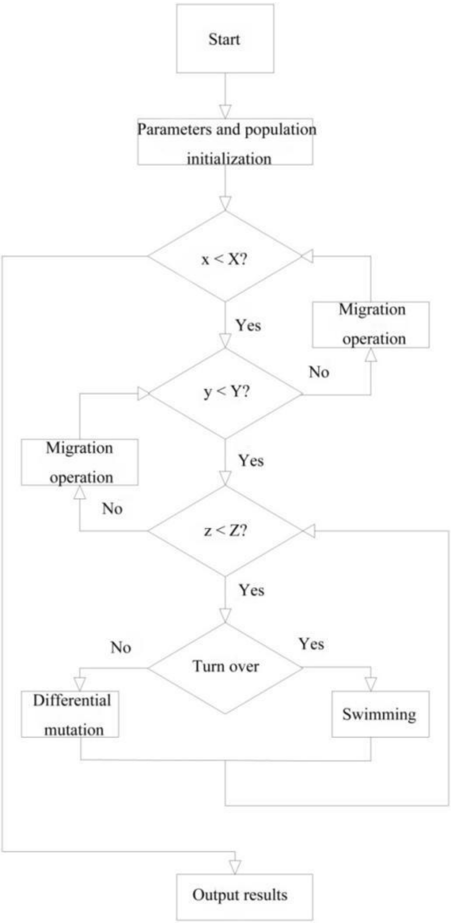


Figure 1: The flow of optimal configuration based on bacterial foraging algorithm.

3.3.2 Assumption

To effectively study resource optimization, the working scheduling of knowledge employees was assumed firstly. The starting time point of tasks was set to 0. Each employee could only do one task simultaneously. Each task could only be fulfilled by one employee. When tasks have been going on, the other tasks should have been waiting. The completion time of each task was associated to the capacity of the employee. Next task could start after last task was completed. Only broad setting was given, thus hypothesis needs to be formulated according to the actual condition of enterprises in practical application.

3.3.3 Setting variables

To simplify modelling, variables were designed after assumption.

(1) M: a set of m projects, $M=\{M_1,M_2,M_3\cdots,M_i,\cdots M_m\}$

(2) N: a set of n knowledge staffs, $N=\{N_1,N_2,N_3\cdots,N_k,\cdots N_n\}$ ($k=1,2,3,\cdots,n$)

(3) P_i : a sequenced set of j tasks of project i, $P_i=\{P_{i1},P_{i2},P_{i3}\cdots P_{ij}\cdots P_{iu}\}$ ($j=1,2,3\cdots,u$)

(4) g_{ijk} : whether knowledge employee N_k could be competent for task P_{ij}. If he could, then $g_{ijk}=1$; otherwise, $g_{ijk}=o$.

(5) t_{ijk} : time for employee N_k completing task P_{ij}

(6) a_{ij} : the starting time of task P_{ij}

(7) b_{ij} : the end time of task P_{ij}

(8) c_{ijk} : cost for employee N_k completing task P_{ij}

For individual employee, IN_k was the set of tasks which needed to be fulfilled:

$IN_k=\{IN_1,IN_2,\cdots,IN_{kr},\cdots,IN_{kk(0)}\}$ ($r=1,2,3,\cdots,k(0)$)

d_{ijkr} stands for whether task P_{ij} could be allocated to employee N_k ; $d_{ijkr}=0$ means the task could not be allocated to employee N_k ; $d_{ijkr}=1$ means the task could be allocated to employee N_k .

H_{kr} refers to the time when employee N_k started the r-th task; L_{kr} refers to the time when employee N_k completed the r-th task.

3.3.4 Establishment of mathematical model

Mathematical expression models were proposed against the three conditions which should be satisfied in dispatching and allocation.

Customer satisfaction could be expressed as:

$$f_1(x)=\min_{1\le i\le m}\left(\max_{1\le j\le u}(b_{iu})\right) \tag{1}$$

The workload of knowledge employees can be expressed as:

$$f_2(x)=\min\sum_{k=1}^{n}\left(t_{ijk}\cdot d_{ijkr}-\sum_{k=1}^{n}t_{ijk}\cdot d_{ijkr}/n\right)^2/n \tag{2}$$

Enterprise cost minimization can be expressed as:

$$f_3(x)=\min\sum_{k=1}^{n}\sum_{r=1}^{k(0)}c_{ijk}\cdot d_{ijkr} \tag{3}$$

The configuration management of enterprises knowledge employees was optimized by three mathematical formulas. The formula of the decision model was:

$$\min\left(f_1(x),f_2(x),f_3(x)\right) \tag{4}$$

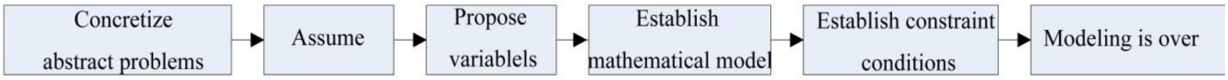

Concretize abstract problems
Assume
Propose variablels
Establish mathematical model
Establish constraint conditions
Modeling is over

Figure 2: The flow of mathematical modelling.

Then equation (4) was processed by normalization using $q_i=\log_{10}\left(f_i(x)\right)$. The three targets were set as θ_1, θ_2 and θ_3. Then the objective function of the model was:

$$F(x)=\min\sum_{i=1}^{3}\theta_i q_i(x) \qquad (5)$$

3.3.5 Establishment of constraint condition

A constraint condition was needed before every time of decision making. When the decision-making objective was formulated, there was also a constraint condition. The constraint condition was:

$$\sum_{i=1}^{m}\sum_{j=1}^{u}\sum_{k=1}^{n}\sum_{r=1}^{k(0)} d_{ijkr}=1 \quad . \qquad (6)$$

Formula (6) means every task was independent and could only be completed by one knowledge employee. The formulas of the starting time points when taking tasks and knowledge employees as subjects respectively are:

$$a_{ij}=\max\left(b_{ij-1},\sum_{k=1}^{n}\sum_{r=1}^{k(0)} L_{kr-1}\Box d_{ijkr}\right) , \qquad (7)$$

$$H_{kr}=\max\left(L_{kr-1},\sum_{i=1}^{m}\sum_{j=1}^{u} b_{ij-1}\Box d_{ijkr}\right) . \qquad (8)$$

The formulas of the completion time points when taking tasks and knowledge employees as subjects respectively are:

$$b_{ij}=aij+\sum_{k=1}^{n}\sum_{r=1}^{k(0)} t_{ijk}\Box d_{ijkr} , \qquad (9)$$

$$L_{kr}=H_{kr}+\sum_{i=1}^{m}\sum_{j=1}^{u} t_{ijk}\Box d_{ijkr} . \qquad (10)$$

Formula (11) was used to ensure allocating a task to a knowledge employee when he was able to complete the task. Formula (12) was used to ensure that the set variable was non-negative.

$$d_{ijkr}\le g_{ijk} \qquad (11)$$

$$b_{ij},a_{ij},t_{ijk},d_{ijkr},g_{ijk},L_{kr},H_{kr}\ge 0 \qquad (12)$$

4 Simulation experiment and analysis of the results

Task assignment and scheduling of employees are most widely used in transportation system [14]. In the field of transportation, they have the same characteristics. They are limited by time and space; each task must take into account the start time and location and ending time and location. It has the most extensive application in the scheduling of airline crew, followed by the scheduling of drivers in the automotive industry. In the field of health care, it is necessary to take into account whether the nurses have appropriate qualifications and whether they are long-term in-service nurses or short-term in-service nurses to ensure the fairness of nurses' shifts at night and on weekends, and the vacation and housing problems of them. To ensure the even distribution of workload, the bacterial foraging method can be used to optimize the configuration management during the task assignment and scheduling. It considers whether it is possible to insert a task into the existing time slot. If it is, then the time can be shortened, and the feasible solution can be optimized. Finally, the management of scheduling can be achieved with the least time and cost.

A project team was set up to solve the performance management related projects of an enterprise. The members of the project team should satisfy the following conditions.

They should know the core knowledge of enterprises operation and development, have strong bearing capacity to shoulder tasks with different difficulties which were allocated by enterprises and be responsible for management works.

According to the characteristics of the team, the members in the team were processed by optimal configuration management. A simulation experiment was performed to verify the role of decision model in optimal configuration of resources.

4.1 Basic data

In the process of experiment, an issue of optimal configuration of knowledge employees was proposed. The issue included two projects, each project contained six tasks, and each task was completed by one employee. The experimental data are shown in Table 1.

M_i		M_1						M_2					
P_{ij}		P_{11}	P_{12}	P_{13}	P_{14}	P_{15}	P_{16}	P_{21}	P_{22}	P_{23}	P_{24}	P_{25}	P_{26}
N_k	N_1	5/0.8	/	6/1.8	/	10/3	6/2.8	11/2.1	/	6/1.8	/	8/1.5	/
	N_2	6/3.8	9/4.8	4/0.8	/	4.1/1.8	11/5.8	10/3.3	5/1	9/3.2	/	10/2.6	7/2
	N_3	4/0.8	/	5/1.2	16/6	/	7/4.1	14/4.2	/	6/1.1	16/7.1	/	9/2.6
	N_4	/	8/3.8	/	13/5.5	5/1.6	/	/	8/2.5	/	18/6.9	13/3.3	/
	N_5	/	7/5.8	/	10/7	8/5.8	9/5.1	/	7/4.4	/	15/5.2	12/2.9	8/5.3
	N_6	6/6.8	8/4.3	7/7	12/5.2	/	/	14/8.1	8/3	10/3	16/5.6	/	10/3.2

Table 1: The time and cost spent on the tasks.
The symbol / means the task cannot be completed by the employee.

4.2 Results analysis

The algorithm was simulated by MATLAB software [10]. The parameters were set according to actual conditions. The maximum number of iteration was 200, the total number of bacteria was 50, there were 10 chemotaxis procedures, and the maximum number of steps was 4. After the forward direction was determined, the forward step length of bacterial individuals was 0.7, there were three copying procedures and three migration procedures, the number of cleavage cells was half of the total number of bacteria, and the probability of migration was 0.6. The simulation results are shown in Figure 3.

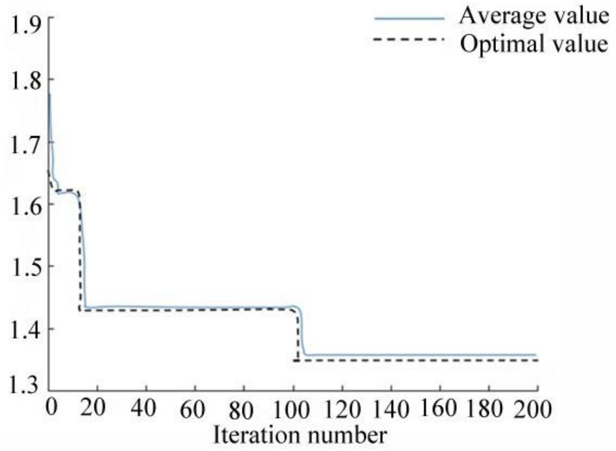


Figure 3: Simulation results.

Figure 3 demonstrates that the average value and optimal value of the model for the optimal configuration of knowledge employees based on bacterial foraging algorithm were basically the same, suggesting the algorithm had favourable convergence. When it was iterated to the 10th generation, the value began to close to the optimal value and maintained at 1.42 all the time. When it was iterated to the 100th generation, the local optimal solution, i.e., 1.32, was obtained. It indicated that the algorithm had strong global searching ability.

4.3 Performance test of bacterial foraging algorithm

To verify the effectiveness of bacterial foraging algorithm in practice, Sphere function, Quartic function and Rastrin function were selected as the benchmark functions, as shown in Table 2. Then the effectiveness of bacterial foraging algorithm and common particle swarm algorithm [15] in the three functions were compared.

Benchmark function	Index range	Optimal solution	Peak value
$sphere(x)=\sum_{i=1}^{n} x_i^2$	[-100,100]	0	Unimodal
$quartic(x)=\sum_{i=1}^{n} ix_i^4+random[0,1]$	[-1.28,1.28]	0	Unimodal
$rastrigin(x)=\sum_{i=1}^{n}[x_i^2-10\cos(2\pi x_i)+10]$	[-5.12,5.12]	0	Unimodal

Table 2: Benchmark functions.

To ensure the fairness of the results, the population number, operation times and maximum iteration times were set the same for the algorithms. Scale of population was set to 30, the maximum iteration times was set to 1,000, and operation times was 50. Chemotaxis times was set to 150, replication times was 8, and migration/dispel times was 5. After the setting, every example was independently operated for 25 times. The results are shown in Table 3.

Benchmark function	Results	Particle swarm algorithm	Bacterial foraging algorithm
$sphere(x)$	Average value	3.247±30.351	0.716±0.198
	Optimal value	3.247	0.416
	Standard deviation	3.268e-4	1.045e-5
$quartic(x)$	Average value	4.1e-2±0.051	1.621e-5±2.18e-5
	Optimal value	4.137e-3	3.529e-5
	Standard deviation	0.902	5.842e-5
$rastrigin(x)$	Average value	1.216e2±20.152	1.548e2±20.421
	Optimal value	1.133e2	1.213e-2
	Standard deviation	0.028	4.489e-2

Table 3: The experimental results of the examples.

When the times of iteration was the same, the optimal value obtained by bacterial foraging algorithm was smaller than that obtained by particle swarm algorithm.

5 Conclusion

Today knowledge employees have been an indispensable part in knowledge-based enterprises. How to optimize the configuration of knowledge employees has been a research task of many experts. Wang and Zheng [11] performed optimal configuration on knowledge employees using particle swarm optimization and found that the algorithm was scientific and effective in the dispatch of knowledge employees. Lin et al. [12] optimized the configuration of knowledge employees in modern enterprises with sliding mode control strategy and found the robustness of knowledge employee system through offline rectangle inequality. This study realized the optimal configuration of knowledge employees in enterprises using bacterial foraging algorithm, investigated the issue of employee dispatch, and established mathematical decision models. A simulation experiment was carried out to prove the effectiveness of the algorithm in optimizing the configuration of knowledge employees. The convergence performance of bacterial foraging algorithm and particle swarm algorithm was compared using three example functions, and the results demonstrated that the optimal value obtained by bacterial foraging algorithm was smaller than that obtained by particle swarm algorithm. This work can provide a reference for the dispatch of knowledge employees in the future.

6 Acknowledgement

This study was supported by 2016 Soft Science Research Project of Shandong Province: Supply Chain Finance Ecological Environment Optimization Research in Shandong Province Under the Background of Supply Side Reform (Project Number: 2016RKA06016).

7 References

[1] Cao Y. (2013). The evaluation research on enterprise staff knowledge collaborative ability based on the collaborative process. International Conference on Information Management, Innovation Management and Industrial Engineering, IEEE, pp. 324-328.

[2] Liu B. (2011). Reasons and responses of loyalty fall to knowledge-type staff in nationalized coal enterprise. *International Conference on Artificial Intelligence, Management Science and Electronic Commerce*, IEEE, pp. 2272-2275.

[3] Jia J. F., Huang S. and Zhu Z. (2012). Internal Relationship of Action-Oriented Competency of Knowledge Workers. Journal of Northeastern University, Vol. 33, No. 4, pp. 601-604.

[4] Rout S. S., Misra B. B. and Samanta S. (2014). Load Allocation in Academic Environment: A Multi Objective PSO Approach. GSTF Journal on Computing, Vol. 3, No. 4, 2014, p. 9.

[5] Wang W. T. and Hou Y. P. (2015). Motivations of employees' knowledge sharing behaviors: A self-determination perspective. Information & Organization, Vol. 25, No. 1, pp. 1-26.

[6] Simeth M. and Lhuillery S. (2015). How do firms develop capabilities for scientific disclosure?", Research Policy, Vol. 44, No. 7, pp. 1283-1295.

[7] Hota P. K., Barisal A. K. and Chakrabarti R. Economic emission load dispatch through fuzzy based bacterial foraging algorithm. Electrical Power and Energy Systems. International Journal of Electrical Power & Energy Systems, Vol. 32, No. 7, 2010, pp. 794-803.

[8] Sathya P. D. and Kayalvizhi R. (2011). Optimal multilevel thresholding using bacterial foraging algorithm. Expert Systems with Applications, Vol. 38, No. 12, pp. 15549-15564.

[9] Vargas A., Boza A., Patel S., Patel D. C., Cuenca L. and Bas A. O. (2016). Inter-enterprise architecture as a tool to empower decision-making in hierarchical collaborative production planning. Data & Knowledge Engineering, Vol. 105, pp. 5-22.

[10] Vedaldi A. and Lenc K. (2015). MatConvNet: Convolutional Neural Networks for MATLAB. ACM International Conference on Multimedia, ACM, pp. 689-692.

[11] Wang Q. and Zheng H. C. (2011). Optimization of task allocation and knowledge workers scheduling based-on particle swarm optimization. Information Technology & Standardization, pp. 574-578.

[12] Liu Y., Tang S., Shang C. and Qi H. (2013). Optimization of knowledge workers system by sliding mode control. Control Conference, IEEE, 8309-8314.

[13] Bogdanowicz M. S. and Bailey E. K. (2013). The value of knowledge and the values of the new knowledge worker: generation X in the new economy. Journal of European Industrial Training, Vol. 26, no. 2/3/4, pp. 125-129.

[14] Demirović E., Musliu N. and Winter F. (2017). Modeling and solving staff scheduling with partial weighted maxSAT. Annals of Operations Research, no. 1, pp. 1-21.

[15] Guo Y. and Nan L. I. (2012). Multiple projects scheduling method based on cloud multi-objective particle swarm optimization. Computer Engineering & Applications, vol. 48, no. 21, pp. 15-20.

[16] Chen Y. P., Li Y., Wang G., Zheng Y. F., Xu Q., Fan J. H. and Cui X. T. (2017). A novel bacterial foraging optimization algorithm for feature selection. Expert Systems with Applications An International Journal, vol. 83, no. C, pp. 1-17.

Application for Viral Hepatitis Infection Risk Assessment - HEPY

Alen Ajanović, Andrej Ulčar, Ana Peterlin, Karolina Počivavšek and Gašper Fele-Žorž
Faculty of Computer and Information Science, Večna pot 113, 1000 Ljubljana, Slovenia
E-mail: polz@fri.uni-lj.si

Anton Gradišek and Matjaž Gams
Institut Jožef Stefan, Jamova cesta 39, 1000 Ljubljana, Slovenia
E-mail: matjaz.gams@ijs.si

Mojca Matičič
Clinic for Infectious Diseases and Febrile Illnesses, University Medical Centre Ljubljana
Japljeva 2, 1525 Ljubljana Slovenia
E-mail: mojca.maticic@kclj.si

Keywords: viral hepatitis, infection, health education, web application, preventive medicine

Received: June 2, 2018

We present a web application to inform users about different types of viral hepatitises. The core of the application is a questionnaire about past behavior and risk factors. Based on the answers, it produces a personalised overview of any risky actions that the user might have taken in the past. The site also contains general information about these diseases, which can help users identify them or seek proper precautions in order to avoid them.

Povzetek: Predstavljamo spletno aplikacijo za informiranje uporabnikov o različnih tipih virusnega hepatitisa. Središče aplikacije je vprašalnik, ki na podlagi odgovorov in dejanj uporabnika poda osebno oceno stopnje nevarnosti za različne okužbe hepatitisa. Stran vsebuje tudi splošne informacije o teh boleznih, ki lahko uporabniku pomagajo pri iskanju hitre in ustrezne pomoči.

1 Introduction

Hepatitis is an inflammation of the liver. Hepatitis viruses are the most common cause of hepatitis. It presents an important global healthcare problem as it has been estimated to affect 330 million people worldwide [1]. In Slovenia, it is estimated that less than 1% of the population is infected with HBV (hepatitis B virus) and around 0.4% with HCV (hepatitis C virus) [2]. They are of greatest concern because of the burden of illness and death they cause and the potential for outbreaks and epidemic spread.

Viral hepatitis may be present as an acute or chronic disease. In acute disease which may occur with mild or no symptoms, or may include symptoms such as jaundice, dark urine, pale stool, extreme fatigue, nausea, vomiting and abdominal pain. Acute HBV and HCV infection are most likely to become ongoing and chronic and are the most common cause of liver cirrhosis and hepatocellular carcinoma, which can lead to liver failure and death [3]. Due to mild and nonspecific signs and symptoms, patients with chronic viral hepatitis typically do not know they are infected and are the main source of spreading the disease, mostly by risky behavior.

The main problem with hepatitis virus infections is that the patients do not know they are infected until the disease has already developed to an advanced stage of liver failure, at which point it becomes difficult to treat. Since serious liver complications of chronic hepatitis infections can be prevented or managed if the disease is detected and treated in its early stages, it is important to identify the infected individuals as soon as possible and act accordingly.

Collecting and identifying valid health information online is difficult. Patients would be much better served by a centralized site which is endorsed by medical professionals and contains all the relevant information in one place. In this paper, we present a web application (https://hepy.mf.uni-lj.si/) that aims to educate users about viral hepatitis infections and to assess possible risks of infection, in a safe anonymous environment. The application is a follow-up of an application to educate about and assess risks for sexually transmitted infections ASPO [4], built on an improved platform with modified goals. We discuss the implementation and the functionality in the subsequent sections.

2 Related work

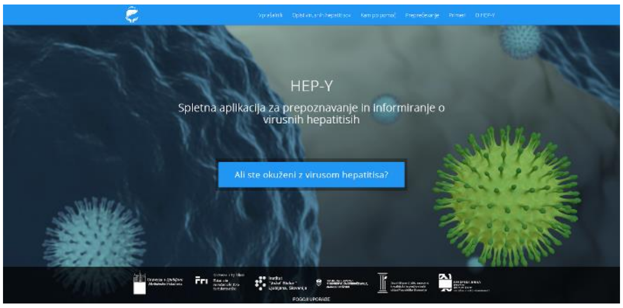
To date, the number of web applications dealing with assessment of risk for viral hepatitis infection and informing general population on different types of viral hepatitises is limited. A descriptive observational study of available viral hepatitis smartphone applications was carried out by Cantudo-Cuenca et al. in 2013 [5]. They identified 232 applications related to viral hepatitis in Google Play Store (Android) and Apple App Store (iOS) of which 33 were selected for further analysis. Most of these apps were uploaded under the medical category.

Only 6 apps had exceeded 1000 downloads. A total of 12 apps were aimed at health professionals, while 4 focused on patients (7 on both of them). The participation of health professionals in the development of apps was 57% [5]. Lack of professional healthcare involvement and lack of public organisation participation in the development of such applications is raising concern regarding the reliability and accuracy of their medical content [6].

3 Website description

3.1 General information

The main purpose of our web application is to provide static information about different hepatitis diseases and general guidelines on how to identify the symptoms, as well as avenues to seek help if need-be. It provides a general overview of each individual disease as well as all recommended steps needed in order to avoid contracting one. This makes it useful both for a potentially sick patient, as well as a healthy person, as it provides necessary curative and preventative information.



HEP-Y

Spletna aplikacija za prepoznavanje in informiranje o virusnih hepatitisih

Figure 1: Landing page of the website.

3.2 Questionnaire

The aim of the questionnaire is to analyse one's symptoms and provide personal feedback based on the given answers. The questions are formed in a way that makes the user reflect on their actions and become acutely aware of the behavior they have exhibited in the past which might have led up to them contracting a potential disease. This is enforced by short messages which appear as the user is answering particular questions, informing them that such behavior is risky and why. At the end, users get a general overview of their answers and how risky their actions were, using color-coding for severity; green, orange, or red.

4 System description

4.1 Frontend

The frontend uses Bootstrap [7] for styling and AngularJS [8] for dynamic content delivery. These technologies allow the page to seamlessly display information and track the user's progress even in the case of a temporary server or internet outage. Each answer is only recorded on the client during the solving of the questionnaire. When finished, the user is prompted and can decide whether to permanently delete the data or to send it anonymously to the server for statistical analysis. If the user does not agree, no data is stored or sent to the server. After completing the questionnaire, they get their final risk assessment that consists of a personalised message based on their answers and risk factors. These may include not being vaccinated, travelling to foreign countries, etc.

4.2 Backend

Django [9] was used for the backend in order to allow a combination of a static website and a dynamic questionnaire, as well as to make potential future improvements easier. The questionnaire contains single and multiple-choice questions. Since there is a large number of questions in total and some are mutually exclusive, answers to certain questions disable other questions. For example, as mandatory vaccination for HAV and HBV was introduced in 1993, people born after this year are not asked about their vaccination status. Similarly, people who did not travel to high-risk countries are not asked about their experience during such trips. This approach allows us to set an intricate web of questions while minimising the amount of time it takes for the user to answer them while still giving full feedback without skimping out on the important details. For easier access and maintenance, rules and other data regarding questions, along with the questions themselves, are stored in a relational database and made accessible via the Django REST framework [10].

5 Methodology

5.1 Designing the database

At first, we had to design our system very generally to allow the administrator to design the website as liberally as possible. This meant designing the structure in a such a way which would allow for possible expansion, as well as allow quick changes to content that is already available on the site. The goal was to create an interface where the administrator could log in with their credentials and change anything on the website at a moment's notice.

Django already provides one such feature in the shape of an administrator page. Once set up, it allows a quick overview of content on the website and running rudimentary queries directly through the page itself (such as inserting new instances of data, altering the existing ones, etc.). The first obstacle we faced is localisation, if we want the web page to be easily translatable and accessible via the admin page, all of the content would have to be stored inside of the database. This is not a problem in and of itself, but it does require some extra work to fetch the content, translate it, and put it back in. We had to sacrifice that accessibility in favour of the content being easily alterable, mostly because we don't expect translations to happen often, but it can be very important to update the content in case of new relevant medical information.

Another obstacle in designing the database was the functioning logic of the questionnaire itself. We could not afford to statically assign the questions as they were continually altered, switched, or completely scrapped even in the process of designing the site. We had to come up with a system that would allow us to assign different types of questions with different values regarding risk assessment, change the order and even which questions can preclude others from being shown based on their answers. All of this had to be easily adjustable by the administrator.

5.2 Designing the questionnaire

In designing the aforementioned system there were a couple of options that we initially considered. We wanted to support different types of questions, so the easiest way to encode that was to have a number of discrete values and some type describing what the question was in the database. We could then programmatically extract the data and insert the appropriate question based on type in our frontend part of the application. This means that inserting a completely new type would require writing code, but for the purposes of our questionnaire, we mainly focused on discrete questions since they allow us to both inform the user, as well as guide them through the process of answering them. This also means that the administrator can switch between types with a single query, for example, changing a “radio” type questions to a “checkbox”.

The second part was enabling a system that could potentially exclude other questions based on the user’s given answers. At first, we wanted questions to be separate entities that would be connected somehow, but we opted for a more flexible approach and utilised a queue. Every question has an assigned id and an order property. When the user begins answering questions, we construct a priority queue based on this order. Each question is comprised of entities called “answers”. Every answer has a property which contains a (possibly empty) set of references (ids) to the questions which should not be should if this answer is chosen. For example, if the user selects a certain answer and its exclusionID is set to 3, we will remove question with id=3 from the queue. As the user progresses through the questionnaire, this question is omitted. This also allows the user to go back and change their answer, making question 3 reappear unless some other answer disables it.

It is also possible to alter the order of the questions. The administrator can do it by hand by simply changing the priority, but it is also possible for a particular answer to trigger the change. We grouped questions into clusters that have similar semantic meanings. If an entity has a special flag that rearranges these, it is possible for an answer to alter the order in which these groups appear in. This is simply done by having another property which tells us which group should come first.

6 Conclusion

As the number of hepatitis infections increases, we need to find a way to inform the general public and provide its members with medically accurate information in order to combat the disease before it develops too far. While the internet holds a lot of information, not all of it is completely medically accurate. The aim of the application is to readily provide all relevant information to users, both identifying potential risk factors or symptoms and learning about preventative measures, while at the same time offering a safe anonymous online environment. With the introduction of this website, we hope to bridge the gap of knowledge and urge users to seek appropriate help and spread this valuable information to others.

Acknowledgement

The project was held under public tender "Po kreativni poti do praktičnega znanja" (Creative path to practical knowledge), funded by Ministry of Education, Science and Sport, and European Union. We thank other participants in the project: Matic Podpadec, Ana Prodan, Saša Rink, and Špela Pišek.

References

[1] WHO http://www.who.int/mediacentre/factsheets/ Accessed: 22-05-2018

[2] J. Tomažič and F. Strle, Infekcijske bolezni, vol. 1. Ljubljana: Združenje za infektologijo, Slovensko zdravniško društvo Ljubljana 2014/2015.

A. Wasley, S. Grytdal, and K. Gallagher, “Surveillance for Acute Viral Hepatitis - United States, 2006,” MMWR Surveillance summaries, 2008.[Online]. Available: https://www.cdc.gov/mmwr/preview/mmwrhtml/ss5702a1.htm. Accessed: 23-05-2018.

[3] N. A. Terrault, N. H. Bzowej, K.-M. Chang, J. P. Hwang, M. M. Jonas, and M. H. Murad, “AASLD guidelines for treatment of chronic hepatitis B,” Hepatology, vol. 63, no. 1, pp. 261–283, Jan. 2016.

[4] Ajanović, A., Konda, J., Fele-Žorž, G., Gradišek, A., Gams, M., Peterlin, A., ... & Matičič, M. Application for sexually transmitted infection risk assessment. *Informatica*, *41*(2), June 2017.

[5] Cantudo-Cuenca MR, Robustillo-Cortés MA, Cantudo-Cuenca MD, Morillo-Verdugo R. A better regulation is required in viral hepatitis smartphone applications. Farm Hosp; 38(2):112-7. April 2014.

[6] S. D. Burdette, T. E. Herchline, R. Oehler, “Surfing the web practicing medicine in a technological age: using smartphones in clinical practice,” Clin Infect Dis; vol. 47, pp. 117-122, Jul. 2008.

[7] Bootstrap CSS, https://getbootstrap.com/css/ Accessed: 2018-19-05.

[8] AngularJS, https://angularjs.org/ Accessed: 2018-19-05.

[9] Django, https://www.djangoproject.com/ Accessed: 2018-19-05.

[10] Django REST framework, http://www.django-rest-framework.org/, Accessed: 2018-19-05.

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 twentyfour 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:

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 2018 (Volume 42) 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.
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 (Simon Dobrišek)
Slovenian Artificial Intelligence Society (Mitja Luštrek)
Cognitive Science Society (Olga Markič)
Slovenian Society of Mathematicians, Physicists and Astronomers (Marej Brešar)
Automatic Control Society of Slovenia (Nenad Muškinja)
Slovenian Association of Technical and Natural Sciences / Engineering Academy of Slovenia (Stane Pejovnik)
ACM Slovenia (Matjaž Gams)

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 42 Number 2 June 2018 ISSN 0350-5596

Informatica

An International Journal of Computing and Informatics

Informatica **42** (2018) Number 2, pp. 145–283