Volume 33 Number 2 May 2009 ISSN 0350-5596

Informatica

An International Journal of Computing and Informatics

1977

EDITORIAL BOARDS, PUBLISHING COUNCIL

Informatica is a journal primarily covering the European computer science and informatics 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.

An Exquisite Mutual Authentication Scheme with Key Agreement Using Smart Card

Chiu-Hsiung Liao
General Education Center
National Chin-Yi University of Technology
Taichung, Taiwan 411, R.O.C.
E-mail: cliao@ncut.edu.tw

Hon-Chan Chen and Ching-Te Wang
Department of Information Management
National Chin-Yi University of Technology
Taichung, Taiwan 411, R.O.C.
E-mail: {chenhc, ctwang}@ncut.edu.tw

Keywords: Diffie-Hellman scheme, transformed identity, authentication, key agreement, password update

Received: June 25, 2008

To access a network system legally, efficiently and securely, the authentication scheme is essential and very important. In this paper, we propose a nonce-based authentication scheme using smart card. We use Diffie-Hellman scheme to enhance the security of our protocol. To lessen the computation load, the remote system alone proceeds the exponentiation computation and it also implements only once. The other computations are all concerned with simple one-way hash functions or exclusive-or operations. No verification table is needed in our protocol. The protocol provides not only mutual authentication between a user and the remote server but also achievement of key agreement. The protocol also supports convenient password update at the user's terminal. To avoid the identity duplication, we introduce the idea of transformed identity in our protocol.

Povzetek: Opisana je nova shema dostopa do omrežja s pomočjo pametne kartice.

1 Introduction

In recent years, people communicate via networks much more frequently than before. The frequency that network users transmit information and share the computing resources increases very quickly. Moreover, with e-commerce being prosperous, people use computers daily to link with server to ask for service. In these situations, the remote authentication and network security become inevitable and very important.

The authentication scheme is an essential part to assure legitimate, secure and efficient access to a network system. Among authentication schemes, password-based authentication is widely used. But password-based authentication is vulnerable to the dictionary attacks [1,2,3,4], i.e. the password guessing attacks, because people are inclined to choose easy-to-remember identities or meaningful phrases. As a result, a number of protocols have been proposed to overcome the guessing attacks [1,5–7]. Some of the improved protocols [1,8–12] use public key encryption in authentication. The others [6,11,12,14] use nonces and one-way hash functions. The nonce-based protocol is more secure because the nonce is randomly generated. As for one-way hash functions, it is irreversible. Thus, the protocol using hash functions and nonces is safe and secure.

Recently, some authentication protocols using smart card have been proposed [6,11,12,14]. Using smart card has many merits. Not only can it implement computations and store a wealth of useful information, like identification number, password and basic personal data, but also it is portable. Although the protocol using public key encryption is much more secure, it may incur a burdensome computation load. Therefore, we propose an authentication protocol using Diffie-Hellman scheme [15] to enhance the security level and efficiency but to reduce the computation load for a smart card. In our method, the smart card is responsible for simple computations and the server is responsible for complicated ones. The proposed scheme also uses the one-way hash function and the exclusive-or operation to maintain security and convenience. To prevent the replay attacks and the synchronization problem, we adopt the nonces in our scheme instead of using time-stamp. Furthermore, we introduce the design of transformed identity [16] in our scheme to avoid the duplication of identities.

The rest of this paper is organized as follows: Some related schemes are reviewed in Section 2. The proposed authentication scheme is described in Section 3. The security analysis of our scheme is discussed in Section 4. The efficiency and specialities of the proposed scheme are discussed in Section 5. The functionality and performance of the proposed scheme are compared with related schemes and the result is listed in Table 1. Finally, the conclusions are given in Section 6.

2 Reviews of related schemes

In this section, we review some related schemes briefly and closely.

2.1 Chien and Jan's scheme (ROSI scheme)

Chien and Jan proposed a nonce-based authentication scheme using smart card: Robust and Simple authentication protocol (ROSI) [6], in 2003. The ROSI scheme consists of two phases: "The registration phase" and "The authentication phase".

In the scheme, a prospective user, u, selects his identity, ID_u, password, PW_u, and an initial nonce, N_1. Then, the user transmits these values to the server, S, in registration phase. After accepting the application, the server stores ID_u and $h^2(PW_u||N_1)$ in its database, where the symbol " $||$ " is the string concatenation. The server also uses its secret key to calculate some parameters and stores them in a smart card. Then, the server issues the smart card to the applicant, u. After the authentication phase, the user and the server can mutually authenticate each other. However, in this scheme, it is necessary to set up a verification table and a legitimate user cannot update his password conveniently and freely when the security faces potential threats.

2.2 Juang's password authenticated key agreement scheme

In Juang's authenticated key agreement scheme using smart card [12], two phases are included: "The registration phase" and "The login and session key agreement phase".

A prospective user submits his identity and password to the server for registration. After getting a smart card, the user can use it to access the server. The user applies his smart card to compute a secret key and uses the key to encrypt a message, which includes a random value and an authentication tag. After receiving the message, the server computes the secret key and decrypts the received message to extract the embedded authentication tag. Then, the server verifies the validity of this tag. In order to attain the shared session key, the user's smart card has to encrypt a forwarding message and decrypt the received message from server to perform a nonce-checking. In this scheme, we found that the smart card should encrypt and decrypt several messages by using the cryptographic scheme. In this situation, the smart card has to compute the modular exponential operations, which require a large amount of computations. These computations may overload the capability of the smart card.

2.3 Hwang et al's remote user authentication scheme

The scheme [14] is comprised of three main phases and an additional one. The main phases are "The registration phase", "The log in phase" and "The authentication phase". The additional phase is "The password changing phase" within the user's discretion.

When a prospective user, u, wants to register with a server, S, he submits his identity, ID_u, and a hash value of password, $h(PW_u)$ to the registration center of S. Then, the center uses the server's secret key, x_s, and the hash value of password to compute a shifted password, $PW_{1u} = h(ID_u \oplus x_s) \oplus h(PW_u)$ and stores it with the hash function, $h(\cdot)$, into a smart card, where " $\oplus$ " is the exclusive-or operation. Then, the smart card is issued to the user.

To access the server, the user connects his smart card to a card reader and keys in his identity and password at the user's terminal. The smart card executes the exclusive-or operation on the shifted password and $h(PW_u)$ to attain a crucial parameter, $h(ID_u \oplus x_s)$. The smart card then combines this parameter with a time-stamp to compute an authenticating value. Next, the user transmits these values to the server for authentication. On receiving the messages, the server executes the verification procedures and performs the authentication. However, although the scheme can verify a legitimate user, the user and the server cannot achieve the mutual authentication and the session key agreement. The scheme cannot avoid the time synchronization problem, either.

2.4 Behind the reviews

In reviewing the related schemes, we are motivated to propose an improved scheme. Not only do we supplement the deficiencies, but we also enhance the efficiency and the functionality. In our scheme, the verification table is not required and the mutual authentication can be achieved. Furthermore, a user is allowed to select and update his password freely. Finally, the computation cost is reduced in the proposed scheme.

3 The proposed authentication scheme

Our authentication scheme consists of four phases: the registration phase, the login and authentication phase, the key agreement phase and the password update phase. As mentioned before, for the sake of security, we prefer to adopt modular exponentiation in registration phase. But, it is performed only at the remote server to reduce the computation load for smart card. The login phase is executed at the user's terminal and the authentication is verified mutually between the user and the server. The key agreement is achieved by the user and the server respectively, and is kept temporarily for mutual communication in the session. As for the password update phase, it is completed only at the user's terminal.

To describe our proposed scheme with ease, we use the following symbols and operations:

1. The operator " $\oplus$ " is the bit-wise exclusive-or operation.
2. The symbol " $||$ " is the string concatenation.
3. The function " h " is a one-way hash function.
4. For the sake of convenience, let the expression " $X \longrightarrow Y : M$ " mean a sender X transmits a message M to a recipient Y.

3.1 The registration phase

The registration phase is performed with the remote server. When a person, u, wants to be a legitimate user to a server, S, he offers an account application to S. The procedure is as follows:

Step 1: $u \longrightarrow S : ID_u, PW_u$. Responding the challenge from the server, the applicant submits his identity, ID_u, and password, PW_u, to the server for registration via a secure communication channel. Both ID_u and PW_u are selected by himself freely.

Step 2: After receiving the response, the server confirms the formats of the submitted identity and password first. Then, the server takes note of the registration time, TS_u and archives the user's ID_u and related TS_u for later authenticating use. Then the server performs the following four processes:

(1) Compute the transformed identity [16], $TID_u = TS_u||ID_u$, automatically by itself. The transformed identity, TID_u, can ensure the uniqueness of the identity. At this stage, the applicant only needs to remember his selected identity, ID_u, and password, PW_u.

(2) Compute $A_u = h(TID_u \oplus x)$, where the parameter x is the secret key of S and is kept confidentially.

(3) Compute $B_u = (g^{A_u} \, mod \, p) \oplus PW_u$, where p is a large prime positive integer and g is a primitive element in Galois field $GF(p)$.

(4) Store the values, TS_u, B_u and $h(\cdot)$, in a smart card and issue the smart card to the applicant.

3.2 The login and authentication phase

When a legitimate user, u, intends to login the server, S, the user's terminal and the server need to mutually authenticate each other.

Step 1: $u \longrightarrow S : M_1 = \{ID_u, NTID_u, C_u\}$.
The user, u, connects his smart card to a reader. The smart card challenges the user for his identity, ID_u, and password, PW_u, which are selected at his application. The smart card automatically performs the following processes:

(1) Generate a nonce, n_u. Store the value, n_u, temporarily until the end of the session.

(2) Retrieve the stored registration time to generate the transformed identity, $TID_u = TS_u||ID_u$.

(3) Compute $NTID_u = TID_u \oplus n_u$.

(4) Compute the value $C_u = h(B_u \oplus PW_u) \oplus n_u$.

(5) Send the message $M_1 = \{ID_u, NTID_u, C_u\}$ to the server, S.

Step 2: $S \longrightarrow u : M_2 = \{D_u, NTID_s\}$.
After receiving the message M_1, S does the following processes:

(1) Retrieve from the database the registration time, TS_u, which is corresponding with the identity, ID_u, of the connecting user. If no such corresponding user matches, the server terminates the connection. Otherwise, it goes on to the next processes.

(2) Compute $TID_u = TS_u||ID_u$, and $n'_u = NTID_u \oplus TID_u$.

(3) Compute $A_u = h(TID_u \oplus x)$ and $g^{A_u} \, mod \, p$, then $h(g^{A_u} \, mod \, p)$.

(4) Compute $n''_u = C_u \oplus h(g^{A_u} \, mod \, p)$. If $n'_u = n''_u$, the received $NTID_u$ is truly sent from u and the parameters n'_u and n''_u should be the same as n_u, which is generated by the smart card at the user's terminal. Hence, the legitimacy of the connecting user is authenticated. See Theorem 1. So, the communication will carry on. On the other hand, if $n'_u \neq n''_u$, the server terminates the connection. Furthermore, the server stores n_u in memory temporarily for later use.

(5) Create a nonce, n_s, randomly. Compute $D_u = C_u \oplus n_u \oplus n_s$ and $NTID_s = TID_u \oplus n_s$. Then the server sends the message, $M_2 = \{D_u, NTID_s\}$, to the connecting user, u.

Theorem 1: If $n'_u = n''_u$, the user, u, is authenticated.

Proof. Since $NTID_u = TID_u \oplus n_u$, thus, $n'_u = NTID_u \oplus TID_u = n_u$.
Also, given $B_u = (g^{A_u} \, mod \, p) \oplus PW_u$, we have

$$\begin{aligned} C_u &= h(B_u \oplus PW_u) \oplus n_u \\ &= h((g^{A_u} \, mod \, p) \oplus PW_u \oplus PW_u)) \oplus n_u \\ &= h(g^{A_u} \, mod \, p) \oplus n_u. \end{aligned}$$

Then,

$$\begin{aligned} n''_u &= C_u \oplus h(g^{A_u}\, mod\, p) \\ &= h(g^{A_u}\, mod\, p) \oplus n_u \oplus h(g^{A_u}\, mod\, p) \\ &= n_u. \end{aligned}$$

It follows that $n'_u = n''_u = n_u$.
The nonce, n_u, is generated at the user terminal when the user, u, inserts his smart card into a card reader. So it is fresh and unique. It is also embedded in $NTID_u$ and never exposed. No one can impersonate it or pry about it. Both TID_u and $NTID_u$ are unique, and $NTID_u$ can be computed by u only. Once $n'_u = n''_u$ is proven, we verify $NTID_u$ is really transmitted by u. Hence, the genuineness of the user, u, is authenticated. □

Step 3: $u \longrightarrow S : M_3 = \{E_u\}$.
When u receives the message M_2, he executes the following processes:

(1) Compute $n'_s = NTID_s \oplus TID_u$ and $n''_s = Cu \oplus n_u \oplus D_u$. If $n'_s = n''_s$, the communication goes on. In this situation, both n'_s and n''_s are equal to n_s, which is generated by the server. Thus, the server is authenticated. See Theorem 2. On the other hand, if $n'_s \neq n''_s$, u ceases the communication. Furthermore, u keeps n_s temporarily at the user terminal for later use.

(2) Compute $E_u = (C_u \oplus n_u)||(n_s + 1)$. Then u sends the message $M_3 = \{E_u\}$ to S. The parameter $n_s + 1$ is the response to the server.

Theorem 2: The server, S, is authenticated if $n'_s = n''_s$.
Proof. Since $NTID_s = TID_u \oplus n_s$, $n'_s = NTID_s \oplus TID_u = n_s$.
Also, since $D_u = C_u \oplus n_u \oplus n_s$, $n''_s = C_u \oplus n_u \oplus D_u = n_s$.
Then, we have $n'_s = n''_s = n_s$.
The nonce, n_s, is immediately generated by S, when S verifies the genuineness of the user, u. So n_s is fresh and unique. The transformed identity, TID_u is also unique. Thus, $NTID_s$ is unique and it can be computed by the server only. Furthermore, D_u is computed with C_u, n_u and n_s. A false server can not forge all of them. Once $n'_s = n''_s$ is proven, the integrity of S is authenticated. □

Step 4: After receiving the message, M_3, the server finds E_u in it. Since $B_u \oplus PW_u = g^{A_u}\, mod\, p$, $C_u = h(B_u \oplus PW_u) \oplus n_u = h(g^{A_u}\, mod\, p) \oplus n_u$. Thus, $C_u \oplus n_u = h(g^{A_u}\, mod\, p)$. So, $E_u = (C_u \oplus n_u)||(n_s + 1) = h(g^{A_u}\, mod\, p)\|(n_s+1)$, and it is really the string concatenation of $h(g^{A_u}\, mod\, p)$ and n_s+1. The server can easily extract $n_s + 1$ from E_u and find n_s in there. At this time, the server ensures that the authenticating user does have the nonce, n_s.

Now, both the user and the server can try for a session key agreement.

3.3 The key agreement phase

After receiving the nonce, n_s, sent from the server, the user creates a session key $SK_u = h((B_u \oplus PW_u)||n_s||n_u)$. Once the server ensures that u has the nonce, n_s, it generates a session key $SK_s = h((g^{A_u}\, mod\, p)||n_s||n_u)$.

Since $B_u = (g^{A_u}\, mod\, p) \oplus PW_u$ is computed in the registration phase,

$$h((B_u \oplus PW_u)||n_s||n_u) = h((g^{A_u}\, mod\, p)||n_s||n_u).$$

Thus, $SK_u = SK_s$. Therefore, the key agreement is achieved and the session key for the session communication is

$$SK = h((B_u \oplus PW_u)||n_s||n_u) = h((g^{A_u}\, mod\, p)||n_s||n_u).$$

3.4 The password update phase

When a user wants to change his password for personal reasons or for the sake of security. He can do so at the user's terminal by performing the following:

Step 1: Insert the smart card into a reader and announce a password update request at the user's terminal.

Step 2: Key in the original password, PW_u. The smart card calculates $B_u \oplus PW_u$.

Step 3: Responding to the challenge of the smart card, the user gives a new password PW^*_u. The smart card calculates $B^*_u = (B_u \oplus PW_u) \oplus PW^*_u$ and then replaces B_u with this new B^*_u. At this time, the password update phase is completed.

4 Security analysis

Not only do we concern with the efficiency and the specialties of our scheme, but also we ask for security and the computational complexity in our proposed scheme. In this section, we will display the strength of our scheme first, and later we discuss the computational complexity. The security analysis is listed as follows:

(1) Our scheme can overcome the guess attacks:
The user is allowed to select his own identity and password freely in our scheme, so he is apt to choose easy-to-remember or meaningful identity and password. In this situation, it seems easy to guess the identity and the password of a legitimate user. However, the construction of transformed identity in our proposed scheme makes the transformed identity be an independent unity. The uniqueness can prevent the transformed identity from being duplicate and resist the guess attacks. An intruder guesses a legitimate

user's identity. The guessed identity can not be converted into a valid transformed identity without the exact registration time, which is stored in the user's smart card. As a result, a intruder's intent to access a remote server should be rejected without a valid transformed identity.

(2) Our scheme is capable of resisting the man-in-the-middle attacks:
A malicious intruder may intercept or eavesdrop on the communication between a legitimate user, u, and the server, S. After intercepting the message M_1 sent by u, he may impersonate u and replay the message to S. Then, he waits for a response message from S. The intruder can not compute the efficient TID_u from the intercepted $NTID_u$ without the nonce, n_u, which is generated randomly by the smart card and is never exposed on the communication. Even though the intruder has the response message, M_2, from S, he can not extract the nonce, n_s, from $NTID_s$, which is included in M_2, because he has no TID_u at hand. The nonce, n_s, is generated by S and is needed to authenticate the server in Step 3(1) in the login and authentication phase. This nonce is also required to achieve the session key agreement. Furthermore, the intruder must respond the server with $n_s + 1$. Because the nonce, n_s, is unavailable, this response can't be completed, either. Eventually, an illegitimate user should be rejected and the connection is terminated.

On the other hand, when a malicious person intercepts the message M_1, he may pretend to be the server that u is connecting to. Furthermore, he has no TID_u and he can't compute it because he has no TS_u at hand. The intruder has no nonce, n_u, either. Thus, he can not send an available parameters to the user u for authenticating the integrity of the server. The communication terminates when the authentication fails.

(3) An intruder can not achieve session key agreement:
The user's password is never exposed in the transmission. An intruder can not intercept the password or any information about it. Meanwhile, the parameter B_u is stored in the user's smart card, no one can access it. So, the parameter $g^{A_u} \, mod \, p$ can not be computed from $B_u \oplus PW_u$. On the other hand, if an intruder intends to compute $g^{A_u} \, mod \, p$ directly. He needs to compute $A_u = h(TID_u \oplus x)$ first. But, the secret key x of the server is kept confidentially. No one can have it. Hence, it is impossible to compute $g^{A_u} \, mod \, p$ directly. Therefore, no session key agreement can be achieved without all of $g^{A_u} \, mod \, p$, n_u and n_s at hand.

(4) An intruder will be confronted with the complexity of the discrete logarithm:
The secret key x of the server is protected by the one-way hash function. It is not possible to derive it from $A_u = h(TID \oplus x)$. Trying to solve out A_u from $g^{A_u} \, mod \, p$ is also impossible, because the adversary will be confronted with the difficulty and the complexity of the discrete logarithm problem. Without secret key x, an adversary can not pretend to be the server, S, in the communication. The parameter, B_u, can't be derived without A_u. Thus, an adversary can not pretend to be the connecting user, u, either.

5 The efficiency and specialties of our scheme

From the procedures of the construction, we point out some merits in our scheme. We concern not only efficiency but also special properties.

(1) No verification table is needed:
Once a prospective user, u, offers his identity, ID_u, and password, PW_u, in registration phase. The server, S, takes note of the registration time, TS_u, to derive the transformed identity, TID_u. Then, S calculates the parameter, B_u, and stores it in a smart card. When the legitimate user wants to access the system, he only gives his selected identity to compute the transformed identity and then transmits it to the remote server. The smart card also generates automatically a nonce, n_u, to compute the authenticating values, C_u and $NTID_u$. Then the values are transmitted to the server. It is not necessary for the remote server to set up any verification table of passwords or other personal information.

(2) The transformed identity is unique:
The construction of transformed identity makes the identity unique. A few users could select the same identities, but the transformed identities should eventually be different since our scheme takes the registration time into account. It prevents the duplication from happening.

(3) The user's identity and password can be selected freely:
Since our proposed scheme uses the transformed identity to discriminate different users, the original identity is allowed to be selected according to the user's preference. Taking into account the registration time, the proposed scheme converts the selected identity into transformed identity. The transformed identities should be different from one another even if the selected identities might be the same. Thus, a user's identity can be selected freely.

The transformed identity is used to compute the parameter A_u. Then, $g^{A_u} \, mod \, p$ is computed. The parameter B_u is generated by performing exclusive-or operation on PW_u and $g^{A_u} \, mod \, p$. Because B_u is stored in the user's smart card, no one can pry about it. Therefore, the password can also be selected freely.

(4) Diffie-Hellman scheme is used:
In registration phase, the server calculates the parameter B_u through Diffie-Hellman scheme to enhance

security. Because the computation of modular exponentiation is burdensome for a smart card, the proposed scheme makes the server execute the operation in order to lessen the troublesome implementation for smart card and to speed up the computation.

(5) The computations proceed very quickly and the load is low:
The modular exponentiation is the only burdensome and time-consuming computation. It is used on the Diffie-Hellman scheme and is performed only once at the remote server. The other computations at both the user's terminal and the remote server are just the one-way hash functions, string concatenations and the exclusive-or operations. The computations proceed very quickly, and the load is extremely low for either of them. The Table 1 demonstrates the computational complexity is simple.

(6) The password can be conveniently updated at the user's terminal:
The server needs no password-verification table to check the a user's genuineness. The proposed scheme allows a user to update his password at his terminal. It is convenient and efficient for users.

(7) The mutual authentication is executed:
The scheme can mutually authenticate each other between the user and the server. From the Theorem 1 and 2, the correct methods of the mutually authentication between the user and the remote server are proven.

At the end of this section, we compare our proposed scheme with some other schemes on the computational complexity and the performances.

The comparison on computational complexity is also listed in Table 1.

From an objective point of view about the performance, we include some criteria in the following items:

Item 1. No verification table needed: At the remote server, a password-verification table is not needed to authenticate the users.

Item 2. Using unique transformed identity: Describe whether a user can choose his identity according to his preference and prevent it from duplication.

Item 3. Choosing a password freely: Display whether a scheme allows a user to choose his password freely or not.

Item 4. Mutual authentication: Demonstrate whether a legitimate user and the remote server can mutually authenticate each other or not.

Item 5. Password update conveniently: Discuss whether a user can conveniently update his password at the user's terminal or not.

Item 6. Session key agreement: Show whether a scheme can achieve the session key agreement or not.

Item 7. Avoiding time synchronization problem: Exhibit whether a scheme can avoid the time synchronization problem or not.

The result of the comparisons on the performances is listed in Table 2.

6 The conclusions

We have proposed an exquisite mutual authentication scheme without verification table of passwords and other users' personal information. The proposed scheme includes session key agreement and convenient password update. Our scheme uses the registration time to create the unique transformed identity in order to discriminate a user from the others efficiently, even if they may choose the same value for their identities. Through the storage of important information in the smart card, the proposed scheme can generate necessary parameters without exposing the password in transmission. Our scheme can withstand the replay attacks and resist the man-in-the-middle attacks. Moreover, the security of our scheme relies on the intractability of discrete logarithm because the Diffie-Hellman scheme is used.

References

[1] S.M. Bellovin, M. Merritt, (1993) Augmented encrypted key exchange: A password-based protocols secure against dictionary attacks and password file compromise, *Proceedings of First ACM Conference on Computer & Communications Security,* pp.244–250.

[2] Y. Ding, P. Horster, (1995) Undetectable on-line password guessing attacks, *ACM Operating Syst. Rev.*, pp.77–86.

[3] DV. Klein, (1990) Foiling the cracker: a survey of, and improvements to password security, *Proceedings of the second USENIX UNIX security workshop,* pp.5–14.

[4] R. Morris, K. Thompson, (1979) Password security: a case history. *Communications of the ACM,* 22(11), pp.594–597.

[5] V. Goyal, V. Kumar, M. Singh, A. Abraham, and S. Sanyal, (2006) A new protocol to counter online dictionary attacks, *Computers & Security*, 25, pp.114–120.

[6] H.Y. Chien, J.K. Jan, (2003) Robust and simple authentication protocol, *Computer Journal*, 46, pp.193–201.

[7] C.L. Lin, H.M. Sun, T. Hwang, (2001) Attacks and solutions on strong-password authentication, *IEICE Trans. Commun.* E84-B, No. 9, pp.2622–2627.

[8] S. Halevi, H. Krawczyk, (1998) Public-key cryptography and password protocols, *Proceedings of the 5th ACM Conference on Computer and Communications Security*, San Francisco, CA, pp.122–131.

[9] C.C. Chang, W.Y. Liao, (1994) A Remote Password Authentication Scheme Based upon ElGamal's Signature Scheme, *Computer & Security*, Vol. 13, pp.137–144.

[10] C.C. Chang, L.H. Wu, (1990) A Password Authentication Scheme Based upon Rabin's Public-Key Cryptosystem, *Proceedings of International Conference on Systems Management '90*, Hong Kong, pp.425–429.

[11] M.S. Hwang, L.H. Li, (2000) A new remote user authentication scheme using smart card, *IEEE Transactions on Consumer Electronics*, 46(1), pp.28–30.

[12] W. S. Juang, (2004) Efficient password authenticated key agreement using smart card, *Computer & Security*, 23, pp.167–173.

[13] Y.C. Chen, L.Y. Yeh, (2005) An efficient nonce-based authentication scheme with key agreement, *Applied Mathematics and Computation*, 169, pp.982–994.

[14] M.S. Hwang, C.C. Lee, Y.L. Tang, (2002) A simple remote user authentication scheme, *Mathematical and Computer Modelling*,36, pp.103–107.

[15] W. Diffie, M. Hellman, (1976) New directions in cryptography, *IEEE Trans. Inform. Theory*, 22, pp.476–492.

[16] C.T. Wang, C.C. Chang, C.H. Lin, (2004) Using IC Cards to Remotely Login Passwords without Verification Tables, *Proceedings of the 18th International Conference on Advanced Information Networking and Application*(*AINA*), Fukoka, Japan, pp.321–326.

Phase	Registration	Login and Authentication	Key Agreement	Password Update
Our scheme	1Co 1Ha 2⊕ 1ME	3Co 2Ha 9⊕ 1ME	Yes	Yes
Chien et al [6]	2Co 3Ha 1⊕	5Co 17Ha 10⊕	No	No
Hwang and Li [11]	1ME	2Ha 2⊕ 5ME 2MM	No	No
Juang's [12]	1Ha 1⊕	1Co 4Ha 1⊕ 3En 3De	Yes	No
Hwang et al [14]	2Ha 2⊕	1Ha 2⊕	No	Yes

Co: concatenation; Ha: one-way hash function; ⊕: exclusive-or;
ME: modular exponentiation; MM: modular multiplication;
En: encryption; De: decryption

Table 1: Comparison on Computational Complexity

Criterion item	No verification table	Using transformed ID	Choosing PW freely	Mutual authentication	PW update	Session key agreement	Avoiding synchronization
Our scheme	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Chien et al's [6]	No	No	Yes	Yes	No	No	Yes
Hwang and Li's [11]	Yes	No	No	No	No	No	No
Juang's [12]	Yes	No	Yes	Yes	No	Yes	Yes
Hwang et al's [14]	Yes	No	Yes	No	Yes	No	No

Table 2: The result of the comparisons on performancesamong schemes

Routing Scalability in Multicore-Based Ad Hoc Networks

Ami Marowka
12 Anna Frank, Ramat-Gan,
52526, Israel
E-mail: amimar2@yahoo.com

Keywords: ad hoc networks, routing speedup, multicore, power management

Received: December 4, 2007

The integration of multicore processors into wireless mobile devices is creating new opportunities to enhance the speed and scalability of message routing in ad hoc networks. In this paper we study the impact of multicore technology on routing speed and node efficiency, and draw conclusions regarding the measures that should be taken to conserve energy and prolong the lifetime of a network.
We formally define three metrics and use them for performance evaluation: Time-to-Destination *(T2D),* Average Routing Speedup *(ARS), and* Average-Node-Efficiency *(ANE). The T2D metric is the time a message takes to travel to its destination in a loaded traffic network. ARS measures the average routing speed gained by a multicore-based network over a single-core based network, and ANE measures the average efficiency of a node, or the number of active cores.*
These benchmarks show that routing speedup in networks with multicore nodes increases linearly with the number of cores and significantly decrease traffic bottlenecks, while allowing more routings to be executed simultaneously. The average node efficiency, however, decreases linearly with the number of cores per node. Power-aware protocols and energy management techniques should therefore be developed to turn off the unused cores.

Povzetek: Narejena je analiza povezljivosti vozlov omrežja.

1 Introduction

The recent emergence of affordable dual-core processors in consumer products will overturn many currently accepted standards for software applications(5). The transition from single to multicore CPUs calls for the parallelization of all applications. Developers will be faced with the challenge of designing single-threaded applications that run efficiently on multiple cores.

Dual-core processors are only the beginning. Chip makers are currently working on the next generation of multicore processors, which will contain 4, 8 or 16 cores on a single die. According to the roadmap introduced by Intel(15), dual core processors are slated to reach mobile devices as well.

The integration of multicore processors into wireless communication and mobile computation devices will in general strengthen the communication infrastructure. Ad hoc wireless networks of mobile devices will also become more robust. An ad hoc network is a self-organized mobile network, whose every node is responsible for both computation and communication operations (1; 11).

In this paper we address the problem of how to measure the gain in routing speedup in ad hoc wireless networks where nodes are equipped with multicore processors, in particular when the network is heavily loaded. Moreover, we analyze the efficiency of multicore nodes in the network and show that the energy consumption of multicore nodes could be dramatically reduced by adapting existing methods and techniques.

We use location-based routing protocols in our traffic simulations(12). Location-based protocols are usually compared and analyzed by means of the “hops count” metric; i.e., the best single-path routing protocol is the one that finds a path from the origin node to destination node with the fewest number of hops. In real networks, however, many routing tasks are carried out at the same time. This is known in the literature as a multiple-sessions scenario, in which several routing sessions compete for the node’s services. Each message wants to reach its destination node without waiting in a queue of routing sessions to be served. In such a scenario, we need a metric to quantify the arrival time of messages to their destinations.

This work therefore starts by defining a routing metric, called the *Time to Destination (T2D).* This metric will quantify the time required for a message to travel along the best path determined by the routing protocol. Then, two other metrics are defined based on the T2D: the *Average Routing Speedup* (ARS) and *Average Node Efficiency* (ANE). The first quantifies the average improvement in message travel time for a multicore network compared to a single-core network. The second measures the average node efficiency in a multicore network, compared to the efficiency of a network with single-core nodes.

These metrics serve to analyze the behavior of routings in our simulations of heavily loaded networks. We derive conclusions regarding the benefits of message routing in multicore networks, and show how power-aware management protocols can reduce the consumption of energy. Moreover, we analyze the effect of mobility on the delivery success rate in multicore ad hoc networks and show that multicore nodes can improve the dependability of mobile networks. To the best of our knowledge, this is the first work in the open literature discussing routing protocols in ad hoc networks of multicore devices.

The design principles behind the speedup and the efficiency metrics present in this work were inspired from the parallel computing literature. However, our evaluation methods using scalable networks for measuring the *real* scalability of routing protocols are novel. In order to explore how protocols scale as the number of the nodes increases and as the average-node-degree increases, our simulator has the capability to generate extendable networks. In the open literature, the scalability is evaluated by measuring the performance of different number nodes such as 10, 20, 30,... where, for example, the 10-nodes network has different topology from the 20-nodes network. Our unique method for generating planar random networks enables us to generate 20-nodes network which is *extension* of the 10-nodes network and therefore to measure and to evaluate *real* scalability. In Section 4 we describe in detail this method, that to the best of our knowledge, we are the first to use. A work-in-progress versions of this paper was presented in (9; 10).

The remainder of the paper is organized as follows. Section 2 describes state-of-the-art works in the area of MIMO ad hoc networks. Section 3 defines the network model. In section 4, the metrics described above are defined. Section 5 details the simulations and analyzes the resulting data. Section 6 summarizes our results.

2 State-of-the-Art of MIMO MANET

Multiple-input Multiple-output (MIMO) wireless communication systems are the most promising multiple antenna technology today (16; 17). The advantages of MIMO communication, which exploits the physical channel between many transmit and receive antennas, are currently receiving significant attention (18). The integration of an air interface technology, such as MIMO, with a modulation scheme called orthogonal frequency division multiplexing (OFDM) (19) has the potential to lay the foundation for the data rate and capacity gains that will be needed for years to come. Since multiple data streams are transmitted in parallel from different antennas there is a linear increase in throughput with every pair of antennas added to the system. MIMO systems do not increase throughput simply by increasing bandwidth. They exploit the spatial dimension by increasing the number of unique spatial paths between the transmitter and receiver.

MIMO-OFDM combines OFDM and MIMO techniques thereby achieving spectral efficiency and increased throughput. A MIMO-OFDM system transmits independent OFDM modulated data from multiple antennas simultaneously. At the receiver, after OFDM demodulation, MIMO decoding on each of the sub-channels extracts the data from all the transmit antennas on all the sub-channels. The IEEE 802.16e standard incorporates MIMO-OFDMA.

The MIMO capability of antenna arrays has been studied at the physical layer and over a single link. There are few research directions that have studied MIMO in a multi-hop network from the perspective of higher layers. The research in (20) proposed a scheduling algorithm to offer fair medium access in a network where nodes are equipped with MIMO antennas. The model under study provides a simple abstraction of the physical layer properties of MIMO antennas. At the routing layer, a routing scheme to exploit MIMO gains is proposed (21). The idea is to adaptively switch the transmission/reception strategy using MIMO so that the aggregate throughput at the routing layer is increased. At each hop along a route this decision is made dynamically based on network conditions such as node density and traffic load.

At the transport layer, TCP performance over MIMO communications was studied (22). Focusing on the two architectures previously proposed to exploit spatial multiplexing and diversity gains (namely BLAST and STBC), the authors studied how the ARQ and packet combining techniques impact on the overall TCP performance. Their results indicate that, from the standpoint of TCP performance, the enhanced reliability offered by the diversity gain is preferable to the higher capacities offered by spatial multiplexing.

3 A Multicore Network Model

A highly realistic network model would take into account many complexities, such as the control traffic overhead, traffic congestion, mobility of the nodes, the irregular shape of radio coverage areas, and the intermittence of communication due to weather conditions and interference from preexisting infrastructure (power lines, base stations, etc.). Including *all* these details in the network model, however, would make it extremely complicated and scenario-dependent. This would hamper the derivation of meaningful and sufficiently general analytical results. It was shown that a simple parameterized model can accurately reflect the simulations (14). The model defined here, which is used in our simulations, therefore makes some widely accepted simplifying assumptions.

We formally define a multicore network model as follows:

Definition 1: A multicore network model is an undirected graph $G \equiv (V, E, D, M)$, where:

1. *V is the set of nodes;*
2. *$E \subseteq VxV$ is the set of undirected edges i.e., $(u, v) \in E$ if u is able to transmit to v;*

3. D is the average node degree;
4. M is the number of cores in a node;
5. $P_{i,j}$ is the link probability of retransmission;
6. For $v \in V$, the node mobility is determined every pause time pau(v) by its velocity vel(v) towards destination des(v).

A realistic multicore model must reflects the potential cost of retransmissions required to recover from link errors. We use a linear transmission cost function: $T_{i,j} = \frac{\lambda L}{1-P_{i,j}}$ where a link is assumed to exist between node pair (i, j) as long as node j lies within the transmission range of node i, λ is the transmission rate, L is the message size and $P_{i,j}$ is the link packet error probability associated with that link. This cost function captures the cumulative delay expended in reliable data transfer, for both reliable and unreliable link layers.

Node mobility is based on the *random waypoint* parameterized model (2): a node chooses a destination uniformly at random in the simulated region, chooses a velocity uniformly at random, and then moves to that destination at the chosen velocity. Upon arriving at the chosen waypoint, the node pauses for a period before repeating the same process. In this model, the pause time represents the degree of mobility in a simulation; a longer pause time amounts to more nodes being stationary for more of the simulation.

The nodes communicate using omnidirectional antennas with maximum range r. We assume that all the nodes are equipped with identical transceivers, and further that each $M - core$ node is equipped with M transceivers and M antennas. Each core has multiple wireless channels and identical hardware and software mechanisms. A multicore node can hence transmit multiple packets at the same time to different nodes. The network is assumed to be homogeneous, consisting of nodes with the same transmission range, number of cores, and battery power. Imposing a common transmission range induces a strongly connected communication graph, often called a *unit disk graph* in the literature of routing protocols.

We assume that the traffic in the network is highly loaded, and that routing sessions are issued in a random manner. In other words, the origin and destination nodes of each routing are chosen randomly and no *a priori* knowledge is available regarding future sessions. Each node maintains a queue, so that incoming routing sessions are served on a FIFO basis. For simplicity it is assumed that the queue is large enough to store all arriving messages, so no messages are lost due to overflow. If the network consists of multicore nodes with M cores, then each node can serve M routing requests simultaneously.

The model assumes that different networks may have different ratios of computation time to communication time, and that communication and computation overlap. The communication time is the amount of time it takes for a packet transmitted by one node to be received by the next node on the path. The computation time is the amount of time it takes for a packet to be processed, from the time it is received by a node to the time it is transmitted to the next node. This interval includes computationally intensive functions such as signal processing, encoding and decoding, and encrypting and decrypting. It also includes relatively lightweight functions such as next-hop routing decisions and channel access delay.

Routing requests in our model are managed by localized, location-based protocols (12). In such protocols the location of the destination node is known, and the distance to neighboring nodes can be estimated on the basis of incoming signal strengths. The routes between nodes are created through a series of localized hop decisions; each node decides which neighbor will receive the message based on its own location, its neighboring nodes, and the message's destination. In our simulations we use the *Most Forward within Radius* (MFR) protocol (13), which forwards the message to the neighbor that maximizes its progress.

4 Routing Speedup and Efficiency

In measuring the scalability of parallel applications, the most commonly used practical metrics are *relative speedup* and *relative efficiency* (7; 8). *Relative Speedup* is the ratio of a parallel application's run time on a single processor to its run time on N processors; it is important to emphasize that the application and its test problem are identical in both situations. *Relative Efficiency* is the relative speedup divided by the number of processors N. Researchers use the speedup metric to check the performance of their applications on multiple platforms; it is a natural choice, since it is dimensionless and captures the relative benefit of solving a problem in parallel.

Motivated by these definitions, we define similar metrics for measuring the scalability of routing in heavily loaded, multi-session ad hoc networks. First, we define a time-based routing metric called the *Time-to-Destination* (T2D). Second, we define the *Routing-Speedup* (RS) and the *Average-Routing-Speedup* (ARS) metrics. Finally, we define the *Node-Efficiency* (NE) and *Average-Node-Efficiency* (ANE) metrics.

The metrics used in simulations of wireless ad hoc networks usually reflect the goal of the network design protocol. Most routing schemes thus use *hop count* as their metric, where hop count is the number of transmissions along a given route from source to destination. This choice agrees with the assumption that delay is proportional to hop count, which is reasonable when the impact of congestion is not significant. However, this assumption is not warranted for realistic ad hoc network scenarios in which routing sessions are issued from many and various origin nodes simultaneously and to random destinations. We therefore need a metric that reflects the delay caused by traffic congestion in a wireless network.

We formally define the Time-to-Destination metric as follows:

Definition 2: Time-to-Destination (T2D).
Assume a unit disk multicore graph G, and a route

$R = v_s, ..., v_d$ from source node v_s to destination node v_d. The Time-to-Destination (T2D) is the aggregate time taken to route a message from the origin node to the destination node.

The T2D metric is also known in the open literature as the end-to-end latency. We assume that different networks have different ratios of computation time to communication time, and that communication and computation overlap. Thus, in the case of a routing with delay times $d_s, ..., d_d$ at each hop the total time it takes for a message to traverse the route is given by

$T2D = L * (T_{comm} + T_{comp}) + \sum_{i=s}^{d} d_i$,

Where T_{comm} and T_{comp} are the total communication time and total computation time respectively. L is the length of the route, the number of hops from the source node to the destination node.

Definition 3: Routing Speedup (RS).

Given a unit disk multicore graph G and a route $R = v_s, ..., v_d$ from source node v_s to destination node v_d, the Routing Speedup (RS) of R is the ratio of message routing time for a network with single-core nodes to the message routing time for a network with M-core nodes:

$RS = \frac{T2D_1}{T2D_m}$,

Where the subscripts 1 and m indicate the single-core and M-core networks respectively. Since it is more practical to measure the average speedup in a scenario with multiple sessions, we also define the *Average Routing Speedup* as follows:

Definition 4: Average Routing Speedup (ARS).

The Average Routing Speedup (ARS) is the average speedup over R routings in G: $ARS = \frac{1}{n} \sum_{i=1}^{n} RS_i$.

The ARS metric is a practical tool for evaluating the speed gained by moving from an ad hoc network with single-core nodes to one with multicore nodes. ARS does not, however, measure the efficiency of a multicore node. (By node efficiency, we mean the average number of cores that are busy per node.) As will be shown in the next section, knowing the node efficiency permits a dramatic reduction of power consumption in each node and thus increases the lifetime of the whole network.

Definition 5: Node Efficiency (NE).

Assume a unit disk multicore graph G, and R routings executed in G over a period Δt. The Node Efficiency (NE) of a given node during Δt is the ratio of T^1_{comp}, its aggregate computation time in the case of a single-core network, to M times T^m_{comp}, its aggregate computation time in the case of an M-core network.

$NE = \frac{T^1_{comp}}{M * T^m_{comp}}$,

For the same practical reasons mentioned with respect to the RS and ARS metrics, we define the *Average Node Efficiency* as follows:

Definition 6: Average Node Efficiency (ANE).

The Average Node Efficiency (ANE) in G of $|V| = n$ nodes during time Δt is: $ANE = \frac{1}{n} \sum_{i=1}^{n} NE_i$.

It is important to notice that NE and ANE do not take into account idle times, only those times when the nodes are involved in the routing process. The aim of these metrics is to measure the efficiency of the cores within a node, rather than a given node's dominance over other nodes in the routing process. Although this information is important for intelligent power management and energy conservation, the efficiency metrics we define here focus on what happens inside a multicore node.

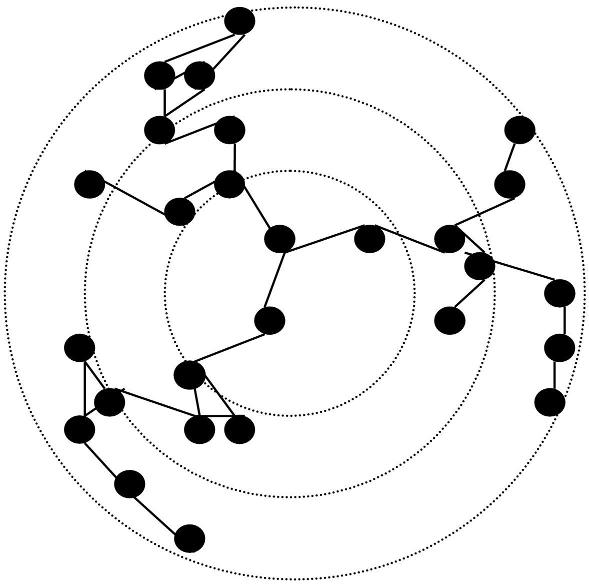


Figure 1: Illustration of Scalable Planar Network of 27 nodes with c=3.

In the next section we use these metrics to evaluate and analyze the implications of multicore nodes on position-based routing protocols and power management strategies in ad hoc networks.

5 Simulator and Simulations

A discrete event simulator was developed in order to monitor, observe, and measure ARS and ANE in multicore ad hoc networks. We generated a database with hundreds of random unit graphs, with values of V and D spanning a wide range of sparse and dense networks. The results shown in this paper are only a representative sample of the many simulations performed, and each result is averaged over many runs.

The network generator first partitions the plane into k regions: an innermost disk whose radius is equal to the transmission range r, and a series of concentric annuli of width r surrounding the disk. The number of nodes in each annulus is proportional to its area. If c nodes are randomly located in the inner disk of a network with k regions, then a total of $c * k^2$ nodes will be randomly placed in the entire network. For example, if the innermost region of a network has 3 nodes (c=3) then 9, 15, and 21 nodes will be located in the

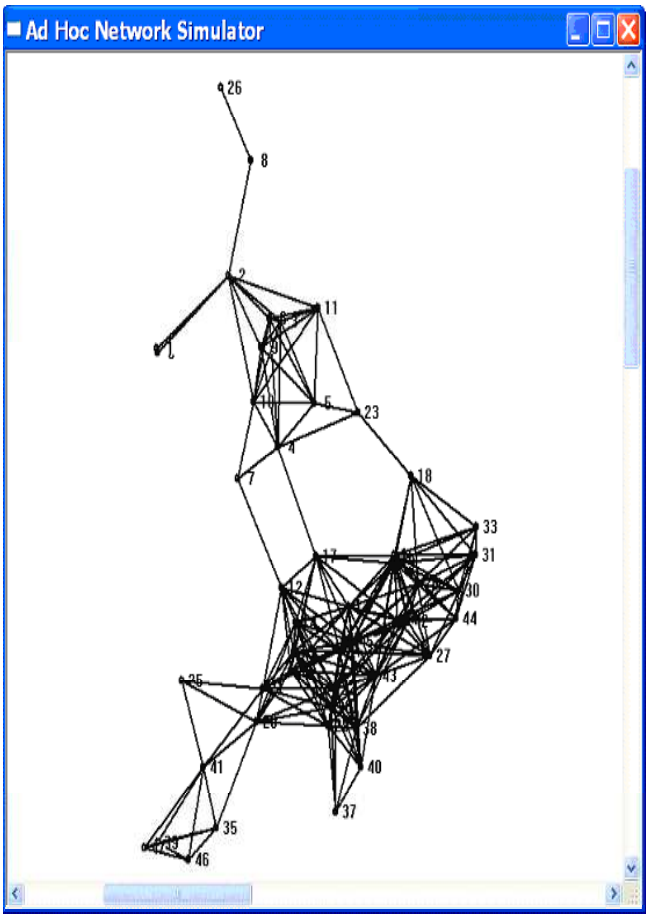


Figure 2: A randomly generated 4-regions network of 48 nodes and average node degree of 4.

first three rings respectively. Figure 1 illustrates a network of 27 nodes for c=3. The networks are thus generated incrementally, ring after ring. A network with k regions is just an extension of the network with $(k-1)$ regions. In this way we can calculate the scalability of our protocols exactly.

For small networks (up to 50) we choose the value of c to be 3 and for large networks the value was 4. The networks were extension of a base network of 27 nodes (in case of c=3) or 36 nodes (in case of c=4). The average-node-degree of the base network was preserved in the extended networks. Usually, the topology of randomly generated scalable networks are not symmetrical like the illustration shown in Figure 1. An example of a randomly generated 4-region network of 48 nodes and average-node-degree of 4 is shown in Figure 2.

All the simulations shown in this paper were carried out on a network of 108 nodes, with an average node degree of 7. Measurements were performed for three traffic loads: 100, 1K and 10K routings. The source and destination nodes of each route were randomly chosen, and all routings were issued simultaneously. The scalability of routing was tested assuming values of 2, 4, 8, and 16 cores per node. Each simulation shown in this section (except Fig. 5) was performed assuming that the ratio of computation time to communication time is 1. The routing protocol used in our benchmarks is the *Forward within Radius (MFR)* protocol.

For the simulations of our multicore network model we used parameter values that are the average values of the IEEE 802.11-based interfaces as appeared in (4) and summarized in table 1. Each core is assumed to be able to transmit and receive 2Mbps. The packet size was of 64KB and the link probability of retransmission was set to 0.25.

For the network mobility it was assumed that all nodes move according to the random waypoint model. First, a node chooses a destination uniformly at random in the simulated region. The area $A(N, R)$ of a simulated region is determined relative to the number of nodes in the network ($N = c * k^2$) and the data transmission range R. Thus, $A(N, R) = (k * R)^2 * \pi = (N/c) * R^2 * \pi$. For example, a scalable network of 48 nodes with $c = 3$ has 4 rings and thus the area of the network region is $(4R)^2\pi$. In the simulations we used data transmission range of 250m. Next, the node chooses a velocity uniformly at random, with a maximum velocity of 10 m/s, and moves to that destination at the chosen velocity. Upon arriving at the chosen waypoint, the node pauses for a period before repeating the same process. We simulate pause times in the range of 0-10 seconds.

Table 1: Parameter values used in the simulations.

Communication	
Packet size	64KB
Retransmission probability	0.25
Mobility	
Simulated area	$(N/c) * R^2 * \pi$
Transmission range	250m
Velocity	0-10m/s
Pause time	0-10s

Figures 3 and 4 depict the routing scalability that can be expected from a multicore-based network. Figure 3 plots ARS as a function of the number of routings for nodes with 2, 4, 8 and 16 cores. Figure 4 plots ARS as function of the number of cores per node for traffic loads of 100, 1K and 10K routings. We choose to present the same information in two different ways to make all the relationships clearly visible.

Analysis of these results leads to the following findings. First, ARS increases as the number of the routings increases; this is obvious from Figure 3. For 10K routings the ARS increases linearly with the number of cores, up to values of 1.96, 3.82, 7.26 and 13.27 for 2, 4, 8 and 16 cores respectively. For 100 routings, however, the ARS reaches its maximum value of 1.35 at only 4 cores (Figure 4). Adding more cores does not increase the ARS unless there is more traffic to handle. These encouraging results show that mul-

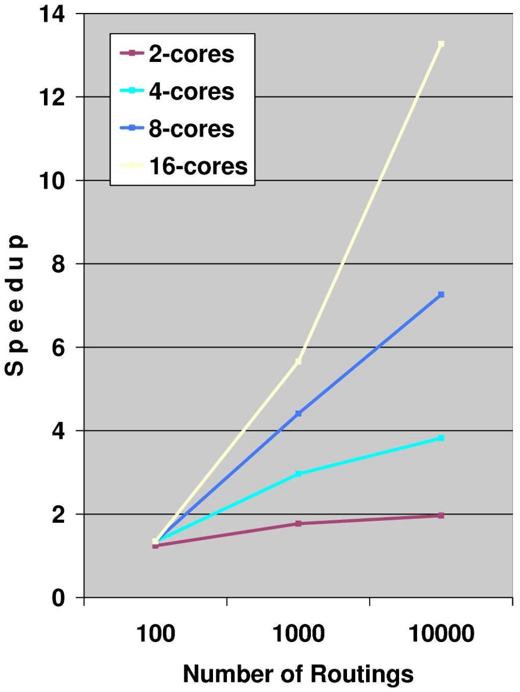


Figure 3: Graphs of Average-Routing-Speedup as function of number of routings.

ticore nodes decrease the congestion of loaded networks, increase the routing speedup, and decrease the travel time of messages.

Figure 5 plots the ARS as function of the number of cores per node and the ratio of computation time to communication time. The goal of this paper is to study the impact of various multicore ad hoc networks on routing scalability, so we varied the ratio of computation time to communication time in the range 0.5 to 3. Delays in computation and communication can be incurred from many sources: next-hop routing decisions, channel access delays, transmission delays, traffic load, the sizes of contending packets, the medium access control algorithm used by the nodes, the modulation and symbol rate of the packets, and finally the distance that the packets must travel. Moreover, wireless communication usually requires several network-dependent, computationally-intensive functions such as signal processing, encoding and decoding, and encrypting and decrypting. Figure 5 shows that as the ratio of computation time to communication time increases, the degree of speedup for large core numbers improves. In the case of 16 cores the ARS increases by 34% as the computation time to communication time ratio is varied in the range 0.5 to 3. In the case of 8 cores the improvement is only by 8% and for 2 and 4 cores there is no improvement at all. However, in the case of 2, 4 and 8 cores the ARS is high. This phenomenon matches also the results shown in Figure 8. The network reaches its load balancing equilibrium point when the number of cores reaches 16. At this point each node has enough available cores to amortize the increase in the ratio of computation time to communication time.

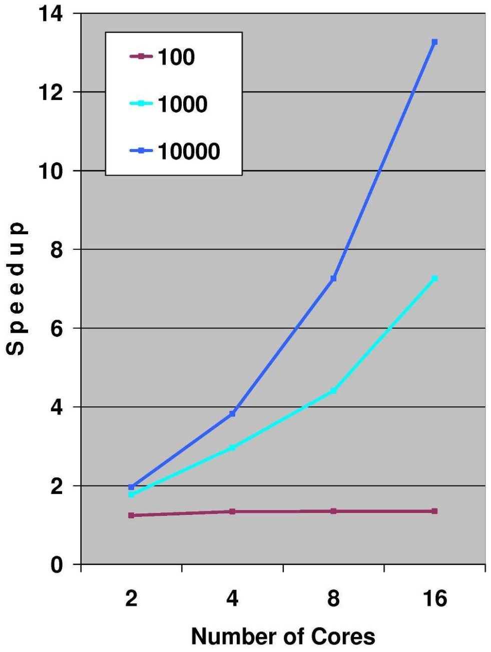


Figure 4: Graphs of Average-Routing-Speedup as function of number of cores.

Figures 6, 7 and 8 describe the node efficiency of multicore-based networks. Figure 6 graphs ANE as a function of core number for the cases of 100, 1K and 10K routings. Figure 6 graphs ANE as a function of the number of routings for the cases of 2, 4, 8, and 16 cores. Once again, we choose to present the same information in two different ways to clarify the relations. Figure 8 is a histogram showing the distribution of routing loads over all nodes for the case of 10K simultaneous, random routings.

Analysis of these results reveals the following relationships. The ANE asymptotically increases with the number of routings (Figure 6). For example, in the case of 8-core network ANE approaches the values 0.18, 0.54 and 0.70 for 100, 1K, and 10K routings respectively. However, the ANE decreases dramatically as the number of cores increases (Figure 7). For example, in the case of 10K routings ANE approaches the values 0.88, 0.78, 0.70 and 0.63 for 2, 4, 8 and 16 cores respectively. In other words, as the number of cores increases more cores will remain idle.

Figure 8 presents another view of this phenomenon. This histogram shows how the routing load is distributed among the nodes. For example, in the case of a single-core network most nodes (80 of 108) were not busy at least 40% of the time (marked *low* in the histogram). 18 nodes were busy between 40% to 80% of the time (marked *moderate*), and 10 nodes were busy more then 80% of time (marked *high*). These 10 nodes are the *dominant set* of the network, through which most of the traffic passes. However, as the number of cores increases more nodes become busy more

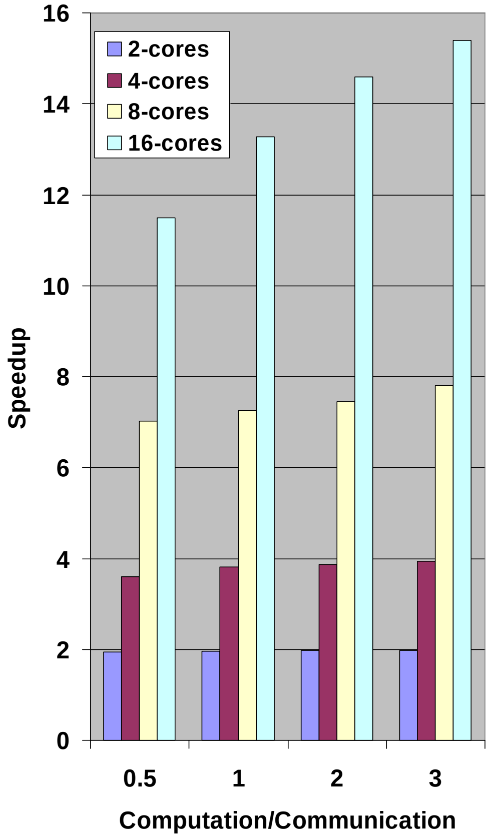


Figure 5: Graphs of Average-Routing-Speedup as function of number of cores and computation to communication ratio.

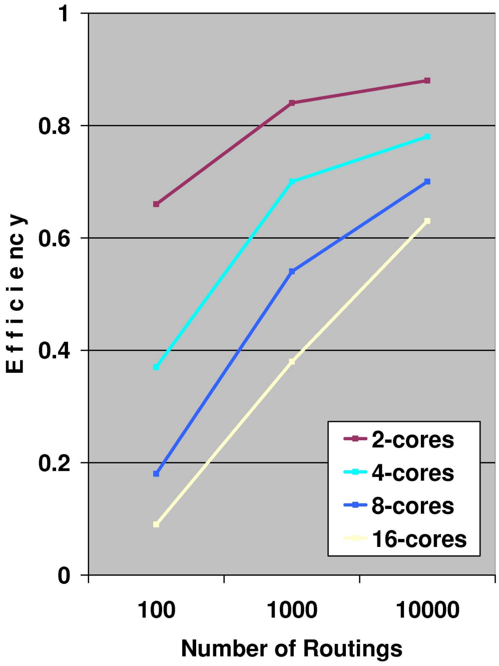


Figure 6: Graphs of Average-Node-Efficiency as function of number of routings.

of the time. This indicates improved load balancing and decreased traffic congestion. For example, in the case of 16 cores only 42 nodes are idle most of the time, 28 nodes are busy about half of the time, and 38 are busy most of the time.

These observations have important implications for the power management strategies that should be taken to reduce energy consumption in multicore-based networks. Many design techniques for reducing the power consumption of mobile devices have been introduced over the last decade (4; 6; 3). Among these are technologies which enable the processor to operate at multiple voltages and frequencies depending on the workload required by the user. Thus, when workload drops the processor steps down to a lower voltage and frequency, conserving battery power. Processors including a power-optimized system bus with a low-power L2 cache, which turn off parts of the high-speed memory when it isn't needed, also result in an overall reduction of the power consumption. A *minimization of capacitance* can also be achieved by relying on chip-based resources such as caches and registers.

Our measurements of Average Node Efficiency show that it is essential to develop power-aware protocols and complementary hardware capabilities that will permit a multicore node to dynamically adjust the number of active cores. As can be seen from Figure 6, a 16-core network under traffic loads of 10K and 1K routings has ANE values of 0.6 and 0.4 respectively. In other words, energy savings of up to 40% to 60% could be achieved by allowing individual cores to be turned off or put in a sleep mode.

Figure 9 and Figure 10 depict the effect of the network topology characteristics on the routing scalability. Figure 9 plots the speedup as function of number of nodes, 16 cores each, for 100, 1K and 10K routings. It can be observed from the graphs that for high traffic load (10K routings) the speedup decreases from 15.4 to 13.27 when the number of nodes increases from 27 to 108 respectively. Since low traffic load leads to low speedup, as shown in Figure 3, increasing the number of nodes, for the same global traffic load, decreases the local traffic load in each node and thus decreases the speedup. For low traffic loads (100 and 1K routings) the impact of the number of nodes on the speedup is marginal. Figure 10 plots the speedup as function of average-node-degree for network of 108 nodes, 16 cores each, and for 100, 1K and 10K routings. Since low average-node-degree increases the traffic load in the nodes, it is expected that the speedup will increase as well. Figure 10 shows that for high traffic load (10K routings) the speedup increases from 12.2 to 14.8 when the average-node-degree decreases from 9 to 4 respectively as expected. For traffic load of 1K routings the speedup increases slightly and for 100 routings the speedup remains 1.35 for 4, 7 and 9 degrees.

Figure 11 depicts the effect of mobility on the delivery

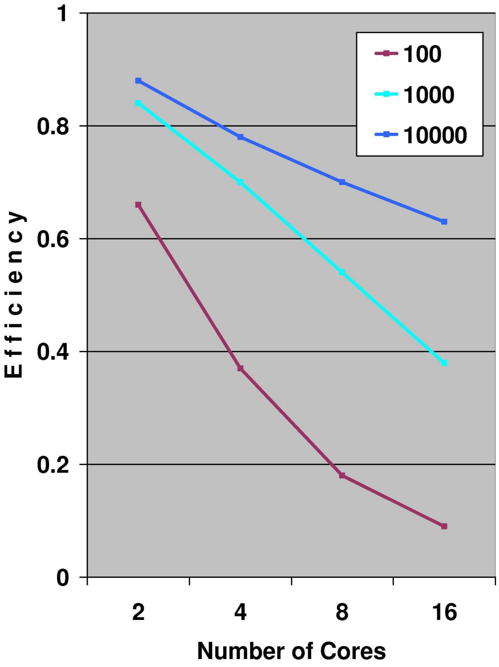


Figure 7: Graphs of Average-Node-Efficiency as function of number of cores.

success rate in multicore ad hoc networks. Figure 11 plots the routing delivery success rate as a function of pause time for 1, 2, 4, 8 and 16 cores per node. The delivery success rate decreases as the number of cores per node decreases since less cores means higher traffic load. High traffic load increases the time-to-destination a message travels and thus increases the possibility that the message will not reach its destination due to nodes mobility. For example, for the case of pause time of 0 (continuous mobility) the delivery success rate increases from 0.75 to 0.85 when the number of cores per node increases from 1 to 16 respectively. Moreover, increase in the number of cores per node may achieves the desired delivery success rate although the mobility rate increases. For example, for the case of pause time of 10 seconds and single core nodes, the success delivery rate is 0.85. Increasing the mobility rate, by decreasing the pause time to 5 seconds, and using 8 cores per node, instead of one, yield success delivery rate of 0.89. The conclusion arises from these results is that multicore nodes decrease the the number of messages that do not reach their destination due to mobility and thus increase the dependability of ad hoc networks.

6 Conclusions

We have studied the effects of introducing multicore nodes on the routing performance of wireless ad hoc networks. First, we formally defined a multicore-based network model and three metrics as evaluation tools: Time-to-Destination, Routing Speedup, and Node Efficiency. Next we evaluated the routing speedup over a wide range of network configurations through intensive routing simulations.

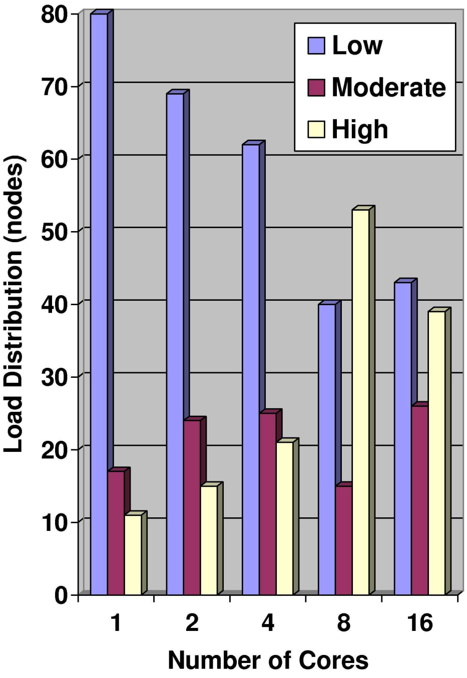


Figure 8: Histogram of routings load distribution. The nodes are grouped into three load levels: low (nodes that were busy between 0% and 40% of the time), moderate (40%-80%) and high (80%-100%).

We discovered that a network with more cores in each node has a larger routing speedup and handles more routings efficiently. Multicore networks decreased traffic congestion, balanced the load among the nodes and improve the dependability of mobile networks. However, there are side effects that must still be resolved. Adding more cores also decreases node efficiency; not all the node's cores are used all the time. This phenomenon calls for the development of better power-saving protocols and energy management strategies in order to conserve battery power.

References

[1] S. Basgani, M. Conti, S. Giordano, and I. Stojmenoviv, *Mobile ad hoc networking*, IEEE Press, 2004.

[2] J. Broch, D. Maltz, D. Johnson, Y. Hu, and J. Jetcheva, *A performance comparison of multi-hop wireless ad hoc network routing rotocols*, ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom '98), August 1998.

[3] R. Chary and J. Gilbert, *Intel pca: Advanced power management for the mobile platform*, Intel Developer UPDATE Magazine, August 2002.

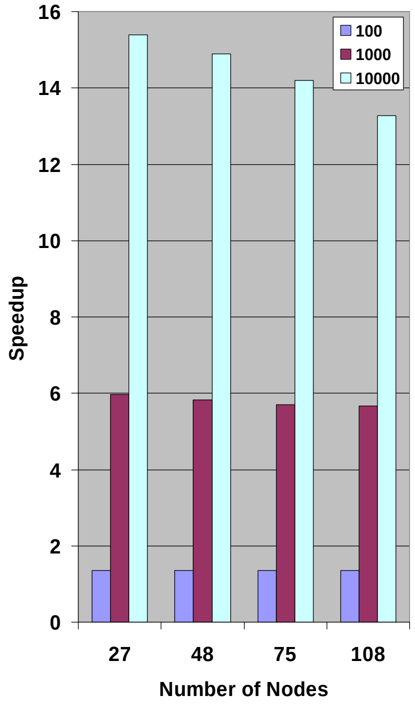


Figure 9: Graphs of Average-Routing-Speedup as function of number of nodes for 16-cores nodes and for 100, 1K and 10K routings requests.

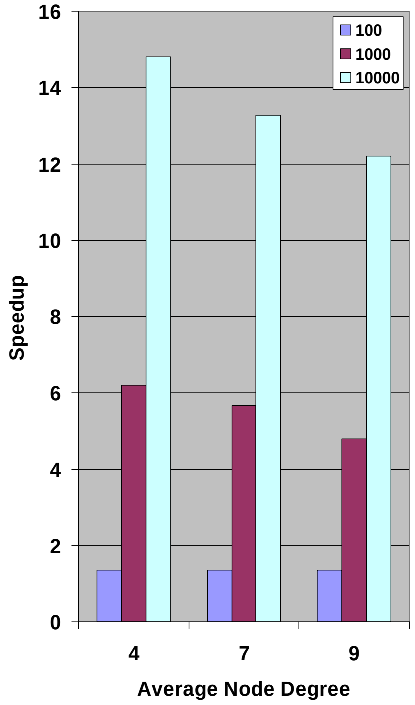


Figure 10: Graphs of Average-Routing-Speedup as function of average-node-degree for network of 108(16) nodes(cores) and for 100, 1K and 10K routings requests.

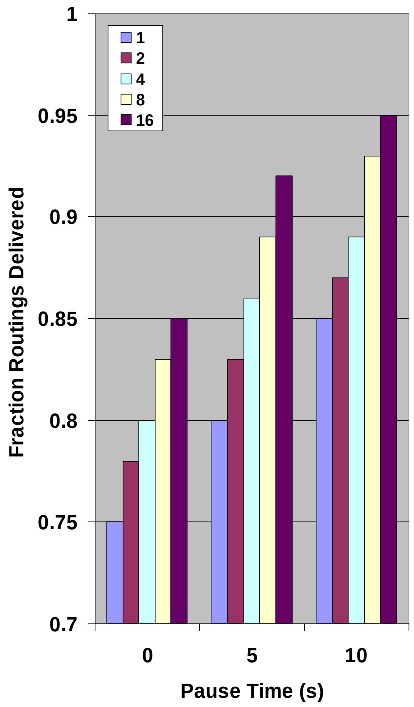


Figure 11: Routing delivery success rate as a function of pause time for 1, 2, 4, 8 and 16 cores.

[4] L. M. Feeney, *Energy-efficient communication in ad-hoc wireless networks*, In Mobile Ad Hoc Networking Editors S. Basagni, M. Conti, S. Giordano and I. Stojmenovic, pages 301–327, IEEE-Wiley, 2004.

[5] D. Geer, *Chip makers turn to multicore processors*, IEEE Computer, May 2005.

[6] P. J. M. Havinga and G. J. M. Smit, *Design techniques for low-power systems*, Journal of Systems Architecture, 46(1):1–21, 2000.

[7] K. Hwang and Z. Xu, *Scalability and speedup analysis*, Scalable Parallel Computing, McGraw Hill, pages 134–148, 1998.

[8] V. Kumar, A. Grama, A. Gupta, and G. Karipis, *Performance and scalability of parallel systems*, Introduction to Parallel Computing, Benjamin/Cummings, pages 117–146, 2004.

[9] A. Marowka, *Routing speedup and node efficiency in multicore ad hoc networks*, 4th International Conference on Performance Modeling and Evaluation of Heterogeneous Networks (HET-NETs'06), pages P02/1–P02/11, September 2006.

[10] A. Marowka, *Routing speedup in multicore-based ad hoc networks*, 6th International Symposium on Parallel and Distributed Computing (ISPDC'07), July 2007.

[11] C. Siva, R. Murthy, and B. S. Manoj, *Ad hoc networks, architecture and protocols*, Prentice Hall, 2004.

[12] I. Stojmenovic, *Position-based routing in ad hoc networks*, IEEE Communications Magazine, July 2002.

[13] H. Takagi and L. Kleinrock, *Optimal transmission ranges for randomly distributed packet radio terminals*, IEEE Transactions on Communications, 32(3):246–257, 1984.

[14] M. Viennot, T. Clausen, and P. Jacquet, *Analyzing control traffic overhead versus mobility and data traffic activity in mobile ad-hoc network protocols*, ACM Wireless Networks journal, 10(4), July 2004.

[15] U. Weiser, *Future direction in microprocessors*, In: Intel ŚTechnology in MotionŠ conference, May 30 2005, Ramat-Gan, Israel.

[16] E. Biglieri, R. Calderbank, A. Constantinides ,A. Goldsmith, A. Paulraj, H. Poor, *MIMO Wireless Communications*, Cambridge University Press, 2007

[17] C. Oestges, B. Clerckx, *MIMO Wireless Communications : From Real-world Propagation to Space-time Code Design*, Academic, 2007.

[18] D. Gesbert, M. Kountouris, R. Heath, C. Chae, T. Salzer, *Shifting the MIMO Paradigm: From Single User to Multiuser Communications*, IEEE Signal Processing Magazine, vol. 24, no. 5, pp. 36-46, Oct., 2007.

[19] A. Bahai, B. Saltzberg, M. Ergen, *Multi Carrier Digital Communications: Theory and Applications of OFDM*, Springer, 2004.

[20] K. Sundaresan, R. Sivakumar, M. Ingram, T. Chang, *A fair medium access control protocol for ad-hoc networks with MIMO links*, In IEEE Infocom, 2004.

[21] K. Sundaresan and R. Sivakumar, *Routing in ad-hoc networks with MIMO links*, In IEEE ICNP, 2005.

[22] A Toledo and X Wang, *TCP performance over wireless MIMO channels with ARQ and packet combining*, IEEE Transactions on Mobile Computing, 5(3):208-223, 2006.

Similarity Measures for Relational Databases

Melita Hajdinjak
University of Ljubljana, Faculty of Electrical Engineering, Tržaška 25, 1000 Ljubljana, Slovenia
melita.hajdinjak@fe.uni-lj.si and http://matematika.fe.uni-lj.si/

Andrej Bauer
University of Ljubljana, Faculty of Mathematics and Physics, Jadranska 21, 1000 Ljubljana, Slovenia
andrej.bauer@fmf.uni-lj.si and http://andrej.com/

Keywords: relational algebra, related answers, similarity search

Received: November 5, 2008

We enrich sets with an integrated notion of similarity, measured in a (complete) lattice, special cases of which are reflexive sets *and* bounded metric spaces*. Relations and basic relational operations of traditional relational algebra are interpreted in such richer structured environments. An canonical similarity measure between relations is introduced. In the special case of reflexive sets it is just the well known* Egli-Milner ordering *while in the case of bounded metric spaces it is the* Hausdorff metric*. Some examples of how to perform approximate searches (e.g.,* similarity search *and* relaxed answers*) are given.*

Povzetek: Z željo po iskanju bližnjih informacij in relaksiranih odgovorov množice obogatimo z merami podobnosti. Interpretiramo relacije in operacije relacijske algebre.

1 Introduction

The *relational algebra* (4; 15), a relational data model with five basic operations on relations, i.e., *Cartesian product* $\times$, *projection* π, *selection* σ, *union* $\cup$, and *set difference* $-$, and several additional operations such as *θ-join* or *intersection*, has three main advantages over non-relational data models (13):

- From the point of view of *usability*, the model has a simple interpretation in terms of real-world concepts, i.e., the essential data structure of the model is a relation, which can be visualized in a tabular format.
- From the point of view of *applicability*, the model is flexible and general, and can be easily adapted to many applications.
- From the point of view of *formalism*, the model is elegant enough to support extensive research and analysis.

Hence, the relational data models have gained acceptance from a broad range of users, they have gained popularity and credibility in a variety of application areas, and they facilitate better theoretical research in many fundamental issues arising from database query languages and dependency theory.

However, there are several applications that have evolved beyond the capabilities of traditional relational data models, such as applications that require databases to *cooperate* with the user by suggesting answers which may be helpful but were not explicitly asked for. The *cooperative-behaviour* or *cooperative-answering* techniques (5) may be differentiated into the following categories:

i.) consideration of specific information about a user's state of mind,

ii.) evaluation of presuppositions in a query,

iii.) detection and correction of misconceptions in a query,

iv.) formulation of intensional answers,

v.) generalization of queries and of responses.

The cooperative behaviour plays an important part, for instance, in information-providing dialogue systems (7), where the most vital cooperative-answering technique leading to user satisfaction is *generalization of queries and of responses* as shown by Hajdinjak and Mihelič (8). Generalization of queries and of responses, the aim of which is to capture possibly relevant information, is often achieved by *query relaxation* (6).

Another kind of applications not suitable for the traditional relational data models are applications which require the database to be enhanced with a notion of *similarity* that allows one to perform *approximate* searches (9). The goal in these applications is often one of the following:

i.) Find objects whose feature values fall within a given range or where the distance from some query object falls into a certain range (range queries).

ii.) Find objects whose features have values similar to those of a given query object or set of query objects (nearest neighbour queries and approximate nearest neighbour queries).

iii.) Find pairs of objects from the same set or different sets which are sufficiently similar to each other (closest pairs queries).

Examples of such approximate-matching or similarity-search applications are databases storing images, fingerprints, audio clips or time sequences, text databases with typographical or spelling errors, text databases where we look for documents that are similar to a given document, and computational-biology applications where we want to find a DNA or a protein sequence in a database allowing some errors due to typical variations.

Persuaded that many applications will never reach the limitations of the widespread relational data model this article focuses on traditional relational algebra equipped with extra features that allow query relaxation and similarity searches. Although a large body of work has addressed how to extend the relational data model to incorporate cooperativity, neighbouring information, and/or orderings (2; 3; 10; 11; 13), neither of them have succeeded to fit into the representational and operational uniformity of traditional relational algebra or even to reach a certain degree of generality.

Therefore, we are going to talk about domains, similarity, approximate answers, and nearness of data in a highly systematic and comprehensive way, which will lead us towards an usable, applicable, and a formaly strong generalization of the relational data model.

2 Sets with similarity

Most applications and proposed solutions of non-exact matches and similarity search, which are not covered by traditional relational algebra, have some common characteristics – there is a universe of objects and a non-negative distance or distance-like function defined among them. The distance function measures how close are the non-exact matches to the exact specifications that were given by the user willing to accept approximate answers.

Instead of restricting only to distance metrics, we consider more general *similarity measures* that satisfy the only condition of being *reflexive*, i.e., every object is most similar to itself. Hence, rather than focusing on (ordinary sets) or metric spaces, we will consider more general *sets with similarity*, where a measure of similarity assigns to a pair of objects a similarity value, which tells us how similar they are. Note, we speak of *similarity* instead of distance – if a point x moves toward a point y, the distance between x and y gets smaller, but their similarity gets larger.

For the domain of possible similarity values we choose *complete lattices*, i.e., partially ordered sets in which all subsets have both a least upper bound (*join*) and a greatest lower bound (*meet*).

Definition 1. A *set with similarity* is an ordered triple

$$\underline{A} = (A, L_A, \rho_A),$$

where A is the *underlying set*, L_A is a complete lattice with the least element 0_A and the greatest element 1_A, and

$$\rho_A : A \times A \to L_A$$

is a *measure of similarity* in A satisfying the *reflexivity* condition

$$\rho_A(x, x) = 1_A$$

for all $x \in A$.

In the trivial case, if we take the complete lattice L_2 of boolean values $\{0, 1\}$ equipped with minimum and maximum as the operations meet and join, respectively, ordered with relation $\leq$, and define the similarity by

$$\rho(x, y) = \begin{cases} 1, & \text{if } x = y \\ 0, & \text{if } x \neq y, \end{cases}$$

the resulting set with similarity gains no additional structure. That is, it is equivalent to the underlying set.

There are many non-trivial similarities and, consequently, non-trivial sets with similarity, such as *reflexive sets* and *bounded metric spaces*.

Definition 2. A *reflexive set* is an ordered pair $(A, \lhd_A)$, where A is the underlying set, and

$$\lhd_A : A \times A \to L_2$$

is a reflexive relation in A. Habitually, instead of $\lhd_A(x, y) = 1$ we write $x \lhd_A y$.

Since $x \lhd_A x$ for all $x \in A$, the reflexive relation $\lhd_A$ can be understood as a special case of a measure of similarity, thus the reflexive set $(A, \lhd_A)$ can be transformed to the set with similarity $(A, L_2, \lhd_A)$ and embedded into sets with similarity.

Definition 3. A *bounded metric space* is an ordered pair (A, d_A), where A is the underlying set, and

$$d_A : A \times A \to [0, \infty]$$

is a distance function, which satisfies the conditions of non-negativity, symmetry, and triangle inequality:

a.) $d(x, y) \geq 0$ (non-negativity)
and $d(x, y) = 0 \iff x = y$,

b.) $d(x, y) = d(y, x)$, (symmetry)

c.) $d(x, y) \leq d(x, y) + d(y, z)$. (triangle inequality)

In an arbitrary metric space the distance is measured by values strictly smaller than ∞. By allowing ∞ as a distance we have in effect restricted to bounded metric spaces. While the modest generalization of allowing ∞ as a similarity value does not pose a serious restriction (databases are usually built from finite, and therefore bounded sets of data), it makes the set $[0, \infty]$, when ordered by the usual $\geq$ relation, a complete lattice $L_{[0,\infty]}$ as required in sets with similarity. Meet and join are computed as supremum and infimum, respectively. Note that we turned $[0, \infty]$ *upside down* so that the least element is ∞ and the greatest is 0.

Hence the metric d_A is again a special case of a measure of similarity because $d_A(x, x) = 0$, which is the greatest element of the complete lattice $L_{[0,\infty]}$. Thus the bounded metric space (A, d_A) can be transformed to the set with similarity $(A, L_{[0,\infty]}, d_A)$ and embedded into sets with similarities.

3 Tables, relations, and basic relational operations

A relational database is composed of several relations in the form of two-dimensional tables of rows and columns containing related tuples. The rows (tuples) are called *records* and the columns (fields in the record) are called *attributes*. Each attribute has a data type that defines the set of possible values. Thus a relation is a subset of a Cartesian product of sets (value domains).

3.1 Cartesian products and subsets

In order to use sets with similarity instead of (ordinary) sets we need a suitable notion of relation between sets with similarity. Hence we first need to know how to interpret Cartesian products and subsets of sets with similarity in a natural and effective way.

Definition 4. The *Cartesian product* of sets with similarity $\underline{A} = (A, L_A, \rho_A)$ and $\underline{B} = (B, L_B, \rho_B)$ is the set with similarity

$$\underline{A} \times \underline{B} = (A \times B, L_A \times L_B, \rho_{A\times B}),$$

where $A \times B$ is the Cartesian product of sets, $L_A \times L_B$ is the product of complete lattices, and the measure of similarity $\rho_{A\times B}$ is given by

$$\rho_{A\times B}((x_1, y_1), (x_2, y_2)) = (\rho_A(x_1, x_2), \rho_B(y_1, y_2)).$$

The corresponding *canonical projections* are $(\pi_1, p_1) : \underline{A} \times \underline{B} \to \underline{A}$ and $(\pi_2, p_2) : \underline{A} \times \underline{B} \to \underline{B}$, where π_1 and π_2 are projections of sets, but p_1 and p_2 are projections of complete lattices.

This interpretation of Cartesian products of sets with similarity is sound since a product of complete lattices is a complete lattice (14) and $\rho_{A\times B}$ satisfies the condition of being a measure of similarity:

$$\begin{aligned} \rho_{A\times B}((x, y), (x, y)) &= (\rho_A(x, x), \rho_B(y, y)) \\ &= (1_A, 1_B) \\ &= 1_{A\times B}, \end{aligned}$$

where 1_A is the greatest element of L_A, 1_B is the greatest element of L_B, and $1_{A\times B}$ is the greatest element of the complete lattice $L_A \times L_B$.

Further, we have decided to consider only those subobjects or substructures $\underline{I}$ of the set with similarity $\underline{A} = (A, L_A, \rho_A)$ whose similarity measure is induced by the structure of $\underline{A}$. That is, the underlying set is a subset of A but the measure of similarity and the corresponding lattice are inherited from $\underline{(A)}$. Even though the domain of the measure of similarity has changed from $A \times A$ to $I \times I$, we will keep the notation ρ_A and write $\underline{I} = (I, L_A, \rho_A)$.

Definition 5. (i.e., the Egli-Milner ordering and the Hausdorff metric) A *subset* of the set with similarity $\underline{A} = (A, L_A, \rho_A)$ is a set with similarity $\underline{I} = (I, L_A, \rho_A)$, where $I \subseteq A$. Subsets of sets with similarity will also be called *induced subobjects*.

3.2 Relations and basic relational operations

The family of subsets of $\underline{A}$, denoted by $IndSub(\underline{A})$, is essentially just the power set $\mathcal{P}(A)$.

Theorem 1. The induced subobjects of a set with similarity $\underline{A} = (A, L_A, \rho_A)$ form a complete boolean algebra equivalent to $\mathcal{P}(A)$, in which all the basic relational operations can be properly interpreted.

The formal proof is given in (7). However, the Boolean lattice $IndSub(\underline{A})$ is ordered with the usual subset relation, where the least element is the empty subobject $\underline{\emptyset} = (\emptyset, L_A, \rho_A)$ and the greatest element is $\underline{A}$. Hence selection, union, and difference are calculated as usual $(A_1, A_2 \subseteq A)$:

$$\begin{aligned} &\sigma_F(A_1, L_A, \rho_A) = (\sigma_F(A_1), L_A, \rho_A), \\ &(A_1, L_A, \rho_A) \cup (A_2, L_A, \rho_A) = (A_1 \cup A_2, L_A, \rho_A), \\ &(A_1, L_A, \rho_A) - (A_2, L_A, \rho_A) = (A_1 - A_2, L_A, \rho_A). \end{aligned}$$

Moreover, Cartesian products, projections, selections, unions, and differences of induced subobjects satisfy all the abstract properties that are axiomatized by relational calculus (15; 16).

A relation between two objects of the category of similarities, namely $\underline{A} = (A, L_A, \rho_A)$ and $\underline{B} = (B, L_B, \rho_B)$, is now determined by a subset $R \subseteq A \times B$, which induces a subobject $(R, L_A \times L_B, \rho_{A\times B})$ of the Cartesian product $\underline{A} \times \underline{B}$. Hence tables and answers to queries are modeled as induced subobjects.

4 Similarity of relations

Sets with similarity enjoy additional constructions, which do not exist at the level of underlying sets. For instance, a suitable notion of similarity $\lhd$ between induced subobjects can be defined.

In the case of the reflexive set $(A, \lhd_A)$, which is equipped with a reflexive relation $\lhd_A$ establishing connections between certain elements of A, we propose to take the naturally integrated *Egli-Milner ordering*. Its importance in data models was also recognized by Buneman, Jung, and Ohori (1).

Definition 6. Let $\underline{A_1} = (A_1, \lhd_A)$ and $\underline{A_2} = (A_2, \lhd_A)$ be two induced subobjects of the reflexive set $\underline{A} = (A, \lhd_A)$. The *Egli-Milner ordering* is given as follows:

$$\begin{aligned}\underline{A_1} \lhd \underline{A_2} \quad \Longleftrightarrow \quad & (\forall x \in A_1\ \exists y \in A_2 : x \lhd_A y) \text{ and} \\ & (\forall y \in A_2\ \exists x \in A_1 : x \lhd_A y).\end{aligned}$$

On the other hand, in the case of the bounded metric space (A, d_A), which is equipped with a distance function d_A, we propose to take the well-known *Hausdorff metric*. It has several applications, for instance, in fractal geometry, in numerical mathematics, and in pattern recognition.

Definition 7. Let $\underline{A_1} = (A_1, d_A)$ and $\underline{A_2} = (A_2, d_A)$ be two induced subobjects of the bounded metric space $\underline{A} = (A, d_A)$. The *Hausdorff metric* is defined as follows:

$$\begin{aligned}d(\underline{A_1}, \underline{A_2}) = \quad & \max\{\sup_{x \in A_1} \inf_{y \in A_2}\{d_A(x,y)\}, \\ & \sup_{y \in A_2} \inf_{x \in A_1}\{d_A(x,y)\}\}.\end{aligned}$$

Note, in the trivial example of ordinary sets (without similarity), it is straightforward that two induced subobjects (ordinary subsets) of a given set can only be similar if they are equal, i.e., if they share all the elements.

The following theorem generalizes the above-defined, special notions of similarity between induced subobjects and proposes a similarity measure in $IndSub(\underline{A})$.

Theorem 2. Let $\underline{A_1} = (A_1, L_A, \rho_A)$ and $\underline{A_2} = (A_2, L_A, \rho_A)$ be two induced subobjects of the set with similarity $\underline{A} = (A, L_A, \rho_A)$. The Egli-Milner ordering from reflexive sets and the Hausdorff metric from bounded metric spaces can be generalized to sets with similarity as follows:

$$\begin{aligned}&\rho(\underline{A_1}, \underline{A_2}) = \\ &\quad = (\bigwedge_{x \in A_1} \bigvee_{y \in A_2} \rho_A(x,y)) \wedge (\bigwedge_{y \in A_2} \bigvee_{x \in A_1} \rho_A(x,y)),\end{aligned}$$

where all the meets and joins are computed in the complete lattice L_A.

The proof of this theorem and some highly-desirable properties of the generalized similarity measure ρ, such as

i.) the empty induced subobject is completely dissimilar to any other induced subobject and

ii.) every induced subobject is most similar to itself,

are given in (7). Note, if infinite sets are allowed, theorem 2 requires from the sets with similarity to be equipped with complete (!) lattices (see definition 1).

5 Approximate searches

As already explained, tables and answers to queries are modeled as induced subobjects. Each column is equipped with its own measure of similarity (integrated within sets with similarity), and from all these we build the measure of similarity for the whole table (see definition 4), which can be used to make comparisons between pairs of rows, find rows whose distance from some origin falls into a certain range, find nearest neighboring rows or closest pairs of rows. Hence we can perform all types of similarity search.

Moreover, the measure of similarity ρ (see theorem 2) between induced subobjects could serve to measure the nearness or exchangeability of the exact and the *relaxed answer* to a query, for comparing instances of a time-dependent table, or track changes made to a table. While in the special case of reflexive sets, the Egli-Milner relation tells us only when a table or an answer is interchangeable with another one, in the special case of bounded metric spaces, the Hausdorff metric allows a more fine-grained control of relaxation.

Example 1. Let tables 1 and 2 contain data about the users of an Internet forum at two consecutive days (day 1 and day 2).

NAME	NICK	CITY
Marko	obi	Maribor
Maja	maja	Ljubljana
Darko Koren	dare	Koper

Table 1: Table of users at day 1.

NAME	NICK	CITY
Hujs Marko	marko	Pragersko
Maja	maja	Ljubljana
Darko Koren	dare	Koper
Meta Novak	metan	Maribor
Jernej	jernej	Kranj

Table 2: Table of users at day 2.

The relational schema of tables 1 and 2 is

$$[\mathcal{P} : \text{NAME}, \mathcal{N} : \text{NICK}, \mathcal{C} : \text{CITY}],$$

where the domains corresponding to the atributes $\mathcal{P}$, $\mathcal{N}$, and $\mathcal{C}$ are the following sets with similarity:

$$\begin{aligned}\text{NAME} &= (\text{Strings}, L_{\text{Strings}}, \rho_{\text{Strings}}),\\ \text{NICK} &= (\text{Strings}, L_{\text{Strings}}, \rho_{\text{Strings}}),\\ \text{CITY} &= (\text{Cities}, L_{[0,\infty]}, d_{Cities}).\end{aligned}$$

Here, Strings is the set of all strings of maximum length 30 and Cities is the set of all possible cities, towns, and villages in the world. The complete lattice L_{Strings} is the linearly ordered set $\{0, 1, \ldots, 30\}$ with the order relation $\geq$ and $L_{[0,\infty]}$ is the complete lattice from definition 3. The measures of similarity are defined as follows:

- The measure of similarity ρ_{Strings} is the Damerau-Levenshtein distance (12), given as the minimum number of operations needed to transform one string into the other, where an operation is an insertion, deletion, or substitution of a single character or a transposition of two characters. Since the length of the strings is bounded by 30, the Damerau-Levenshtein distance is at least 0 (the greatest element in the lattice L_{Strings}) and at most 30 (the least element in the lattice L_{Strings}).

- The measure of similarity $d_{Slovenia}$ calculates the similarity of two cities as their air distance given as the Euclidean distance (in kilometres) between the Gauss-Krüger coordinates of the city centers (we have used the tool from http://www2.arnes.si/).

The measure of similarity corresponding to the Cartesian product

$$\underline{\text{USERS}} = \text{NAME} \times \text{NICK} \times \text{CITY}$$

of given sets with similarity is defined in accordance with definition 4:

$$\begin{aligned}&\rho_{\text{Users}}((p1, n1, c1), (p2, n2, c2)) =\\ &\quad = (\rho_{\text{Strings}}(p1, p2), \rho_{\text{Strings}}(n1, n2), d_{Cities}(c1, c2)),\end{aligned}$$

where $(p1, n1, c1)$ and $(p1, n1, c1)$ are two rows of the given table instance, i.e., elements of the Cartesian product of sets:

$$\text{Users} = \text{Strings} \times \text{Strings} \times \text{Cities}.$$

For instance, the similarity between the first rows of the two tables 1 and 2 is equal to $(5, 5, 18.3)$, but the similarity of the last two rows of table 2 is equal to $(9, 5, 189.9)$. Clearly, $(9, 5, 189.9) \geq (5, 5, 18.3)$, which means that the first pair of rows is more similar than the second one, i.e., the similarity value of the first pair is higher in the complete lattice

$$L_{\text{Users}} = L_{\text{Strings}} \times L_{\text{Strings}} \times L_{[0,\infty]}$$

than the similarity value of the second pair. Note, since L_{Users} is not linearly ordered, there are also uncomparable elements in the lattice.

Furthermore, the measure of similarity ρ between induced subobjects $\underline{A}$ and $\underline{B}$ of the Cartesian product $\underline{\text{USERS}}$ can be defined in accordance with theorem 2:

$$\begin{aligned}&\rho(\underline{A}, \underline{B}) =\\ &\quad = \Big(\bigwedge_{x\in A} \bigvee_{y\in B} \rho_{\text{Users}}(x, y) \Big) \wedge \Big(\bigwedge_{y\in B} \bigvee_{x\in A} \rho_{\text{Users}}(x, y) \Big),\end{aligned}$$

where all the meets and joins are computed in the complete lattice L_{Users}. Hence the similarity between the given table instances is equal to $(9, 4, 106.9)$. The similarity would certainly decrease if one of the tables would be increased in size by users living far from Slovenia and/or have or use much longer names or nicks. Clearly, if the similarity measures integrated within the sets with similarity were changed, the similarity value between the two table instances would also change and possibly have a different interpretation. Hence the usefullness of the calculated similarity values depends highly on the definitions of the basic similarity measures.

Example 2. Let table 3 contain a portion of public-transport bus routes in Ljubljana (Slovenia).

The relational schema of table 3 corresponding to relation BUSES is

$$\begin{aligned}&[\mathcal{R} : \text{ROUTE}, \mathcal{D} : \text{DEPATURE}, \mathcal{DT} : \text{DTIME},\\ &\mathcal{A} : \text{ARRIVAL}, \mathcal{AT} : \text{ATIME}],\end{aligned}$$

where the domains corresponding to the atributes $\mathcal{R}$, $\mathcal{D}$, $\mathcal{DT}$, $\mathcal{A}$, and $\mathcal{AT}$ are the following sets with similarity:

$$\begin{aligned}\text{ROUTE} &= (\text{Buses}, L_2, \sigma_{\text{Buses}}),\\ \text{DEPARTURE} &= (\text{Stops}, L_{\text{Stops}}, \sigma_{\text{Stops}}),\\ \text{DTIME} &= (\text{Time}, L_{\text{Time}}, \sigma_{\text{Time}}),\\ \text{ARRIVAL} &= (\text{Stops}, L_{\text{Stops}}, \sigma_{\text{Stops}}),\\ \text{ATIME} &= (\text{Time}, L_{\text{Time}}, \sigma_{\text{Time}}).\end{aligned}$$

Here, Buses and Stops are the sets of bus routes and bus stops in Ljubljana, respectively. Time is the set of all possible times of the form HH:MM, where HH denotes hours written as $00, 01, \ldots, 23$ and MM denotes minutes written as $00, 01, \ldots, 59$. The complete lattice L_{Stops} is the linearly ordered set $\{0, 1, \ldots, M, \infty\}$ of non-negative integers (smaller than the number of all bus stops M) and the infinity value ∞ with the order relation $\geq$. The complete lattice L_{Time} is the linearly ordered set Time with 23:59 being the least element and 00:00 being the greatest element. Moreover, lattice L_2 is the lattice of boolean values from definition 2.

The measures of similarity are defined as follows:

- The measure of similarity σ_{Buses} says 1 if the given bus routes are equal and 0 if they are not.

- The measure of similarity σ_{Stops} calculates the similarity of the given bus stops as the minimum number of bus stops needed to pass by bus to come from the first bus stop to the second. If it is impossible to do this, the similarity value given is equal to ∞.

ROUTE	DEPARTURE	DTIME	ARRIVAL	ATIME	Exchangeability
9 (Štep. naselje-Trnovo)	145 (Emona)	9:58	026 (Konzorcij)	10:25	(1,0,23:58,4,00:25)
5 (Štep. naselje-Podutik)	145 (Emona)	10:10	025 (Hotel Lev)	10:26	(1,0,00:10,6,00:26)
13 (Sostro-Bežigrad)	145 (Emona)	10:11	059 (Bavarski dvor)	10:24	(1,0,00:11,5,00:24)
9 (Štep. naselje-Trnovo)	145 (Emona)	10:14	026 (Konzorcij)	10:41	(1,0,00:14,4,00:41)
6 (Črnuče-Dolgi most)	058 (Bavarski dvor)	10:29	034 (Hajdrihova)	10:32	(1,6,00:29,0,00:32)
1 (Vižmarje-Mestni log)	024 (Kolizej)	10:36	034 (Hajdrihova)	10:49	(1,7,00:36,0,00:49)
6 (Črnuče-Dolgi most)	026 (Konzorcij)	10:41	034 (Hajdrihova)	10:46	(1,8,00:41,0,00:46)
1 (Vižmarje-Mestni log)	026 (Konzorcij)	10:44	034 (Hajdrihova)	10:49	(1,8,00:44,0,00:49)
6 (Črnuče-Dolgi most)	026 (Konzorcij)	10:47	034 (Hajdrihova)	10:54	(1,8,00:47,0,00:54)

Table 3: Table of public-transport bus routes in Ljubljana. The last column contains data about the exchangeability of each row with an exact answer to the query given within example 2.

- The measure of similarity σ_{Time} calculates the similarity of two time moments as their difference (second minus first) in form of HH:MM.

Now consider the query *"It is 10 o'clock and I am at the Emona bus stop. Are there any buses to Hajdrihova? I would like to arrive as soon as possible."*, written in the language of relational algebra (4):

$$\sigma_{\mathcal{D}=\text{Emona}\wedge\mathcal{DT}=10{:}00\wedge\mathcal{A}=\text{Hajdrihova}\wedge\mathcal{AT}=10{:}00}(\text{BUSES}).$$

The last column of table 3 contains the calculated similarity or exchangeability values of the exact and the possibly relaxed answer (row). Notice that in table 3 there are no buses satisfying all the conditions given by the user but there are several buses that could be interesting for the user, such as the buses described by the second or the third row. These have a different destination, which is not a real handicap since the user could take another bus to come to Hajdrihova, i.e., bus routes 1 and 6, respectively. However, if we would like to suggest a suitable bus or a sequence of buses, we just need to calculate a θ-join of the given relation with the requirement that the arrival bus stop of the first and the departure bus stop of the second bus are (basically) the same, i.e., there are no bus stops between them, maybe one only needs to cross the street.

6 Conclusion

We have defined the mathematical structure of sets with similarity that allows us to treat the features of richly-structured data, such as order, distance, and similarity, in a theoretically sound and uniform way. The proposed measures of similarity allow us to perform all types of similarity search.

In addition, we now briefly discuss possible implementations of the resulting databases enriched with measures of similarity. Clearly, the user should be able to query approximate or cooperative data from databases without being concerned about the internal structure of data. Hence some default similarity measures should be integrated within the database. But still, the user should have the opportunity to modify the default notions of similarity if he/she is willing to do this.

However, the question that arises is how to store the defined similarity measures. When the size of the data set A is small, the evident way to store a similarity measure $\rho_A : A \times A \to L_A$ is in tabular form, i.e., as a relation

$$\rho_A \subseteq A \times A \times L_A.$$

This kind of representation quickly becomes inefficient since it requires space quadratic in the size of A. Fortunately, in most cases the similarity measure can be easily calculated so that there is no need for storing it.

There are two typical examples of similarity measures that can be computed rather than stored. First, distance-like similarities are computed from auxiliary data, such as geographic location, duration, and various other features that only require a minimal amount of additional storage.

Second, reflexive relations are often defined in terms of deduction rules, e.g., it may be known that the relation is symmetric or transitive. In such cases we only store the base cases in a database, and deduce the rest from them. This is precisely the idea behind deductive database languages, such as *Datalog*.

References

[1] P. Buneman, P. Jung, A. Ohori (1991) Using Powerdomains to Generalize Relational Databases, *Theoretical Computer Science* 9/1, Elsevier, pp. 23–55.

[2] W. W. Chu, Q. Chen (1994) A Structured Approach for Cooperative Query Answering, *IEEE Transactions on Knowledge and Data Engineering* 6/5, IEEE Computer Society, pp. 738–749.

[3] W. W. Chu, H. Jung, K. Chiang, M. Minock, G. Chow, C. Larson (1996) CoBase: A Scalable and Extensible Cooperative Information System, *Journal of Intelligent Information Systems* 6/2-3, Springer US, pp. 223–259.

[4] E. F. Codd (1970) A Relational Model of Data for Large Shared Data Banks, *Communications of the ACM* 13/6, Association for Computing Machinery, pp. 377–387.

[5] T. Gaasterland, P. Godfrey, J. Minker (1992) An Overview of Cooperative Answering, *Journal of Intelligent Information Systems* 1/2, Springer US, pp. 123–157.

[6] T. Gaasterland, P. Godfrey, J. Minker (1992) Relaxation as a Platform of Cooperative Answering, *Journal of Intelligent Information Systems* 1/3-4, Springer US, pp. 293–321.

[7] M. Hajdinjak (2006) *Knowledge Representation and Evaluation of Cooperative Spoken Dialogue Systems*, Ph.D. thesis, Faculty of Electrical Engineering, University of Ljubljana, Ljubljana, Slovenia.

[8] M. Hajdinjak, F. Mihelič (2006) The PARADISE Evaluation Framework: Issues and Findings, *Computational Linguistics* 32/2, MIT Press, pp. 263–272.

[9] G. R. Hjaltason, H. Samet (2003) Index-Driven Similarity Search in Metric Spaces, *ACM Transactions on Database Systems* 28/4, Association for Computing Machinery, pp. 517–580.

[10] A. Motro (1988) VAGUE: A User Interface to Relational Databases that Permits Vague Queries, *ACM Transactions on Office Information Systems* 6/3, Association for Computing Machinery, pp. 187–214.

[11] A. Motro (1990) FLEX: A Tolerant and Cooperative User Interface to Databases, *IEEE Transactions on Knowledge and Data Engineering* 2/2, IEEE Computer Society, pp. 231–246.

[12] G. Navarro (2001) A guided tour to approximate string matching, *ACM Computing Surveys* 33/1, Association for Computing Machinery, pp. 31–88.

[13] W. Ng (2001) An Extension of the Relational Data Model to Incorporate Ordered Domains, *ACM Transactions on Database Systems* 26/3, Association for Computing Machinery, pp. 344–383.

[14] D. E. Rutherford (1965) *Introduction to Lattice Theory*, Oliver & Boyd, Edinburgh, London.

[15] J. D. Ullman (1988) *Principles of Database and Knowledge-Base Systems, Volume I*, Computer Science Press Inc., Rockville, Maryland.

[16] J. D. Ullman (1989) *Principles of Database and Knowledge-Base Systems, Volume II: The New Technologies*, Computer Science Press, Inc., Rockville, Maryland.

Robust H$_\infty$ Control of a Doubly Fed Asynchronous Machine

Gherbi Sofiane
Department of electrical engineering, Faculty of Science of the engineer
20 August 1956 University, Skikda, Algeria
E-mail: sgherbi@gmail.com

Yahmedi Said
Department of electronic, Faculty of Science of the engineer
Badji Mokhtar University, Annaba, Algeria
E-mail: sais.yahmedi@carmail.com

Sedraoui Moussa
Department of electronics, Faculty of Science of the engineer
Constantine University, road of AIN EL BEY Constantine, Algeria
E-mail : msedraoui@gmail.com

Keywords: doubly fed asynchronous machine, robust control, H_∞ control, LMI's

Received: April 25, 2008

The doubly fed asynchronous machine is among the most used electrical machines due to its low cost, simplicity of construction and maintenance [1]. In this paper, we present a method to synthesize a robust controller of doubly fed asynchronous machine which is the main component of the wind turbine system (actually the most used model [2]), indeed: there is different challenges in the control of the wind energy systems and we have to take in a count a several parameters that perturb the system as: the wind speed variation, the consumption variation of the electricity energy and the kind of the power consumed (active or reactive) ...etc.. The method proposed is based on the H_∞ control problem with the linear matrix inequalities (LMI's) solution: Gahinet-Akparian [3], the results show the stability and the performance robustness of the system in spite of the perturbations mentioned before.

Povzetek: Opisana je metoda upravljanja motorja vetrnih turbin.

1 Introduction

From all the renewable energy electricity production systems, the wind turbine systems are the most used specially the doubly fed asynchronous machine based systems, the control of theses systems is particularly difficult because all of the uncertainties introduced such as: the wind speed variations, the electrical energy consumption variation, the system parameters variations, in this paper we focus on the robust control (H_∞ controller design method) of the doubly fed asynchronous machine which is the most used in the wind turbine system due to its low cost, simplicity of construction and maintenance [1].

This paper is organised as follow:

Section 2 presents the wind turbine system equipped with the doubly fed asynchronous machine and then the mathematical electrical equations from what the system is modelled (in the state space form) are given.

The section 3 presents the H_∞ robust controller design method with the LMI's solution used to control our system.

The section 4 presents a numerical application and results in both the frequency and time plan are presented And finally a conclusion is given in section 5.

2 System presentation and modelling

The following figure represents the wind turbine system

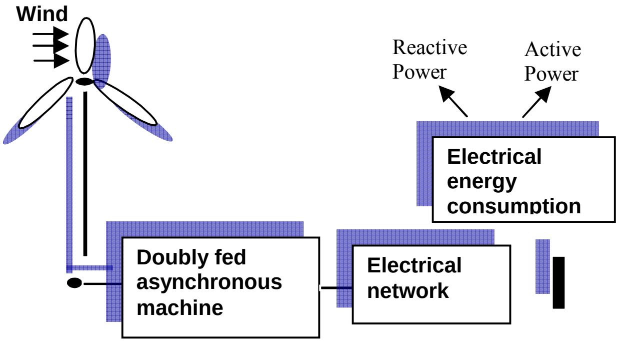

Wind
Reactive Power
Active Power
Electrical energy consumption
Doubly fed asynchronous machine
Electrical network

Figure 1: The Wind turbine system

The system use the wind power to drag the double fed asynchronous machine who acts as a generator, the output power produced must have the same high quality when it enters the electrical network, i.e.: 220 volts amplitude and 60 Hz frequency and the harmonics held to a low level in spite of wind speed changes and electrical energy consumption in active or reactive power form. References [4], [5], [6] describe detailed models of wind turbines for simulations, we use the model equipped with the doubly fed induction generators (asynchronous machine) (for more details see [7]), the system electrical equations are given in (d,q) frame orientation, then the stator voltage differential equations are:

$$V_{ds} = R_s.I_{ds} + \frac{d}{dt}\Phi_{ds} - w_s \cdot \Phi_{qs} \qquad (1)$$

$$V_{qs} = R_s.I_{qs} + \frac{d}{dt}\Phi_{qs} + w_s \cdot \Phi_{ds} \qquad (2)$$

The rotor voltage differential equations are:

$$V_{dr} = R_r.I_{dr} + \frac{d}{dt}\Phi_{dr} - w_r \cdot \Phi_{qr} \qquad (3)$$

$$V_{qr} = R_r.I_{qr} + \frac{d}{dt}\Phi_{qr} + w_r \cdot \Phi_{dr} \qquad (4)$$

The stator flux vectors equations are:

$$\Phi_{ds} = L_s.I_{ds} + M.I_{dr} \qquad (5)$$

$$\Phi_{qs} = L_s.I_{qs} + M.I_{qr} \qquad (6)$$

The rotor flux vectors equations:

$$\Phi_{dr} = L_r.I_{dr} + M.I_{ds} \qquad (7)$$

$$\Phi_{qr} = L_r.I_{qr} + M.I_{qs} \qquad (8)$$

The electromagnetic couple flux equation :

$$C_{em} = p.\frac{M}{L_s}(\Phi_{ds}.I_{qr} - \Phi_{qs}.I_{dr}) \qquad (9)$$

The electromagnetic couple mecanic equation :

$$C_{em} = C_r + J\frac{d\Omega}{dt} + f.\Omega \qquad (10)$$

With:

V_{ds}, V_{qs} : Statoric voltage vector components in 'd' and 'q' axes respectively.

V_{dr}, V_{qr} : Rotoric voltage vector components in 'd' and 'q' axes respectively.

I_{ds}, I_{qs} : Statoric current vector components in 'd' and 'q' axes respectively.

I_{dr}, I_{qr} : Rotoric current vector components in 'd' and 'q' axes respectively.

Φ_{ds}, Φ_{qs} : Statoric flux vector components in 'd' and 'q' axes respectively.

Φ_{dr}, Φ_{qr} : Rotoric flux vector components in 'd' and 'q' axes respectively.

R_s, R_r : Stator and rotor resistances (of one phase) respectively.

L_s, L_r : Stator and rotor cyclic inductances respectively.

w_s, w_r : Statoric and rotoric current pulsations respectively.

M : Cyclic mutual inductance.

p : Number of pair of the machine poles.

C_r : Resistant torque.

f : Viscous rubbing coefficient.

J : Inertia moment.

2.1 State space model

In order to apply the robust controller design method, we have to put the system model in the state space from; we consider the rotoric voltage V_{dr}, V_{qr} as the inputs and the statoric voltage V_{ds}, V_{qs} as the outputs, i.e. we have to design a controller who acts on the rotoric voltages to keep the output statoric voltages at $220volts$ and $50Hz$ frequency in spite of the electric network perturbations (demand variations … etc) and the wind speed variations (see figure.2).

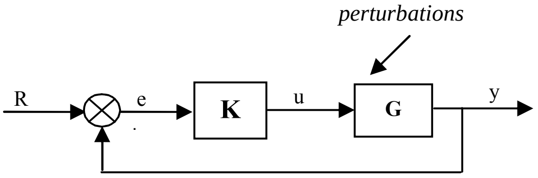


Figure 2: A Doubly fed wind turbine system control configuration

Where: u, y and e are the rotoric voltage vector (control vector), statoric output voltage vector and the error signal between the input reference and the output system respectively. K, G are the controller and the wind turbine system respectively. R: is the statoric voltage references vector and *perturbations* are the electric energy demand variations, wind speed variations …etc.

Let us consider $x = [\Phi_{dr} \quad \Phi_{qr}]^T$ as a state vector, and $u = [I_{ds} \quad I_{qs} \quad V_{ds} \quad V_{qs}]^T$ as the command vector, the stator flux vector is oriented in d axis of Parks reference frame then : $\Phi_{qs} = 0$ and I_{ds}, I_{qs} are considered constant in the steady state i.e.: $\dot{I}_{ds} = \dot{I}_{qs} = 0$.

We use the folowing doubly fed asynchronous machine parameters:

$R_s = 5\Omega$; $R_r = 1.0113\Omega$; $M = 0.1346H$
$L_s = 0.3409H$; $L_r = 0.605H$; $w_r = 146.6Hz$;
$w_s = 2\pi \cdot 50Hz$

Let $w = w_s - w_r$ and $\sigma = 1 - \frac{M^2}{L_s \cdot L_r}$.

The state space (11) can be obtained by the combining of the equations (1) to (8) as follow:

$$\begin{cases} \dot{x} = A \cdot x + B \cdot u \\ y = C \cdot x + D \cdot u \end{cases} \qquad (11)$$

Where:

$$x = [\phi_{dr} \quad \phi_{qr}]^T$$

$$u = [I_{ds} \quad I_{qs} \quad V_{dr} \quad V_{qr}]^T$$

$$y = [V_{ds} \quad V_{qs}]^T$$

And:

$$A = \begin{bmatrix} \frac{-R_r}{L_r} & w_r \\ -w_r & \frac{-R_r}{L_r} \end{bmatrix} \quad B = \begin{bmatrix} \frac{R_r.M}{L_r} & 0 & 1 & 0 \\ 0 & \frac{R_r.M}{L_r} & 0 & 1 \end{bmatrix}$$

$$C = -\frac{M}{L_r} \cdot \begin{bmatrix} \frac{R_r}{L_r} & w \\ w & -\frac{R_r}{L_r} \end{bmatrix}$$

$$D = \begin{bmatrix} R_s + \frac{M^2}{L_r^2} \cdot R_r & -\sigma \cdot L_s.W & \frac{M}{L_r} & 0 \\ \sigma \cdot L_s.W & R_s + \frac{M^2}{L_r^2} \cdot R_r & 0 & \frac{M}{L_r} \end{bmatrix}$$

3 The H_∞ controller design method

It is necessary to recall the basics of a control loop (figure.3). With G': the perturbed system.

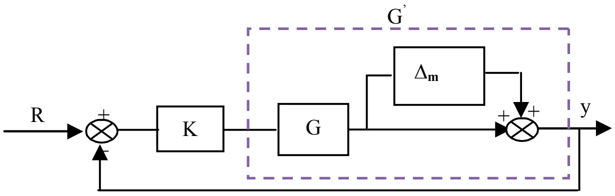


Figure 3: The control loop with the output multiplicative uncertainties

The multiplicative uncertainties at the process output which include all the perturbations that act in the system are then : $\Delta_m = (G' - G).G^{-1}$, with $G' = G(I + \Delta_m)$: is the perturbed system, figure.4 show the singular values plot at the frequency plan of Δ_m, we can see that the uncertainties are smaller at low frequencies and grow at the medium and high frequencies, this mean a strong perturbation at high frequencies (the transient phase), we also note a pick at: $\omega = 260 rad/s$, this is due to the fact that the system is highly coupled at this pulsation.

We can bound the system uncertainties by the following weighting matrix function:

$$W_t(jw) = \begin{bmatrix} \frac{0.55(0.02\,jw+1)}{(1+0.0001\,jw)} & 0 \\ 0 & \frac{0.55(0.02\,jw+1)}{(1+0.0001\,jw)} \end{bmatrix} \quad (12)$$

The figure.5 show that the singular values of $W_t(jw)$ bounds the maximum singular values of the uncertainties in the entire frequency plan.

The robust stability condition [11] is then:

$$\bar{\sigma}[T(jw) \cdot W_t(jw)] \prec 1 \quad (13)$$

Or:

$$\bar{\sigma}[T(jw)] < \bar{\sigma}[W_t(jw)]^{-1} \quad (14)$$

Where: $\bar{\sigma}$ is the maximum singular value and $T(jw)$is the nominal closed loop transfer matrix defined by:

$$T(jw) = G(jw) \cdot K(jw) \cdot [I + G(jw) \cdot K(jw)]^{-1} \quad (15)$$

The equations (13) allow us to guaranty the stability robustness, in other hand we most guaranty satisfying performances (no overshoot, time response …etc) in the closed loop (performances robustness), this can by done by the performance robustness condition [8]:

$$\bar{\sigma}[S(jw) \cdot W_p(jw)] \prec 1 \quad (16)$$

Or:

$$\bar{\sigma}[S(jw)] < \bar{\sigma}[W_p(jw)]^{-1} \quad (17)$$

Where:

$S(jw)$is the sensitivity matrix given by:

$$S(jw) = [I + G(jw) \cdot K(jw)]^{-1} \quad (18)$$

$W_P(jw)$ is a weighting matrix function designed to meet the performance specifications desired in the frequency plan, we choose the following matrix function:

$$W_p(jw) = \begin{bmatrix} \frac{(0.005\,jw+1)}{0.05\,jw} & 0 \\ 0 & \frac{(0.005\,jw+1)}{0.05\,jw} \end{bmatrix} \quad (19)$$

The figure.6 represent the singular values of $W_P(jw)$ in the frequency plan, one notice that the specifications on the performances are bigger in low frequencies (integrator frequency behaviour), and this guaranty no static error.

Then the standard problem of H∞ Control theory is then:

$$\min_{K\_stabili\sin g} \left\| \begin{matrix} T(jw) \cdot W_t(jw) \\ S(jw) \cdot W_p(jw) \end{matrix} \right\|_\infty \quad (20)$$

i.e.: to find a stabilising controller K that minimise the norm (20).

With: $\| \quad \|_\infty$ is The Hinfinity norm.

4 Application

The minimisation problem (20) is solved by using two Riccati equations [9] or with the linear matrix inequalities approach. For our system, we use the linear matrix inequalities solution (for more details see [10]). The solution (controller) can be obtained via the Matlab instruction *hinflmi* available at ‘LMI Toolbox’ of Matlab® Math works Inc [11].

The figure 7 and the figure 8 show the satisfaction of the stability and performances robustness conditions (14) and (17).

The figure.9 show the step responses step responses of the closed loop controlled nominal system with: $R = \left(V_{ds\_ref} = {}^{1}\!/_{0} \quad V_{qs\_ref} = {}^{0}\!/_{1} \right)$ respectively.

The Outputs V_{ds} and V_{qs} follow the references with a good time response and no overshoot.

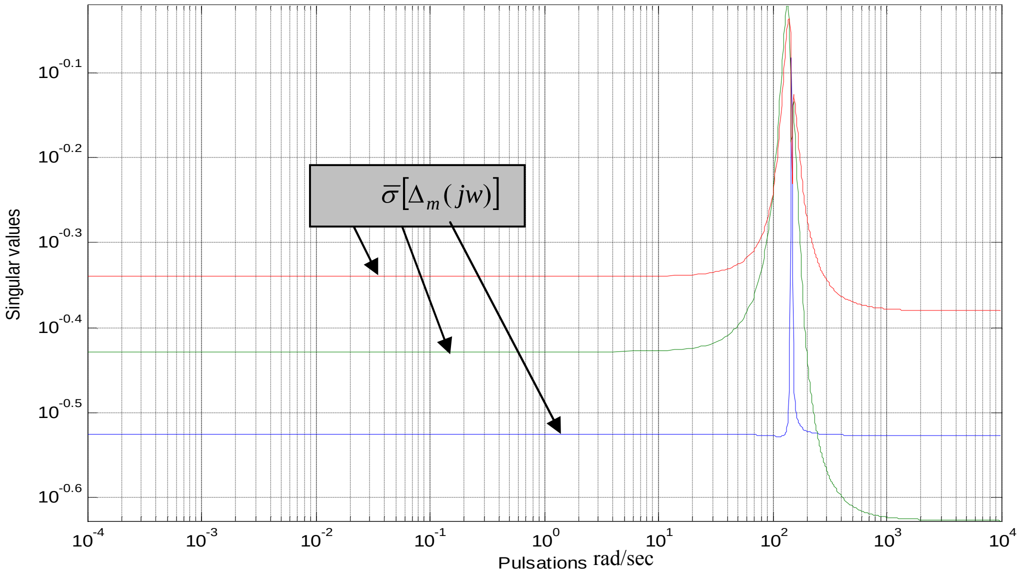


Figure 4: The system uncertainties maximum singular values

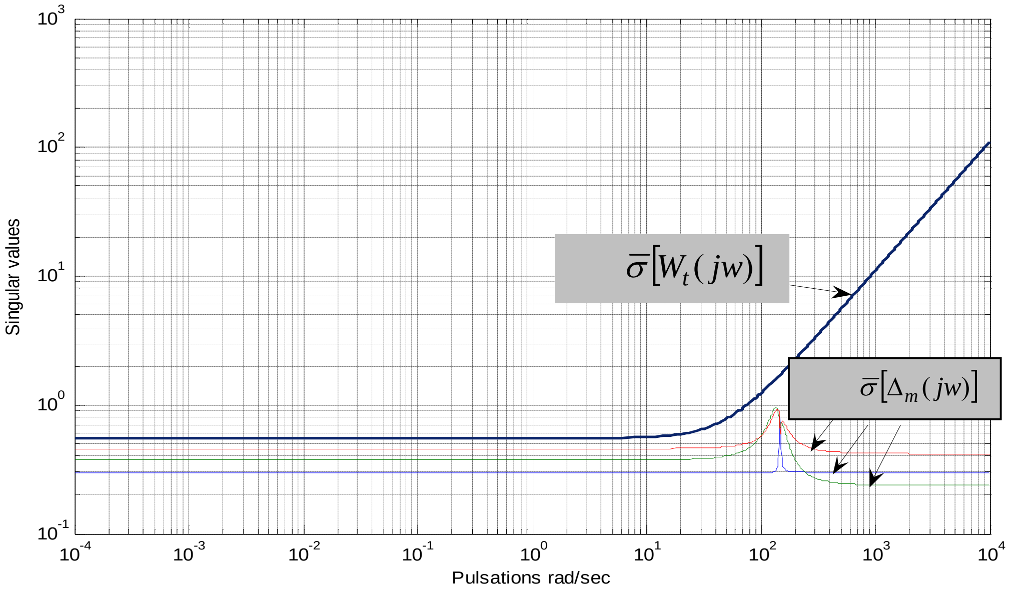


Figure 5: Maximum singular values of the system uncertainties Δ_m bounded by the singular values of $W_t(jw)$.

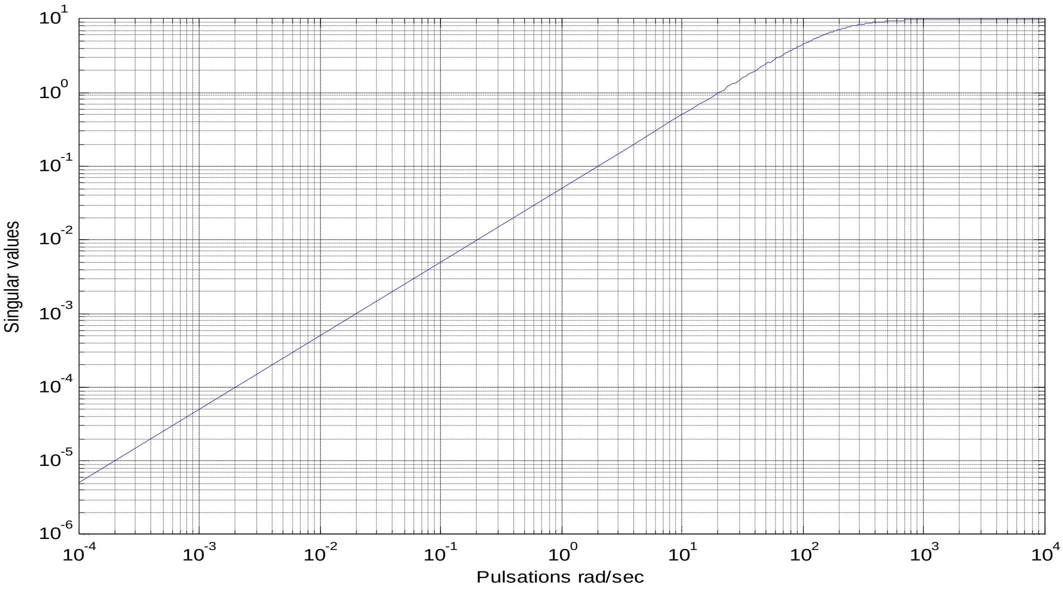


Figure 6: Singular Values of the weighting performance specification

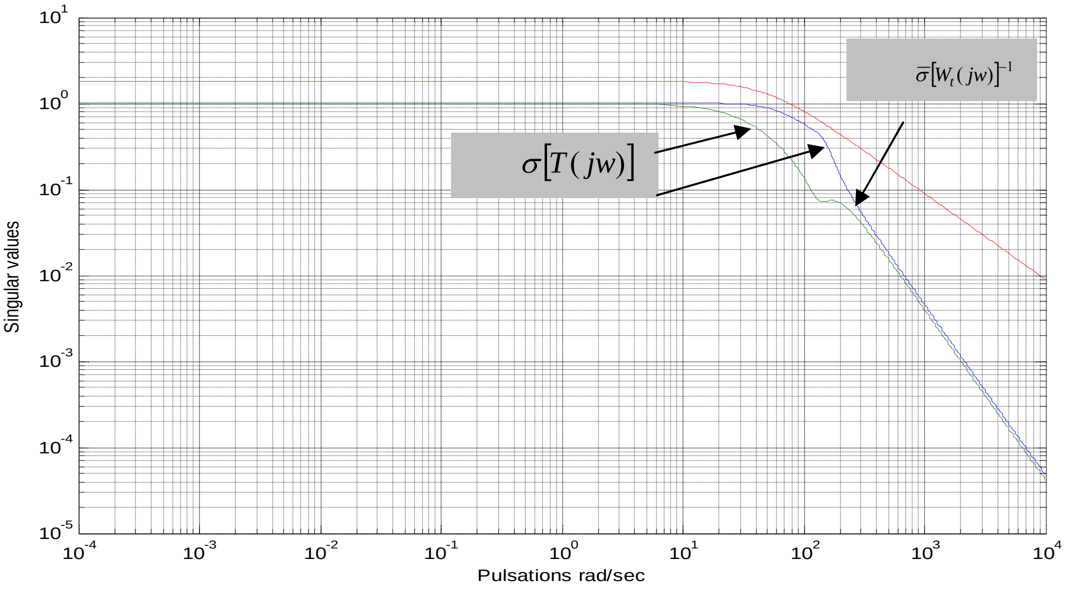


Figure 7: Stability robustness condition

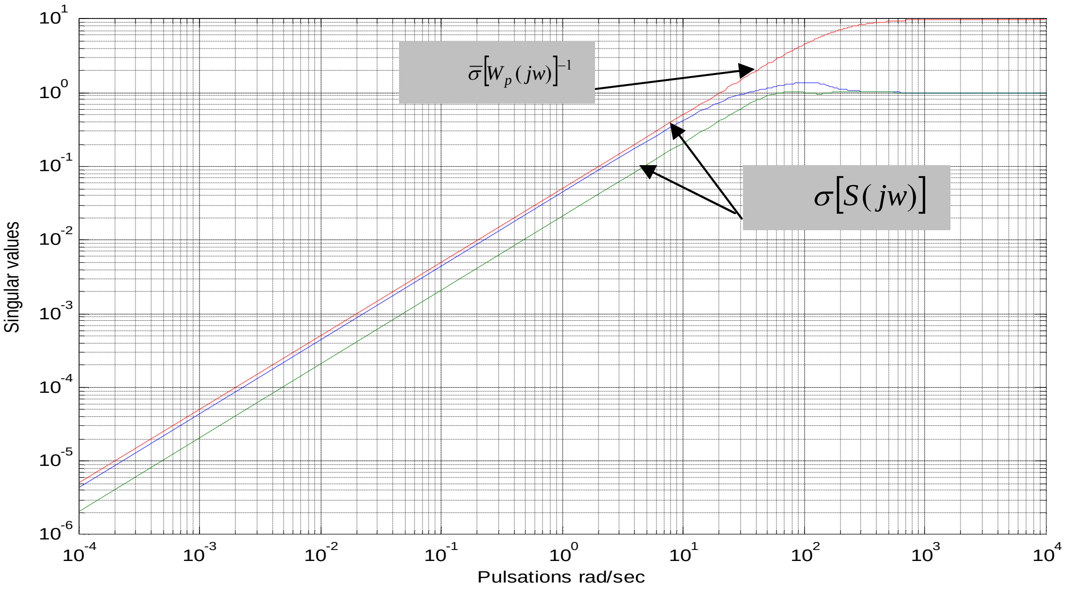


Figure 8: Performances robustness condition

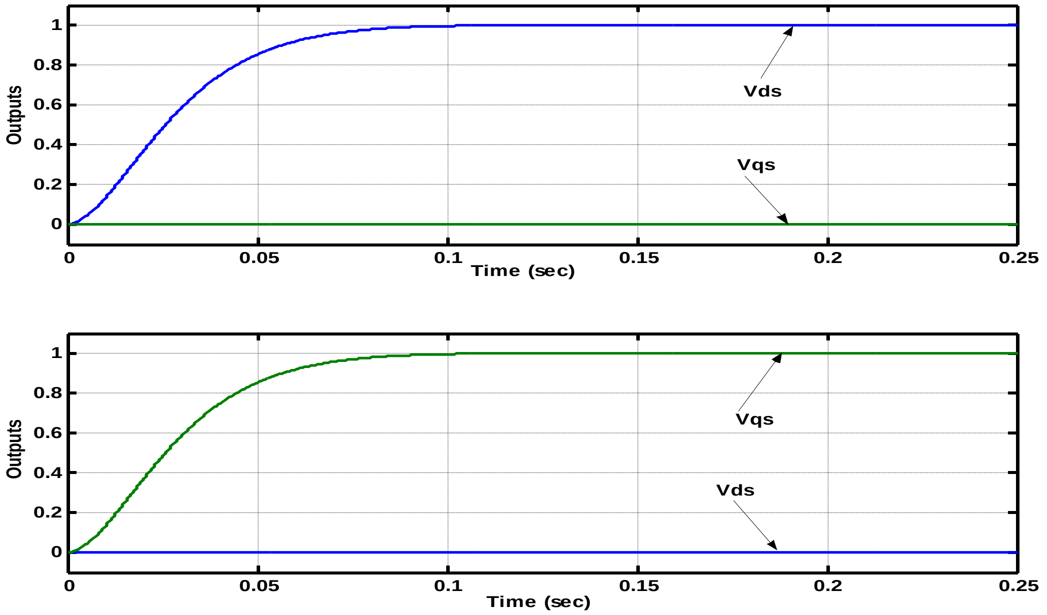


Figure 9: Step response of the controlled closed loop nominal system

5 Conclusion

In this paper we deal with the control problem of a wind turbine equipped with a doubly fed asynchronous machine subject to various perturbations and system uncertainties (wind speed variations, electrical energy consumption, system parameters variations ...etc), we show that the H∞ controller design method can be successfully applied to this kind of systems keeping stability and good performances in spite of the perturbations and system uncertainties.

References

[1] G. L. Johnson (2006), 'Wind energy systems: Electronic Edition', Manhattan, KS, October 10.

[2] 'AWEA Electrical Guide to Utility Scale Wind Turbines', (2005), The American Wind Energy Association, available at http://www.awea.org.

[3] P. Gahinet, P. Akparian (1994), 'A linear Matrix Inequality Approach to H_∞ Control ', Int. J. of Robust & Nonlinear Control", vol. 4, pp. 421-448.

[4] J. Soens, J. Driesen, R. Belmans (2005), ' Equivalent Transfer Function for a Variable-speed Wind Turbine in Power System Dynamic Simulations ', International Journal of Distributed Energy Resources, Vol.1 N°2, pp. 111-133.

[5] 'Dynamic Modelling of Doubly-Fed Induction Machine Wind-Generators' (2003), Dig Silent GmbH Technical Documentation, available at http://www.digsilent.de.

[6] J. Soens, J. Driesen, R. Belmans (2004), ' Wind turbine modelling approaches for dynamic power system simulations **',** IEEE Young Researchers Symposium in Electrical Power Engineering - Intelligent Energy Conversion, (CD-Rom), Delft, The Netherlands.

[7] J. Soens, V. Van Thong, J. Driesen, R. Belmans (2003), ' Modelling wind turbine generators for power system simulations ', European Wind Energy Conference EWEC.

[8] Sigurd Skogestad, Ian Postlethwaite (1996), 'Multivariable Feedback Control Analysis and Design', John Wiley and Sons. pp: 72 to 75

[9] J. C. Doyle, K. Glover, P. P. Khargonekar and Bruce A. Francis (1989), 'State-Space Solution to Standard H_2 and H_∞ Control Problems', IEEE Transactions on Automatic Control, Vol. 34, N°. 8.

[10] D.-W. Gu, P. Hr. Petkov and M. M. Konstantinov (2005), 'Robust Control Design with MATLAB® ', © Springer-Verlag London Limited.pp:27 to 29

[11] P. Gahinet, A. Nemirovski, A. J. Laub, M. Chilali (1995). "LMI Control Toolbox for Use with MATLAB®", User's Guide Version 1, The Math Works, and Inc.

Balancing Load in a Computational Grid Applying Adaptive, Intelligent Colonies of Ants

Mohsen Amini Salehi
Department of Software Engineering, Faculty of Engineering, Islamic Azad University, Mashhad Branch, Iran
E-mail: Amini@mshdiau.ac.ir

Hossein Deldari
Department of Software Engineering, Faculty of Engineering Ferdowsi University of Mashhad, Iran
E-mail: hd@um.ac.ir

Bahare Mokarram Dorri
Management and Planning Organisation of Khorasan, Mashhad, Iran
E-mail: mokarram@mpo-kh.ir

Keywords: Grid computing, Load balancing, Ant colony, Agent-based Resource Management System (ARMS)

Received: July 1, 2007

Load balancing is substantial when developing parallel and distributed computing applications. The emergence of computational grids extends the necessity of this problem. Ant colony is a meta-heuristic method that can be instrumental for grid load balancing. This paper presents an echo system of adaptive fuzzy ants. The ants in this environment can create new ones and may also commit suicide depending on existing conditions. A new concept called Ant level load balancing is presented here for improving the performance of the mechanism. A performance evaluation model is also derived. Then theoretical analyses, which are supported with experiment results, prove that this new mechanism surpasses its predecessor.

Povzetek: Metoda inteligentnih mravelj je uporabljena na problemu razporejanju bremen.

1 Introduction

A computational grid is a hardware and software infrastructure which provides consistent, pervasive and inexpensive access to high end computational capacity. An ideal grid environment should provide access to all the available resources seamlessly and fairly.

The resource manager is an important infrastructural component of a grid computing environment. Its overall aim is to efficiently schedule applications needing utilization of available resources in the grid environment. A grid resource manager provides a mechanism for grid applications to discover and utilize resources in the grid environment. Resource discovery and advertisement offer complementary functions. The discovery is initiated by a grid application to find suitable resources within the grid. Advertisement is initiated by a resource in search of a suitable application that can utilize it. A matchmaker is a grid middleware component which tries to match applications and resources. A matchmaker may be implemented in centralized or distributed ways. As the grid is inherently dynamic, and has no boundary [1], so the distributed approaches usually show better results [2] and are also more scalable. A good matchmaker (broker) should uniformly distribute the requests, along the grid resources, with the aid of load balancing methods.

As mentioned in [1], the grid is a highly dynamic environment for which there is no unique administration. Therefore, the grid middleware should compensate for the lack of unique administration.

ARMS is an agent-based resource manager infrastructure for the grid [3, 4]. In ARMS, each agent can act simultaneously as a resource questioner, resource provider, and the matchmaker. Details of the design and implementation of ARMS can be found in [2]. In this work, we use ARMS as the experimental platform.

Cosy is a job scheduler which supports job scheduling as well as advanced reservations [5]. It is integrated into ARMS agents to perform global grid management [5]; Cosy needs a load balancer to better utilize available resources. This load balancer is introduced in part 3.

The rest of the paper is organized as follows: Section 2 introduces the load balancing approaches for grid resource management. In Section 3, ant colony optimization and self-organizing mechanisms for load balancing are discussed. Section 4 describes the proposed mechanism. Performance metrics and simulation results are included in Section 5. Finally, the conclusion of the article is presented as well as future work related to this research.

2 Load balancing

Load balancing algorithms are essentially designed to spread the resources' load equally thus maximizing their utilization while minimizing the total task execution time [7]. This is crucial in a computational grid where the most pressing issue is to fairly assign jobs to resources. Thus, the difference between the heaviest and the lightest resource load is minimized.

A flexible load sharing algorithm is required to be general, adaptable, stable, scalable, fault tolerant, transparent to the application and to also induce minimum overhead to the system [8]. The properties listed above are interdependent. For example, a lengthy delay in processing and communication can affect the algorithm overhead significantly, result in instability and indicate that the algorithm is not scalable.

The load balancing process can be defined in three rules: the location, distribution and selection rule [7]. The location rule determines which resource domain will be included in the balancing operation. The domain may be local, i.e. inside the node, or global, i.e. between different nodes. The distribution rule establishes the redistribution of the workload among available resources in the domain, while the selection rule decides whether the load balancing operation can be performed preemptively or not [7].

2.1 Classification of load balancing mechanisms

In general, load balancing mechanisms can be broadly categorized as centralized or decentralized, dynamic or static [10], and periodic or non-periodic [11].

In a centralized algorithm, there is a central scheduler which gathers all load information from the nodes and makes appropriate decisions. However, this approach is not scalable for a vast environment like the grid. In decentralized models, there is usually not a specific node known as a server or collector. Instead, all nodes have information about some or all other nodes. This leads to a huge overhead in communication. Furthermore, this information is not very reliable because of the drastic load variation in the grid and the need to update frequently.

Static algorithms are not affected by the system state, as their behaviour is predetermined. On the other hand, dynamic algorithms make decisions according to the system state. The state refers to certain types of information, such as the number of jobs waiting in the ready queue, the current job arrival rate, etc [12]. Dynamic algorithms tend to have better performance than static ones [13].

Some dynamic load balancing algorithms are adaptive; in other words, dynamic policies are modifiable as the system state changes. Via this approach, methods adjust their activities based on system feedback [13].

3 Related works

Swarm intelligence [14] is inspired by the behaviour of insects, such as wasps, ants or honey bees. The ants, for example, have little intelligence for their hostile and dynamic environment [15]. However, they perform incredible activities such as organizing their dead in cemeteries and foraging for food. Actually, there is an indirect communication among ants which is achieved through their chemical substance deposits [16]. This ability of ants is applied in solving some heuristic problems, like optimal routing in a telecommunication network [15], coordinating robots, sorting [17], and especially load balancing [6, 9, 18, 19].

Messor [20] is the main contribution to the load balancing context.

3.1 Messor

Messor is a grid computing system that is implemented on top of the Anthill framework [18].

Ants in this system can be in Search–Max or Search–Min states. In the Search–Max state, an ant wanders around randomly until it finds an overloaded node. The ant then switches to the Search–Min state to find an underloaded node. After these states, the ant balances the two overloaded and underloaded nodes that it found. Once an ant encounters a node, it retains information about the nodes visited. Other ants which visit this node can apply this information to perform more efficiently. However, with respect to the dynamism of the grid, this information cannot be reliable for a long time and may even cause erroneous decision-making by other ants.

3.2 Self-Organizing agents for grid load balancing

In [6], J.Cao et al propose a self-organizing load balancing mechanism using ants in ARMS. As this mechanism is simple and inefficient, we call it the “seminal approach”. The main purpose of this study is the optimization of this seminal mechanism. Thus, we propose a modified mechanism based on a swarm of intelligent ants that uniformly balance the load throughout the grid.

In this mechanism an ant always wanders ‘2m+ 1’ steps to finally balance two overloaded and underloaded nodes.

As stated in [6], the efficiency of the mechanism highly depends on the number of cooperating ants (n) as well as their step count (m). If a loop includes a few steps, then the ant will initiate the load balancing process frequently, while if the ant starts with a larger m, then the frequency of performing load balancing decreases. This implies that the ant’s step count should be determined according to the system load. However, with this method, the number of ants and the number of their steps are defined by the user and do not change during the load balancing process. In fact, defining the number of ants and their wandering steps by the user is impractical in an environment such as the grid, where users have no background knowledge and the ultimate goal is to introduce a transparent, powerful computing service to end users.

Considering the above faults, we propose a new mechanism that can be adaptive to environmental conditions and turn out better results. In the next section, the proposed method is described.

4 Proposed method

In the new mechanism, we propose an echo system of intelligent ants which react proportionally to their conditions. Interactions between these intelligent, autonomous ants result in load balancing throughout the grid.

In this case, an echo system creates ants on demand to achieve load balancing during their adaptive lives. They may bear offspring when they sense that the system is drastically unbalanced and commit suicide when they detect equilibrium in the environment. These ants care for every node visited during their steps and record node specifications for future decision making. Moreover, every ant in the new mechanism hops 'm' steps (the value of 'm' is determined adaptively) instead of '2m+1'. At the end of the 'm' steps, 'k' overloaded are equalized with 'k' underloaded nodes, in contrast to one overloaded with one underloaded according to the previous method. This results in an earlier convergence with fewer ants and less communication overhead.

In the next sections, the proposed method is described in more detail.

4.1 Creating ants

If a node understands that it is overloaded, it can create a new ant taking only a few steps to balance the load as quickly as possible. Actually, as referred in [2], neighbouring agents, in ARMS, exchange their load status periodically. If a node's load is more than the average of its neighbours, for several periods of time, and it has not been visited by any ant during this time, then the node creates a new ant itself to balance its load throughout a wider area.

Load can be estimated several ways by an agent to distinguish whether a node is overloaded or not. For the sake of comparison with similar methods, the number of waiting jobs in a node is considered the criterion for load measurement.

4.2 Decision-making

Every ant is allocated to a memory space which records specifications of the environment while it wanders. The memory space is divided into an underloaded list (Min List) and an overloaded list (Max List). In the former, the ant saves specifications of the underloaded nodes visited. In the latter, specifications of the overloaded nodes visited are saved.

At every step, the ant randomly selects one of the node's neighbours.

4.2.1 Deciding algorithm

After entering a node, the ant first checks its memory to determine whether this node was already visited by the ant itself or not. If not, the ant can verify the condition of the node, i.e. overloaded, underloaded or at an equilibrium, using its acquired knowledge from the environment.

As the load quantity of a node is a linguistic variable and the state of the node is determined relative to system conditions, decision making is performed adaptively by applying fuzzy logic [21, 22].

To make a decision, the ant deploys the node's current workload and the remaining steps as two inputs into the fuzzy inference system. Then, the ant determines the state of the node, i.e. Max, Avg or Min. The total average of the load visited is kept as the ant's internal knowledge about the environment. The ant uses this for building membership functions of the node's workload, as shown in Figure1.a. The membership functions of Remain steps and Decide, as the output, are on the basis of a threshold and are presented in Figures 1.b, 1.c:

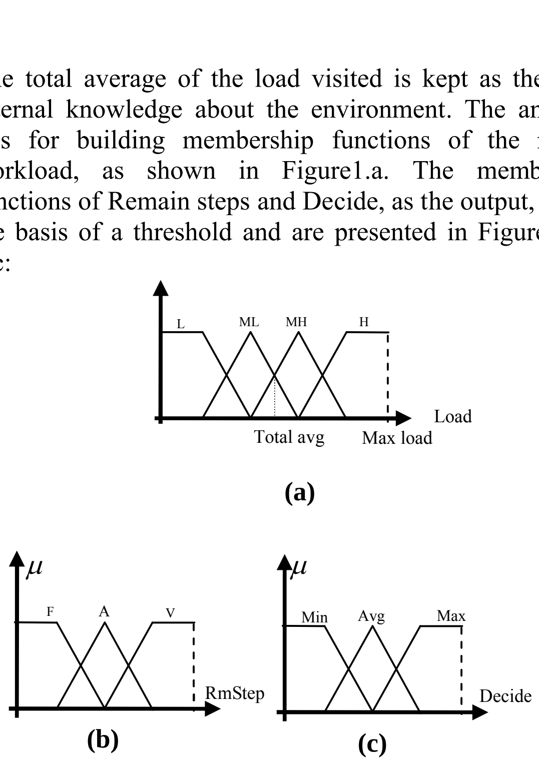


Figure 1: Membership functions of fuzzy sets
a) The Node's Load, b) Remain steps, c) Decide.

The inference system can be expressed as the following relation:

$$R_A : Load< L,ML,MH,H > *RmStep< F,A,V > \rightarrow Decide< Min,Avg,Max> \quad (1)$$

Where L, ML, MH, H in Figure1.a indicates Low, Medium Low, Medium High, High respectively and F, A, V in Figure 1.b indicates Few, Average and Very.

Thus, the ant can make a proper decision. If the result is "Max" or "Min", the node's specifications must be added to the ant's max-list or the min-list. Subsequently, the corresponding counter for Max, Min, or Avg increases by one. These counters also depict the ant's knowledge about the environment. How this knowledge is employed is explained in the next sections.

4.2.2 Ant level load balancing

In the subtle behaviour of ants and their interactions, we can see that when two ants face each other, they stop for a moment and touch tentacles, probably for recognizing their team members. This is what inspired the first use of ant level load balancing.

With respect to the system structure, it is probable that two or more ants meet each other on the same node. As mentioned earlier, each of these ants may gather specifications of some overloaded and underloaded nodes. The amount of information is not necessarily the same for each ant, for example one ant has specifications of four overloaded and two underloaded while the other has two overloaded and six underloaded nodes in the same position. In this situation, ants extend their knowledge by exchanging them. We call this *“ant level load balancing.”* In the aforementioned example, after ant level load balancing of the two co-positions, the ants have specifications of three overloaded and four underloaded nodes in their memories. This leads to better performance in the last step, when the ants want to balance the load of ‘k’ overloaded with ‘k’ underloaded nodes. This operation can be applied to more than two ants.

Actually, when two or more co-positioned ants exchange their knowledge, they extend their movement radius to a bigger domain, thus improving awareness of the environment. Another idea is taken from the ant’s pheromone deposits while travelling, which is used by ants to pursue other ants. This is applied in most ant colony optimization problems [23, 24]. There is, however, a subtle difference between these two applications. In the former the information retained by the ant may become invalid over time. This problem can be solved by evaporation [23, 24]. Evaporation, however, is not applicable in some cases, e.g. in the grid, where load information varies frequently. On the other hand, in the latter application, the knowledge exchanged is completely reliable.

4.2.3 How new ants are created

In special conditions, particularly when the its life span is long, the ant’s memory may fill up, even though the ant may still be encountering nodes which are overloaded or underloaded. In this situation, if a node is overloaded, the ant bears a new ant with predefined steps. If encountering an underloaded node, the ant immediately exchanges the node’s specification with the biggest load on the list of underloaded elements. This results in better balancing performance and adaptability to the environment. Here, adaptability translates into increasing the number of ants automatically, whenever there is an abundance of overloaded nodes.

4.3 Load balancing, starting new itineration

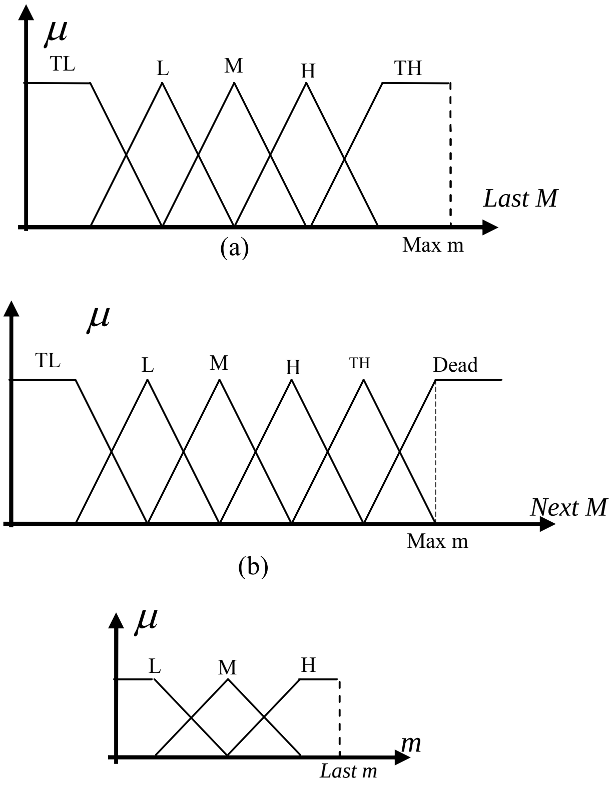
When its journey ends, the ant has to start a balancing operation between the overloaded (Max) and underloaded (Min) elements gathered during its roaming. In this stage, the number of elements in the Max-list and Min-list is close together (because of ant level load balancing). To achieve load balancing, the ant recommends underloaded nodes to the overloaded nodes and vice versa. In this way, the amount of load is dispersed equally among underloaded and overloaded nodes.

After load balancing, the ant should reinitiate itself to begin a new itineration. One of the fields that must be reinitiated is the ant’s step counts. However, as stated in previous sections, the ant’s step counts (m) must be commensurate to system conditions [6]. Therefore, if most of the visited nodes were underloaded or in equilibrium, the ant should prolong its wandering steps i.e. decrease the load balancing frequency and vice versa. Doing this requires the ant’s knowledge about the environment. This knowledge should be based on the number of overloaded, underloaded and equilibrium nodes visited during the last itineration.

Because of fuzzy logic power in the adaptation among several parameters in a problem [22] and the consideration of the step counts (m) as a linguistic variable, e.g. short, medium, long, it is rational to use fuzzy logic for determining the next itineration step counts.

Actually, this is an adaptive fuzzy controller which determines the next itineration step counts (NextM, for short) based on the number of overloaded, underloaded and equilibrium nodes visited, along with the step counts during the last itineration (LastM). In other words, the number of overloaded, underloaded and equilibrium nodes encountered during the LastM indicate the recent condition of the environment, while the LastM, itself, reports the lifetime history of the ant.

The membership functions of the fuzzy sets are shown in Figure 2.



(c)

Figure 2: Membership functions of fuzzy sets
a) Last m, b) Next m, c) count of Max, Min and Average nodes visited

Where TL, L, M, H, TH shows Too Low, Low, Medium, High and Too High in Figure 2.a, 2.b and L, M, H indicates Low, Medium and High in Figure 2.c.

This fuzzy system can be displayed as a relation and a corresponding function as follows:

$$R_B : MaxCount< l,m,h > *MinCount< l,mh> \\ *Avg< l,m,h > \rightarrow NextM< TL,L,M, \\ H,TH,Dead> \qquad (2)$$

$$f(x) = \frac{\sum_{r=1}^{135} \bar{y}^r * \prod_{i=1}^{4} \mu_{B_i^r}(x_i)}{\sum_{r=1}^{135} \prod_{i=1}^{4} \mu_{B_i^r}(x_i)} \quad (3)$$

Where x_i shows the input data into the system, $\bar{y}^r$ is the centre of the specific membership function declared in rule r. $\mu_{B_i^r}(x_i)$ indicates the membership value of the i th input in membership functions of the r th rule. In this inference system, we also have 4 inputs and 135 rules defined, as stated in (3).

In this system, a large number of underloaded and, especially, equilibrium elements indicate equilibrium states. Consequently, the NextM should be prolonged, thus lowering the load balancing frequency. One can say that, if an ant's step counts extend to extreme values, its effect tends to be zero. Based on this premise, we can conclude that an ant with overly long step counts does not have any influence on the system balance. Rather, the ant imposes its communication overhead on the system. In this situation, the ant must commit suicide. This is the last ring of the echo system. Therefore, if the *NextM* is fired in the *"Dead"* membership function, the ant does not start any new itineration.

Below is a diagram exhibiting the ant's behaviour in different environmental conditions. Figure 3.a shows the relation between the *LastM* and the amount of overloaded nodes visited, while Figure 3.b illustrates the relation between the *Last*M and the number of equilibrium nodes visited.

5 Performance valuations

In this section, several common statistics are investigated, which show the performance of the mechanism.

5.1 Efficiency

To prove that the new mechanism increases efficiency, it should be compared with the mechanism described in [4]. First, we introduce some of the most important criteria in load balancing:

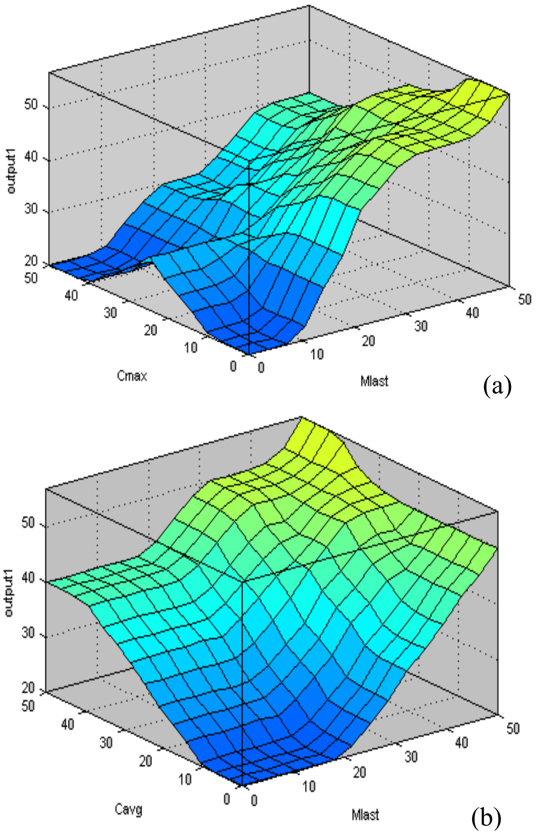


Figure 3: Schematic view of adaptive determining of next itineration step counts. a) LastM –MaxCount–output, b) LastM – avgCount–output

Let P be the number of agents and Wpk where (p: 1, 2... P) is the workload of the agent p at step k. The average workload is:

$$\overline{W_k} = \frac{\sum_{p=1}^{P} W_{pk}}{P} \quad (4)$$

The mean square deviation of Wpk, describing the load balancing level of the system, is defined as:

$$L_k = \sqrt{\frac{\sum_{p=1}^{P} (\overline{W_k} - W_{pk})^2}{P}} \quad (5)$$

Finally, The system load balancing efficiency (e) is defined as:

$$e_k = \frac{L_0 - L_k}{C_k} \quad (6)$$

Where ek means efficiency at step k and Ck is the total number of agent connections that have achieved a load balancing level Lk. To compare the efficiency of these two mechanisms, we should consider $e_{k_{new}} / e_{k_{Trad}}$.

As L0 indicates the load balancing level at the beginning of the load balancing process and is also equal in both new and seminal mechanisms, we shall discuss the value of Lk. For the sake of simplicity, assume that every node gets to $\overline{W}_k$ after the balancing process and no longer requires balancing, i.e.

$$\overline{W}_k - W_{pk} = 0 \qquad (7)$$

On the other hand, after the k stage, if the memory space considered for overloaded and underloaded elements is equal to 'a' (a>2), then we have ka elements balanced:

$$L_{k_{new}} = \sqrt{\frac{\sum_{p=1}^{p-ka}(\overline{W}_k - W_{pk})^2}{P}} \qquad (8)$$

While in the seminal approach we have:

$$L_{k_{Trad}} = \sqrt{\frac{\sum_{p=1}^{p-2k}(\overline{W}_k - W_{pk})^2}{P}} \qquad (9)$$

If we suppose that a>2, we can conclude:

$$P-2k > P-ka \qquad (10)$$

After the k stages, the difference in the balanced nodes in these two mechanisms is:

$$P-2a-P+ka = k(a-2) \qquad (11)$$

Then:

$$L_{k_{Trad}} = \sqrt{\frac{\sum_{p=1}^{p-ka}(\overline{W}_k - W_{pk})^2}{P} + \frac{\sum_{p=ka}^{p-2k}(\overline{W}_k - W_{pk})^2}{P}} \qquad (12)$$

$$L_{k_{new}} = \sqrt{\frac{\sum_{p=1}^{p-ka}(\overline{W}_k - W_{pk})^2}{P}} \qquad (13)$$

$$L_{k_{Trad}} > L_{k_{new}} \to \frac{L_{k_{new}}}{L_{k_{Trad}}} < 1 \qquad (14)$$

With respect to (14), we have:

$$\frac{e_{k_{new}}}{e_{k_{Trad}}} = \frac{2(L_0 - L_{k_{new}})}{L_0 - L_{k_{Trad}}} \Rightarrow \frac{e_{k_{new}}}{e_{k_{Trad}}} > 2 \qquad (15)$$

One of the most important parameters in the efficiency of the new mechanism is the ant's memory space (a). In an extreme case, if a=2, then the mechanism resembles the seminal one, with half steps (S), i.e.

$$S_{new} = 1/2 * S_{Trad} \qquad (16)$$

Consider that memory space (a) is effective if and only if it can be filled during the ant's wandering steps. Therefore, if a increases, then the amount of steps (S) must increase accordingly to prevent performance degradation. This means that:

If $a \to \infty$ then $S \to \infty$ (17)

Increasing S causes a decrease in load balancing frequency and consequently an increase in convergence time.

Overly long trips also lead to many reserved nodes. At the same time, there may be other roaming ants looking for free, unbalanced nodes. On the other hand, expanding the ant's memory leads to occupying too much bandwidth as well as increasing processing time. Actually, there is a trade-off between the step counts (S) and the memory allocated to each ant (a).

If a<<S, then the memory allocated expires rapidly and the ant is compelled to generate new ants. This explodes the ant population, subsequently augments their communication and the remaining pheromone and finally leads to an increase in time. However, as the probability of balancing every node more than once rises, the load balancing level falls.

On the other side, if a→S, then the probability of creating new ants lessens. Subsequently, the ant's population is reduced. Cutting down on the ant population results in faster speed, diminished communication and the pheromone left by the ants. The final result, however, is not satisfactory (final load balancing level is high). Due to the reasons discussed and with respect to several experiments shown in Figures 4, 5 and Table 1, we deduce that, in order to satisfy the different parameters mentioned, it is better to set the allocated memory at about half of the step counts.

$$a \cong S/2 \qquad (18)$$

Experiments achieved with a different memory size allocated, where S=15 initially, are reported here.

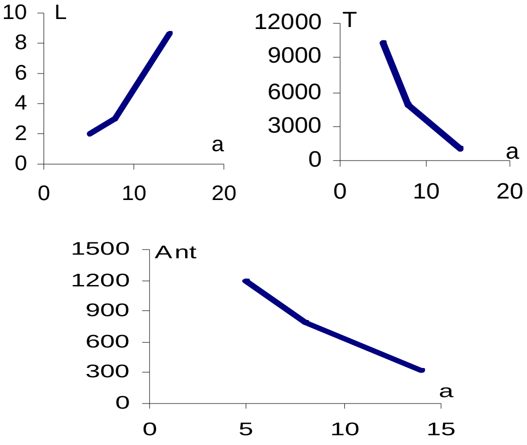


Figure 4: Relation between memory allocated to each ant (a) and a) load balancing level(L) b) time(T) base on millisecond c) Ant no (Ant), where the Ant initial Step Counts (S)=1

a	Time(ms)	Level	Ant No
5	10274	1.9455	1197
8	4906	3.0363	797
14	971	8.6015	325

Table 1: Relation between ants' memory size (a) and Ants with initial Step Counts (S=15).

5.2 Load balancing speed

Adaptively determining the step counts, actually, causes a differentiation in load balancing frequency over time. In other words, as time increases, the whole system approaches convergence and the load balancing frequency lessens, hence postponing the final convergence time. On the contrary, the new mechanism imposes less overhead as the system nears the balance state. In reality, in an environment such as the grid, attaining final convergence is impractical because of its inherent dynamism and vastness. However, if balancing occurs, it would not last long.

Figure 5 shows a schematic comparison for load balancing frequency between the new and the seminal mechanism.

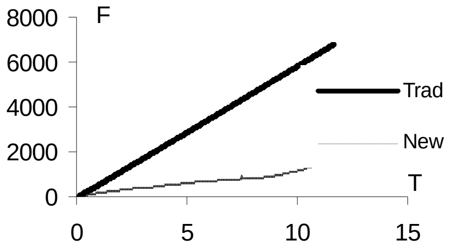


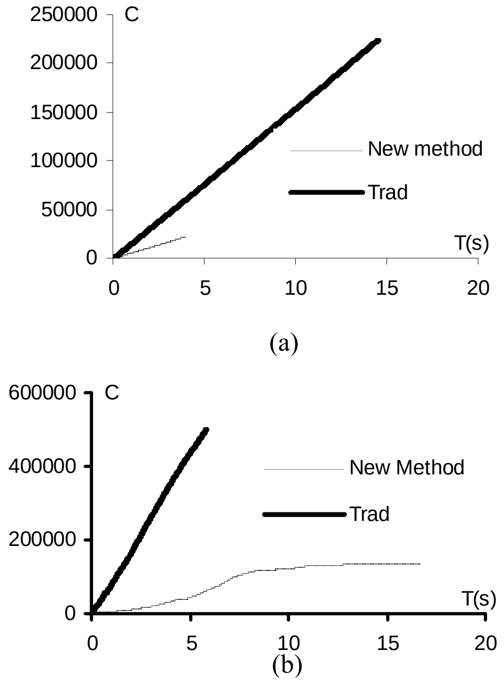
Figure 5: Comparison between the Seminal (Trad) and the new method’s load balancing frequency (F).

5.3 Experimental results

Experiments are achieved according to the specifications of Simorgh mini-grid [25]. This mini-grid will include different clusters throughout Ferdowsi University. However, as this mini-grid is under construction now, we have simulated its behaviour. In this simulation, Agent system, Workload, and Resources are modelled as follows:

- Agents. Agents are mapped to a square grid. This simplification has been done in similar works [6, 13]. All of experiments described later include 400 agents.
- Workload. A workload value and corresponding distribution are used to characterize the system workload. The value is generated randomly in each agent.
- Resources. Resources are defined in the same way as workload.

The first experiment involves total network connections. In this experiment, as shown in Figure 6.a, ant communication in the new mechanism is drastically less than in the seminal approach. This is mainly because every time an ant wanders ‘S’ steps in the new method, it balances ‘k’ elements. In the traditional method, however, the ant wanders ‘2S+1’ steps and then balances only two elements. Therefore, as seen below, with an equal initial step count (S=15), the ant in the new mechanism only goes through 2,000 stages to achieve final convergence, while in the traditional method, the ant passes 7,000 stages. Figure 6.b illustrates the comparison between a colony of ants using S=15 and a memory size=7. This figure elucidates that, in the new mechanism, the communication count goes flat. This occurs when the step counts enlarge and load balancing frequency decreases, i.e. in the last seconds.



(a)

(b)

Figure 6: Comparing agent communications (C) between the new and seminal (Trad) method. Final results using. a) One ant S=15, a=7 b) a colony of ants, N=220, S=15, a=7

The second experiment focuses on the relation between load balancing levels and the number of dead ants. As illustrated in Figure 7, as the number of dead ants rises, the load balancing level declines, i.e. it approaches final convergence. This experiment is conducted with different initial ants. Repeating the experiment with a different number of initial ants proves that, deploying more ants would result in better balancing level.

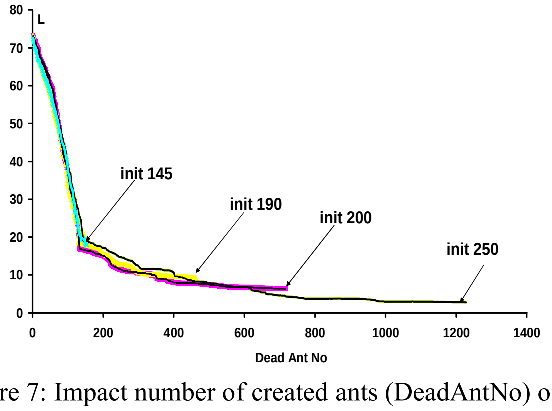


Figure 7: Impact number of created ants (DeadAntNo) on the load balancing level (L), the experiment achieved with different numbers of initial ants (init).

The third experiment concentrates on the correlation between an ant’s step counts and the load balancing level. The average step counts of the swarm over time are used for measurement. As Figure 8 shows, the step count increases by approaching convergence. This results in delay to achieve final convergence.

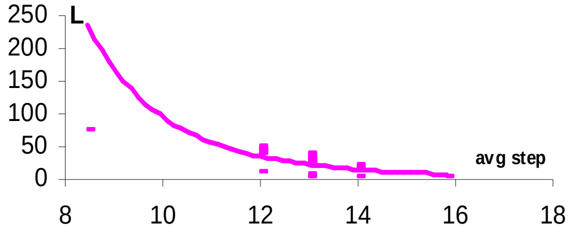


Figure 8: Relation between average Step count (avgStep) and load balancing Level (L)

The fourth experiment indicates the effect of proposed load balancing method on the final job distribution. As understood from Figure 9, ant level load balancing produces a better convergence.

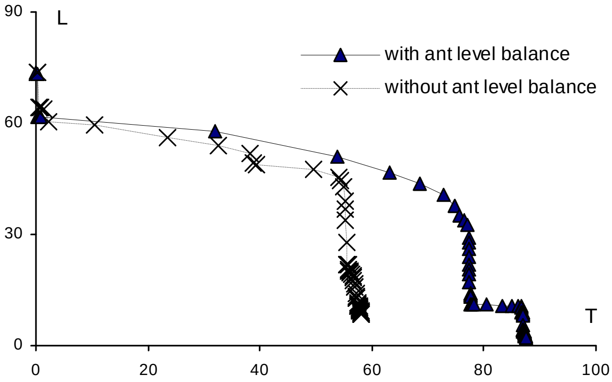


Figure 9: Comparison between effects of using ant level load balancing on final balancing level (L) during the time (T).

It is clear that this load balancing method cannot be achieved without any cost. As illustrated in Figure 9, although the proposed method results are better than previous ones, it consumes more time. We must acknowledge that the new method enables the ant to obtain global information even while moving locally. Furthermore, the validity of the exchanged information is guaranteed in contrast to using the pheromone, which is not, even with evaporation.

The fifth experiment presents the efficiency of the new method in comparison with the seminal approach. Figure 10 illustrates that the new method, with different initial steps and different memory allocated, is more efficient than the seminal one.

On the other hand, comparing the new method's efficiencies, with different initial step counts(S) and different memory allocated shows the effect of the trade-off in determining the memory allocated to each ant (a). In this case, if the memory allocated is high, e.g. a=10, then the probability of creating new ants decreases. So, the probability of visiting a node by different ants is decreased which causes a fall in efficiency. In the other way, as mentioned earlier, low values for memory allocated (a), e.g. a=5, increase the ant population and consequently their interconnections (Ck). This again results in decreasing the final efficiency in regard to (6).

Consider that stages do not have a completely standard meaning in our method. Thus, we think of periods of time as stages (k).

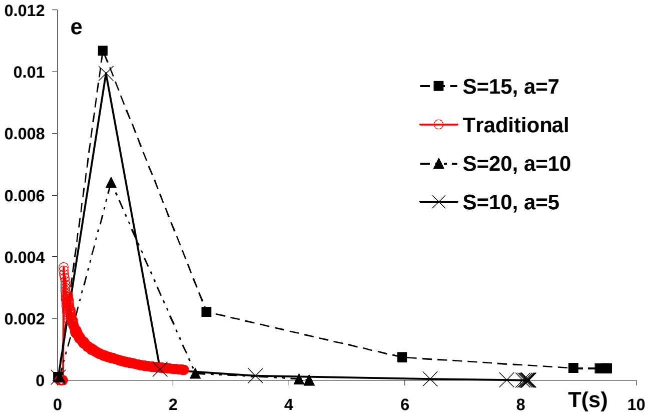


Figure 10: Efficiency (e) comparison between the traditional and the new method with different step counts and memory allocated, during the time (T).

6 Conclusion

As described in the previous sections, equalizing the load of all available resources is one of the most important issues in the grid. In this way, with respect to grid specifications, an echo system of autonomous, rational and adaptive ants is proposed to meet the challenges of load balancing. There are great differences between the proposed mechanism and similar mechanisms which deploy ant colony optimization. We believe that ant level load balancing is the most important difference.

In future work, we plan to extend the applications of ant level load balancing in addition to implementing this mechanism in a more realistic environment. Promoting ant intelligence and adaptation, establishing billing contracts among resources as they exchange customer loads, as well as overcoming security concerns are other future work.

References

[1] F. Berman, Anthony J.G. Hey, Geoffrey C. Fox (2003) *Grid Computing Making the Global Infrastructure a Reality* WILEY SERIES IN COMMUNICATIONS NETWORKING & DISTRIBUTED SYSTEMS.

[2] J. Cao, S. A. Jarvis, S. Saini, D. J. Kerbyson, and G. R. Nudd (2002) *ARMS: an Agent-based Resource Management System for Grid Computing*, Scientific Programming, Special Issue on Grid Computing Vol. 10, No. 2 pp. 135-148.

[3] M. Baker, R. Buyya, and D. Laforenza (2002) *Grids and Grid Technologies for Wide-area Distributed Computing*, Software: Practice and Experience Vol. 32, No. 15 pp. 1437-1466.

[4] J. Cao, D. J. Kerbyson, G. R. Nudd (2001) *Performance evaluation of an agent-based resource management infrastructure for grid computing* in

Proc. 1st IEEE Int. Symp. on Cluster Computing and the Grid, pp. 311-318.

[5] J. Cao and F. Zimmermann (2004) *Queue Scheduling and Advance Reservations with COSY* in Proc. of 18th IEEE Int. Parallel and Distributed Processing Symp pp. 120-128.

[6] J. Cao (2004) *Self-Organizing Agents for Grid Load Balancing* Proc. of the 5th IEEE/ACM Int. Workshop on Grid Computing, pp.168-176.

[7] A. Y. Zomaya, and Y. The (2001) *Observations on using genetic algorithms for dynamic load-balancing*, IEEE Trans. on Parallel and Distributed Systems, Vol. 12, No. 9, pp. 899-911.

[8] O. Remien, J. Kramer (1992) *Methodical analysis of adaptive load sharing algorithms*, IEEE Trans. on Parallel and Distributed Systems, Vol. 3, No: 11, pp. 747-760.

[9] Bing Qi Chunhui Zhao (2007) Ant Algorithm Based Load Balancing for Network Sessions, *ICNC 2007*, 3th Int. Conference on Natural Computation, pp. 771-775.

[10] Y. Lan, T. Yu (1995) *A Dynamic Central Scheduler Load-Balancing Mechanism*, Proc. 14th IEEE Conf. on Computers and Communication, Tokyo, Japan, pp. 734-740.

[11] H.C. Lin, C.S. Raghavendra (1992) *A Dynamic Load-Balancing Policy with a Central Job Dispatcher (LBC),* IEEE Transaction on Software Engineering, Vol. 18, No. 2, pp. 148-158.

[12] Z. Zeng, B. Veeravalli (2004) *Rate-Based and Queue-Based Dynamic Load Balancing Algorithms in Distributed Systems*, *Proc.* 10th IEEE Int. Conf. on Parallel and Distributed Systems, pp. 349- 356.

[13] M. Amini, H. Deldari (2006) *Grid Load Balancing Using an Echo System of Ants*, Proc. Of 24th IASTED Int. Conf, Innsbruck, pp. 47–52.

[14] E. Bonabeau, M. Dorigo, G. Theraulaz (1999) *Swarm Intelligence: from natural to artificial systems*, Oxford University Press, pp: 75-98.

[15] M. T. Islam, P. Thulasiraman, R. K. Thulasiram (2003) *A Parallel Ant Colony Optimization Algorithm for All-Pair Routing in MANETs*, Proc. 3th Int. Symp., Parallel and Distributed Processing, pp. 259-270.

[16] Siriluck Lorpunmanee, Mohd Noor Sap, Abdul Hanan Abdullah, and Chai Chompoo-inwai (2007) *An Ant Colony Optimization for Dynamic Job Scheduling in Grid Environment*, Int. Journal of Computer and Information Science and Engineering Volume 1 Number 4, pp. 207-214.

[17] J. Deneubourg, S. Goss, N. Franks (1990) *The dynamics of collective sorting robot-like ants and ant-like robots*, Proc. of the 1st Int. Conf. on Simulation of Adaptive Behavior, pp. 356-363.

[18] Ö. Babaoglu, H. Meling, A. Montresor (2002) *Anthill: A Framework for the Development of Agent-Based Peer-to-Peer Systems*, in Proc. of 22th IEEE Int. Conf. on Distributed Computing Systems, Vienna, Austria, pp. 15-22.

[19] J. Liu, X. Jin, and Y. Wang (2005) *Agent-Based Load Balancing on Homogeneous Minigrids: Macroscopic Modeling and Characterization*, IEEE TRANS. ON PARALLEL AND DISTRIBUTED SYSTEMS, VOL 16, NO 7, pp.586-598.

[20] A. Montresor, H. Meling, and Ö. Babaoglu (2002), Messor: *Load-Balancing through a Swarm of Autonomous Agents*, in Proc. of 1st ACM Int. Joint Conf. on Autonomous Agents and Multi-Agent Systems, Bologna, Italy, pp.112-120.

[21] A. Shaout, P. McAuliffe (1998) *Job scheduling using fuzzy load balancing in distributed system*, ELECTRONICS LETTERS, Vol. 34, No. 20, pp. 56-62.

[22] Hai Zhuge, Jie Liu (2004) *A fuzzy collaborative assessment approach for Knowledge Grid*, Int. Journal of Future Generation Computer Systems, Vol. 2, No 20, pp. 101-111.

[23] M. Dorigo, L. Maria (1997) *Ant Colony System: A Cooperative Learning Approach to the Traveling Salesman Problem*, Transactions ON EVOLUTIONARY COMPUTATION, VOL. 1, NO. 1, pp. 53-66.

[24] M. Dorigo, G. Carol (1999) *Ant Colony Optimization: A New Meta-Heuristic*, proc. Of 3th Fuzzy Sets and Systems, pp. 21–29.

[25] http://profsite.um.ac.ir/~hpcc

Improving Part-of-Speech Tagging Accuracy for Croatian by Morphological Analysis

Željko Agić and Zdravko Dovedan
Department of Information Sciences, Faculty of Humanities and Social Sciences, University of Zagreb
Ivana Lučića 3, HR-10000 Zagreb, Croatia
E-mail: {zeljko.agic, zdravko.dovedan}@ffzg.hr

Marko Tadić
Department of Linguistics, Faculty of Humanities and Social Sciences, University of Zagreb
Ivana Lučića 3, HR-10000 Zagreb, Croatia
E-mail: marko.tadic@ffzg.hr

Keywords: part-of-speech tagging, morphological analysis, inflectional lexicon, Croatian language

Received: September 12, 2008

This paper investigates several methods of combining a second order hidden Markov model part-of-speech (morphosyntactic) tagger and a high-coverage inflectional lexicon for Croatian. Our primary motivation was to improve tagging accuracy of Croatian texts by using our newly-developed tagger CroTag, currently in beta-version. We also wanted to compare its tagging results – both standalone and utilizing the morphological lexicon – to the ones previously described in (Agić and Tadić 2006), provided by the TnT statistical tagger which we used as a reference point having in mind that both implement the same tagging procedure. At the beginning we explain the basic idea behind the experiment, its motivation and importance from the perspective of processing the Croatian language. We also describe tools – namely tagger and lexicon – and language resources used in the experiment, including their implementation method and input/output format details that were of importance. With the basics presented, we describe in theory four possible methods of combining these resources and tools with respect to their operating paradigm, input and production capabilities and then put these ideas to test using the F-measure evaluation framework. Results are then discussed in detail and conclusions and future work plans are presented.

Povzetek: Za hrvaški jezik je razvita metoda za označevanje besedila.

1 Introduction

After obtaining satisfactory results of the preliminary experiment with applying a second order hidden Markov model part-of-speech/morphosyntactic tagging paradigm by using TnT tagger on Croatian texts, we decided to attempt reaching a higher level of accuracy based on these results. Detailed description of the previous experiment is given in (Agić and Tadić 2006) and TnT tagger is described in (Brants 2000). Please note that abbreviation HMM is used instead hidden Markov model and PoS (MSD) tagging instead part-of-speech (morphosyntactic) tagging further in the text.

In the section about our future work plans in (Agić and Tadić 2006), we provided two main directions for further enhancements:

a. Producing new, larger and more comprehensive language resources, i.e. larger, more precisely annotated and systematically compiled corpora of Croatian texts, maybe with special emphasis on genre diversity and
b. Developing our own stochastic tagger based on HMMs (being that TnT is available to public only as a black-box module) and then altering it by adding morphological cues about Croatian language or other rule-based modules.

We considered both courses of action as being equally important. HMM PoS/MSD trigram taggers make very few mistakes when trained on large and diverse corpora encompassing most of morphosyntactic descriptions for a language and, on the other hand, they rarely seem to surpass 98% accuracy on PoS/MSD, excluding the tiered tagging approach by (Tufis 1999.) and (Tufis and Dragomirescu 2004), not without help of rule-based modules, cues from morphological lexica or other enhancements which in fact turn stochastic tagging systems into hybrid ones. We have therefore chosen to undertake both courses of action in order to create a robust version of Croatian PoS/MSD tagger that would be able to provide us with high-quality MSD-annotated Croatian language resources automatically.

However, knowing that manual production of MSD-tagged corpora takes substantial amounts of time and human resources, we put an emphasis on developing and fine-tuning the trigram tagger in this experiment. Here we describe what is probably the most straightforward of currently available fine-tuning options for Croatian – combining CroTag tagger and Croatian morphological lexicon. The lexicon itself is described in (Tadić and Fulgosi 2003) and implemented in form of Croatian lemmatization server, described in (Tadić 2006) and available online at http://hml.ffzg.hr. Our notion of tagger-lexicon combination in this paper refers to several possibilities of utilizing high coverage of the lexicon on Croatian texts in order to assist CroTag where it makes most errors, namely while tagging tokens that were not encountered by its training procedure.

Section 2 of the paper describes all the tools, language resources, annotation standards, input and output formats used in the experiment, while section 3 deals in theory with four conceptually different but functionally similar methods of pairing CroTag tagger and Croatian morphological lexicon. Section 4 defines the evaluation framework that would finally provide us with results. Discussion and conclusions along with future plans are given in sections 5 and 6.

2 Resources and tools

In this section, we give detailed insight on tools and resources used in the experiment, along with other facts of interest – basic characteristics of available annotated corpora and input-output file format standard used.

2.1 Inflectional lexicon

At the first stage of the experiment, we had available the Croatian morphological lexicon in two forms – one was the generator of Croatian inflectional word forms, described in (Tadić 1994) and another was the Croatian lemmatization server, detailed in (Tadić 2006). As it can be verified at http://hml.ffzg.hr, the server takes as input a UTF-8 encoded verticalized file. File verticalization is required because the server reads each file line as a single token which is used as a query in lemma and MSD lookup. Output is provided in form of a text file and an equivalent HTML browser output. Figure 2.1 represents a simplified illustration of this output: first token is the word form given at input and it is followed by pairs of lemmas and corresponding morphosyntactic descriptors compliant to MULTEXT-East v3 specification, given by e.g. (Erjavec 2004).

```
da [da2 Qr] [dati Vmia2s] [dati Vmia3s]
       [dati Vmip3s] [da1 Css]
```

Figure 2.1: Output of inflectional lexicon (illustration).

Therefore, a text document was extracted from the server containing all (lemma, token, MSD) triples and any computer program or a programming library implementing fast search capability over this document could be utilized in our experiment as a black-box module. For this purpose, we used the Text Mining Tools library (TMT), described in (Šilić et al. 2007), that had implemented a very fast and efficient dictionary module based on finite state automata, storing triples of word forms, lemmas and tags into an incrementally constructed deterministic automaton data structure. This TMT dictionary module has thus provided us with the needed object-oriented interface (conveniently developed in C++, same as CroTag) that we could use to get e.g. all lemmas and MSDs for a token, all MSDs for a (token, lemma) pair etc. By utilizing this library, a working inflectional lexicon interface was at our disposal to be used both as an input-output black-box and rule-based module for integration with CroTag at runtime.

2.2 Stochastic tagger

Stochastic PoS/MSD trigram tagger for Croatian (or just CroTag from this point on) was developed and made available in form of an early beta-version for purposes of validation in this experiment, enabling us to envision future improvement directions and implementation efforts. Although many stochastic taggers have been made available to the community for scientific purposes during the years – for example, the TnT tagger (Brants 2000) and its open source reimplementation made in OCaml programming language, named HunPos (Halacsy et al. 2007) – and could be utilized in research scheme of our experiment, we still chose to develop our own trigram HMM tagger. This enabled us to alter its operation methods whenever required and also allowed us to integrate it with larger natural language processing systems that are currently under development for Croatian, such as the named entity recognition and document classification libraries. CroTag is developed using standard C++ with some helpful advice from the HunPos development team and additional interpretation of the OCaml source of HunPos tagger itself.

At this moment, the tagger implements only a second order hidden Markov model tagging paradigm (trigram tagging), utilizing a modified version of the Viterbi algorithm (Thede and Harper 1999), linear interpolation, successive abstraction and deleted interpolation as smoothing and default unknown word handling paradigms. These are de facto standard methods, also found in both TnT and HunPos. CroTag presumes token emission upon reached state and is trained as a visible Markov model, i.e. on pre-tagged corpora, from which it acquires transition and emission probability matrices, as described in e.g. (Manning and Schütze 1999).

Input and output formats of CroTag are once again virtually identical to ones of TnT and HunPos The training procedure takes a verticalized, sentence delimited corpus and creates the language model – i.e. tag transition and token emission probability matrices – while the tagging procedure takes as input a verticalized, sentence delimited, non-tagged text and utilizes the language model matrices to provide an output formatted identical to that required for training input: verticalized text containing a token and MSD per line.

Since CroTag is still under heavy development taking several different implementation directions, tagging procedures do not offer any possibility of setting the parameters to the user at the moment, although implementation of these options is placed on our to-do list. Once we develop a final version of CroTag, it will be made available to the community as a web service and possibly as an open source project as well. Additional work planned for CroTag beta is discussed in section 6 together with other possible research directions.

2.3 Annotated corpus

The Croatia Weekly 100 kw newspaper corpus (CW100 corpus further in the text) consists of articles extracted from seven issues of the Croatia Weekly newspaper, which has been published from 1998 to 2000 by the Croatian Institute for Information and Culture. This 100 kw corpus is a part of Croatian side of the Croatian-English parallel corpus, as described by (Tadić 2000).

PoS	Corpus %	Different MSD
Noun	30.45	119
Verb	14.53	62
Adjective	12.06	284
Adposition	09.55	9
Conjunction	06.98	3
Pronoun	06.16	312
Other	20.27	107

Table 2.1: PoS distribution on the CW100 corpus.

The CW100 corpus was manually tagged using the MULTEXT-East version 3 morphosyntactic descriptors specification, detailed in (Erjavec 2004) and encoded using XCES encoding standard (Ide et al. 2000). The corpus consists of 118529 tokens, 103161 of them being actual wordforms in 4626 different sentences, tagged by 896 different MSD tags. Nouns make for a majority of corpus wordforms (30.45%), followed by verbs (14.53%) and adjectives (12.06%), which is in fact a predictable distribution for a newspaper corpus.

Some details are provided in Table 2.1. Please note that PoS category Other includes acronyms, punctuation, numerals, etc. A more detailed insight on the CW100 corpus stats and pre-processing methods can be found in (Agić and Tadić 2006).

3 Combining lexicon and tagger

Four different methods were considered while planning this experiment. They all shared the same preconditions for input and output file processing, as described in the previous section. We now describe in theory these methods of pairing our trigram tagger and morphological lexicon.

3.1 Tagger resolving lexicon output

The first idea is based on very high text coverage displayed by the inflectional lexicon (more than 96.5% for contemporary newspaper texts documented). The text, consisting of one token per line to be tagged, could serve as input to the lexicon, providing all known MSDs given a wordform in each output line. The tagger would then be used only in context of tag sequence probabilities obtained by the training procedure and stored in the transition probability data structure. Namely, a program module could be derived from basic tagger function set, using tagger's tag transition probabilities matrix to find the optimal tag sequence in the search space, narrowed by using output of the inflectional lexicon instead of a generally poor lexical database stored in the emission probability matrix acquired at training.

3.2 Lexicon handling unknown words

A second-order HMM tagger such as CroTag is largely (almost exclusively) dependent of matrices of transition and emission probabilities, both of which are usually obtained from previously annotated corpora by a training procedure. As mentioned before, both CroTag and TnT (and HunPos, for that matter) use visible Markov model training procedures. It is well-known that it this case a large gap occurs when comparing PoS/MSD tagging accuracies on tokens known and unknown to the tagger in terms of the training procedure. If the training procedure encounters wordforms and discovers their respective tag distributions at training, error rates for tagging these words decrease substantially compared to tagging words that were not encountered at training. Improving trigram tagger accuracy therefore often means implementing an advanced method of guessing distributions of tags for unknown wordforms based on transition probabilities and other statistical methods, e.g. deleted interpolation, suffix tries and successive abstraction. Namely, TnT tagger implements all the methods listed above. However, most of these heuristic procedures frequently assign MSD tag distributions containing morphosyntactic descriptions having no linguistic sense for given unknown wordforms. We based our second method of pairing CroTag and inflectional lexicon on that fact alone; it would be worth investigating whether lexicon – as a large, high-coverage database of wordforms and associated lemmas and MSDs – could serve as unknown word handling module for the tagger at runtime. As it is expected that in most cases lexicon would recognize more word forms than tagger, implementation of this setting seemed to us as a logical and feasible course of action.

Suffix trie	Lexicon	Distribution
$p(tag_{i1}\|suff_i)$ ... $p(\boxed{tag_{ii}}\|suff_i)$ $p(\boxed{tag_{ij}}\|suff_i)$ $p(\boxed{tag_{ik}}\|suff_i)$ ... $p(tag_{in}\|suff_i)$	$(w_i\ l_1\ tag_1)$ ... $(w_i\ l_1\ \boxed{tag_{ii}})$ $(w_i\ l_1\ \boxed{tag_{ij}})$ $(w_i\ l_2\ \boxed{tag_{ik}})$ ... $(w_i\ l_m\ tag_1)$	$p'(tag_{ii}\|w_i)$ $p'(tag_{ij}\|w_i)$ $p'(tag_{ik}\|w_i)$ $\sum p' = \sum p$

Table 3.1: Lexicon improving the suffix trie.

In more detail, the idea builds on (Halacsy et al. 2006) and (Halacsy et al. 2007) and is basically a simple extension of the unknown word handling paradigm using suffix tries and successive abstraction (Samuelsson 1993). Trigram tagger such as TnT uses algorithms to disambiguate between tags in tag lists provided by emission probability matrix for a known wordform. Upon encountering an unseen wordform, such a list cannot be found in the matrix and must be constructed from another distribution, e.g. based on wordform suffixes acquired from specific types of encountered wordforms and implemented in the suffix trie data structure. Successive abstraction module contributes by iteratively choosing a more general distribution, i.e. distribution for shorter suffixes, shortening until a distribution of tags for a matching is finally assigned to the unknown token. This results in large and consequently low-quality distributions of MSD tag probabilities for unknown word forms, resulting in lower tagging accuracy. Taking high coverage of the inflectional lexicon into consideration, our idea was to choose from the suffix trie distribution only those MSDs on which both lexicon and suffix trie intersect, falling back to suffix tries and successive abstraction alone when both lexicon and tagger fail to recognize the wordform. By this proposition, we utilize wordform and tag probabilities as given by the suffix trie and yet choose only meaningful wordform and tag pairs, i.e. pairs confirmed by reading the lexicon. Probabilities of tags that remain in distributions after the selection are recalculated, increasing and thus becoming more reliable for calculating the optimal tag sequence. Table 3.1 illustrates this principle: if suffix trie tag and lexicon tag for an unknown token match, this tag is chosen for the new emission distribution of the previously unknown wordform and emission probability is recalculated.

3.3 Lexicon as pre-processing module

In this method, we train CroTag and obtain matrices containing transition and emission probabilities. The latter one, emission probability matrix, links each of the tokens found in the training corpus to its associated tags and counts, i.e. probabilities as is shown in Figure 3.1. The figure provides an insight on similarities and differences of storing language specific knowledge of tagger and inflectional lexicon.

```
%% ...
ime 26 Ncnsa 24 Ncnsn 2
imena 8 Ncnpa 1 Ncnpg 1 Ncnpn 3 Ncnsg 3
imenima 2 Ncnpd 1 Ncnpi 1
imenom 3 Ncnsi 3
imenovan 2 Vmps-smp 2
imenovana 1 Vmps-sfp 1
imenovanja 3 Ncnpg 2 Ncnsg 1
imenovanje 1 Ncnsv 1
imenovanjem 1 Ncnsi 1
imenovanju 4 Ncnsl 4
   %% ...
```

```
%% ...
ime ime Ncnsa ime Ncnsn ime Ncnsv
imenima ime Ncnpd ime Ncnpi ime Ncnpl
imenom ime Ncnsi
%% ...
```

Figure 3.1: Emission probability matrix file and lexicon output file comparison.

It was obvious that inflectional lexicon and tagger lexicon acquired by training have common properties, making it possible to create a lexicon-derived module for error detection and correction on the acquired lexicon used internally by the tagger. From another perspective, inflectional lexicon and tagger lexicon could also be merged into a single resource by some well-defined merging procedure.

3.4 Lexicon as post-processing module

Similar to using language knowledge of the inflectional lexicon before tagging, it could also be used afterwards. Output of the tagger could then be examined in the following manner:

1. Input is provided both to tagger and inflectional lexicon, each of them giving an output.
2. The two outputs are then compared, leading to several possibilities and corresponding actions:
   a. Both tagger and lexicon give an answer. Lexicon gives an unambiguous answer identical to the one provided by the tagger. No action is required.
   b. Both tagger and lexicon give an answer. Lexicon gives an unambiguous answer and it is different from the one provided by the tagger. Action is required and we choose to believe the lexicon as a manually assembled and thus preferred source of language specifics.
   c. Both tagger and lexicon give an answer. Lexicon gives an ambiguous answer, i.e. a sequence of tags. One of the tags in the sequence is identical to taggers answer. We keep the tagger's answer, being now confirmed by the lexicon.
   d. Both tagger and lexicon give an answer. Lexicon gives an ambiguous answer and none of the tags in the sequence matches the one provided by the tagger. A module should be written that takes into account the sequence provided by the lexicon and does re-tagging in a limited window of tokens in order to provide the correct answer. Basically, we define a window sized 3 tokens/tags and centred on the ambiguous token, lookup the most frequent of various trigram combinations available for the window (these are given by the lexicon!) in transition probability matrix of the tagger and assign this trigram to the window, disambiguating the output. By this we bypass tagger knowledge and once again choose to prefer lexicon output, unfortunately disregarding the fact that Viterbi algorithm outperforms this simple heuristic disambiguation.
   e. Tagger provides an answer, but token is unknown to the lexicon. We keep the tagger's answer, this being the only possible course of action.
   f. Tagger does not provide an answer and lexicon does. If its answer is unambiguous, we assign it

to the token. If it is ambiguous, we apply the procedure described in option 2d.

3. Final output produced by the merge is then investigated by the evaluation framework.

It should by all means be noted that each of the presented paradigms had to undergo a theoretical debate and possibly – if considered to be a reasonable course of action – a full sequence of tests described in section 4 in order to be accepted or rejected for introducing overall improvement of tagging accuracy or creating additional noise, respectively. Details are given in the following sections.

4 Evaluation method

As a testing paradigm, we chose the F-measure framework for evaluation on specific PoS and general accuracy for overall tagging performance. Firstly, we provide a comparison of CroTag beta and TnT: overall PoS vs. MSD accuracy and also F-measures on nouns, pronouns and adjectives, proven to be the most difficult categories in (Agić and Tadić 2006). We then discuss the proposed tagger-lexicon combinations and provide the measures – overall accuracy and F1-scores for those methods judged as suitable and meaningful at the time of conducting the investigation.

Each test consists of two parts: the worst-case scenario and the default scenario. Worst-case is a standard tagging accuracy measure scenario created by taking 90% of the CW100 corpus sentences for training and leaving the other 10% for testing. Therefore, in a way, this scenario guarantees the highest number of unknown words to be found at runtime given the corpus. The default scenario chooses 90% of sentences from the CW100 pool for training and then 10% for testing from the same pool, making it possible for sentences to overlap in these sets. The default scenario is by definition not a standard measure scenario and was introduced in order to respect the nature of random occurrences in languages, leaving a possibility (highly improbable) of tagger encountering identical sentences at training and at runtime. Also, we argue that investigating properties of errors occurring on highest accuracy scores, derived by the default testing scenario, provides additional insight on properties of trigram tagging in general.

Note that we do not include testing scenarios debating on training set size as a variable: in this test, we consider improving overall tagging accuracy and not investigating HMM tagging paradigm specifics as in (Agić and Tadić 2006), being that conclusions on this specific topic were already provided there.

5 Results

The first set of results we present is from the set of tests evaluating overall tagging accuracy of CroTag on full MULTEXT East v3 MSD and on PoS information only (by PoS we imply the first letter of the MSD tag – not comparable to English PoS of e.g. English Penn Treebank). Acquired results are displayed in Table 5.1.

It could be stated from this table that results on TnT and CroTag are virtually identical and the differences exist merely because testing environment – mainly the number of unknown words – was variable. It is however quite apparent that CroTag outperformed TnT on part-of-speech, especially regarding unknown tokens, but this should be taken with caution as well, being that CroTag dealt with fewer unknowns in that specific test.

Second testing case considers combining CroTag and the inflectional lexicon. Before presenting the results and in order to interpret them correctly, it should be stated that only two of the four initially proposed merging methods were chosen to proceed to the practical testing session: method (3.2) using the inflectional lexicon as an unknown world handler (3.4) using the inflectional lexicon as a postprocessing module to resolve potential errors produced by the tagger. We rejected applying (3.1) tagger as a disambiguation module for inflectional lexicon output because it would be costly to develop yet another tagger-derived procedure to handle transition probabilities only. This procedure would, in fact, do nothing different than a common HMM-based tagger does with its own acquired lexicon: disambiguates its ambiguous entries upon encountering them in the text and applying the transition probability matrix and handling procedures on unknown words.

		TnT		**CroTag**	
		MSD	**PoS**	**MSD**	**PoS**
Worst case	Overall	86.05	96.53	86.05	96.84
	Known	89.05	98.29	89.26	98.42
	Unknown	66.04	86.02	65.95	87.29
	Corp. unk.	13.07	14.40	13.77	14.11
Default case	Overall	97.54	98.51	97.51	99.31
	Known	98.04	98.74	98.05	99.43
	Unknown	62.21	83.11	63.75	88.39
	Corp. unk.	01.42	01.51	01.59	01.13

Table 5.1: Overall tagging accuracy on MSD and PoS.

The idea of inflectional lexicon as preprocessing module (3.3) was also rejected, mainly because we were unable to define precisely how to merge its database to the one acquired by tagger at training procedure. Being that tagger training procedure assigns each entry with a number of its occurrences overall and number of occurrences under various MSDs, in order to apply the inflectional lexicon as proposed by (3.3), we would have to assign these numbers so the tagger could understand the new entries. If we assign all to 1, it does not contribute and is redundant and if we assign any other number, we are in fact altering the tagging procedure outcome in such a manner that is not in any way bound by the language model, i.e. the training corpus. Therefore, we proceed with considering proposed cases (3.2) and (3.4) only.

We have also omitted PoS results from this testing case because TnT and CroTag are both able to achieve an accuracy over 95% without additional modules so we were focused in investigating MSD accuracy, keeping in mind that most errors do not occur on PoS but on sub-PoS levels resolvable by the lexicon. Details are provided by Table 5.2.

The first apparent conclusion is that method (3.4) that cleans up the errors on tagger output has failed and that it has failed on unknown words – where we may have expected it (or hoped for it) to perform better. The reason is, on the other hand and second thought, quite obvious: the tagger applies a tag to an unknown word using transition probabilities and smoothing procedures that are proven to operate quite satisfactory in TnT, HunPos and CroTag. When the postprocessing lexicon-based module encounters a word tagged as unknown, this word is rarely unambiguous in the inflectional lexicon. Therefore, a resolution module using transition probabilities has to be applied quite frequently and this module clearly and expectedly does not outperform default unknown word handling procedures.

		TnT	**CroTag +3.2**	**CroTag +3.4**
Worst case	Overall	86.05	85.58	83.94
	Known	89.05	88.84	88.18
	Unknown	66.04	65.13	57.38
	Corp. unk.	13.07	13.77	13.77
Default case	Overall	97.54	97.97	97.88
	Known	98.04	98.53	98.51
	Unknown	62.21	63.49	59.40
	Corp. unk.	01.42	01.59	01.59

Table 5.2: Tagging accuracy with (3.2) unknown word handler and (3.4) postprocessing.

Based on other stats in Table 5.2, we could end the section by stating that CroTag, when combined with the inflectional lexicon in such a manner that the lexicon provides morphological cues to the tagger upon encountering unknown words, outperforms TnT by a narrow margin on the default MSD test case. However, a more sincere and exact statement – taking in regard all section 5 tables – would be that both TnT and CroTag share the same functional dependency regarding the number of unknown words they encounter in the tagging procedure. That is, CroTag outperforms TnT when less unknown tokens occur for him at runtime and vice versa, the inflectional lexicon contributing for around 1.3% improvement on unknown words. We can thus argue that our beta-version of CroTag tagger performs as well as TnT tagger and that we succeeded in implementing a state-of-the-art solution for tagging large-scale corpora of Croatian, given the test environment we had at hands, its drawbacks noted and hereby included.

In Table 5.3 we present results of evaluation broken down by three most difficult PoS categories: adjectives, nouns and pronouns. Data and analysis is given for PoS information only, as mentioned before.

		Adjective	**Noun**	**Pronoun**
TnT	Worst case	64.56	81.63	75.42
	Default case	94.79	96.75	96.94
CroTag	Worst case	65.31	80.85	74.62
	Default case	95.86	97.40	95.88
CroTag +3.2	Worst case	66.72	82.61	77.32
	Default case	95.06	96.79	95.82

Table 5.3: Tagging accuracy with adjectives, nouns and pronouns.

It can be clearly noticed that suggested combination mode (3.2) outperforms both TnT and CroTag in the worst case scenario on all parts of speech since it has the support of HML when handling unknown words, that obviously do occur somewhat more frequently in this scenario. In the default case scenario, results are expectedly more even and inconclusive – default CroTag actually outperforms lexicon combination (3.2) because unknown tokens were found in small numbers in the test sets, much too small for the inflectional lexicon to contribute significantly to overall tagging accuracy.

6 Conclusion

In this contribution we have presented CroTag – an early beta-version of statistical PoS/MSD tagger for Croatian and proposed combining it with a large scale inflectional lexicon of Croatian, creating a hybrid system for high-precision tagging of Croatian corpora. We have presented several possible types of combinations, tested and evaluated two of them using the F-measure evaluation framework. CroTag provided results virtually identical to TnT, differing only in fractions of percentage in both directions in different evaluating conditions. This way we have shown that CroTag functions at the level of state-of-the-art regarding HMM-based trigram tagging and PoS/MSD-tagging in general.

Our future directions for improvement of this system could and probably can and probably will fall into several different research pathways.

The first of them should be analyzing tagging accuracy on morphological (sub-part-of-speech) features in more detail and fine-tuning the tagger accordingly.

Various parameterization options could also be provided at tagger runtime. Such options could include parameters for unigram, bigram and trigram preference or implementing token emissions depending on previously encountered sequences (multiword unit dependencies). As was previously mentioned, once we remove the beta-version appendix from CroTag by implementing these features and optimizing and tidying its source code, it will firstly be made available as a web service and then most probably as a freely-downloadable open source project on the web.

Fine-tuned rule-based modules for Croatian language specifics could also be considered and applied before or after the statistical procedure. Another option would be integration of inflectional lexicon into tagger as they have been programmed as separate modules, inducing some overhead to execution speed.

The next direction would be to build a full lemmatizer which, unlike inflectional lexicon presented in this paper, gives fully disambiguated lemmas as output relying on the results of the tagger. Selection of proper lemmas from sets of possible ones would be done on the basis of tagger output, once again fine-tuning levels of confidence between tagger and lemmatizer similar to section 3 of the paper.

It should also be noted that (Agić and Tadić 2008) takes into account an entirely different approach, putting an emphasis on corpora development. Namely, all the methods presented in previous sections are made exclusively for handling unknown word occurrences and all of them required lots of time and human effort to be implemented. On the other hand, manual corpora development – although obviously also requiring time and effort – is by definition a less demanding and at the same time reasonable course of action: larger, better and more diverse corpora are always a necessity for any language, necessity that implicitly resolves many unknown wordform issues as well. Courses of action could therefore be argued; we decided to take most of them throughout our future work in order to additionally improve tagging accuracy on Croatian texts.

Acknowledgement

This work has been supported by the Ministry of Science, Education and Sports, Republic of Croatia, under the grants No. 130-1300646-1776, 130-1300646-0645 and 036-1300646-1986.

References

[1] Agić, Ž., Tadić, M. (2006). Evaluating Morphosyntactic Tagging of Croatian Texts. In Proceedings of the Fifth International Conference on Language Resources and Evaluation. ELRA, Genoa – Paris 2006.

[2] Agić, Ž., Tadić, M. (2008). Investigating Language Independence in HMM PoS/MSD-Tagging. In Proceedings of the 30th International Conference on Information Technology Interfaces. Cavtat, Croatia, 2008, pp. 657-662.

[3] Brants, T. (2000). TnT – A Statistical Part-of-Speech Tagger. In Proceedings of the Sixth Conference on Applied Natural Language Processing. Seattle, Washington 2000.

[4] Erjavec, T. (2004). Multext-East Version 3: Multilingual Morphosyntactic Specifications, Lexicons and Corpora. In Proceedings of the Fourth International Conference on Language Resources and Evaluation. ELRA, Lisbon-Paris 2004, pp. 1535-1538.

[5] Halácsy, P., Kornai, A., Oravecz, C. (2007). HunPos - an open source trigram tagger. In Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics Companion Volume Proceedings of the Demo and Poster Sessions. Association for Computational Linguistics, Prague, Czech Republic, pp. 209-212.

[6] Halácsy, P., Kornai, A., Oravecz, C., Trón, V., Varga, D. (2006). Using a morphological analyzer in high precision POS tagging of Hungarian. In Proceedings of 5th Conference on Language Resources and Evaluation (LREC). ELRA, pp. 2245-2248.

[7] Ide, N., Bonhomme, P., Romary, L., (2000). An XML-based Encoding Standard for Linguistic Corpora. In Proceedings of the Second International Conference on Language Resources and Evaluation, pp. 825-830. (see also at http://www.xces.org).

[8] Manning, C., Schütze, H. (1999). Foundations of Statistical Natural Language Processing, The MIT Press, 1999.

[9] Samuelsson, C. (1993). Morphological tagging based entirely on Bayesian inference. 9th Nordic Conference on Computational Linguistics NODALIDA-93. Stockholm University, Stockholm, Sweden.

[10] Šilić, A., Šarić, F., Dalbelo Bašić, B., Šnajder, J. (2007). TMT: Object-Oriented Text Classification Library. Proceedings of the 29th International Conference on Information Technology Interfaces. SRCE, Zagreb, 2007. pp. 559-566.

[11] Tadić, M. (1994). Računalna obrada morfologije hrvatskoga književnog jezika. Doctoral thesis. Faculty of Humanities and Social Sciences, University of Zagreb, 1994.

[12] Tadić, M. (2000). Building the Croatian-English Parallel Corpus. In Proceedings of the Second International Conference on Language Resources and Evaluation. ELRA, Paris-Athens 2000, pp. 523-530.

[13] Tadić, M., Fulgosi, S. (2003). Building the Croatian Morphological Lexicon. In Proceedings of the EACL2003 Workshop on Morphological Processing of Slavic Languages. Budapest 2003, ACL, pp. 41-46.

[14] Tadić, M. (2006). Croatian Lemmatization Server. Formal Approaches to south Slavic and Balkan Languages. Bulgarian Academy of Sciences, Sofia, 2006. pp. 140-146.

[15] Thede, S., Harper, M. (1999). A second-order Hidden Markov Model for part-of-speech tagging. In Proceedings of the 37th annual meeting of the Association for Computational Linguistics, pp. 175-182.

[16] Tufiş, D. (1999). Tiered Tagging and Combined Classifiers. In F. Jelinek, E. Nöth (Eds.) Text, Speech and Dialogue, Lecture Notes in Artificial Intelligence 1692, Springer, 1999, pp. 28-33.

[17] Tufiş, D., Dragomirescu, L. (2004). Tiered Tagging Revisited. In Proceedings of the 4th LREC Conference. Lisbon, Portugal, pp. 39-42.

Applying SD-Tree for Object-Oriented Query Processing

I. Elizabeth Shanthi
Dept of Computer Science
Avinashilingam University for Women, Coimbatore, India.
E-mail: shanthianto@yahoo.com

R. Nadarajan
Dept of Mathematics and Computer Applications
PSG College of Technology, Coimbatore, India.
E-mail: nadarajan_psg@yahoo.co.in

Keywords: signature files, B+ tree, indexing, object oriented data bspases

Received: April 14, 2008

We follow signature-based approach to object-oriented query handling in this paper. The use of signature files as an index for full text search has been widely known and used. Signature file based access methods initially applied on text have now been used to handle set-oriented queries in Object-Oriented Data Bases (OODB). All the proposed methods use either efficient search method or tree based intermediate data structure to filter data objects matching the query. Use of search techniques retrieves the objects by sequentially comparing the positions of 1s in it. Such methods take longer retrieval time. On the other hand tree based structures traverse multiple paths making comparison process tedious. In this paper we describe a new indexing technique for representing signature file using the dynamic balancing of B+ tree called Signature Declustering tree (SD-tree). The structure has the positions of 1s in the signatures distributed over a set of leaf nodes. Using this for a given query signature all the matching signatures can be retrieved cumulatively from a single node. Also for signature insertion an optimal search path is calculated by keeping a threshold value and by using forward pointers in leaf nodes. To promote optimal search between subsequent queries the backward leaf node pointers are used. Experiments have been conducted to analyze the time and space overhead of the SD-tree by varying the signature length and the distribution of signature weight for varying query signature patterns. Also, to validate the proposed structure a hypothetical object schema is considered and sample queries evaluated.

Povzetek: SD drevesa so uporabljena za objektno obravnavo vprašanj.

1 Introduction

The advent of internet has made the volume of data going high everyday in all computer-based applications. This has entrusted researchers to design more powerful techniques to generate and manipulate large amounts of data to derive useful information. Indexing plays a vital role in the fast recovery of required data from large databases.

Among the many indexing techniques reported the signature file approach is preferred for its efficient evaluation of set-oriented queries and easy handling of insert and update operations. Initially applied on text data [2, 3, 12, 17, 21] it has now been used in other applications like office filing[6], relational and Object-Oriented Databases[14,16,26] and hypertext[9].

Signatures are hash coded abstractions of the original data. It is a binary pattern of predefined length with fixed number of 1s. The attributes' signatures are superimposed to form object's signature. To resolve a query, the query signature say Sq is generated using the same hash function and compared with signatures in the signature file for 1s sequentially and many non-qualifying objects are immediately rejected.

If all the 1s of Sq matches with that of the signature in the file it is called a drop. The signature file method guarantees that all qualifying objects will pass through the filtering mechanism; however some non-qualifying objects may also pass the signature test. The drop that actually matches the Sq is called an actual drop and drop that fails the test is called false drop. The next step in the query processing is the false drop resolution. To remove false drops each drop is accessed and checked individually. The number of false drops can be statistically controlled by careful design of the signature extraction method [7] and by using long signatures [3,6].

1.1 Related work

Different approaches have been discussed by researchers to represent Signature file in a way conducive for evaluating queries, such as Sequential Signature File[31], Bit-Slice Signature file[31], Multilevel Signature file[25], Compressed Multi Framed Signature file[23], Parallel Signature file[20], S-Tree and its variants[13,24], Signature Graph[28] and Signature tree[27,29,30].

1.2 Motivation for the current work

The signature tree developed by Chen [30] is having the following drawbacks:

- Signatures are inserted considering both 0s and 1s whereas actual weight age is for set bits only.
- Insertion path is dictated by the existing tree structure.
- To process a query, bits appearing in the tree from root node are compared with query signature pattern for 0s and 1s and not by its set bits.
- For a 0-bit in the query both left and right sub tree is followed leading to multiple traversals.

These observations laid the foundation for the current work. We study a new indexing technique for OODBSs using the dynamic balancing of B+ tree called Signature Declustering (SD)-tree in which the positions of 1s in the signatures are distributed over a set of leaf nodes. Using this for a given query signature all the matching signatures can be retrieved cumulatively in a single node.

The rest of the paper is organized as follows. In section 2 we discuss briefly the different approaches used to represent the signature file. In section 3 the structure of the proposed SD-tree is shown. Section 4 is devoted to the algorithms for insert, search and delete operations. Section 5 proposes a sample data set and queries to validate the new structure. Section 6 reports the results of the experiments conducted with the analytical comparison of SD-tree with that of Signature tree [30]. Finally section 7 concludes the work with further outlook on the work.

2 A summary of signature file techniques

2.1 Signature files

A Signature is a bit string formed from a given value. Compared to other index structures, signature file is more efficient in handling new insertions and queries on parts of words. Other advantages include its simple implementation and the ability to support a growing file. But it introduces information loss which can be minimized by carefully selecting the signature extraction method.

Techniques for signature extraction such as Word Signature (WS) [2,3,4,6], Superimposed Coding (SC) [1,2,3,4,5,6,10], Multilevel Superimposed Coding [12], Run Length Encoding (RL) [3,4,6], Bit-block Compression (BC) [3,4], Variable Bit-block Compression (VBC) [4,6] have been reported. The encoding scheme sets a constant number say m, of 1s in the range [1..F], where F is the length of the signature. The resulting binary pattern with m number of 1s and (F-m) number of 0s is called a word signature. The signature of a text block or object can be obtained by superimposing (logical OR operation) all its constituent signatures (i.e) word signatures for text block and attributes' signatures for object. The set of all signatures form a signature file. An example of Superimposed Coding and a sample query evaluation is given below.

Information Retrieval	0010 0100 0100 0001
Block Signature	0110 0101

Sample queries

Matching query		
Keyword = Information	0010 0100	
Query descriptor	0010 0100	
Block signature matches		(Actual Drop)
False Match query		
Keyword = Coding	0010 0001	
Query descriptor	0010 0001	
Block signature matches but keyword does not		(False Drop)
Non-matching query		
Keyword = Information	0010 0100	
Keyword = Science	0000 0110	
Query descriptor	0010 0110	
Block signature does not match		

2.2 Applications of signatures

Signatures are applied mainly in database access methods useful for text retrieval such as Full text scanning, Inversion, Clustering, Multi attribute retrieval methods like hashing and signature files. Such applications are discussed in [3,5,21]. Here the documents are stored sequentially in the "Text File".

Signatures which are abstractions of the documents are stored in the "Signature File". The latter serves as a filter on retrieval. It helps in discarding a large number of non-qualifying documents. Signatures have been applied in areas rich in text documents like telephone directory [1], office systems [3,4,6], Optical and Magnetic disk access [8], Data base Management system and Library automation.

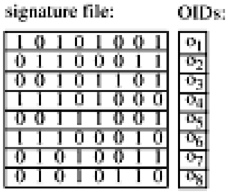
Other applications include
Access method for documents
Indexing method for large text file [17, 18].
Access method for formatted data[1].
To speed up searching in editor[7].
To compress a vocabulary[7].
For a spelling checking program[7].
In differential file[7].

2.3 Physical representations of signature files

This section discusses briefly the various techniques used to represent signature files. We follow the lead of [30] for figures in this section.

2.3.1 Sequential Signature File (SSF)

The signatures are sequentially stored in a file called SSF as in Fig. 1. In single level signature methods every signature must be accessed and tested. Since signatures are abstractions of original data with smaller size, the method is faster than sequential scan of objects themselves. This method is easy to implement and requires low storage space and low update cost. The disadvantage is that more the number of objects exist, the more is the time spent on scanning signature file [8,31]. Therefore it is generally slow in retrieval. To support faster access multilevel signature methods are suggested.

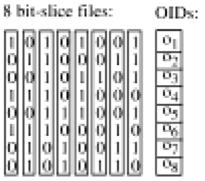


signature file:								OIDs:
1	0	1	0	1	0	0	1	o_1
0	1	1	0	0	0	1	1	o_2
0	0	1	0	1	1	0	1	o_3
1	1	1	0	1	0	0	0	o_4
0	0	1	1	1	0	0	1	o_5
1	1	1	0	0	0	1	0	o_6
0	1	0	1	0	0	1	1	o_7
0	1	0	1	0	1	1	0	o_8

Figure 1: A typical Sequential signature file.

2.3.2 Bit-Sliced Signature File (BSSF)

BSSF stores signatures in a column-wise manner as illustrated in Fig. 2. Thus F bit-slice files one for each bit position of the set signatures are used. In retrieval only a part of the F bit-slice files have to be scanned and hence the search cost is lower than SSF. However update cost is higher. This is because a new signature insertion requires about F disk accesses one for each Bit-slice file [8,31].



8 bit-slice files:								OIDs:
1	0	1	0	1	0	0	1	o_1
0	1	1	0	0	0	1	1	o_2
0	0	1	0	1	1	0	1	o_3
1	1	1	0	1	0	0	0	o_4
0	0	1	1	1	0	0	1	o_5
1	1	1	0	0	0	1	0	o_6
0	1	0	1	0	0	1	1	o_7
0	1	0	1	0	1	1	0	o_8

Figure 2: A typical Bit-Sliced Signature File.

2.3.3 Compressed Bit-Sliced Signature File (CBSSF)

By choosing a proper hashing function for signature extraction the number of 1s is forced to be one. Here, the signature length should be increased to maintain the false drop probability at minimum. This creates a sparse matrix which is easy to compress [8, 23]. A simple way to compress is to replace each 1 with its corresponding physical address.

In Fig. 3 the hash table has a list of pointers pointing to the heads of linked list [8].

For example assume that the word *Text* has its first bit set to 1 and it appears at the 50th byte of text file then searching the first bucket list, we find the position of the word *Text*. Although this approach gives some space saving, the number of false drops will definitely be increased due to sparse signature files.

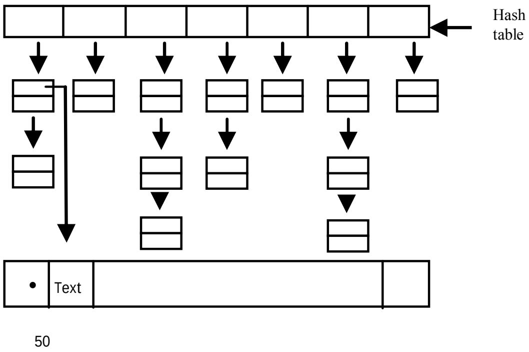


Figure 3: A typical Compressed Bit-Sliced Signature File organization.

2.3.4 S-Tree

S-Tree [24] is a B+ tree like structure [11,22] with leaf nodes containing a set of signatures with their Object Identifiers (OIDs). The internal nodes are formed by superimposing the lower level nodes as shown in Fig. 4. For example to retrieve a query signature Sq = 11000000, we search S-ree top down. From root node v1 we compare and move to v6. In the next level both v7 and v8 match and finally we end up with signatures o11, o12 in v7 and o13 in v8.

The advantage is simple tree searching way of obtaining signatures rather than searching the whole signature file.

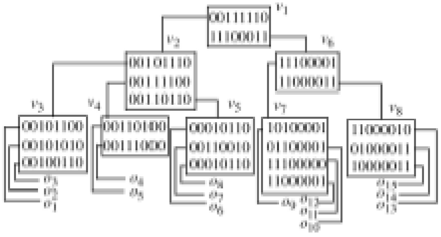


Figure 4: A typical S-tree organization.

The disadvantage is that due to superimposing, internal nodes in the upper level tend to have more weight which ultimately decreases selectivity. The S-Tree has been further improved in [13, 14]. In [13] a number of new split methods namely Linear split, Quadratic split, Cubic split and hierarchical clustering for S-tree is proposed to improve query response time. In [14] a new hybrid scheme combining linear hashing, S-tree and parametric weighted filter is used to evaluate subset-superset queries.

2.3.5 Multilevel Signature file

The structure is similar to S-Tree. However a signature at non-leaf node is formed by superimposed coding from all text blocks indexed by the subtree of which the signature is the root. Fig. 5 shows multilevel signature file for the set of signature values depicted in

4 Fig 4. Though this method improves selectivity in an n internal node, it requires more space. An improved e method for multilevel signature file is discussed in [25].

2.3.6 Signature Graph

The signature file is organized as a trie like structure [28,30]. However, the path visited in the graph to find a signature that matches a given query signature corresponds to a signature identifier which is not a continuous piece of bits, differentiating the signature graph from trie.

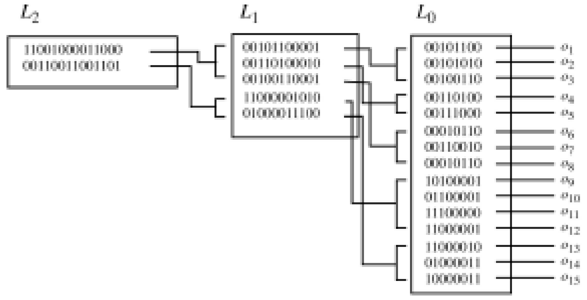


Figure 5.A: Typical Multilevel Signature file organization.

Though signatures are represented compactly, the search path length is not same for all queries. In other words the graph is not balanced. In worst case it degrades to a signature file. Fig. 6. shows the signature file and the corresponding signature graph.

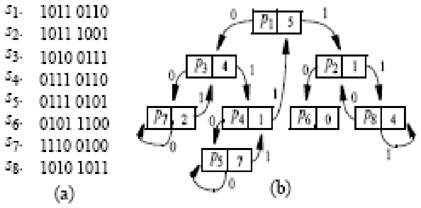


Figure 6: A typical signature file and signature graph.

2.3.7 Signature Tree

The signature tree is a binary tree like structure with nodes representing the bit positions and left and right sub tree followed for binary values 0 and 1 respectively. Each signature is identified from the root by checking the bit positions dictated by the nodes. Nevertheless for a query signature the tree is searched top to bottom according to the bit positions dictated by the nodes rather than the 1s in query signature. Also, for a match with bit 1, searching follows the right sub tree and for 0 at a node both left and right sub trees are followed.

That is for a balanced signature tree more than one path is traversed. Fig. 7. indicates a signature file and its corresponding signature tree. The thick lines in Fig 7(b) indicates the signature identifier that corresponds to S3. In the following section we discuss a quite different method called SD-tree which considers only the positions of 1s in a given signature and decluster them over a set of leaf nodes so that query response time is improved.

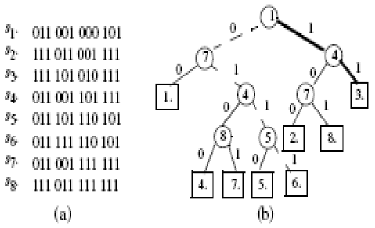


Figure 7: A typical signature file and signature tree.

3 The Structure of SD-tree

In this section we describe the structure of SD tree. There are three types of nodes:

- Internal nodes
- Leaf nodes
- Signature nodes

The internal nodes and leaf nodes are somewhat similar to the internal nodes and leaf nodes of B+ trees respectively. The internal nodes form the upper tree and leaf nodes at last but one level. The signature nodes are at the bottom level of the SD-tree. We will now explain the structure of the nodes in detail. To make discussion simple, we assume the tree order as 3 for a signature file with 8 block signatures of length 12.

3.1 Structure of Internal node

An internal node of SD-tree is illustrated in Fig. 8. Like B+ tree internal node pointers and keys alternate each other. For a tree of order 3 the internal node has two keys K1 and K2 and three pointers P1, P2, P3. These pointers are tree pointers pointing to the nodes at the lower level.

While searching, the left tree pointer is followed for values less than or equal to the node value, else right pointer is followed for values greater than the node value as in B+ tree.

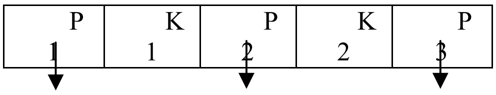


Figure 8: A typical structure of an internal node.

3.2 Structure of a leaf node

The leaf nodes appear in the last but one level of the SD tree. Like B+ tree all the key values appear in ascending order of their values in the leaf nodes and are connected to promote sequential search. But unlike B+ tree in SD-tree each value is followed by a signature node instead of data pointer. This is depicted in Fig 2. Pointers P1 and P2 point to the corresponding signature nodes for K1, K2; P3 is the sequential pointer to next leaf node and P4 is the backward pointer from next leaf node.

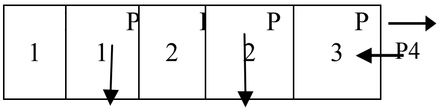


Figure 9: A typical leaf node entry.

3.3 Structure of signature node

The structure of a signature node is shown in Fig. 10. The signature node for Ki has 2^{i-1} binary combinations denoting the possible prefixes. When a signature S_u with 1 in the i th position is to be inserted the intermediate prefix formed (explained in section 4) is compared with the binary combinations in the signature node at Ki and u is inserted in the list.

B1	Signatures having prefix B1
B2	Signatures having prefix B2
...	...
Bn	Signatures having prefix Bn

Figure 10: A typical signature node entry.

3.4 Overall structure of SD-tree

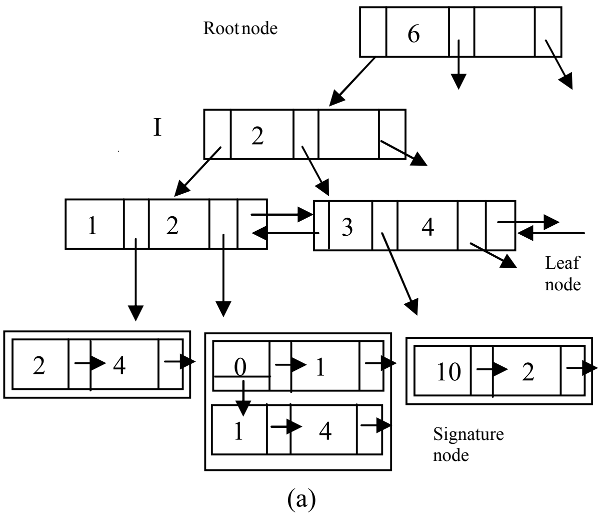
Consider the partially filled SD-tree shown in Fig 11(a) for discussion. The tree has been constructed for signature length F = 12. It is obvious from Fig. 11(a) tree of order 3 has height 2. Consider the signature file in Fig 11(b). To insert signature S1 with first occurrence of 1 at position 2, access the leaf node with value 2, follow its signature node and write the signature value as 1 (for S1) with the prefix 0 (for bit 1). In the same way S2 is inserted in signature node 1 with no prefix and signature node 3 with prefix 10 (for bits 1 and 2). S4 will be inserted at signature node 1 with no prefix and signature node 2 with prefix 1.

While inserting a signature there are two possible options to move to next 1s position in the leaf node. One is via the sequential pointer between leaf nodes and the other is the top-down traversal of tree from root. To ensure optimal search path the threshold (Th) is fixed at h+1 (h being the tree height plus one more level for accessing signature nodes). As long as the number of sequential pointers traversed in leaf nodes is within the specified Th value, follow the sequential pointers.

This is calculated by finding the difference (d) between two consecutive 1s in a signature divided by the number of entries per leaf node. This is given by the condition

$$\frac{d}{p-1} \le Th \quad \text{, where p is the order of the tree.}$$

The division here is integer division. When the above condition is not true, a new tree access is initiated from root. Similarly to promote optimal query processing the query signature value Sq is taken in decimal form. The occurrence of the last one (in the least significant position) is found by performing D Mod 2 operation. The remaining binary prefix is formed in the process of decimal–to-binary conversion of D. The procedures for tree maintenance are explained in the following section.



(a)

S1 : 010……….
S2 : 101……….
S3 : 000……….
S4 : 110……….

(b)

Figure 11: Overall structure of SD-tree.

To minimize the number of false drops the database size is selected according to [4, 27, 30] as

$$F \ln_2 = mD \qquad \ldots\ldots\ldots\ldots\ldots\ldots\ldots (1)$$

where F is the signature length, m the number of set bits and D the average data block size.

4 Tree Search and Updates

This section lists the algorithms for signature insert, delete and search operations on SD-tree. A global flag F is set to 0 indicating the search path from the root of the tree by default. In the procedure after the first 1's insertion, depending on the d value F may be set to 1.

4.1 Insertion

The algorithm for signature insertion is outlined in this section. The call to procedure New(node) returns a new link in the signature node.

```
Insert ($S_u$)
    Input : The signature to insert $S_u$ ;
        1. Let $i_1, i_2, \ldots.. i_n$ be the positions of 1 in $S_u$ ;
           F ← 0;  Th = h+1;  // for tree of order p
        2. Move $i_2$ to $i_n$ to queue Q.  B = NULL;
        3. If ($i_1$ = 1) then begin
        Access leaf node $i_1$;  // from root
        New(node);
        insert u;
        B = strcat ( B, '1'); // to denote bit 1 position
          end
            else begin
              for k = 1 to $i_1 - 1$ do
              B = strcat ( B, '0');
            Access leaf node $i_1$;   // from root
            New(node); write (B);
            insert u ;
            B = strcat ( B, ‘1’);
          end
          f = $i_1$; // store the current bit position
          F ← 1; // enable sequential search in leaf nodes
    4. While Q not empty do
        begin
            read x from Q;
            if (x = f+1) then
            begin
                Access leaf node x;  // via leaf node pointers
                If (not(B)) then  New(node); // create node
              Write(B);
              Write u @ prefix B;
            end
          else begin
                d = x – f;
                If (d/(p-1)) > Th then  F ← 0;
                for k = f +1 to x-1 do
                    B = strcat ( B, ‘0’);
                Access leaf node x;
                If (not(B)) then  New(node);
              Write(B);
               Write u @ prefix B
                 end
               B = strcat (B, ‘1’);
               f = x;
          end.  // until queue is empty
```

4.2 Searching

The following algorithm outlines the steps to search for signatures matching a given query signature Sq. In the procedure F ← 0 always and the algorithm lands up directly in the signature node corresponding to last 1 from root. The match here is the exact match and the optimal signature length selected in equation (1) minimizes the false drops.

Search(S_q)

Input : The (query) signature to search as decimal D.
Output : The list of signatures matching the given signature.

```
1  Repeat  i = D mod 2; D = D div 2; until i = 1
2  Let n be the corresponding bit position of i.
3  Let B = toBinary(D);  // convert D to binary
4  Access leaf node n  // from root node
5  Search for B.
6  If Found() then read and output the list of signatures.
```

4.3 Deletion

The algorithm to delete a signature from SD-tree is described below.

Delete (S_u)

Input : S_u, the signature to delete.

1. Let i_1, i_2, ….. i_n be the positions of 1 in S_u.
2. For each i_k ($1 \leq k \leq n$) form prefix B as in Insert (S_u).
3. Access the leaf node and follow the signature node;
4. Access prefix B and search for u.
5. If present, delete it .
6. Repeat steps (2) through (5) for all i_ks.

5 A sample validation model

This section discusses the evaluation of sample Object-Oriented queries on a hypothetical object base. Fig 12 shows the class diagrams in UML notation [15,19]. The classes and their relationships are listed in Table 1.

S. No	Class 1	Class 2	Relationship
1.	University	Dept	Composition
2.	University	Student	Aggregation
3.	Student	Programme	Association
4.	Dept	Instructor	Association
5.	Student	Male	Generalization
6.	Student	Female	Generalization
7.	Programme	Subject	Association

Table 1: Class relationships

The sample hashing outputs of attributes values are listed in Table 2.

Dept-name	Pgm-name
Mathematics – 1010 0000	M.E – 0001 0100
Computerscience–0100 1000	M.Sc (S.E) – 1000 1000
Physics – 0001 0010	M.Sc (Mat) – 0100 1000
	M.sc (Phy) – 0011 0000

Inst-name	Stud-name	Male
John – 1000 0010	David – 0100 0010	Sex – 0010000
Adams – 1000 1000	Elena – 1001 0000	Female
James – 0000 1001	Maria – 0100 0001	Sex –00001001
Janes – 0110 0000	Peter – 0010 0001	
	Grace – 0000 1100	
	Antony – 0000 0011	

Table 2: Sample attributes

The attributes' hash coded values are superimposed to produce object's signature. The set of all object signatures of a class correspond to the signature file. The signature files created for various attribute combinations of objects are listed in Table 3.

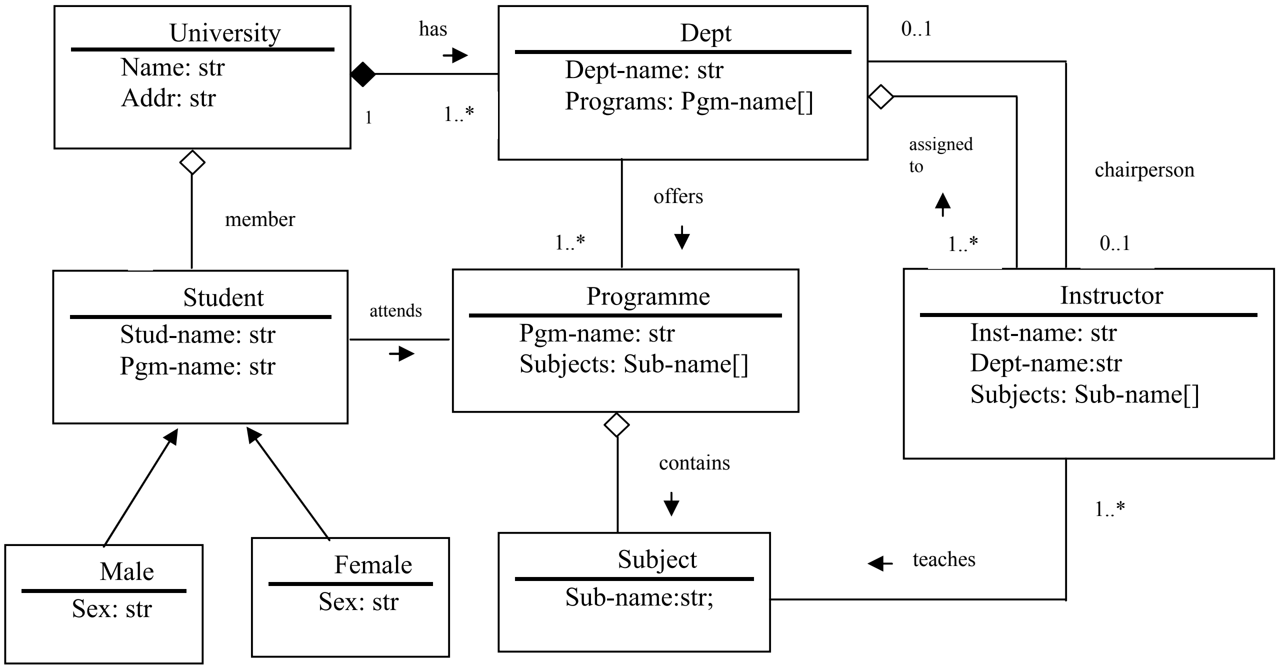


Table 3: Signature files of classes

Class : Subject

Sub-name

1. Software engg – 0100 0001
2. Comp. applns – 0010 1000
3. Comp. engg – 0100 0100
4. Applied Maths – 1000 0001
5. Calculus – 0010 0100
6. Nuclear physics – 0101 0000

Class : Male

Stud-name	Pgm-name	Sex	Object signature
1. David	M.Sc (S.E)	Male	1101 1010
2. Peter	M.E	Male	1011 0101
3. Antony	M.E	Male	1001 0111

Class : Female

Stud-name	Pgm-name	Sex	Obj signature
1. Elena	M.Sc (S.E)	Female	1001 1001
2. Maria	M.Sc (S.E)	Female	1100 1001
3. Grace	M.Sc (Phy)	Female	0011 1101

Class : Dept

Dept-name	Programs	Obj signature
1.Computer science	M.E,M.Sc (S.E)	1101 1100
2. Mathematics	M.Sc (Mat)	1110 1000
3. Physics	M.Sc(Phy)	0011 0010

Figure 12: Sample Object schema.

Class : Programme

Pgm-name	Subjects	Obj. signature
1. M.E	Comp.applns, Applied Maths	1011 1101
2. M.Sc (S.E)	Software engg, Comp. engg	1100 1101
3. M.Sc(Phy)	Comp. applns, Nuclear physics	0111 1000

Class : Instructor

Inst-name	Dept-name	Subjects	Obj signature
1. John	Comp.sc	Software engg, Comp. applns	1110 1011
2. Adams	Mathematics	Applied Mathematics, calculus	1010 1101
3. James	Comp.sc	Software engg	0100 1001
4. Janes	Physics	Nuclear physics	0111 0010

Class : Student

Stud-name	Pgm-name	Obj signature
1. David	M.Sc(S.E)	1100 1010
2. Peter	M.E	0011 0101
3. Antony	M.E	0001 0111
4. Elena	M.Sc(S.E)	1001 1000
5. Maria	M.Sc(S.E)	1100 1001
6. Grace	M.Sc(Phy)	0011 1100

1.	List of students doing a given programme	Pgm-name = M.Sc(S.E)	Student	1000 1000	1100 1010 (David) 1001 1000 (Elena) 1100 1001 (Maria)
2.	Instructors handling a given subject	M.Sc(S.E) in subjects	Instructor	0100 0001	1110 1011 (John) 0100 1001 (James)
3.	Dept(s) offering a given subject	1. Comp. applns in subjects	Programme	0010 1000	1011 1101 (M.E) 0111 1000 (M.ScPhy)
		2. Pgm-name = 00010100, 00110000	Dept		1101 1100 (Comp.sc) 0011 0010 (Phy)
3.	All instructors of a given dept	Dept-name = Comp.sc	Instructor	0100 1000	1110 1011 (John) 0100 1001 (James)
4.	Female students attending a given programme	Pgm-name = M.Sc(S.E)	Female	1000 1000	1001 1001 (Elena) 1100 1001 (Maria)

Table 4: Sample queries

6 Experimental results

To validate the proposed structure we implement SD-tree in Java and for every test run the tree is constructed statically before signature insertion. The parameters considered in the experiments' data sets are Signature length (F), Signature weight (m) and signature weight distribution (swd).

The experiments were carried out in a standalone system with Intel Pentium IV processor. The main memory size is 512 MB and the hard disk capacity is 80 GB.

6.1 Signature tree Versus SD-tree

In this section the parameters which are generally considered in the analysis of indexing structures like time and space complexities are reported. We compare the results of Y. Chen's Signature tree [30] with that of the SD-tree. The observed results are listed in Table 5.

6.2 Time Complexity

Like in other signature applications we use the response time as the performance measure [23]. The time complexity of the insert algorithm initially is the sum of time taken to construct the B+ tree of order p and the time taken for inserting. Here B+ tree is constructed with values 1,2, ….. , F where F is the length of the signature for a given dataset. Hence compared to the use of B+ tree as index structure for large datasets the value F is small which reduces the time taken for SD-tree construction considerably. In algorithm 4.1 the time complexity for insertion is bounded by O(nm) where n is the number of signatures in the file and m is the number of 1s in the given signature as against the O(nF) in the signature tree insertion[30], n being the number of signatures in the file and F is the full length of signature including 0's and 1's. Since deletion follows similar steps as insertion the time complexity is same for both. Another desirable characteristic of SD-tree is that for higher F values, by varying p, the value of h, height of the tree can be kept small to promote faster search. It is observed that the time taken for tree construction when F = 10 and p = 3 is 2015421 nano seconds.

Parameter	Signature tree	SD-tree	Inference
Time complexity	O(nF)	O(nm)	m < F; Faster insertion
Tree height	$O(\log_2 n)$	$O(\log_p (F/(p-1)))$	p>2 ; Shorter tree
Search cost	$O(\lambda.\log_2 n)$	$O(\log_p F+a)$	F < n; Cost < Sig. tree
Space complexity	$n\log_2 F + 2\sum 2^i (i+1)$ i=0 to k	O(F(a+f))	< Sig. tree when k>F

Table 5: Signature tree Vs SD-tree

In Table 5,
n - Number of signatures in signature file
F – Length of signature
m - Number of set bits
p – Order of SD-tree
λ – Number of path traversed in query searching
a – Average no. of signatures / signature node
k – log 2 n
f - Average no. of prefix values / signature node

Similarly search time is the sum of time taken to access the leaf node (Tl) and signature node search time (Tsi). This is given by

$$Ts = Tl + Tsi$$

Here, Tl is constant for all leaf nodes for a dynamic balanced structure like SD- tree and Tsi is directly dependent on the i value and the number of signatures inserted in the signature node. In the worst case the search time is bounded by $O(Tl + 2^{i-1})$.

To analyze the query response time the signature weight distribution was fixed as 100% , 70%, 50% and 30% for signature lengths 10 and 30. The values are plotted in Fig. 13. through Fig. 16. The weight of the signature was biased in upper byte(U), lower byte(L) or uniformly distributed(M) and time values noted.

For 100% swd the insertion and search time is a constant of the signature weight bias. This is depicted in Fig. 13. In the same way the swd was fixed at 70, 50 and 30 and the observed values are plotted in Fig. 14, Fig. 15 and Fig. 16 respectively. The query response time between two consecutive queries is minimized by following the backward pointers in leaf nodes when the following condition is true. That is,

$$\frac{d_{i,j}}{p-1} \leq Th$$

where $d_{i,j}$ is the difference in Sqi's last 1's position and Sqj's first 1's position.

All the graphs show that the time taken for signature insertion grows linearly with the values of swd, F and the weight bias. Insertion time increases in upper nodes due to the complexity of the circuit in creating prefixes.

SD-tree maintenance and space overhead

SD-tree maintenance is quite simple that the tree is not subject to extensive node split or merge. This is because the insertions and deletions do not affect the node values or the height of the tree. Operations are reflected only in the signature node. In the experiments the binary prefix pattern nodes are created dynamically. The space consumption for insertion of a signature number at a signature node depends on the binary prefix length (l), the pointer size (p) and the space for writing the signature number(n). The prefix is created anew each time if it does not exist. Hence, the space complexity to insert with prefix for a signature of weight (w) is given by O[w((l+2p)+(n+p))]. Similarly the space complexity of a signature for which prefix already exists is bounded by O[w(n+p)].

Fig. 17 shows the space overhead of SD-tree. The tree is created statically. Signature nodes are created dynamically and are of fixed size. It is clear from the graph that the space consumption increases linearly with F and w. It is obvious that for all combinations of values the query search time for SD-tree is lesser than signature insertion time.

For swd of 70%, 50% and 30% the signature weight was biased in lower byte, upper byte or uniformly distributed and values noted. All the outputs clearly indicate that the time taken for signature insertion and query response is slightly higher in upper levels.

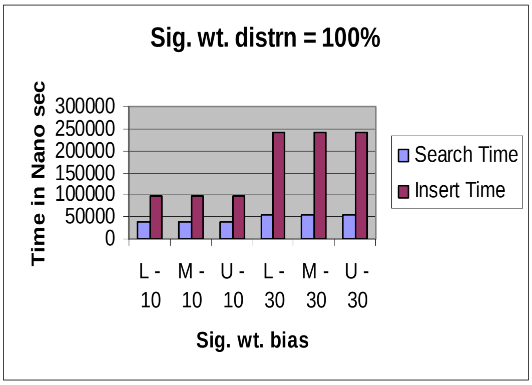


Figure 13: 100% swd.

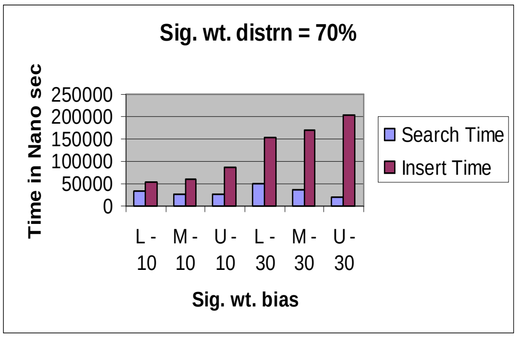


Figure 14: 70% swd.

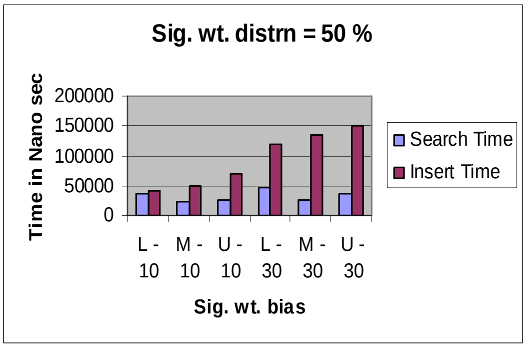


Figure 15: 50% swd.

Nevertheless the query response time is lesser than signature insertion time. As the structure complexity increases in signature nodes in upper levels the swd was analyzed for both ends separately.

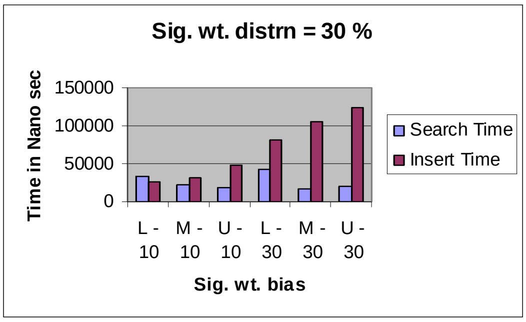


Figure 16: 30% swd.

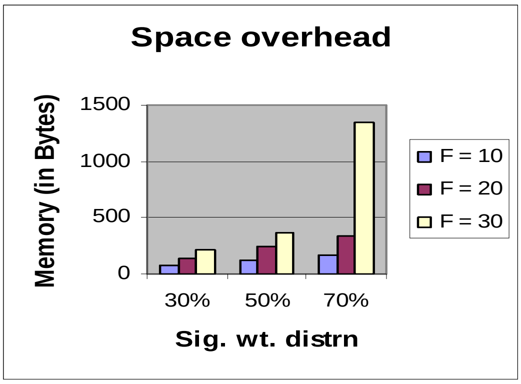


Figure 17: Space overhead of SD-tree.

7 Conclusion and research directions

In this paper we presented a novel way to represent signatures in a B+ tree like structure called SD-tree and analyzed the performance for query response time.

By varying the signature length and distribution of 1s in the signature the query response time was noted and results plotted. It is clear from the graphs that considerable search time is saved.

The space overhead in SD-tree may be higher due to the presence of binary prefixes in higher order signature nodes, but the flexibility provided by the SD-tree outweighs all besides simple maintenance and faster query retrieval time.

The work is proposed to extend in the following directions. The synthetic data sets are to be replaced with, run and verified on a real Object Oriented Data Base system. Another direction is when the signature weight is more than 50%, use 0s so that number of signature nodes accessed for insertion and search is optimal. Also the structure can be modified to support point and range queries in Object Oriented Data Base system.

References

[1] Charles S. Roberts, (1979) *Partial-Match Retrieval via the Method of Superimposed Codes*, Proc. of IEEE, Vol. 67, No. 12, pp 1624 – 1642.

[2] Chris Faloutsos, Stavros Christodoulakis, (1984) *Signature Files: An Access Method for Documents and Its Analytical Performance Evaluation*, ACM Trans on Office Information Systems, Vol.2, No. 4, pp 267 – 288.

[3] Christos Faloutsos, (1985) Access *Methods for Text*", ACM Computing surveys, Vol. 17, No. 1, pp 49 – 74.

[4] Chris Faloutsos, (1985) *Signature files: Design and performance comparison of some signature extraction methods*" Proc. of ACM SIGMOD, pp 63 – 82.

[5] Christos Faloutsos, Stavros Christodoulakis,(1985) *Design of a Signature File Method that Accounts for Non-Uniform Occurrence and Query Frequencies*, Proc. of VLDB, pp 165 – 170.

[6] Christos Faloutsos, Stavros Christodoulakis,(1987) *Description and Performance Analysis of Signature File Methods for Office Filing*, ACM Trans on Office Information Systems, Vol. 5, No. 3, pp 237 – 257.

[7] Christos Faloutsos, Stavros Christodoulakis, (1987) *Optimal Signature Extraction and Information Loss*, ACM Trans. On Database Systems, Vol. 12, No. 3, pp 395 – 428.

[8] Christos Faloutsos, Raphael Chan, (1988) *Fast Text Access Methods for Optical and Large Magnetic Disks: Designs and Performance Comparison*, Proc. of VLDB, pp 280 – 293.

[9] Chris Faloutsos, Raymond Lee, Catherine Plaisant, Ben Shneiderman, (1990)

[10] *Incorporating String Search in a Hypertext System: User Interface and Signature File Design Issues*, Hypermedia, Vol. 2, No. 3, pp 183 – 200.

[11] D. Dervos, Y. Manolopoulous, P. Linardis, (1998) *Comparison of Signature File Methods with Superimposed Coding*, J. Information Processing , 65, pp 101 – 106.

[12] Douglas Comer, (1979) *The Ubiquitos B-Tree*, Computing Surveys, Vol. 11, No. 2, pp121 – 137.

[13] Dik Lun Lee, Young Man Kim, Gaurav Patel, (1995) *Efficient Signature File Methods for Text Retrieval*, IEEE TKDE, Vol. 7, No. 3, pp 423 – 435.

[14] Eleni Tousidou, Alex Nanopoulos, Yannis Manolopoulos, (2000) *Improved Methods for Signature-Tree Construction*, The Computer Journal, Vol. 43, No. 4, pp 301 – 314.

[15] Eleni Tousidou, Panayiotis Bozanis, Yannis Manolopoulos, (2002) *Signature-based structures for objects with set-valued attributes*, Information Systems, Vol. 27, No. 2, pp 93 – 121.

[16] Grady Booch, James Rambaugh, Ivar Jacobson, (2003) *The Unified Modeling Language User Guide*, Pearson Education Pte Ltd.

[17] Hwan-Seung Yong, Sukho Lee, Hyoung-Joo Kim, (1994) *Applying Signatures for Forward Traversal Query Processing in Object-Oriented Databases*, Proc. 10th Intl. conf. Data Engg, pp 518 – 525.

[18] Justin Zobel, Alistair Moffat, Kotagiri Ramamohanarao, (1988) *Inverted Files Versus Signature Files for Text Indexing*, ACM Trans. On Database systems, Vol. 23, No. 4, pp 453 – 490.

[19] A. Kent, R. Sacks-Davis, K. Ramamohanarao, (1990) *A Signature File Scheme Based on Multiple Organizations for Indexing Very Large Text Databases,* J. Am. Soc. Information Science, 1990, Vol. 41, No. 7, pp 508 – 534.

[20] Martin Fowler, Kendall Scott, (2003) *UML Distilled, A brief guide to the standard Object Modeling Language*, II edition, Pearson Education Pte Ltd.

[21] Paolo Ciaccia, Paolo tiberio, Pavel Zezula, (1996) *Declustering of Key-Based Partitioned Signature*

Files, ACM Trans. On Database Systems, Vol. 21, No. 3, pp 295 – 338.

[22] Per-Ake Larson, (1984) *A Method for Speeding up Text Retrieval*, ACM SIGMIS, Database Winter, Vol.15, No. 2, pp 19 – 23.

[23] Rudolf Bayer, Karl Unterauer, (1977) *Prefix B-Trees*, ACM Trans. On Database Systems,Vol. 2, No. 1, pp 11 – 26.

[24] 23. Seyit Kocberber, Fazli Can, (1999) *Compressed Multi-Framed Signature Files: An Index Structure for Fast Information Retrieval*, Proc. ACM Symp. Applied Computing, pp 221 – 226.

[25] 24. Uwe Deppisch, (1986*) S-Tree: A Dynamic Balanced Signature Index for Office Retrieval*, Proc. ACM SIGIR conf, pp 77 – 87.

[26] Walter W. Chang, Hans J. Schek, (1989*) A Signature Access Method for the Starburst Database System*, Proc. of VLDB, 145 – 153.

[27] Wang-chien Lee, Dik L. Lee, (1992) *Signature File Methods for Indexing Object-Oriented Database Systems*, Proc. 2nd Intl. Comp. Sc. Conf, pp 616 – 622.

[28] Yangjun Chen, (2002) *Signature Files and Signature Trees*, Information Processing Letters, 82, pp 213 -221.

[29] Yangjun Chen, Yibin Chen, (2004) *Signature File Hierarchies and Signature Graphs: a New Index Method for Object-Oriented Databases*, Proc. of ACM Symp. on Applied Computing, pp 724 – 728.

[30] Yangjun Chen, (2005) *On the Signature Trees and Balanced Signature Trees*, Proc. of ICDE, pp 742 – 753.

[31] Yangjun Chen, Yibin Chen, (2006) *On the Signature Tree Construction and Analysis*, IEEE TKDE ,Vol.18,No.9, pp 1207 – 1224.

[32] Yoshiharu Ishiwaka, Hiroyuki Kitagawa, Nobuo Ohbo, (1993) *Evaluation of Signature Files as Set Access Facilities in OODBs*, Proc. of ACM SIGMOD, pp 247 – 256.

Improving Design Pattern Adoption with an Ontology-Based Repository

Luka Pavlič, Marjan Heričko, Vili Podgorelec and Ivan Rozman
Institute of Informatics, University of Maribor, FERI
Smetanova ulica 17, SI-2000 Maribor
Slovenia
E-mail: luka.pavlic@uni-mb.si

Keywords: design patterns, semantic web, ontologies, design pattern repository

Received: February 7, 2008

In software engineering, an efficient approach towards reuse has become a crucial success factor. Conceptual simple high level approaches to reuse are the most appropriate for performing it in a useful manner. Design patterns are reliable and an effective high level approach that enables developers to produce high quality software in less time. Unfortunately, the rapidly growing number of design patterns has not yet been adequately supported by efficient search and management tools, making the patterns uninviting for a large part of the software development community. In this way, the issue of managing and selecting design patters in a straight-forward way has become the main challenge.

In this paper, we propose a possible solution for the improvement of design pattern adoption and present a platform that should give design patterns some new and long-overdue momentum. Using our proposed technique for formal design pattern specifications, we have developed an experimental prototype of a new design pattern repository based on semantic web technologies. A new Ontology-Based Design Pattern Repository (OBDPR) has been developed that can also be used as a platform for introducing advanced services. Some fundamental services – searching, design pattern proposing, verification and training services – have already been developed and many others are proposed. Based on the conducted experiments, it is our strong belief that the proposed approach together with the platform's potential -- can significantly contribute to the improvement of design pattern adoption.

Povzetek: Za ponovno uporabo programske opreme je razvita nova metoda z uporabo ontologije in repozitorija.

1 Introduction

Software patterns offer the possibility of achieving reuse in the area of software engineering. In software engineering, several levels of reuse are established. The reuse of concrete software elements such as functions, classes and components have already been well established and practiced on a daily basis. However, if we observe reuse at higher levels of abstraction, i.e. software patterns, reuse is still not practiced on a daily basis.

A pattern is a form of knowledge that is used to capture a recurring successful practice [10]. Basically, a pattern is an idea that has been used in a practical context and probably will be useful in others [24]. As such, software patterns delineate the best practices for solving recurring software design problems and are a proven way of building high quality software [5]. They capture knowledge that experienced developers understand implicitly and facilitate training and knowledge transfer to new developers [17]. One survey [13] has indicated a low adoption of design patterns among practitioners – respondents estimated that no more than half of the developers and architects in their organization knew of, or used, design patterns. Therefore, bridging the gap between the pattern expert communities and the typical pattern user is critical for achieving the full benefits of design patterns [4].

The goal of this paper is to improve design-pattern adoption within the context of a typical pattern use case: a user has to select an appropriate design pattern, understand it and its consequences in detail, and also use it efficiently. To achieve this lofty goal, we will firstly explain how to introduce an appropriate design pattern presentation technique. In order to do this, we will consider several formal design pattern presentation techniques, as presented in this paper. Before we dig deeper, let us clarify the boundaries of our research. While speaking of software patterns we are addressing a whole family of patterns. There are a lot of software pattern types that have been recognized so far. Some authors have proposed a general software-pattern taxonomy [24]:

- Patterns in software analysis, which are the most abstract software patterns;
- Architectural patterns;
- Design patterns;
- Interaction patterns and
- Patterns in software implementation (also known as idioms).

We can, however, split software pattern categories further (e. g. database design patterns, ontology design patterns, communication design patterns etc). It is almost impossible to provide a formal specification for software patterns in general, since every software pattern family addresses a different set of aspects. Software pattern formal specification is, as will be discussed later, necessary in order to provide advanced automatic services. We have limited our research to object-oriented design patterns, since they are the most used and well-known software patterns [27]. This limitation is based on the observation that catalogues with only a few design patterns have clearly been shown to be problematic [5]. However, it is our belief that the approach we propose could eventually also be used with other software patterns, especially when considering the requirement that one could also formalize aspects specific to covering expert knowledge.

In our work, we do not try to formalize all possible aspects. Our goal is to provide a human and machine understandable foundation, primarily to support the design pattern selection process. We do not therefore cope with formalizing the pattern implementation or verification, for instance. Instead, we formalize object-oriented concepts, relationships and expert knowledge on design patterns.

Design patterns improve software design productivity and quality for the following reasons [22]:

- They capture previous design experiences, and make it available to other designers - designers do not need to discover solutions for every problem from scratch.
- They form a more flexible foundation for reuse, as they can be reused in many ways.
- They can be used as a tool for communication among software designers. In fact, this was the original idea of introducing design patterns.

Although design patterns could help significantly in producing high-quality software, developers are continuing to experience more and more difficulties e. g. when finding patterns to match their design problems. It seems that managing and searching facilities are not catching the growing number of design patterns. In this paper, we will also address this challenge. Since we have had many positive experiences in initiating developers to use design patterns, we have also decided to develop an integral web-based platform, primarily to help select design patterns. The platform (Ontology-Based Design Pattern Repository – OBDPR) presented in this paper is a platform and, as such, provides the basis for automatic and intelligent services to be built on top of formally presented design pattern knowledge. We have developed a set of services, also described in this paper, on top of OBDPR:

- design pattern searching service,
- design pattern proposing service, which can also be used as a design pattern suitability verification service,
- training service.

Based on formal design pattern representation, we have introduced capabilities known from the artificial intelligence area into OBDPR and its services. In this paper, we will present the initial experiment performed, which should demonstrate the usefulness of our approach. Moreover, we are planning to perform additional rigorous experiments to indicate if and how much our platform helps software engineers, especially inexperienced ones. We assume that positive experiences with students could also be achieved with full-time developers.

The structure of this paper is as follows: In the “Related work” chapter we present known techniques in formalizing design patterns and their possible and concrete applications. Based on this, we present our own method for formal design pattern representation in chapter three. In that chapter, we also discuss why and how to use ontologies while addressing challenges related to selecting, understanding and using design patterns. Chapter four gives a detailed insight into the conceptual and technical background of OBDPR. The platform’s functionality and additional services are also described. The results of introducing the platform are presented in chapter five. The findings of the initial experiment, where users were exposed to solving design problems with and without our tools, are also presented. Chapter six shows some future trends in our research activities. The final chapter summarizes the most important points of this paper and concludes it.

2 Related work

Since 1994, when design patterns were introduced, many different approaches have been used for documentation purposes. In general, there are three main categories for descriptions:

- informal representations,
- semiformal representations based on graphical notations such as UML and
- various formal representations, which also include notations using semantic web technologies.

Design patterns are traditionally represented by informal, loosely structured documents. These documents are in a canonical form, which consists of a series of fields (name, intent, applicability, structure, participants, consequences, implementation etc), defined by informal descriptions. They help developers understand patterns, but there are glaringly obvious issues regarding advanced knowledge management possibilities. We can also find several semiformal representations, most of them are based on UML [6, 8, 3, 12]. These representations are efficient for a basic understanding of patterns since they cover their structural elements. They are strongly supported by tools, which enable developers to include design patterns in their solutions in a straightforward way. They are successful at capturing structures (static image, usually shown with class diagrams) and behavior (dynamic image, usually shown with sequence or collaboration diagrams). They do not provide information and knowledge on high level aspects such as intent, usability and consequences, on the other hand. For enabling sophisticated services on design patterns, e.g. the ones listed in the prior chapter, we need fully formal representations. The main goals of formalizing design patterns that are recognized within the community are [22]:

- Better understanding of patterns and their composition. It helps to know when and how to use patterns properly in order to take full advantage of them.
- Resolving issues regarding relationships between patterns. It is not only relevant which design patterns are used to solve a problem, it is also important in which order they are applied.
- Allow the development of tool support in activities related to patterns.

In general, when talking about tool support, researchers are currently trying to develop a formal representation of design patterns, primarily for [22]:

- searching for patterns in existing solutions,
- automatic code generation,
- formal solution validation.

A typical use case for this would be the following:

- Developers include a design pattern in their solution. The code is generated automatically. Tool support mostly includes the design pattern in UML diagrams. Other languages are also supported in some tools (e. g. DPML, RSL, RBML, LePUS – see [22]).
- After further development the solution is completed.
- Testers can use tools for formal testing based on design patterns. Tools can find semantic errors in a syntactically perfect solution. Improvements can also be proposed. Since this functionality can really demonstrate the tool's ability to do some inference, it is supported by almost all tools based on languages that enable code X-ray and inference (see Table 1). PEC (Pattern Enforcing Compiler) goes even further – it includes design patterns in the solution at compile time in order to avoid some errors.

There have been several attempts at introducing formal representations in the design patterns area. Some of them are based on pure mathematics, such as first-order logic, temporal logic, object-calculus, ρ-calculus and others [22]. On the other hand, some authors [7][8] are trying to formalize design patterns and keep them understandable for humans at the same time. It is the idea, similar to the semantic web (to keep data semantically understandable both to human and machine). Their representation is mostly supported with ontologies. It is used primarily to describe the structure of source code, which is done according to particular design pattern. One of those used in the "Web of Patterns" (WoP) project [1] has addressed the area of describing knowledge on design patterns.

As stated above, the authors [22] propose several tools; some of them are available for production environments. However, one could also imagine other tools supporting activities regarding design patterns. For instance: before we introduce a design pattern to our solution, how to select an appropriate one? If we have an idea of using a design pattern, one can imagine if the selected pattern would do the desired job. One could also speculate if there is any design pattern that is more suitable than the one currently used. Those were just a few ideas about how to use formal design pattern knowledge in applications. We have not found complete and proven solutions to these challenges, even if there are a few tries, based on keywords rather on design pattern knowledge (for instance [25]). To summarize, for supporting those and other activities, the authors are trying to formalize several aspects of design patterns. They can be divided into the following areas [21]:

- pattern structure (classes, methods, relationships etc),
- pattern behavior (e.g. method call sequence),
- pattern implementation,
- context prerequisites for using design patterns,
- verifying design and implementation based on patterns,
- pattern compositions.

It depends on the formal representation goal for which area of pattern will be formalized (see Table 1). For details on several methods see [22]. In Table 1, we summarize the most important techniques available today. We believe it is important if a method has only theoretical foundations or if it actually has direct tool support. In Table 1, we also show the aspects being formalized by a certain method (we limit the summary to a static and dynamic aspect of a design pattern).

Table 1: The most important formal methods for representing design patterns

Name	**Tool support?**	**Static or dynamic aspect**	**Purpose**
DPML	☑	both	MDA (Model-Driven Architecture)
RSL	☑	both	MDA, code verification
OCSID		both	
SPINE	☑	static	Code verification
SPQR		static	Code X-ray
Object-Calculus		both	
RBML	☑	both	MDA
Slam-SL	☑	both	Inference in general
ODOL (OWL)	☑	static	Pattern Repository
URN		both	Ease of use
PEC	☑	static	Enforcing compiler
LePUS	☑	static	Pattern Repository
TLA		dynamic	

FOL		static	
Prolog		static	
BPSL		both	Inference in general

After reviewing related works and the benefits of using ontologies, which will be explained later, we also decided to employ them in our work. Although there are some ontologies available (e.g. ODOL), we did not use any existing one. Having a separate ontology is not a problem, since there is a possibility of connecting ontologies in a straightforward way. As will be seen in subsequent chapters, we can benefit from combining our solution with others – especially the WoP project [1].

3 The role of ontology in OBDPR

3.1 Using semantic web technologies in OBDPR

The idea of the semantic web allows automatic, intelligent inferring of knowledge, supported by ontologies. The basic idea of the semantic web is a different organization and storage of data and, subsequently, new possibilities for using this data [19]. The barrier that prevents more advanced usage of available data is believed to be the semantic poorness of today's solutions. The vast majority of data is presented as a very simple, non-structured human readable and human understandable material. The result is an inability to make real use of the enormous amount of available "knowledge". In order to overcome these difficulties, the concept of meta-data was introduced into the core of the semantic web. Using meta-data, so called smart agents can be used to search for information by content and to infer on gathered concepts. As a foundation, there has been a lot of work done with regard to common formats for the interchange of data and the common understanding of common concepts. This allows a person to browse, understand and use data in a more straightforward way, and a machine to perform some intelligent tasks on data automatically. Furthermore, semantic web ideas can be used in an internal enterprise information system for knowledge management in a different way to introduce new intelligent services. In the semantic web, knowledge is represented as graphs, and written down in an XML-based language called RDF (Resource Description Framework) [16]. RDF deals with URIs (another W3C standard for naming resources globally unique). The advanced use of semantically annotated data can only be accomplished using ontologies in RDFS or OWL (Web Ontology Language) [14] documents. There is also a language for efficiently querying RDF-represented knowledge, SPARQL [18]. The whole stack of semantic web technologies is available and described in [20] (see Figure 1).

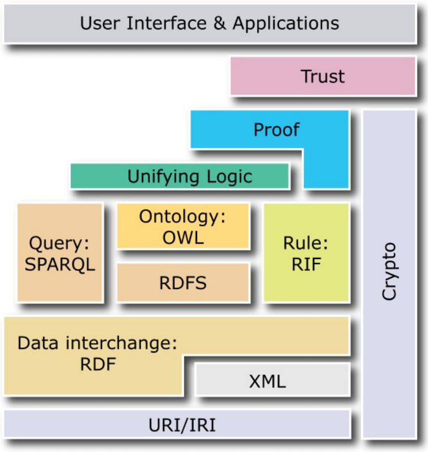


Figure 1: The semantic web technologies stack [20]

3.2 A new ontology

The semantic web allows knowledge to be expressed in a way that enables machine processing and its use in web environments by both intelligent agents and human users [23]. It is considered to provide an efficient way of presenting data, information and knowledge on the internet or in the scope of a global interconnected database. Since many semantic web technologies have reached high community consolidation and have become W3C standards (including RDF and OWL) it can also be considered a long-term platform for intelligent services based on a common knowledge base [20].

One of the enabling approaches used in the semantic web is metadata. It is supported by the concept of ontologies and has its foundation in W3C standards. Ontology describes the subject domain using the notions of concepts, instances, attributes, relations and axioms. Among others, concepts can also be organized into taxonomies whereby inheritance mechanisms can be used in ontology. Ontologies are built on description languages, such as RDF(S) and OWL, and add semantics to the model representation. Their formal, explicit and shared nature makes them an ideal object repository for catalogues.

With the presented facts, we also justify our decision to use ontologies as well as other semantic web technologies to provide a basis, not only for design pattern descriptions, but also for future intelligent services:

- Ontologies in the semantic web has its foundation in W3C standards.
- Ontology-based design pattern descriptions are computer readable and therefore suitable for automated (computer) processing.
- Transforming OWL and RDF based design pattern representations into other kinds of representations (in textual or graphical form) can be achieved easily with simple transformations.
- Enabling technologies are well established, recognized and extendable.

- They enable the exchangeability of design pattern descriptions in a straightforward way.
- The semantic web introduces technology that enables knowledge to be distributed.
- More and more OWL–enabled tools are available which can use and manipulate an ontology-enabled knowledge base.

OBDPR's underlying ontology is implemented using OWL. A core ontology fragment is shown in Figure 2. We use a hierarchical organization of pattern containers. Every pattern container may contain several pattern containers and patterns. This enables us to capture several divisions of design patterns, not only those found in fundamental literature. Every pattern can be included in several containers; the same is true for containers. Patterns themselves are connected in a more logical way by means of related, similar, composed patterns and pattern hierarchies (also mentioned as a pattern language by some authors). Not only patterns and pattern containers themselves are included in the ontology, but there are also real-world examples using patterns to give more meaning to the OBDPR user ("TestCase" class).

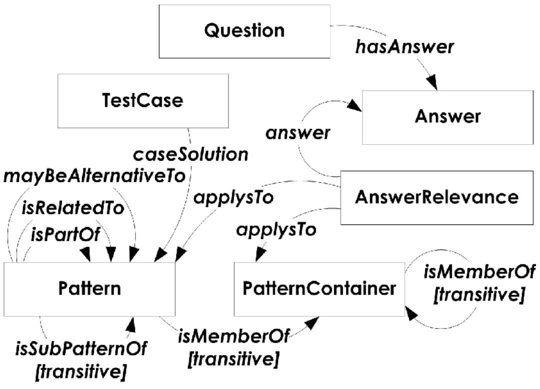


Figure 2: Core of OBDPR ontology

There are many benefits to using such ontology. Beware of transitive relations. Using a relation which has transitive properties can help significantly when dealing with design patterns and design-pattern containers. For instance: a service, built on top of OBDPR, has direct access to all members of a particular pattern container – without performing advanced searching. A pattern language (i. e. interrelated patterns) can also be presented in straightforward way. The solution is also prepared for connecting our own ontology and ODOL (ontology in WoP project [1]). We can introduce the relation theSameAs (a relation supported by OWL) between our Pattern concept and Pattern in ODOL. So we have automatic access to a formal representation not only to expert knowledge, but also the structure and behavior of a particular design pattern. Those were just a couple of strong mechanisms supported by the presented ontology.

Furthermore, the expert knowledge aspect is also supported by the presented ontology. Design pattern experts can provide experiences in question-answer pairs, which enables them to capture their implicit knowledge on design patterns. Not only experts can give experiences to tell which design pattern is used in a particular real-life situation ("Question" class), but they can also specify more possible solutions to a real-life situation ("Answer") with specified probability ("AnswerRelevance"). This value ranges from 0% to 100% and tells the user how likely it is that their particular candidate ("Pattern" or "PatternContainer") is used when the answer to a given question is confirmed as positive. Answers and possible candidates can easily be updated or added to questions at any time with the aid of a rich user-friendly web interface. For instance:
Question: *How do you want to create objects?*
Possible answer: *Separate construction of a complex object from its representation so that the same construction process can create different representations.* → You should use the *Builder pattern (100%).*
Possible answer: *Ensure that the class has only one instance.* → You should use the *Singleton pattern (100%).*
Possible answer: *Create objects without prior knowledge about their concrete classes.* → You can use several design patterns: *Prototype (33%)*, *Abstract Factory(33%)* or *Factory Method(33%).*

This knowledge can also be used by services, run on top of OBDPR in order to achieve intelligent functionalities, such as a guided question-answer dialogue for selecting patterns or verifying design decisions.

4 OBDPR – repository and platform

OBDPR is completely based on semantic web technologies. As a data store, it uses RDF. Since we do not want to rediscover all design pattern knowledge from scratch, we have also integrated knowledge found in other data sources (e. g. Wikipedia, Sun J2EE BluePrints, GoF online patterns etc). These are transformed to RDF, integrated and supported by the presented ontology (see Figure 3). Furthermore, OBDPR is not just a design pattern repository. It is a platform for building intelligent services to improve design pattern adoption. As such, it includes several functionalities:

- It holds design pattern descriptions, containers and an expert knowledge repository.
- Allows design pattern experts to annotate patterns with additional knowledge.
- Integrates knowledge on a particular design pattern from the web (Wikipedia, Sun Blueprints etc) and additional data sources.
- User-friendly transformations of raw RDF data.
- Indexes all the integrated data for supporting full text-search capabilities.
- Full access to RDF data to services built on the platform including questions and answers, which will enable intelligent services to use expert system-like proposing or validating services.

- A set of real world examples and appropriate design patterns solutions in order to enable services to be used to train users or to demonstrate the appropriate use of design patterns in real-world examples.

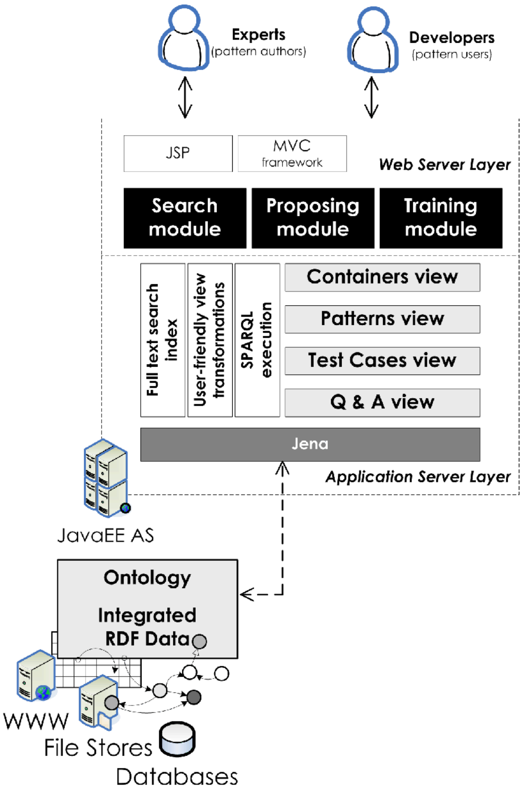


Figure 3: OBDPR architecture

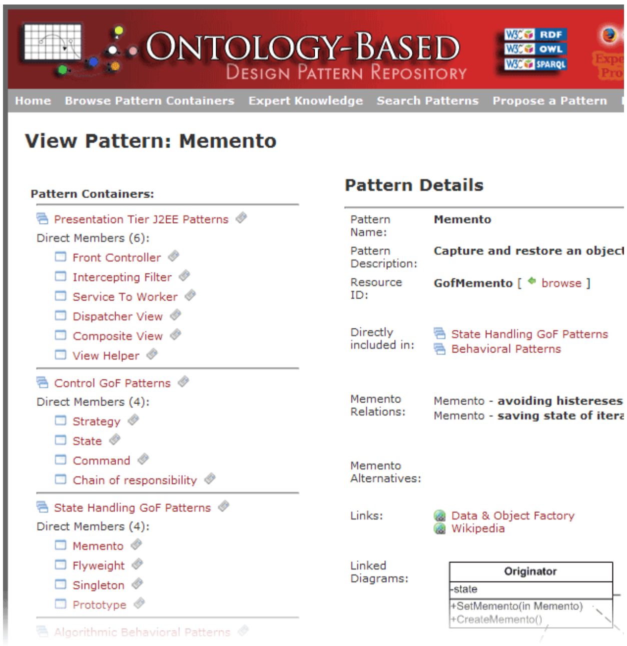
To achieve all the above-mentioned functionalities efficiently, we have also used other standard-based tools. For example, to access data written in RDF we use the Jena framework [9]. It is also exposed for services that will run on top of a platform. Figure 4 shows the user interface that uses an RDF presentation of design patterns, accessed with SPARQL queries. The data is then transformed to show a user-friendly view on design pattern container structure and selected design pattern details. This view is provided with minimum coding effort and is truly one of the most successful experiences within OBDPR.

The current OBDPR prototype implements all functionalities mentioned at the start of this chapter. It also offers services built on top of them:

- A full-text search service,
- A design pattern proposing service and
- A training service

All of them are primarily intended to help the design pattern novice.

At the moment, the OBDPR prototype includes all design patterns found in GoF [5] and J2EE [2] design pattern catalogues. It is not limited to those since it is possible to include additional patterns – even those



View Pattern: Memento

Pattern Containers:

Presentation Tier J2EE Patterns
Direct Members (6):
Front Controller
Intercepting Filter
Service To Worker
Dispatcher View
Composite View
View Helper

Control GoF Patterns
Direct Members (4):
Strategy
State
Command
Chain of responsibility

State Handling GoF Patterns
Direct Members (4):
Memento
Flyweight
Singleton
Prototype

Algorithmic Behavioral Patterns

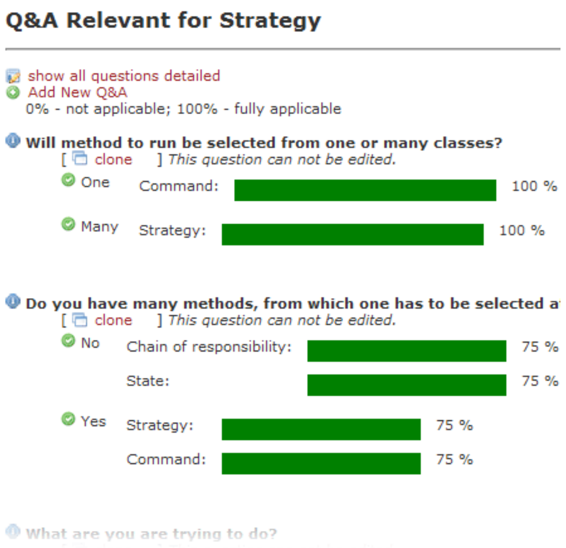
Pattern Details

Pattern Name:	Memento
Pattern Description:	Capture and restore an object
Resource ID:	GofMemento [browse]
Directly included in:	State Handling GoF Patterns Behavioral Patterns
Memento Relations:	Memento - avoiding histereses Memento - saving state of itera
Memento Alternatives:	
Links:	Data & Object Factory Wikipedia

Linked Diagrams:	Originator
	-state
	+SetMemento(in Memento)
	+CreateMemento()

Figure 4: Simple pattern view

recognized in a particular enterprise. Additional expert knowledge can also be provided. Implementing this knowledge is supported for services as well as for browsing with a user-friendly interface – using relatively simple SPARQL queries and transformations (see Figure 5).



Q&A Relevant for Strategy

show all questions detailed
Add New Q&A
0% - not applicable; 100% - fully applicable

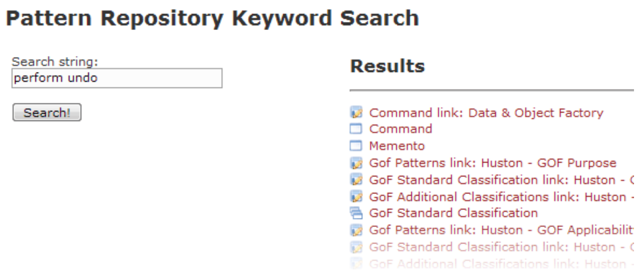
Will method to run be selected from one or many classes?
[clone] *This question can not be edited.*
One Command: 100 %
Many Strategy: 100 %

Do you have many methods, from which one has to be selected a
[clone] *This question can not be edited.*
No Chain of responsibility: 75 %
State: 75 %
Yes Strategy: 75 %
Command: 75 %

What are you are trying to do?

Figure 5: Expert knowledge view

The implementation technology for OBDPR is Java EE with a Jena [9] framework for accessing and performing core semantic operations on data and ontology. A simple user interface framework with basic functionalities like a raw and user-friendly view on the repository is prepared as previously shown. A framework is fully prepared to host additional services, which are developed in the future. To address this requirement efficiently, we have implemented an MVC design pattern into our solution. As a case, we have implemented three services on top of the platform. Since they are the core ones, they can also be used by other services. They enable us to use full text search capabilities in OBDPR (Figure 6) as well as training (Figure 5) and proposing services (figure 7). Not only is the data in OBDPR indexed for a full text search, but also data from the web, such as design-pattern-related content from Wikipedia and other design pattern related pages. The underlying ontology also improves the full text search capabilities.

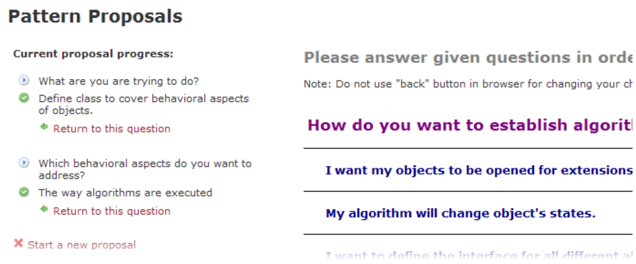

Pattern Repository Keyword Search

Search string: perform undo

Results

- Command link: Data & Object Factory
- Command
- Memento
- Gof Patterns link: Huston - GOF Purpose
- GoF Standard Classification link: Huston - (
- GoF Additional Classifications link: Huston -
- GoF Standard Classification
- Gof Patterns link: Huston - GOF Applicabilit
- GoF Standard Classification link: Huston - (
- GoF Additional Classifications link: Huston -

Figure 6: Search module


Pattern Proposals

Current proposal progress:

- What are you are trying to do?
- Define class to cover behavioral aspects of objects.
  - Return to this question
- Which behavioral aspects do you want to address?
- The way algorithms are executed
  - Return to this question

✖ Start a new proposal

Please answer given questions in orde

Note: Do not use "back" button in browser for changing your ch

How do you want to establish algorit

I want my objects to be opened for extensions

My algorithm will change object's states.

I want to define the interface for all different al

Figure 7: Proposing module

5 Preliminary experiment

We preliminary tested our approach and a platform for advising on design patterns by conducting an experiment on a group of software developers who had different levels of expertise in the field of design patterns. A series of 19 design problems were presented to each of them. Each participant tried to identify the most appropriate design pattern to solve each of the design problems. The experiment was done in four phases: in the first phase, participants had to answer a few questions concerning their development background and level of expertise. In the second phase, they were given the opportunity to solve their design problems without any assistance and/or tools. In the third phase, the platform was provided to help them solve the same set of design problems. In the end, a post-experiment survey was conducted to gather participants' opinions regarding the usefulness of the platform.

We invited 10 software developers to participate in the experiment. According to their own assessment of their design patterns expertise, five of them self-described as "good", two as "very good" and two as "excellent". Only one participant claimed poor knowledge of GoF design patterns. The level of expertise was assessed in terms of how many patterns of the GoF catalogue a participant could identify. A comparison of the results achieved in the second and third phase showed that only one participant did not make any progress when using the platform. For the rest of the participants, the platform helped identify, at a minimum, an additional 50 percent in correct solutions. Using the statistical analysis of the results (paired t-test) we determined that the difference between the number of correct solutions found both with and without the platform is statistically significant (P = 0.000706). The results of the post-experiment survey have shown that only one of the participants found the use of the platform to be less efficient than searching for an appropriate solution without the platform. A decreased standard deviation in the results showed that the efficiency of the less-experienced developers (according to design patterns) became more similar to experienced developers. At the same time, the mean value of successfully solved problems rose significantly. We can conclude that developers with less experience in the area of design patterns benefited the most from using the platform. By comparing the frequency of the correct solutions representing a particular design pattern and the frequency[1] of use for the same pattern in practice, we found that they did not directly correlate. This can be at least partly explained by the fact that the design problems presented to participants were not of equal complexity. The second possible explanation is that a higher frequency of use does not necessarily mean that the design pattern is better known or indeed easier to understand.

It should be noted that the results have to be taken with some caution, since the number of participants was quite small. Nevertheless, the experiment shows that our platform for advising is helpful to those developers who have a relatively poor knowledge of design patterns. For developers who already have solid expertise, this approach does not offer as many advantages, because the questionnaire does not follow their thought processes (mind maps). However, should a wider variety of design patterns be covered by the platform, the differentiation between experienced developers and those who lack expertise would presumably decrease.

For detailed information about experiment please see [26]. In that paper you can find detailed experiment structure, proposing service is also discussed in depth.

6 Further work

We have developed an integral web-based platform, primarily to help select design patterns. To strengthen confidence in the results, some rigorous real life experiments should be performed in addition to this initial one. They might show if and how much OBDPR helps with formally presented design patterns when adopting design patterns. As previously shown, some

[1] Frequency of use was taken from http://www.dofactory.com/Default.aspx

preliminary experiments have already been performed. Since the results were promising (design pattern adoption rose significantly) we are quite confident that we are on the right track. More rigorous experiments are planned for the future.

Besides experimenting with a repository, there is also another idea to expose OBDPR to simple software interfaces. It would not only enable further integration but can also enable the development of plug-ins for the most popular development tools, such as Eclipse, NetBeans or Visual Studio. Having OBDPR always at hand during development would certainly seem beneficial.

Even existing services need some improvements before going into production. For example, we are trying to personalize the proposing service. The proposing component could learn about the user from past proposals and, for instance, ask personalized questions or ask more questions to verify possibly contradictory answers. The idea behind the proposing service includes verifying if the developer knows what a certain question mean. This could be achieved with question redundancy: if the developer answers a question with an option that prefers pattern A and another question with an option that does not prefer pattern A, it is possible that the developer is confused. With this in mind, we can reduce questions asked during the proposing service, if we consider the developer's past dialogues. OBDPR enables the analysis of exhaustive logs of usage. We have data on each proposing process if a selection is well done. If not, we can review which question was shown to be problematic and where the user starts to get confused (by measuring several attributes for each question including repetition number, time spent, premature finishing etc). This can be used as guidance for experts to review and improve questions and answers or to provide more questions connecting particular candidates.

After performing research activities by means of experimenting with the platform on industry developers, we plan to develop a holistic methodology for design pattern selection. It will include both a design pattern expert and user activities. OBDPR will be given the role of an enabling tool for the developed methodology. To take full advantage of formalized design patterns aspects, there is basically no limitation for creating additional services on top of the platform.

7 Conclusions

The platform (Ontology-Based Design Pattern Repository – OBDPR) presented in this paper is the basis for automatic and intelligent services built on top of formally presented design pattern knowledge. The main aim of OBDPR is to introduce formal methods of design pattern representation in order to drawn upon capabilities known from the area of artificial intelligence. It also simultaneously keeps patterns in human-friendly form. Therefore, the semantic web approach and technology were used. OBDPR addresses the challenges of selecting, understanding and using design patterns in the rapidly increasing number of design patterns.

In the paper, we have presented several important components for our approach:

- The proposed formal design pattern presentation technique can easily be used by automatic intelligent services as well as by human users. It can additionally be integrated with existing presentations, especially ODOL [1].
- A new, fully functional OBDPR with the capability to serve as a basis for more advanced services (we have so far developed a searching, proposing, training and validating service).
- The platform and services have initially been exposed to real-world usage. The initial experiment has encouraged us to carry on with our work and perform additional, rigorous experiments.

We are confident that our work can contribute to an increase in using design patterns, especially by helping to find a suitable design pattern for a given situation. This issue constitutes a great challenge for the typical developer. OBDPR was therefore developed primarily to capture design patterns, explicit and implicit expert knowledge, and to enable the further development of intelligent services and to test our belief that we can improve design pattern adoption.

Introducing semantic web concepts and technology into the design pattern field has revealed itself to be the correct solution so far. It creates new possibilities for making design patterns more approachable for software engineers.

8 References

[1] A. H. Eden et al, Precise Specification and Automatic Application of Design Patterns, International Conference on Automated Software Engineering, IEEE Press, 1997.

[2] Core J2EE Patterns, http://java.sun.com/blueprints/corej2eepatterns.

[3] D. K. Kim at al, A UML-based Metamodeling Language to Specify Design Patterns, Proceedings of the Workshop Software Model Eng. (WiSME) with Unified Modeling Language Conf. 2003, October 2003.

[4] D.Manolescu, W. Kozaczynski, A. Miller, J. Hogg, "The Growing Divide in the Patterns World", IEEE Software, Vol. 24, No. 4., July/August 2007, pp. 61-67.

[5] E. Gamma et al, Design patterns: Elements of reusable object orientated software, Addison Wesley Longman, 1998.

[6] Gerson Sunyé et al, Design Pattern Application in UML, ECOOP'00, http://www.ifs.uni-linz.ac.at/~ecoop/cd/papers/1850/18500044.pdf.

[7] J. M. Rosengard, M. F. Ursu, Ontological Representations of Software Patterns, KES'04, Lecture Notes in Computer Science, Springer-Verlag, 2004, http://w2.syronex.com/jmr/pubs/2004/ontology-pattern.pdf.

[8] J. M. Rosengard, M. F. Ursu, Ontological Representations of Software Patterns, KES'04, Lecture Notes in Computer Science, Springer-Verlag, 2004, http://w2.syronex.com/jmr/pubs/2004/ontology-pattern.pdf.

[9] Jena Semantic Web Framework, http://jena.sourceforge.net.

[10] L. Rising, "Understanding the Power of Abstraction in Patterns", IEEE Software, July/August 2007, Vol. 24, No. 4., pp. 46-51.

[11] L. Rising, The Pattern Almanac 2000, Addison Wesley, 2000.

[12] Marcus Fontoura and Carlos Lucena , Extending UML to Improve the Representation of Design Patterns, Computer Science Department, Pontifical Catholic University of Rio de Janeiro.

[13] Microsoft, Microsoft Patterns & Practices, http://msdn.microsoft.com/practices.

[14] OWL Web Ontology Language Overview, http://www.w3.org/TR/owl-features.

[15] R. Singh, Drive: An RDF Parser for .NET, http://www.driverdf.org/.

[16] RDF/XML Syntax Specification, http://www.w3.org/TR/rdf-syntax-grammar.

[17] Schmidt, D.C., "Using Design Patterns to Develop Reusable Object-Oriented Communication Software", Communications of the ACM, October 1995.

[18] SPARQL Query Language for RDF, http://www.w3.org/TR/rdf-sparql-query.

[19] T. Berners-Lee, "Business Model for the Semantic Web", http://www.w3.org/ DesignIssues/Overview .html.

[20] W3C, "Semantic Web", http://www.w3.org/2001/ sw/.

[21] J. Helin, P. Kellomäki, T. Mikkonen. Patterns of Collective Behavior in Ocsid. p. 73-93. Design Patterns Formalization Techniques. IGI Publishing. March 2007.

[22] T. Taibi, Design Patterns Formalization Techniques, United Arab Emirates University, UAE, Preface, IGI Publishing. December 2006.

[23] T. Berners-Lee, J. Hendler, O. Lassila, The Semantic Web, Scientific American, 284 (5) (2001) 28-37.

[24] P. Bonillo, N. Zambrano y Eleonora Acosta, Methodologic Proposal for Business Process Management sustained in the use of Patterns, Journal of Object Technology, vol. 7, no. 7, September - October, pp. 131-145, http://www.jot.fm/issues/issue_2008_09/article5.

[25] P. Gomes, F.C. Pereira, P. Paiva, N. Seco, P. Carreiro, J. Ferreira, C. Bento, Selection and Reuse of Software Design Patterns Using CBR and WordNet, 15th International Conference on Software Engineering and Knowledge Engineering, SEKE 2003, Proceedings, (SEKE'03), pp. 289-296.

[26] M. Heričko, I. Brejc, L. Pavlič, V. Podgorelec, A Question-Based Design Pattern Advisement Approach, Journal of Systems and Software, Submitted to publication in November 2008.

[27] U. Zdun, Systematic Pattern Selection Using Pattern Language Grammars and Design Space Analysis, Software: Practice & Experience, vol. 37, no. 9, p 983-1016, Wiley, 2007.

Historical Impulse Response of Return Analysis Shows Information Technology Improves Stock Market Efficiency

William Leigh
University of Central Florida
College of Business Administration
PO Box 161400
Orlando, FL 32816-1400, USA
E-mail: wleigh@bus.ucf.edu; http://www.bus.ucf.edu/leigh/

Russell Purvis
Clemson University
College of Business & Behavioural Science
165 Sirrine Hall
Clemson, SC 29634, USA
E-mail: rlpurvi@clemson.edu; http://people.clemson.edu/~rlpurvi/

Keywords: empirical impulse response function, impact of information technology, market efficiency, information diffusion, econophysics, stock market as information system

Received: March 9, 2007

We compare average impulse response of rate of return curves computed from more than sixty-seven years of historical Dow Jones Industrial, Transportation, and Utility Average closing values. The curves are relatively consistent in shape until the 1990s, when marked changes, indicative of improved market efficiency, occur for the Dow Jones Industrial Average. We argue that the effect is a result of the increased availability and reduced cost of online stock trading and of the more rapid dissemination and diffusion of information made possible by information technology. The basis for this argument is that: 1) the effect occurs only for the Industrial Average, which is comprised of stocks which are most well-known to the investing public, and not for the Transportation or Utility Averages, which are comprised of less well-known stocks; and 2) the effect is progressive and contemporaneous with the growth of the use of personal computing and the internet.

Povzetek: Z analizo prejšnjih dogajanj je pokazano, da informacijske tehnologije izboljšujejo učinkovitost borz.

1 Introduction

The output signal response from a continuous system to a narrow pulse input signal is the system's *impulse response*. Physical systems may be analyzed by observing the system's response to a sharp input, by striking a bell, for example, and observing the movement of points on the surface of the bell. Economic systems are not amenable to the sort of laboratory analysis that may be applied to physical systems, but an investigator of the stock market may detect narrow pulses of increased (or decreased) rate of return in the time series of daily rates of return for the stock market and then observe the behavior of the subsequent daily returns, thus employing a synthetic impulse response analysis.

The assumption of *market efficiency* is important in the theory of the academic discipline of Finance. A perfectly efficient market adjusts its prices to new information instantaneously and correctly. This implies that for a perfectly efficient market, all relevant information available in the past history of stock price as well as any newly revealed information is considered and reflected in the current price, and this means that historical price information alone may not be used to forecast future prices with any useful degree of success. Perfect market efficiency requires the instantaneous availability and instantaneous correct application of perfect information by all market participants, and these assumptions are not true in the real world of the stock market. Thus, as friction is assumed to be negligible in order to consider an ideal "F=MA", the effects of investor psychology and of non-instantaneous and non-homogeneous information communication and application are assumed to be negligible by the assumption of a perfectly efficient market. Brynjolfsson and Smith (2000) consider internet retail as a "frictionless" commerce.

Society is interested in the fairness and efficiency of markets because society makes resource allocation decisions based on market prices. In practice our markets may not be truly efficient, but perfect efficiency is a benchmark by which we can evaluate how well a market serves society. Perfect stock market efficiency is a worthwhile ideal, though perfect market efficiency is impossible. Kauffman and Walden (2001) identify the effect of the widespread use of modern information technology on market efficiency as a topic for future study.

Successes with the application of theory and techniques from physics, such as impulse response analysis, to understanding and predicting the behavior of the stock market are reported primarily in the physics journals (for example, Sornette and Zhou 2006). There are some reports in the economics and finance journals of the use of techniques from physics for modeling economic and financial systems (for example, Sornette and Zhou 2005), but as most academics in economics and finance hold to the assumption that the stock markets are perfectly efficient, and as the physics models tend to be useful only in modeling markets which are not perfectly efficient, the physics journals (where this work is sometimes called "econophysics") are a more likely place to find the application of physics models to the stock markets than are the economics and finance journals.

Donella Meadows (1999) identifies "leverage points", places to intervene in a system to effect change. These leverage points are, in increasing order of effectiveness:

- Constants, parameters.
- Sizes of buffers and stocks.
- Structure of material stocks and flows.
- Lengths of delays.
- Strength of negative feedback loops.
- Gain around driving positive feedback loops.
- Structure of information flows (who does and does not have access to what kinds of information.)
- Rules of system.
- Power to add or change the system structure.
- Goals of the system.
- Mindset out of which the system arises.
- Power to transcend paradigms.

The introduction of innovations in the stock market system is ongoing and continuous (White 2003). The behavior of the stock market may be affected by changes in the structure of the market itself, in the form of new products and new regulations, or by changes in the information systems used by the market participants. Offering new stock market products constitutes level 10 leverage point change to the stock market system. New regulations, including the spate of new regulations enacted after the 1987 Crash (Lindsey and Pecora 1998), are examples of leverage point change no more powerful than level 8. The increased availability and reduced cost of online stock trading and the more rapid dissemination of information, all made possible by the personal computer and internet information technology revolution, constitute more powerful level 6 change at least, and may involve change as powerful as level 4. "Day trading", enabled by information technology, may constitute change as strong as level 2.

Diffusion models are used to model the adoption of innovation, and this is well-studied; Rogers (1983) is a seminal survey. Physicists model radiation, dispersion, and diffusion phenomena, as surveyed in Beltrami (2002): for example, Frick's Law of Diffusion, Newton's Law of Cooling, Fisher's Equation. These models from physics may be applicable to information diffusion in the stock market system. Houthakker and Williamson (1996) survey the research concerning the characteristics of the stock market traders' information medium and how these characteristics may affect the diffusion of information. Surowiecki (2004) surveys the research explaining how markets can arrive at correct prices and can be efficient if the market participants receive timely information.

An impulse response of return graph may be constructed with rate of return for a period (40 trading days in this study) in the vertical axis and time in the horizontal axis. If there is no persistence of the effect of the impulse, then the average impulse response of return behavior for the period is "flat", in graphical terms, after the impulse, and we may say that the market is perfectly efficient. If the graph is not flat then the market is not perfectly efficient.

Change in price in the stock market is the effect of information on the traders' value estimates for the stocks. Lags in the diffusion and adoption of new information can explain non-flat response of return behavior. Using the terminology of the innovation diffusion literature, "early adopters" may receive information early and act quickly, while "late adopters" may receive new information late and/or act slowly.

2 Method

The results in this paper are from the use of a method for deriving an empirical impulse response function of return, such as is discussed in Koop et al (1996). The method is applied to rolling 5000 trading day intervals of closing values of three Dow Jones Averages (Industrial -- DJIA, Transportation -- DJTA, and Utility -- DJUA) for the period 5/28/1936 to 1/26/2004, which comprises 17,000 trading days.

For each trading day t, a return, r_t, is computed, which is the change in closing price over the 40 trading days following as a fraction of price, p_t. Next we determine a trading day history window for p_t such that p_t is not included in calculating return values for prices in that history window. Since any return for trading days $t-40$ through t includes p_t in the calculation, a 400 trading day history window preceding trading day t that does not include p_t would begin at day $a = t-400-40 = t-440$ and end at day $b = t-40-1 = t-41$. This presentation uses 40 trading days as the return horizon and 400 trading days as the history period throughout.

Hence we have:

$t = 1,\ldots,17000$ trading days

$p_t =$ closing value of Dow Jones Average on trading day t

$r_t = (p_{t+40} - p_t) / p_t$ return for trading day t

$R_{a,b} = \{ r_t \mid a \leq t \leq b \}$ a set of returns for trading days a to b

$R_{t-440,t-41} = \{ r_t \mid t-440 \leq t \leq t-41 \}$ Historical Return Set for p_t

A positive impulse is identified if a return, r_t, exceeds the average of the returns in p_t's historical return set $R_{t-440,t-41}$ by a multiple of the standard deviation of the returns in that historical return set. A value of 1.5 is used for this multiplier throughout this paper.

$\check{r}_t = {}^1/_n\sum_{k=1}^{n} r_k$ the average return for the historical return set, $R_{t-440,t-41}$

$s_t = \sqrt{[{}^1/_{n-1}\sum_{k=1}^{n} (r_k - \check{r}_t)^2]}$ the standard deviation for the historical return set, $R_{t-440,t-41}$

$m = 1.5$ a constant multiplier

$i_t = \begin{cases} 1, & \text{if } r_t > \check{r}_t + m \cdot s_t \\ 0, & \text{otherwise} \end{cases}$ the impulse for trading day t

Table 1 contains the beginning and ending trading day numbers and the corresponding dates for the intervals used. For each interval the table lists the number of positive impulses which occurred in that interval for each of the three Dow Jones Averages.

Interval	Trading Day		Date		Impulses		
	Begin	End	Begin	End	DJIA	DJTA	DJUA
36-56	1	5000	05/28/36	05/21/56	415	427	395
56-76	5001	10000	05/22/56	05/03/76	376	528	500
60-80	6001	11000	05/11/60	04/17/80	409	522	491
64-84	7001	12000	05/04/64	03/30/84	469	496	481
68-88	8001	13000	04/24/68	03/16/88	541	510	507
72-92	9001	14000	05/16/72	02/28/92	497	466	477
76-96	10001	15000	05/04/76	02/13/96	514	446	441
80-00	11001	16000	04/18/80	01/31/00	525	393	458
84-04	12001	17000	04/02/84	01/26/04	470	378	521

Table 1: Intervals used in terms of trading days and the number of impulses in each interval for each Dow Jones Average.

The method defines the "response" to the impulse in terms of time lags expressed in trading days. This impulse response function is derived from an aggregation of trading days in a response set. Returns for trading days are identified for inclusion in the impulse response return set by relating them to the return for the trading day preceding them by a lag of between 1 and 200 trading days. If a trading day's return is identified as a return associated with an impulse, then the lagged trading day return is a member of the response set. There is an impulse response return set, $I_{a,b,L}$, for each of the 200 lag values for trading day interval from *a* to *b*.

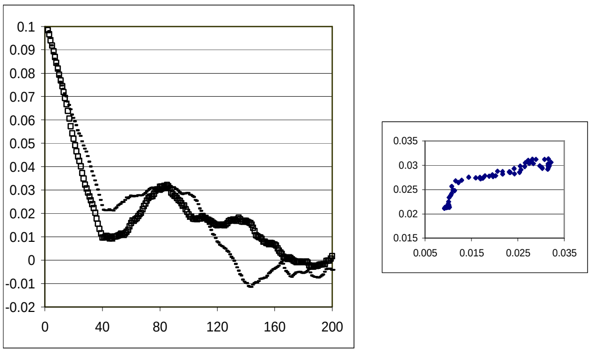
That is, we have:

$L = 1, \ldots, 200$ the lag in trading days

$I_{a,b,L} = \{ r_k \mid i_{k-L} = 1 \text{ and } a \leq k \leq b \}$ the impulse response return set given L from trading day a to trading day b

$\check{\imath}_{a,b,L} = {}^1/_n\sum_{k=1}^{n} r_k$ where r_k is in $I_{a,b,L}$ the average impulse response return in $I_{a,b,L}$ for lag L

Figure 1(a) shows a graph of impulse responses for the time intervals 36-56 (the 200 $\check{\imath}_{1,5000,L}$ values for the years 1936 to 1956, denoted by the squares in the graph) and 56-76 (the 200 $\check{\imath}_{5001,10000,L}$ values for the years 1956 to 1976, denoted by dashes) for lags of 1 to 200. Figure 1(b) graphs the impulse response values for interval 36-56 on the x-axis and interval 56-76 on the y-axis for lags of 40 to 100. The method is ex-ante for lags of 40 and up, and so it is the impulse responses for those lags that are of interest for forecasting and trading applications.



(a) (b)

Figure 1: (a) Impulse response graphs for intervals 36-56 (squares) and 56-76 (dashes) for lags 1 to 200. (b) Graph of impulse response values for interval 36-56 against impulse response values for interval 56-76 for lags of 40 to 100.

3 Historical comparison

Table 2 shows correlation coefficient values calculated for each interval, for each of the three Dow Jones Averages, for lags 40 to 100, and for lags 40 to 200. The correlation coefficient is calculated for the impulse response curve for interval 36-56 for the respective Dow Jones Average as correlated against each of the other intervals for the same Dow Jones Average. The correlation coefficients then are:

$c_{X,Y}$ = correlation coefficient as computed from the sets X and Y of values paired as (x_L, y_L) where L is the lag value

C_{a,b,c,d,L_1,L_n} = correlation coefficient as computed from $c_{\breve{\imath}_{a,b},\breve{\imath}_{c,d}}$ ({ $\breve{\imath}_{a,b,L}$ | $L_1 \le L \le L_n$}, { $\breve{\imath}_{c,d,L}$ | $L_1 \le L \le L_n$})

For example, the top value in the second column of Table 2 is 0.907. This is the evaluation of $C_{1,5000,5001,10000,40,100}$ for the Dow Jones Industrial Average.

Interval	DJIA		DJTA		DJUA	
	40-100	40-200	40-100	40-200	40-100	40-200
56-76	.907	.734	.434	.627	.654	.385
60-80	.943	.700	.667	.547	.376	.161
64-84	.923	.765	.661	.594	.343	.298
68-88	.910	.805	.700	.642	.334	.269
72-92	.863	.751	.600	.498	.077	.235
76-96	.893	.828	.490	.215	-.125	.124
80-00	-.496	.706	.330	.426	.355	.276
84-04	-.475	.391	.490	.026	.617	.301

Table 2: Correlation coefficient values for the impulse response function of each interval correlated with the first interval 36-56 for each Dow Jones Average.

The correlation coefficient is useful for comparing the impulse response curves in a rough-and-ready way. In Table 2 it may be seen that the correlation coefficient values for lags 40 to 100 change markedly in sign and magnitude for the DJIA for intervals 80-00 and 84-04 as compared with the correlation coefficient values for the intervals which went before. Note the negative correlation values for the Industrial Average in the last two rows of Table 2.

Table 3 is prepared in a similar way to Table 2 except that the 36-56 interval for the Dow Jones Industrial Average is used as the correlation partner for each of the other intervals (instead of the 36-56 interval in the respective Dow Jones Average.)

Interval	DJTA		DJUA	
	40-100	40-200	40-100	40-200
56-76	.426	.816	.953	.217
60-80	.850	.812	.865	.203
64-84	.875	.827	.865	.395
68-88	.970	.895	.870	.459
72-92	.927	.727	.718	.557
76-96	.803	.537	.600	.683
80-00	.581	.594	.861	.749
84-04	.714	.367	.938	.609

Table 3: Similar to Table 2 but 36-56 interval for DJIA is used as correlation partner for each of the intervals in the DJTA and in the DJUA.

In Table 3 the correlation coefficient values are generally higher for the DJTA and DJUA than they are in Table 2. The correlations continue to be relatively strong through the 80-00 and 84-04 intervals.

Figure 2 shows graphs of the impulse response functions for all of the intervals for the DJIA, except that the curves are offset by the average profit in each of the intervals. Thus, on the y-axis Figure 2 shows excess profit, which is the difference between $\breve{\imath}_{a,b,L}$, the average impulse response return in the set $I_{a,b,L}$ for trading days a to b and for lag L, and the average of returns from trading days a to b, that is the set $R_{a,b}$. A market timing rule which would have worked before 1990 on the DJIA and still worked on the DJTA and DJUA to the end of the study: "Buy at impulse lag day 80 and sell after holding for 40 days".

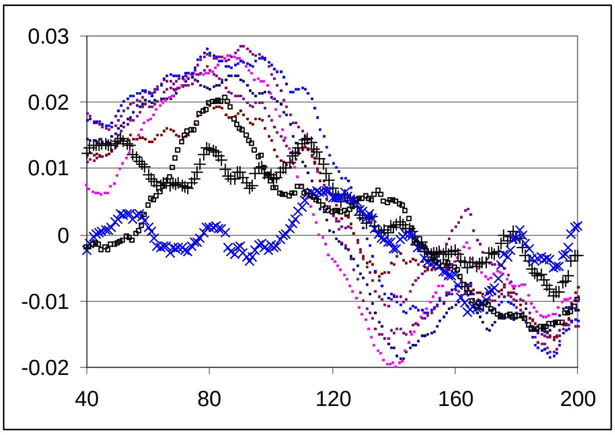


Figure 2: Impulse response for DJIA for each interval, adjusted by average profit in interval. Values for interval 36-56 are denoted by the square, for interval 80-00 by "+", and for interval 84-04 by "x".

4 Conclusion

The stock market acts like an elastic medium in time which transmits shock waves, in this case for rate of return shocks. The impulse response behavior of the stock market information transmission medium was relatively homogeneous from 1936 up until about 1990. In Figure 2, an oscillating wave is seen, with over-reaction and under-reaction out to 200 days. About 1990, the stock market price information transmission medium changed drastically for the DJIA, with the response from the shock dampened out immediately. No similar dampening is seen for the DJTA and DJUA, though perhaps we can expect the DJTA and DJUA to be transformed in the same way in the future.

The progressive and distinct flattening of the impulse response curves for DJIA intervals 80-00 and 84-04 (see Figure 2) can be considered to indicate increasing efficiency in the market underlying the DJIA, showing that the market mechanism has been improved. The changes in kind and velocity of information processing resulting from the computer and communications technology revolution of the last few decades is truly new to the stock market, and only for this information technology revolution is it possible that the impact be sufficient to have caused the effect observed. This with the fact that the effect on stock market information processing reported in this paper is exactly contemporaneous with this revolution in information technology and that the effect is pronounced in the case of the stocks most widely known and traded by the segment of the stock trading population which has been newly empowered by the information technology revolution leads us to make the argument that the effects are primarily the result of information technology.

This work investigated a trading horizon of 40 trading days. An objective of future work will be to determine the effect on impulse response of information technology through the full spectrum of trading horizons, from 1 day to 100 days or more. An interesting hypothesis, consistent with the idea that the technology and the traders use of it become more effective as time goes on, is that the "flattening" begins at the longer trading horizons and over time, progresses, to the shorter trading horizons.

References

[1] Beltrami, E. (2002). *Mathematical Models for Society and Biology*. Academic Press, San Diego. 199 pages.

[2] Brynjolfsson, E. and M. Smith (2000). Economics and electronic commerce: survey and research directions. *International Journal of Electronic Commerce*, 5(4), 5-117.

[3] Houthakker, H.S. and P.J. Williamson (1996). *The Economics of Financial Markets*. Oxford University Press, Oxford. 361 pages.

[4] Kauffman, R.J. and E.A.Walden (2001).Economics and electronic commerce: survey and research directions. *International Journal of Electronic Commerce.* 5(4), 5-117.

[5] Koop, G., M.H. Pesaran and S.M. Potter (1996). Impulse response analysis in nonlinear multivariate models. *Journal of Econometrics*, 74(1), 119-148.

[6] Lindsey, R.R. and A.P. Pecora (1998). Ten years after: regulatory developments in the securities markets since the 1987 market break. *Journal of Financial Services Research*, 13(3), 283-314.

[7] Meadows, D. (1999). *Leverage Points Places to Intervene in a System*. The Sustainability Institute, Hartland, Vermont. 19 pages.

[8] Rogers, E.M. (1983). *Diffusion of Innovations*. The Free Press, New York. 453 pages.

[9] Sornette, D. and W Zhou (2005). Non-parametric determination of real-time lag structure between two time series: the "optimal thermal causal path" method. *Quantitative Finance,* 5, 577-591.

[10] Sornette, D. and W Zhou (2006). Importance of positive feedbacks and overconfidence in a self-fulfilling Ising model of financial markets. *Physica A: Statistical and Theoretical Physics,* 370(2), 704-726

[11] Surowiecki, J. (2004). *The Wisdom of Crowds*. Doubleday, New York. 296 pages.

[12] White, L.J. (2003). Technological change, financial innovation, and financial regulation in the U.S.: the challenges for public policy. In G. Constantinides, M. Harris, and R.M. Stulz (Ed.), *Economics of Finance: Financial Markets and Asset Pricing*, Vol. 1B. North Holland, New York.

Fall Detection and Activity Recognition with Machine Learning

Mitja Luštrek and Boštjan Kaluža
Jožef Stefan Institute, Department of Intelligent Systems
Jamova cesta 39, SI-1000 Ljubljana, Slovenia
E-mail: mitja.lustrek@ijs.si, bostjan.kaluza@ijs.si

Keywords: fall detection, activity recognition, posture and movement reconstruction, machine learning

Received: July 16, 2008

Due to the rapid aging of the European population, an effort needs to be made to ensure that the elderly can live longer independently with minimal support of the working-age population. The Confidence project aims to do this by unobtrusively monitoring their activity to recognize falls and other health problems. This is achieved by equipping the user with radio tags, from which the locations of body parts are determined, thus enabling posture and movement reconstruction. In the paper we first give a general overview of the research on fall detection and activity recognition. We proceed to describe the machine learning approach to activity recognition to be used in the Confidence project. In this approach, the attributes characterizing the user's behavior and a machine learning algorithm must be selected. The attributes we consider are the locations of body parts in the reference coordinate system (fixed with respect to the environment), the locations of body parts in a body coordinate system (affixed to the user's body) and the angles between adjacent body parts. Eight machine learning algorithms are compared. The highest classification accuracy of over 95 % is achieved by Support Vector Machine used on the reference attributes and angles.

Povzetek: Članek opisuje zaznavanje padcev in prepoznavanja aktivnosti nasploh ter izvedbo prepoznavanja aktivnosti s strojnim učenjem za potrebe projekta Confidence.

1 Introduction

The European population is aging due to the increase in life expectancy and decrease in birth rate. The percentage of population aged over 65 years is anticipated to rise from 17.9 % in 2007 to 53.5 % in 2060 [7]. As a consequence, the number of the elderly will exceed the society's capacity for taking care of them. Thus an effort needs to be made to ensure that the elderly can live longer independently with minimal support of the working-age population. This is the primary goal of the EC Seventh Framework project Confidence [4].

The Confidence project will develop a ubiquitous care system to unobtrusively monitor the user, raise an alarm if a fall is detected and warn of changes in behavior that may indicate a health problem. This will improve the chances of a timely medical intervention and give the user a sense of security and confidence, thus prolonging his/her independence.

The user of the Confidence system will wear small inexpensive wireless tags on the significant places on the body, such as wrists, elbows, shoulders, ankles, knees and hips. The precise number and placement of tags will be defined during development. The tags may even be sewn into the clothes. The locations of the tags will be detected by a base station placed in the apartment and a portable device carried outside. This will make it possible to reconstruct the user's posture and movement and to recognize his/her activity. Some tags may be placed in the user's environment at locations such as bed and chair to recognize activities such as the user lying in a bed and sitting in a chair. Finally, the user's behavior will be interpreted as normal or abnormal. An alarm or warning will be raised in the latter case.

This paper describes machine learning methods for activity recognition [12][13] to be used in the Confidence project. We focus on the selection of attributes and machine learning algorithm to maximize the recognition accuracy. The activities to recognize are falling, the process of lying down, the process of sitting down, standing/walking, sitting and lying. Falling is important in itself because fall detection is one of the main goals of the project. For the processes of lying down and sitting down, we wanted to see whether they can be distinguished from falling. The recognition of standing/walking, sitting and lying is needed to detect changes in behavior, such as the user walking less and lying more, which may indicate a health problem.

The paper is structured as follows. Section 2 gives a detailed overview of related work on fall detection and activity recognition [9]. Section 3 describes the recordings of user behavior used as input data. Section 4 lists the attributes extracted from the input data that are fed into the machine learning algorithms. Section 5 presents the experiments in which the various attributes and machine learning algorithms are compared. Finally, Section 6 concludes the paper in outlines the future work.

2 Related work

We divide the work on fall detection and activity recognition into four approaches presented in the following four subsections. They are distinguished by the equipment used and by the features extracted from sensor data.

The first approach is based on *accelerometers*. An accelerometer is a device for detecting the magnitude and direction of the acceleration along a single axis or along multiple axes. Three-axis accelerometers are typically used. By detecting the acceleration caused by the earth's gravity, one can also compute the accelerometer's angle with respect to the earth.

The second approach uses *gyroscopes*, which measure orientation. A gyroscope consists of a spinning wheel whose axle is free to take any orientation. It can measure the orientation along one axis or multiple axes. By equipping an object with the gyroscope(s) to measure the orientation along three axes, it is possible to exactly determine the object's orientation and the changes in orientation, from which the angular velocity can be computed.

The third approach is denoted *visual detection without posture reconstruction*. It is based on extracting input data from still images or from video. Various computer vision techniques are applied to the input data, but the human posture is not reconstructed explicitly.

The fourth approach, named *visual detection with posture reconstruction*, is based on 3D locations of markers placed on an object, typically human body. The approach also uses video recordings, but, in contrast to the third approach, the visual information is used only to reconstruct the 3D locations of the markers. Additional processing uses the markers' coordinates as input data. If a sufficient number of markers are provided, it is possible to reconstruct the shape of an object, which in our case means the human posture.

2.1 Accelerometers

The most common and simple methodology for fall detection is using a tri-axial accelerometer with threshold algorithms [3][10]. Such algorithms simply raise the alarm when the threshold value of acceleration is reached. There are several sensors with hardware built-in fall detection [1][5][15], having the accuracy of over 80 %.

Zhang et al. [25] designed a fall detector based on Support Vector Machine (SVM) algorithm. The detector was using one waist-worn accelerometer. The features for machine learning were the accelerations in each direction, changes in acceleration etc. Their method detected falls with 96.7 % accuracy. Researches embedded an accelerometer in a cell phone [24] and detected falls with the proposed method. The cell phone was put in a pocket of clothes or hanged around the neck, which made the detection more difficult as with the body-fixed sensor. The cell-phone system correctly raised the alarm in 93.3 % of the cases.

Tapia et al. [18] presented a real-time algorithm for automatic recognition of not only physical activities, but also, in some cases, their intensities, using five wireless accelerometers and a wireless heart rate monitor. The accelerometers were placed at shoulder, wrist, hip, upper part of the thigh and ankle. The features, e.g., FFT peaks, variance, energy, correlation coefficients, were extracted from time and frequency domains using a predefined window size on the signal. The classification of activity was done with C4.5 and Naïve Bayes classifiers into three groups: postures (standing, sitting etc.), activities (walking, cycling etc.) and other activities (running, using stairs etc). For these three classes they obtained the recognition accuracy of 94.6 % using subject-dependent training and 56.3 % using subject-independent training.

Willis [21] developed a fall detection system based on belief network models, which enable probabilistic modeling of scenarios (e.g., normal walking, tripping/stumbling and running) and the transitions between them. The sensors were placed under the heel and toe, which made it possible to reconstruct gait cycle and to detect falls. The accuracy was not reported.

Researchers using accelerometers give a lot of attention to the optimal sensor placement on the body [3][10]. A head-worn accelerometer provides excellent impact detection sensitivity, but its limitations are usability and user acceptance. A better option is a waist-worn accelerometer. The wrist did not appear to be an optimal site for fall detection. Some researchers made a step further and used accelerometers for trying to recognize the impact and posture after the fall [11].

In the Confidence system, accelerations could in principle be derived from the movement of tags. However, we believe this approach to be unreliable: first, because the acceleration is the second derivative of tag location and as such strongly affected by sensor noise, and second, because the data acquisition frequency in Confidence is expected to be relatively low. The studies of sensor placement may be valuable for deciding where to place tags in Confidence.

2.2 Gyroscopes

Bourke and Lyons [2] introduced a threshold algorithm to distinguish between normal activities (sitting down and standing up, lying down and standing up, getting in and out of a car seat, walking etc.) and falls. The ability to discriminate was achieved using a bi-axial gyroscope mounted on the torso, measuring pitch and roll angular velocities. They applied a threshold algorithm to the peaks in the angular velocity signal, angular acceleration and torso angle change. The system proved 100 % successful in fall detection.

The Confidence system derives velocities from the movement of tags. The velocity, being the second derivative of tag location and being less affected by the low data acquisition frequency, is more reliable than acceleration. However, since the data available in Confidence is much richer than that provided by gyroscopes, we decided against simple threshold-based fall detection.

2.3 Visual detection without posture reconstruction

Vishwakarma et al. [20] presented a video approach for fall detection. First, they eliminated the background of the video and extracted a set of features from the remaining objects' bounding boxes, e.g., the aspect ratio, horizontal and vertical gradients etc. In the next step they detected falls based on the angle between an object's bounding box and the ground. The final step was fall confirmation, which was rule-based, e.g., the abovementioned angle had to be less than 45°. The method achieved 95 % accuracy on single-object fall detection and 64 % accuracy on multiple objects.

Fu et al. [8] described a vision system designed to detect accidental falls in elderly home care applications. They used a temporal contrast vision sensor, which extracts changing pixels from the background. An algorithm was observing the dynamic of motion and reported falls when it indicated significant changes in the vertical downward direction. They were able to distinguish falls from normal human behaviors, such as walking, crouching down and sitting down. The accuracy was not reported.

The proposed methods are quite capable of dealing with fall detection, but it is not clear how to adapt them to the sensor data available in the Confidence system.

2.4 Visual detection with posture reconstruction

Wu [23] studied unique features of the velocity during normal and abnormal (i.e. fall) activities so as to make the automatic detection of falls during the descending phase of a fall possible. Normal activities included walking, rising from a chair and sitting down, descending stairs, picking up an object from the floor, transferring in and out of a tub and lying down on a bed. The study provides exhaustive velocity parameters for fall detection, gathered by three markers placed on the posterior side of the torso, recorded by three cameras with the sampling rate of 50 Hz. The aim of the study was to suggest velocity characteristics, so the author did not actually implement automatic fall detection.

Qian et al. [16] introduced a gesture-driven interactive dance system capable of real-time feedback. They used 41 markers on the body recorded by 8 cameras with the frame rate of 120 Hz to construct a human body model. The model was used to extract features such as torso orientation, angles between adjacent body parts etc., which was used to represent different gestures. Each gesture was statistically modeled with a Gaussian random vector defined as the statistical distribution of the features for that gesture. To recognize a new pose, the likelihood of its feature vector given the vector of each known gesture was computed. The new pose was classified as the gesture for which this likelihood was the largest. Experimental results with two dancers performing 21 different gestures achieved gesture recognition rate of 99.3 %.

Sukthankar and Sycara [17] presented a system that reconstructs the users' posture and recognizes pre-defined behaviors. The data were captured with 43 body markers and 12 cameras with the sampling rate of 120 Hz. They constructed a human body model from the raw marker coordinates, and computed features, e.g. the angles between body parts, limb lengths, range of motion etc. from the model. Learning was performed using SVM. The method achieved 76.9 % accuracy in detecting the following elementary activities: walking, running, sneaking, being wounded, probing, crouching, and rising. Behavior was defined as a sequence of elementary activities and was modeled with Hidden Markov models. The authors defined a number of behavior models and classified a new sequence of activities into the model that fit it best.

The markers in the proposed systems have the same role as the tags in the Confidence system. The methods by Qian et al. and even more so by Sukthankar and Sycara inspired the approach we used for activity recognition in Confidence. We are not aware of anybody having used this kind of methods for fall detection, though.

3 Input data

The goal of our research was to classify the user's behavior into one of the following activities: falling, lying down, sitting down, standing/walking, sitting and lying. To obtain training data for a classifier to recognize these activities, we recorded 45 examples of the behavior of three persons. Each recording consisted of multiple activities:

- 3 × 15 recordings of falling, consisting of standing/walking, falling and lying.
- 3 × 10 recordings of lying down, consisting of standing/walking, lying down and lying.
- 3 × 10 recordings of sitting down, consisting of walking, sitting down and sitting.
- 3 × 10 recordings of walking.

The recordings consisted of the coordinates of 12 body tags attached to the shoulders, elbows, wrists, hips, knees and ankles. This is the full complement of tags that will probably be reduced in the future. Since the equipment with which the Confidence system will acquire tag coordinates is still under development, the commercially available Smart infrared motion capture system [6] was used instead. The coordinates were acquired with 60 Hz. The frequency was afterwards reduced to 10 Hz, which is the expected Confidence data acquisition frequency. To make the recordings even more similar to what we expect of the Confidence equipment, we added Gaussian noise to them. The standard deviation of the noise was 4.36 cm horizontally and 5.44 cm vertically. This corresponds to the noise measured in the Ubisense real time location system [19]. The Ubisense system is similar to the equipment planned for acquiring tag coordinates in Confidence. The noise in the recordings was smoothed with Kalman filter [14].

4 Attributes for machine learning

Finding the appropriate representation of the user's behavior activity was probably the most challenging part of our research. The behavior needs to be represented with simple and general attributes, so that the classifier using these attributes will also be general and work well on behaviors different from those in our recordings. It is not difficult to design attributes specific to our recordings; such attributes would work well on them. However, since our recordings captured only a small part of the whole range of human behavior, overly specific attributes would likely fail on general behavior.

The attribute vector from which the classifier infers the user's activity consists of ten consecutive snapshots of the user's posture, describing one second of activity. When multiple activities took place within a given second, the attribute vector was assigned the longest one.

We designed three sets of attributes describing the user's behavior. *Reference attributes* are expressed in the reference coordinate system, which is fixed with respect to the user's environment. *Body attributes* are expressed in a coordinate system affixed to the user's body. *Angle attributes* are the angles between adjacent body parts.

4.1 Reference attributes

When selecting reference attributes, we ignored x and y coordinates. These coordinates describe the user's location in the environment, but the activities of interest can generally take place at any location.

In the list of reference attributes, the upper index t indicates the time within the one-second interval: t = 1 ... 10. The lower index i indicates the tag: i = 1 ... 12. The lower index R indicates the reference coordinate system and distinguishes reference attributes from those belonging to the other two sets.

- $z^t_{i\mathrm{R}}$ … z coordinate of tag i at time t
- $v^t_{i\mathrm{R}}$ … the absolute velocity of the tag
- $v^t_{\mathrm{z}i\mathrm{R}}$ … the velocity of the tag in the z direction
- $d^t_{ij\mathrm{R}}$ ... the absolute distance between the tags i and j; $j = i + 1$ ... 12
- $d^t_{\mathrm{z}ij\mathrm{R}}$ ... the distance between tags i in j in the z direction

4.2 Body attributes

Body attributes are expressed in a coordinate system affixed to the user's body. This makes it possible to observe x and y coordinates of the user's body parts, since these coordinates no longer depend on the user's location in the environment.

The *body coordinate system* is shown in Figure 1. Its origin O is at the mid-point of the line connecting the hip tags (H_R and H_L for the right and left hip respectively). This line also defines the y axis, which points towards the left hip. The z axis is perpendicular to the y axis, touches the line connecting both shoulder tags (S_R and S_L for the right and left shoulder respectively) at point S_z, and points upwards. The x axis is perpendicular to the y and z axes and points forwards.

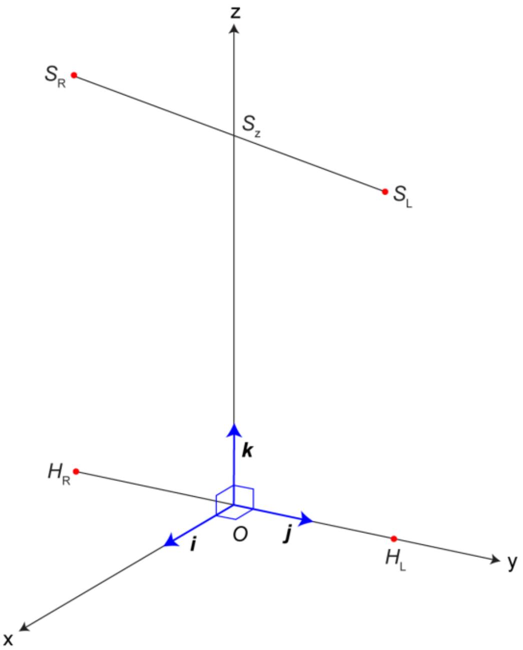


Figure 1: The body coordinate system.

In order to translate reference coordinates into body coordinates, we need to express the origin O and basis ($\boldsymbol{i}$, $\boldsymbol{j}$, $\boldsymbol{k}$) of the body coordinate system in the reference coordinate system. Note that bold type denotes vectors and $\boldsymbol{x}$ denotes a vector from the origin to the point X. Equation (1) expresses the origin of the body coordinate system in the reference coordinate system.

$$\boldsymbol{o} = \frac{\boldsymbol{h}_\mathrm{L} + \boldsymbol{h}_\mathrm{R}}{2} \quad (1)$$

Equation (2) gives us the basis vector $\boldsymbol{j}$.

$$\boldsymbol{j} = \frac{\boldsymbol{h}_\mathrm{L} - \boldsymbol{o}}{|\boldsymbol{h}_\mathrm{L} - \boldsymbol{o}|} \quad (2)$$

To obtain $\boldsymbol{k}$, Equation (3) is first used to calculate $\boldsymbol{s}_\mathrm{z}$.

$$\begin{aligned} &\boldsymbol{s}_\mathrm{z} = \boldsymbol{s}_\mathrm{R} + a(\boldsymbol{s}_\mathrm{L} - \boldsymbol{s}_\mathrm{R}) \\ &(\boldsymbol{s}_\mathrm{z} - \boldsymbol{o})(\boldsymbol{h}_\mathrm{L} - \boldsymbol{h}_\mathrm{R}) = 0 \\ &a = \frac{(\boldsymbol{s}_\mathrm{R} - \boldsymbol{o})(\boldsymbol{h}_\mathrm{L} - \boldsymbol{h}_\mathrm{R})}{(\boldsymbol{s}_\mathrm{L} - \boldsymbol{s}_\mathrm{R})(\boldsymbol{h}_\mathrm{L} - \boldsymbol{h}_\mathrm{R})} \end{aligned} \quad (3)$$

Once $\boldsymbol{s}_\mathrm{z}$ is calculated, Equation (4) gives us $\boldsymbol{k}$.

$$\boldsymbol{k} = \frac{\boldsymbol{s}_\mathrm{z} - \boldsymbol{o}}{|\boldsymbol{s}_\mathrm{z} - \boldsymbol{o}|} \quad (4)$$

Finally we obtain $\boldsymbol{i}$ using Equation (5).

$$\boldsymbol{i} = \boldsymbol{j} \times \boldsymbol{k} \quad (5)$$

We also experimented with a variant of *body coordinate system with the reference z axis*, which is shown in Figure 2. Its origin O is again at the mid-point of the line connecting the hip tags. The z axis is the z axis of the reference coordinate system. The y axis is perpendicular to the z axis, lies on the plane defined by the hip tags and a point on the z axis, and points towards the left hip. The x axis is perpendicular to the y and z axes and points forwards when the user is upright (in general it points in the direction of the cross product of the basis vectors $\boldsymbol{j}$ and $\boldsymbol{k}$).

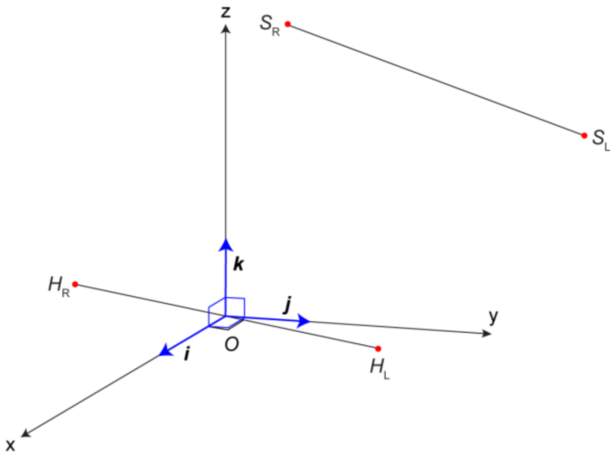


Figure 2: The body coordinate system with reference z axis.

In the body coordinate system with the reference z axis, the origin is again calculated with Equation (1). The basis vector $\boldsymbol{k}$ equals the basis vector $\boldsymbol{k}$ in the reference coordinate system: $\boldsymbol{k}$ = (0, 0, 1). The basis vector $\boldsymbol{i}$ is perpendicular to $\boldsymbol{k}$ and to the vector from O to H_L, which is expressed with Equation (6).

$$\boldsymbol{i} = \frac{\boldsymbol{k} \times (\boldsymbol{h}_\mathrm{L} - \boldsymbol{o})}{|\boldsymbol{k} \times (\boldsymbol{h}_\mathrm{L} - \boldsymbol{o})|} \qquad (6)$$

The basis vector $\boldsymbol{j}$ is obtained with Equation (7).

$$\boldsymbol{j} = \boldsymbol{k} \times \boldsymbol{i} \qquad (7)$$

To finally translate the coordinates in the reference coordinate system into the coordinates in either of the body coordinate systems, Equation (8) is used. The vector $\boldsymbol{p}_\mathrm{R}$ = (x_R, y_R, z_R, 1) corresponds to the point (x_R, y_R, z_R) in the reference coordinate system. The vector $\boldsymbol{p}_\mathrm{B}$ = (x_B, y_B, z_B, 1) corresponds to the point (x_B, y_B, z_B) in a body coordinate system. $\boldsymbol{T}_{\mathrm{R}\rightarrow\mathrm{B}}$ is the transformation matrix from the reference to the body coordinate system. Notation $\boldsymbol{i}_{(\mathrm{B})\mathrm{R}}$ refers to the basis vector $\boldsymbol{i}$ belonging to the body coordinate system, expressed in the reference coordinate system.

$$\boldsymbol{p}_\mathrm{B} = \boldsymbol{T}_{\mathrm{R}\rightarrow\mathrm{B}}\boldsymbol{p}_\mathrm{R}^\mathrm{T}$$

$$\boldsymbol{T}_{\mathrm{R}\rightarrow\mathrm{B}} = \begin{bmatrix} x_{\boldsymbol{i}(\mathrm{B})\mathrm{R}} & y_{\boldsymbol{i}(\mathrm{B})\mathrm{R}} & z_{\boldsymbol{i}(\mathrm{B})\mathrm{R}} & -\boldsymbol{o}_{(\mathrm{B})\mathrm{R}}\boldsymbol{i}_{(\mathrm{B})\mathrm{R}} \\ x_{\boldsymbol{j}(\mathrm{B})\mathrm{R}} & y_{\boldsymbol{j}(\mathrm{B})\mathrm{R}} & z_{\boldsymbol{j}(\mathrm{B})\mathrm{R}} & -\boldsymbol{o}_{(\mathrm{B})\mathrm{R}}\boldsymbol{j}_{(\mathrm{B})\mathrm{R}} \\ x_{\boldsymbol{k}(\mathrm{B})\mathrm{R}} & y_{\boldsymbol{k}(\mathrm{B})\mathrm{R}} & z_{\boldsymbol{k}(\mathrm{B})\mathrm{R}} & -\boldsymbol{o}_{(\mathrm{B})\mathrm{R}}\boldsymbol{k}_{(\mathrm{B})\mathrm{R}} \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad (8)$$

Body attributes (in either of the body coordinate systems) are labeled with a lower index B:

- ($x^t_{i\mathrm{B}}$, $y^t_{i\mathrm{B}}$, $z^t_{i\mathrm{B}}$) ... coordinates of the tag i at the time t
- $v^t_{i\mathrm{B}}$ … absolute velocity of the tag
- ($\varphi^t_{i\mathrm{B}}$, $\theta^t_{i\mathrm{B}}$) … the angles of movement of the tag with respect to the z axis and xz plane

If a body coordinate system is used, the attributes describing its location, orientation and movement with respect to the reference coordinate system are added to the attribute vector:

- z^t_OR ... z coordinate of the origin of the body coordinate system
- (Φ^t_OR, Θ^t_OR) ... the direction of the x axis of the body coordinate system with respect to the z axis and xz plane
- v^t_OR … absolute velocity of the origin of the body coordinate system
- (φ^t_OR, θ^t_OR) ... the angles of movement of the origin of the body coordinate system with respect to the z axis and xz plane

So far we expressed body attributes in the body coordinate system of each snapshot of the user's posture. However, the attributes in all ten snapshots within a one-second interval can be expressed in the coordinate system belonging to the first snapshot in the interval. This captures the changes in the x and y coordinates between snapshots within the interval. *First-snapshot body attributes* are the same as body attributes, except that they are labeled with Bf instead of B. The attributes describing the location and orientation of the first-snapshot body coordinate system with respect to the reference coordinate system are somewhat different, though:

- z_OfR … z coordinate of the origin of the first-snapshot body coordinate system
- (Φ_OfR, Θ_OfR) … the direction of the x axis of the first-snapshot body coordinate system with respect to the z axis and xz plane

4.3 Angle attributes

The paper will not delve into the details of the computation of body angles. The angles between body parts that rotate in more than one direction are expressed with quaternions:

- q^t_SL and q^t_SR ... left and right shoulder angles with respect to the upper torso at the time t
- q^t_HL and q^t_HR ... left and right hip angles with respect to the lower torso
- q^t_T ... the angle between the lower and upper torso
- α^t_EL, $\alpha^t_\mathrm{ER,}$ α^t_KL and α^t_KR ... left and right elbow angles, left and right knee angles

5 Machine learning experiments

We tried various machine learning algorithms to train classifiers for classifying the behavior into the six activities (falling, lying down, sitting down, standing/walking, sitting and lying). To do so, sections of the 135 recordings described in Section 3 were first manually labeled with the activities. Afterwards, the recordings were split into overlapping one-second intervals (one interval starting every one-tenth of a second). The attributes described in Section 4 were extracted from these intervals. This gave us 5,760 attribute vectors consisting of 240–2,700 attributes each (depending on the combination of attributes used). An activity was then assigned to each attribute vector. Finally these vectors were used as training data for eight machine learning algorithms: C4.5 decision trees, RIPPER decision rules, Naive Bayes, 3-Nearest Neighbors, Support Vector Machine (SVM), Random

Forest, Bagging and Adaboost M1 boosting. The algorithms were implemented in Weka [22], an open-source machine learning suite. Default parameter settings were used in all cases, except for Adaboost M1, where the algorithm to train the base classifier was replaced with Fast Decision Tree Learner. Machine learning experiments proceeded in two steps.

In the first step of machine learning experiments we compared the classification accuracy of the eight machine learning algorithms and of all single attributes sets described Section 4: reference, body, body with reference z, first-snapshot body, first-snapshot body with reference z and angles. The results are shown in Table 1. The accuracy was computed with ten-fold cross-validation. The accuracy of the best attribute set for each algorithm is in bold type; the accuracy of the best algorithm for each attribute set is on gray background.

Attribute set **Algorithm**	reference	body	body with reference z	first-snapshot body	first-snapshot body with reference z	angles
	Clean data					
C4.5 decision trees	**94.1**	92.8	93.7	92.9	93.2	91.8
RIPPER decision rules	**93.1**	91.4	92.8	92.0	93.0	90.9
Naive Bayes	89.5	88.7	**90.6**	86.8	88.2	76.7
3-Nearest Neighbor	**97.1**	92.0	82.8	88.1	85.1	96.9
SVM	**97.7**	94.4	95.0	94.1	94.3	90.5
Random Forest	**97.0**	96.5	96.8	96.0	96.0	96.8
Bagging	**95.9**	95.3	95.7	95.4	94.9	94.5
Adaboost M1 boosting	**97.7**	94.9	95.3	94.7	94.7	94.4
	Noisy data					
C4.5 decision trees	**90.1**	88.4	89.9	88.9	90.0	80.8
RIPPER decision rules	87.5	84.7	88.1	86.2	**88.6**	80.0
Naive Bayes	83.9	79.1	**84.0**	81.0	82.2	78.2
3-Nearest Neighbor	**95.3**	74.6	79.7	73.4	74.7	93.3
SVM	**96.3**	87.2	91.6	89.9	91.1	87.2
Random Forest	**93.9**	90.5	93.4	91.9	93.2	90.5
Bagging	**93.6**	91.8	93.3	92.3	93.5	89.1
Adaboost M1 boosting	**93.2**	92.0	93.1	92.1	92.9	88.4

Table 1: Classification accuracy for all the algorithms and all single attribute sets.

For the next step of machine learning experiments, we retained the best algorithms and the best attribute sets. To rank them, we compared the classification accuracies of all pairs of algorithms and all pairs of attribute sets. Table 2 shows the number of comparisons in which a given algorithm statistically significantly ($p < 0.05$) wins over another algorithm, minus the number of comparisons where it loses. Table 3 shows the same for the attribute sets. The accuracies of the algorithms and attribute sets selected for the second step are on grey background; the accuracies of the best algorithm and attribute set are in bold type. Since the second step consisted of combining the attribute sets, the selection of the sets to retain was based more on redundancy than classification accuracy. Thus we retained angles, but not the two first-snapshot body attributes (even though the latter have a higher accuracy), because first-snapshot body attributes are very similar to regular body attributes. We chose body attributes with the body z axis over body attributes with the reference z axis (even though the latter again have a higher accuracy), because the reference z coordinates are already included in the reference attributes. The comparison between every-snapshot and first-snapshot body attributes slightly favors the latter, but we nevertheless retained the former because they are computed more quickly.

Algorithm	**Wins – losses**	
	Clean	**Noisy**
C4.5 decision trees	–12	–10
RIPPER decision rules	–18	–21
Naive Bayes	–38	–34
3-Nearest Neighbor	–13	–16
SVM	13	11
Random Forest	**38**	23
Bagging	17	**25**
Adaboost M1 boosting	13	22

Table 2: The number of wins – the number of losses of every algorithm against the others for clean and noisy data

Attribute set	**Wins – losses**	
	Clean	**Noisy**
reference	**25**	**28**
body	–2	–21
body with reference z	9	20
first-snapshot body	–11	–9
first-snapshot body with reference z	–2	12
angles	–19	–30

Table 3: The number of wins – the number of losses of every single attribute set against the others for clean and noisy data

After selecting the best algorithms and attribute sets, we proceeded with the second step of machine learning experiments. In this step we tried combinations of attribute sets. Table 4 shows the classification accuracy for the four algorithms we retained and all the reasonable combinations of the remaining attribute sets. The accuracy of the best combination of attributes for each algorithm is in bold type; the accuracy of the best algorithm for each combination of attributes is on gray background.

Attribute set combination / Algorithm	reference + body	reference + body with reference z	reference + angles	body + angles	body with reference z + angles	all	all (reference z)
	Clean data						
SVM	96.6	96.9	**97.7**	95.3	95.5	96.7	96.9
Random Forest	97.0	97.0	**97.2**	96.7	96.9	97.1	97.0
Bagging	96.1	96.0	96.1	95.6	95.7	**96.3**	96.0
Adaboost M1 boosting	**95.7**	95.6	95.5	95.3	95.3	95.6	95.5
	Noisy data						
SVM	95.5	95.4	**96.5**	91.9	92.5	95.6	95.5
Random Forest	93.8	**94.2**	94.1	91.8	93.5	93.9	94.0
Bagging	93.8	**94.1**	93.7	92.4	93.4	93.8	**94.1**
Adaboost M1 boosting	93.6	**93.7**	93.2	93.2	93.3	93.6	**93.7**

Table 4: Classification accuracy for the retained algorithms and combinations of attribute sets.

6 Conclusion

We first investigated the work done so far in the area of fall detection and activity recognition. Fall detection methods were based on the accelerations and velocities of body parts and on visual cues. These data will not be available in the Confidence system, at least not directly. What will be available are the locations of body parts. Accelerations and velocities can be computed from the changes in these locations, but with questionable accuracy. We decided to use velocities, since they are expected to be more accurate than accelerations, and the locations of body parts themselves. Some work on activity recognition was also based on accelerations and velocities, but there were approaches better suited to Confidence as well. We were mostly inspired by the work of Sukthankar and Sycara [17], who used machine learning on attributes representing the body posture.

We then examined various attributes and machine learning algorithms to detect six common activities. The attributes were the coordinates of body parts in the reference coordinate system, the coordinates of body parts in four different body coordinate systems and the angles between adjacent body parts. We first compared the attribute sets in isolation and then in combinations. The reference coordinates were the best single attribute set. In combination with the angles, they gave the highest overall classification accuracy, although it should be noted that all the combinations were close in performance. We compared eight machine learning algorithms, from which Support Vector Machine produced the most accurate classifier: the accuracy on clean data was 97.7 % and on noisy data 96.5 %. It was closely followed by Random Forest, Bagging and Adaboost M1 boosting.

There are four directions for future work. The first is tuning the machine learning algorithms discussed in this paper and augment them with feature selection techniques. This is done relatively easily, but will probably not contribute much to the classification accuracy. The second direction is to take into account the temporal information: each activity takes usually lasts for some time and some transitions between activities are more likely than others. This information can help us correct some erroneous classifications, e.g., a single falling in a long sequence of walking must be an error. The third direction is using fewer than 12 tags, since a potential product resulting from the Confidence project is unlikely to use the full complement of tags. The last direction for future work is experimenting with recordings of additional behaviors. These may be variations of the existing ones to test the robustness of the classifier or entirely new activities to increase the classifier’s scope.

Concerning the Confidence project, the results described in this paper are encouraging. The classification accuracy of over 95 % leads us to believe that once the planned improvements are implemented, the frequency of false alarms will be low enough for the Confidence system to be useful.

Acknowledgement

This work was supported by the Slovenian Research Agency under the Research Programme P2-0209 Artificial Intelligence and Intelligent Systems. The research leading to these results has also received funding from the European Community's Framework Programme FP7/2007–2013 under grant agreement nº 214986. Consortium: CEIT (coordinator), Fraunhofer Institute for Integrated Circuits (IIS), Jožef Stefan Institute, Ikerlan, COOSS Marche, University of Jyväskylä, Umeå Municipality, eDevice, CUP2000 S.p.A/Ltd., ZENON S.A. Robotics & Informatics. We would like to thank Matjaž Gams for suggestions and discussion and Barbara Tvrdi for help with programming.

References

[1] AlertOne Services, Inc. *iLife™ Fall Detection Sensor.* http://www.falldetection.com, 2008-07-18.

[2] Bourke, A. K., and Lyons, G. M. *A threshold-based fall-detection algorithm using a bi-axial gyroscope sensor*. Medical Engineering & Physics, vol. 30, issue 1, pp. 84–90, 2006.

[3] Bourke, A. K., Scanaill, C. N., Culhane, K. M., O'Brien, J. V., and Lyons, G. M. *An optimum accelerometer configuration and simple algorithm for accurately detecting falls*. In Proceedings of the 24th IASTED international Conference on Biomedical Engineering, pp. 156–160, 2006.

[4] Confidence. http://www.confidence-eu.org, 2008-09-15

[5] Doughty, K., Lewis, R., and McIntosh, A. *The design of a practical and reliable fall detector for community and institutional telecare*. Journal of Telemedicine and Telecare, vol. 6, pp. 150–154, 2000.

[6] eMotion. Smart motion capture system. http://www.emotion3d.com/smart/smart.html, 2008-09-15.

[7] Eurostat. http://epp.eurostat.ec.europa.eu, 2008-09-09.

[8] Fu, Z., Culurciello, E., Lichtsteiner, P., and Delbruck, T. *Fall detection using an address-event temporal contrast vision sensor*. In Proceedengs of the IEEE International Symposium on Circuits and Systems – ISCAS 2008, pp. 424–427, 2008.

[9] Kaluža, B., and Luštrek M.. *Fall Detection and Activity Recognition Methods for the Confidence Project: A Survey*. In Proceedings of the 12th International Multiconference Information Society 2008, vol. A, pp. 22–25, 2008.

[10] Kangas, M., Konttila, A., Lindgren, P., Winblad, P., and Jamsa, T. *Comparison of low-complexity fall detection algorithms for body attached accelerometers*. Gait & Posture, vol. 28, issue 2, pp. 285–291, 2008.

[11] Kangas, M., Konttila, A., Winblad, I., and Jamsa, T. *Determination of simple thresholds for accelerometry-based parameters for fall detection*. In Proceedings of the 29th Annual International Conference of the IEEE, Engineering in Medicine and Biology Society, pp. 1367–1370, 2007.

[12] Luštrek, M., and Gams, M. *Posture and movement recognition from locations of body tags*. European Conference on Ambient Intelligence, "Ambient Assisted Living" and "Personal Health" – between Paragdigms, Projects and Products workshop, 2008.

[13] Luštrek, M., and Gams, M. *Prepoznava položaja telesa s strojnim učenjem*. In Proceedings of the 12th International Multiconference Information Society 2008, vol. A, pp. 30–33, 2008.

[14] Maybeck, P. S. Stochastic models, estimation, and control. Mathematics in Science and Engineering 141, 1979.

[15] Noury, N., Barralon, P., Virone, G., Boissy, P., Hamel. M., and Rumeau, P. *A smart sensor based on rules and its evaluation in daily routines*. In Proceedings of the 25th Annual International Conference of the IEEE, Engineering in Medicine and Biology Society, vol. 4, pp. 3286–3289, 2003.

[16] Qian, G., Guo, F., Ingalls, T., Olson, L., James, J., and Rikakis, T. *A Gesture-Driven Multimodal Interactive Dance System*. In Proceedings of the International Conference on Multimedia and Expo, Taipei, Taiwan, 2004.

[17] Sukthankar, G., and Sycara, K. *A Cost Minimization Approach to Human Behavior Recognition*. In Proceedings of the Fourth international Joint Conference on Autonomous Agents and Multiagent Systems 2005, pp. 1067–1074, 2005.

[18] Tapia, E. M., Intille, S. S., Haskell, W., Larson, K., Wright, J., King, A., and Friedman, R. *Real-Time Recognition of Physical Activities and Their Intensities Using Wireless Accelerometers and a Heart Rate Monitor.* In Proceedengs of the 11th IEEE International Symposium on Wearable Computers, pp. 37–40, 2007.

[19] Ubisense. http://www.ubisense.net, 2008-09-15.

[20] Vishwakarma, V., Mandal, C., and Sura, S. *Automatic Detection of Human Fall in Video.* Pattern Recognition and Machine Intelligence: Automatic Detection of Human Fall in Video, pp. 616–623, 2007.

[21] Willis, D. J. *Ambulation Monitoring and Fall Detection System using Dynamic Belief Networks*. PhD Thesis, School of Computer Science and Software Engineering, Monash University, 2000.

[22] Witten, I. H., and Frank, E. Data Mining: Practical machine learning tools and techniques, 2nd Edition. Morgan Kaufmann, San Francisco, USA, 2005.

[23] Wu, G. *Distinguishing fall activities from normal activities by velocity characteristics*. Journal of Biomechanics, vol. 33, issue 11, pp. 1497–1500, 2000.

[24] Zhang, T., Wang, J., Liu, P., and Hou, J. *Fall Detection by Embedding an Accelerometer in Cellphone and Using KFD Algorithm.* International Journal of Computer Science and Network Security, vol. 6, issue 10, 2006.

[25] Zhang, T., Wang, J., Liu, P., and Hou, J.. *Fall Detection by Wearable Sensor and One-Class SVM Algorithm*. Lecture Notes in Control and Information Science, issue 345, pp. 858–863, 2006.

Rajan Transform and its Uses in Pattern Recognition

Ekambaram Naidu Mandalapu
Professor, TRR College of Engineering, Hyderabad, India
Research Scholar of the University of Mysore
E-mail: menaidu2004@yahoo.co.in

Rajan E. G.
Professor and Dean
JB Institute of Engineering & Technology, Hyderabad
President, Pentagram Research Foundation
An Undertaking of Pentagram Research Centre (P) Limited, India
E-mail: president@pentagramresearch.com

Keywords: transform, pattern classification, image processing, permutation invariant systems

Received: June 13, 2006

This transform was introduced in the year 1997 by Rajan [2], [4] on the lines of Hadamard Transform. This paper presents, in addition to its formulation, the algebraic properties of the transform and its uses in pattern recognition. Rajan Transform (RT) is a coding morphism by which a number sequence (integer, rational, real or complex) of length equal to any power of two is transformed into a highly correlated number sequence of the same length. It is a homomorphism that maps a set consisting of a number sequence, its graphical inverse and their cyclic and dyadic permutations, to a set consisting of a unique number sequence ensuring the invariance property under such permutations. This invariance property is also true for the permutation class of the dual sequence of the number sequence under consideration. A number sequence and its dual are like an object and its mirror image. For example, the four point sequences x(n) = 3, 1, 3, 3 and y(n) = 2, 4, 2, 2 are duals to each other. Observe that the sum of each sequence is 10 and one sequence could be obtained from the other by subtracting the elements of the other sequence from 5, which is half of its sum. Since RT of a number sequence is an organized number sequence with a high degree of correlation, it is suitable for effective data compression. This paper describes in detail the techniques of using RT for pattern recognition purposes. For example, pattern recognition operations like extracting lines, curves, isolated points and points of intersection of lines from a digital gray or colour image could be carried out using RT based fast algorithms.

Povzetek: Prispevek opisuje Rajanovo transformacijo in njeno uporabo v prepoznavanju vzorcev.

1 Introduction

Pattern recognition is essentially a classification process. Set-theoretically, it is partitioning of domain set by a rule. One can also visualize pattern recognition as a homomorphic map connecting domain and range sets, while isomorphic maps are used for the purpose of analysis. Mostly, mathematical transforms are associated with inverses and hence they are isomorphic in nature. For example, Discrete Fourier Transform is defined as an Analysis-Synthesis pair like many other transforms. It is reasonable to raise a question here whether at all it is possible to develop a homomorphic transform, which would advocate operations related to pattern recognition. It is after a prolonged research and study of related techniques like Permutation Invariant Systems and Number Theory, that Professor Rajan and his research team introduced in the year 1997 a novel algorithm for classification purposes, which they called as Rapid Transform. Subsequent research on the algebraic properties of this transform has exposed its richness and recently it was renamed as Rajan Transform (RT). The purpose of this paper is to report the research carried out till date on the algebraic properties of RT and its role in developing high- speed pattern recognition algorithms like contour detection, thinning, corners detection, lines detection, curves detection, and detection of isolated points in a given grey or colour image. Some algebraic properties like Regenerative Property of RT are found to have potential applications related to Cryptography.

2 Rajan Transform

Rajan Transform is essentially a fast algorithm developed on the lines of Decimation-In-Frequency (DIF) Fast Fourier Transform algorithm, but it is different from the DIF-FFT algorithm. Given a number sequence x(n) of length N, which is a power of 2, first it is divided into the first half and the second half each consisting of (N/2) points so that the following hold good.

$$g(j) = x(i)+x(i+(N/2)) \quad ; \quad 0 \leq j \leq N/2 \ ; \ 0 \leq i \leq N/2$$
$$h(j) = |x(i) - x(i\text{-}N/2))| \quad ; \quad 0 \leq j \leq N/2 \ ; \ (N/2) \leq i \leq N$$

Now each (N/2)–point segment is further divided into two halves each consisting of (N/4) points so that the following hold good.

$g_1(k) = g(j) + g(j + (N/4))$; $0\leq k\leq (N/4)$; $0\leq j\leq (N/4)$
$g_2(k) = |g(j) - g(j - (N/4))|$; $0\leq k\leq (N/4)$; $(N/4)\leq j\leq (N/2)$
$h_1(k) = h(j) + h(j + (N/4))$; $0\leq k\leq (N/4)$; $0\leq j\leq (N/4)$
$h_2(k) = |h(j) - h(j - (N/4))|$; $0\leq k\leq (N/4)$; $(N/4)\leq j\leq (N/2)$

This process is continued till no more division is possible. The total number of stages thus turns out to be $\log_2 N$. Let us denote the sum and difference operators respectively as + and ~.

Then the signal flow graph for the Rajan Transform of a number sequence of length 8 would be of the form shown in figure 2.1.

If x(n) is a number sequence of length $N = 2^k$; K>0, then its Rajan Transform is denoted as X(k). RT is applicable to any number sequence and it induces an isomorphism in a class of sequences, that is, it maps a domain set consisting of the cyclic and dyadic permutations of a sequence on to a range set consisting of sequences of the form X(k)E(r) where x(k) denotes the permutation invariant RT and E(r) an encryption code corresponding to an element in the domain set. This map is a one-to-one and on to correspondence and an inverse map also exists. Hence it is viewed as a transform.
Consider a sequence x(n) = 3, 8, 5, 6, 0, 2, 9, 6. Then X(k) = 39, 5, 13, 9, 13, 1, 7, 5. The signal flow graph for this transform together with the encryption key (number 0 or 1 inside the brackets) is given in figure 2.1.

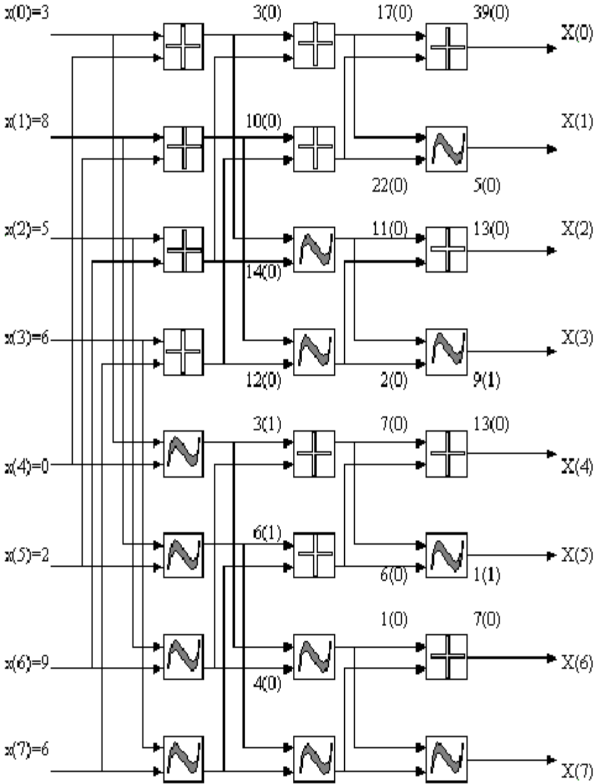


Figure 2.1: Signal flow diagram of one-dimensional 8-point RT.

Observe from the diagram that E(r) is a union of three sequences $E_1(r)$ = 0, 0, 0, 0, 1, 1, 0, 0 , $E_2(r)$ = 0, 0, 0, 0, 0, 0, 0, 1 and $E_3(r)$ = 0, 0, 0, 1, 0, 1, 0, 0. That is, $E(r) = E_1(r)E_2(r)E_3(r)$. Now, the sequence X(k)E(r) = 39, 5, 13, 9, 13, 1, 7, 5, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 1, 0, 0 is the RT of the input sequence x(n) = 3, 8, 5, 6, 0, 2, 9, 6. In general, the first point in a RT is called ‘Cumulative point Index’ (CPI).

A closed form expression for RT is described below. RT is applicable to sequences consisting of both positive as well as negative numbers. The forward transform can be obtained by operating the input sequence using a matrix operator called **R** matrix, recursively by dividing input sequence as well as by partitioning the **R** matrix in the following manner.

….2.1

$$A_{N\times 1} = R_N \times X_{N\times 1}$$
$$\text{where } R_N = \begin{bmatrix} I_{N/2} & I_{N/2} \\ -e_k I_{N/2} & e_k I_{N/2} \end{bmatrix} \text{ of size } N\times N$$

X is the column matrix containing input sequence of length N. $I_{N/2}$ is the identity matrix of size N/2, that is, half the size of the input sequence matrix X_{NX1}. Basic identity matrix $I_1=1$; and e_k is the encryption function with k as encryption value which is defined as

$e_k = (-1)^k$ such that
k = 1; for $x(i + N/2) < x(i)$; $0\leq i\leq N/2$ ... 2.2
= 0; otherwise,

and x(i) is the i^{th} element of matrix X_{NX1}. Now divide the matrix A_{NX1} into two column matrices, namely A_{00} and A_{01} of size N/2 and treating these matrices as two input sequences, compute the two new A_{10} and A_{11} matrices by using the operator $R_{N/2}$ for both the sequences. Note that size of the R matrices used is one half of the original R_N matrix. Continue the above procedure iteratively till the size of the sub matrices is reduced to two. So, we have N = 2^{n-p}, for p^{th} iteration stage, with p = 1, 2,…, n. All the encryption binary values, that is, k values thus generated during the process of computation per iteration are to be associated with the final spectrum, to recover the original sequence through inverse transformation technique. Note that this closed form expression precisely defines the algorithm stated in the beginning of this section.

3 Inverse Rajan Transform

Retrieval of the information or the signal x(n) can be done by Inverse Rajan Transform (IRT) [2], [4]. The basic requirements for the IRT computation are the RT coefficients associated with the encryption values (k values) that are generated by encryption function while computing forward RT.

The strategy adopted here is to retrace the forward transform signal flow graph. It was observed that the point-wise addition of a constant value, say K to the sample sequence x(n) while computing forward RT, changes only the DC component X_0, that is, the CPI to X_0+NK, which is the first coefficient of the newly computed RT, and the remaining spectral values remain the same of the original spectrum. For example, the RT spectrum X(k) of the sequence x(n) = 3, 8, 5, 6, 0, 2, 9, 6 is 39, 5, 13, 9, 13, 1, 7, 5. Now let us build a new sequence $x_1(n)$ = 4, 9, 6, 7, 1, 3, 10, 7 by adding K=1 to the sequence x(n) = 3, 8, 5, 6, 0, 2, 9, 6. The RT spectrum of the new sequence is $X_1(k)$ = 47, 5, 13, 9, 13, 1, 7, 5.

Now, in order to work with sequences containing negative sample values, we proceed as usual in the case of forward transform. But, the inverse transform is calculated just by adding a constant value $N(2^M\text{-}1)$ to the CPI value of the spectrum. M is the bit length required to represent the maximum quantization level of the samples and N is the length of the sequence. This constant factor $K = (2^M\text{-}1)$ is chosen such that all the maximum possible negative values of the sequence x(n) are level shifted to 0 or above. This DC shift is required, because we hide the sign of the negative values that are generated while computing the forward RT. As mentioned earlier, RT induces an isomorphism between the domain set consisting of the inverse, cyclic, dyadic and dual class permutations of a sequence on to a range set consisting of sequences of the form X(k)E(r) where X(k) denotes the permutation invariant RT and E(r) an encryption code corresponding to an element in the domain set. This map is a one-to-one and on-to correspondence and an inverse map also exists. Thus RT is viewed as a transform. Now we provide a technique for obtaining the inverse of Rajan Transform.

Inverse Rajan transform (IRT) is a recursive algorithm and it transforms a RT code X(k)E(r) of length N(1+m) where $N = 2^m$ and m is the number of stages of computation, into one of its original sequences belonging to its permutation class depending on the encryption code E(r). The computation of IRT is carried out in the following manner. First the input sequence is divided into segments each consisting of two points so that either

$$g(2j+1) = (X(2k)+X(2k+1))/2$$

$$g(2j) = \max (X(2k), X(2k+1))-g(2j+1)$$

$$\text{if } E_1 (2r)= 0 \text{ and } E_1(2r+1) = 0;$$

$$0\le j<N;\ 0\le k\le N;\ 0\le r\le N,$$

or

$$g(2j) = (X(2k)+X(2k+1))/2$$

$$g(2j+1) = \max (X(2k), X(2k+1) - g(2j)$$

$$\text{if } E_1 (2r) = 1 \text{ or } E_1 (2r+1) = 1;$$

$$0\le j\le N;\ 0\le k\le N;\ 0\le r\le N.$$

The resulting sequence is divided into segments each consisting of four points. Each 4-point segment is synthesized as per the above procedure. The resulting sequence is further divided into segments each consisting of eight points and the same procedure is carried out. This process is continued till no more division is possible. Consider X(k) = 39, 5, 13, 9, 13, 1, 7, 5. Then its IRT is x(n) = 3, 8, 5, 6, 0, 2, 9, 6. The inverse x(n) is obtained from the given X(k)E(r) as shown in figure 3.1.

The symbols ^ > and ~ respectively denote the operators average of two, maximum of two and difference of two. Note that IRT will work only in the presence of encryption sequence E(r) and for every member of the permutation class there would be a unique encryption sequence. Study of the class of encryption sequences corresponding input sequences itself is a field of active research.

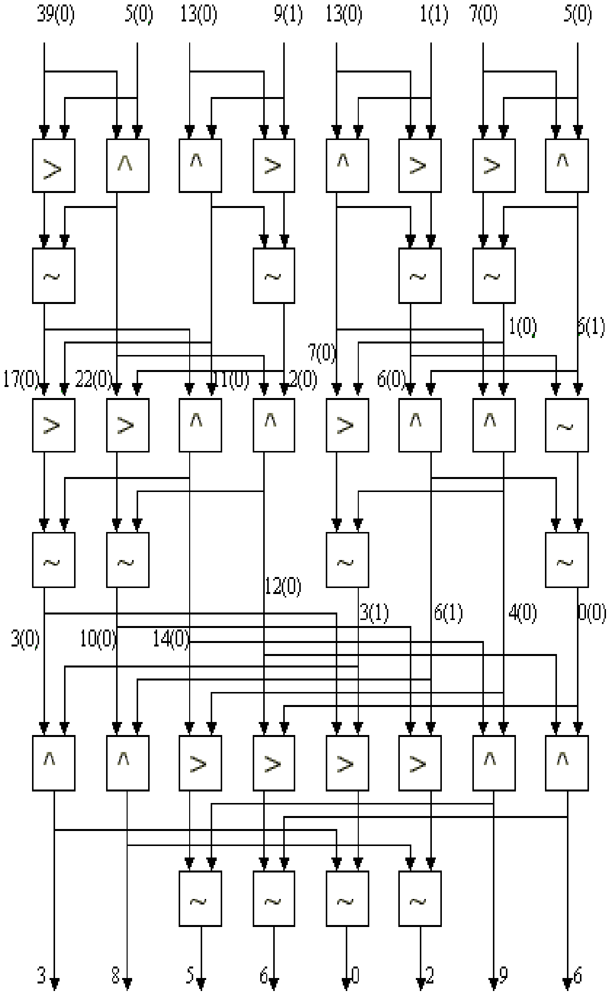


Figure 3.1: Signal flow diagram of IRT of a spectral sequence.

As already outlined that RT hides negative signs and so it is important to add $N(2^M\text{-}1)$ to X(0) in order to get back the original sequence. Let X_{NX1} be the column matrix of the RT coefficients of length N, and the associated encryption values generated per iteration are orderly arranged in the form of column matrices $E^1, E^2, \ldots, E^{n-1}$, of length N/2, where $n = \log_2 N$. For N = 8, we have E^1 and E^2, two encryption matrices of size 4X1 with binary encryption values as elements. Here X_{NX1} is the RT coefficient matrix. R_{NX1} is the intermediate matrix. A_{NX1} is the matrix with the elements positioned in their appropriate positions using the encryption matrix E^r corresponding to the current iteration. $\bar{E}^r$ is the matrix obtained by element wise complementing the E^r matrix, where r = n-1, n-2, …, 1.

Since we trace back the forward algorithm, we start initially with $N=2^p$, with p = 1, 2, 3, … , n. Hence, we use initially the encryption matrix that is generated at the last stage of forward RT computation and further use the remaining matrices in the reverse order, for IRT computation. Each encryption matrix is to be partitioned as per the requirement.

The closed form expression shown as equation 3.1 represents the Inverse Rajan Transform. For example, let us consider a sequence x(n) which has a maximum sample value of –15. The RT of this sequence is X(k).

$$[R_{N\times 1}] = \begin{bmatrix} \max(x_1, x_{1+N/2}) \\ \cdot \\ \cdot \\ \max(x_{N/2}, x_N) \\ 0 \\ \cdot \\ \cdot \\ \text{upto}\,\frac{N}{2}\,(\text{zeros}) \end{bmatrix} - \frac{1}{2}\begin{bmatrix} I_{N/2} & I_{N/2} \\ -I_{N/2} & -I_{N/2} \end{bmatrix} [X_{N\times 1}] \qquad \text{... 3.1}$$

$$[A_{N\times 1}] = \begin{bmatrix} I_{N/2} & I_{N/2} \\ I_{N/2} & I_{N/2} \end{bmatrix} [R_{N\times 1}] - \begin{bmatrix} [E^r_{N/2}][I_{N/2}] & [\overline{E^r_{N/2}}]_{L*N/2} \\ [\overline{E^r_{N/2}}][I_{N/2}] & [E^r_{N/2}][I_{N/2}] \end{bmatrix} [R_{N\times 1}]$$

……3.1

Now one can assume the value of M to be 4 so that the maximum negative sample value in the sequence is –15. Addition of +15 to each sample value in an 8-point input sequence x(n) will lead to the sequence $x_1(n)$ = x(0)+15, x(1)+15, … , x(7)+15, which is level shifted to 0 and above. Now the spectrum of this level shifted sequence $x_1(n)$ would be $X_1(k) = X_1(0), X_1(1), \ldots, X_1(7)$, where $X_1(0) = X(0)+120$, $X_1(1) = X(1), \ldots, X_1(7) = X(7)$.

A specific example would make things easy. Let us consider a sequence x(n) = 3, 8, 5, 6, 0, 2, 9, 6 and its RT, X(k) = 39, 5, 13, 9, 13, 1, 7, 5. Now addition of 15 to each sample point in x(n) would yield a sequence $x_1(n)$ = 18, 23, 20, 21, 15, 17, 24, 21 and its RT would be $X_1(k)$ = 159, 5, 13, 9, 13, 1, 7, 5. Observe that $X_1(0)$ = X(0) + 120 = 39 + 120 = 159. As outlined earlier, one can add $N(2^M-1)$ to X(0) of a spectral sequence and compute the IRT as usual. Then it is important to subtract 2^M-1 from every sample point so as to obtain the actual sample domain sequence. Let us take the case of X(k) = 39, 5, 13, 9, 13, 1, 7, 5. Now let us add 120 only to X(0) so that the spectral sequence becomes 159, 5, 13, 9, 13, 1, 7, 5. The IRT of this sequence is 18, 23, 20, 21, 15, 17, 24, 21. By subtracting 15 from each sample value we obtain the actual sequence 3, 8, 5, 6, 0, 2, 9, 6.

4 Algebraic Properties of Rajan Transform

RT has interesting algebraic properties like permutation invariance, scalar property and linear pair forming property. All such properties are explained in detail in the following.

4.1 Cyclic shift invariance property

Let us consider the same sequence x(n) = 3, 8, 5, 6, 0, 2, 9, 6. Using this sequence one can generate seven more cyclic shifted versions such as $x_{c1}(n)$ = 6, 3, 8, 5, 6, 0, 2, 9 , $x_{c2}(n)$ = 9, 6, 3, 8, 5, 6, 0, 2 , $x_{c3}(n)$ = 2, 9, 6, 3, 8, 5, 6, 0 , $x_{c4}(n)$ = 0, 2, 9, 6, 3, 8, 5, 6 , $x_{c5}(n)$ = 6, 0, 2, 9, 6, 3, 8, 5 , $x_{c6}(n)$ = 5, 6, 0, 2, 9, 6, 3, 8 and $x_{c7}(n)$ = 8, 5, 6, 0, 2, 9, 6, 3. It is obvious that the cyclic shifted version of $x_{c7}(n)$ is x(n) itself. One can easily verify that all these eight sequences have the same X(k), that is, the sequence 39, 5, 13, 9, 13, 1, 7, 5 but different E(r).

4.2 Graphical inverse invariance property

The sequence x(n) = 3, 8, 5, 6, 0, 2, 9, 6 has the graphical inverse $x^{-1}(n)$ = 6, 9, 2, 0, 6, 5, 8, 3. Using this sequence one can generate seven more cyclic shifted versions such as $x_{c1}^{-1}(n)$ = 3, 6, 9, 2, 0, 6, 5, 8 , $x_{c2}^{-1}(n)$ = 8, 3, 6, 9, 2, 0, 6, 5 , $x_{c3}^{-1}(n)$ = 5, 8, 3, 6, 9, 2, 0, 6 , $x_{c4}^{-1}(n)$ = 6, 5, 8, 3, 6, 9, 2, 0 , $x_{c5}^{-1}(n)$ = 0, 6, 5, 8, 3, 6, 9, 2 , $x_{c6}^{-1}(n)$ = 2, 0, 6, 5, 8, 3, 6, 9 and $x_{c7}^{-1}(n)$ = 9, 2, 0, 6, 5, 8, 3, 6. It is obvious that the cyclic shifted version of $x_{c7}^{-1}(n)$ is $x^{-1}(n)$ itself. One can easily verify that all these eight sequences have the same X(k), that is, the sequence 39, 5, 13, 9, 13, 1, 7, 5 but different E(r)

4.3 Dyadic shift invariance property

The term ‘dyad’ refers to a group of two, and the term ‘dyadic shift’ to the operation of transposition of two blocks of elements in a sequence. For instance, let us take x(n) = 3, 8, 5, 6, 0, 2, 9, 6 and transpose its first half with the second half. The resulting sequence $T_d^{(2)}[x(n)]$ = 0, 2, 9, 6, 3, 8, 5, 6 is the 2-block dyadic shifted version of x (n). The symbol $T_d^{(2)}$ denotes the 2-block dyadic shift operator. In the same manner, we obtain $T_d^{(4)}[T_d^{(2)}[x(n)]]$ = 9, 6, 0, 2, 5, 6, 3, 8 and $T_d^{(8)}[T_d^{(4)}[T_d^{(2)}[x(n)]]]$=6, 9, 2, 0, 6, 5, 8, 3. One can easily verify that all these dyadic shifted sequences have the same X(k), that is, the sequence 39, 5, 13, 9, 13, 7, 5 but different E(r). There is yet another way of dyadic shifting the input sequence x(n) to $T_d^{(2)}[T_d^{(4)}[T_d^{(8)}[x(n)]]]$. Let us take x(n) = 3, 8, 5, 6, 0, 2, 9, 6 and obtain the following dyadic shifts: $T_d^{(8)}[x(n)]$ = 8, 3, 6, 5, 2, 0, 6, 9 , $T_d^{(4)}[T_d^{(8)}[x(n)]]$ = 6, 5, 8, 3, 6, 9, 2, 0 and $T_d^{(2)}[T_d^{(4)}[T_d^{(8)}[x(n)]]]$ = 6, 9, 2, 0, 6, 5, 8, 3. Note that $T_d^{(2)}[T_d^{(4)}[T_d^{(8)}[x(n)]]] = T_d^{(8)}[T_d^{(4)}[T_d^{(2)}[x(n)]]]$. One can easily verify from the above that other than $T_d^{(4)}[T_d^{(2)}[x(n)]]$ and $T_d^{(8)}[x(n)]$, all other dyadically permuted sequences fall under the category of the cyclic permutation class of x(n) and $x^{-1}(n)$. This amounts to saying that the cyclic permutation class of x(n) has eight non-repeating independent sequences, that of $x^{-1}(n)$ has eight non-repeating independent sequences and the dyadic permutation classes of x(n) has two non-repeating independent sequences. To conclude, all these 18 sequences could be seen to have the same X(k). Each of these 18 sequences has an independent encryption key E(r).

4.4 Dual class invariance

Given a sequence x(n), one can construct another sequence y(n) consisting of at least one number which is not present in x(n) such that X(k) = Y(k). In such a case, y(n) is called the dual of x(n). Consider two sequences x(n) = 2, 4, 2, 2 and y(n) = 3, 1, 3, 3. Then X(k) = Y(k) = 10, 2, 2, 2. Figure 4.4.1 shows the symmetry property of the sequences x(n) and y(n). The term Differential Mean (DM) refers to the mean about which the two sequences are symmetrical to each other.

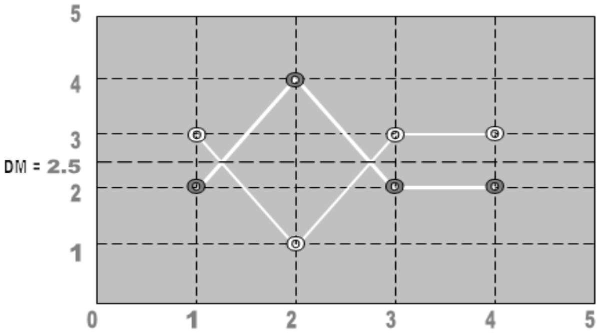


Figure 4.4.1: Symmetry property of dual sequences (DM=2.5).

The major contribution of this paper is a characterization theorem (theorem 4.4.1) for classifying sequences, which have dual sequences.

Theorem 4.4.1: A sequence of length N=2n is said to have a dual if and only if its CPI is an even number and is divisible by N/2.

Corollary 4.4.1.1: Theorem 4.4.1 advocates a necessary condition but not a sufficient condition. This means that all sequences that satisfy theorem 4.4.1 need not form valid dual pairs with other sequence vide definition of dual. For example, let us consider the sequence x(n) = 6, 8, 2, 0. This satisfies theorem 4.4.1. That is its CPI is 16 and the value of CPI/(N/2) is 8. Now its dual is computed as y(n) = 2, 0, 6, 8 , which is not a dual of x(n) as per the definition.

PROPERTIES OF DUAL SEQUENCES

1. If y(n) is a dual of x(n), then x(n) is also called the dual of y(n). Hence the pair <x(n), y(n)> is called 'dual pair'.
2. Dual of a sequence, say y(n) will necessarily exhibit geometric symmetry together with the original sequence x(n).
3. Each dual pair has a value called 'Differential Mean' (DM), which is equal to (|x(i)-y(i)|)/2; $0 \le i \le (N-1)$. It is obvious that the DM could be a real number.

4.5 Regenerative property

As per theorem 4.4.1, an arbitrary sequence x(n) of length $N=2^n$ is said to have a dual y(n) if and only if its CPI is an even number and is divisible by N/2. In other words, x(n) is said to form a dual pair with y(n). It was observed that the sequence |x(0)-y(0)|, |x(1)-y(1)|, |x(2)-y(2)|, ….., |x(N-2)-y(N-2)|, |x(N-1)-y(N-1)| also satisfies theorem 4.4.1 and is eligible to form a dual pair with yet another sequence. For example let us consider a sequence x(n) = 3, 1, 3, 3. This satisfies theorem 4.4.1 since its CPI is 10 and it is divisible by N/2, that is, 2 yielding the value 5. Now one can obtain the dual sequence y(n) = 2, 4, 2, 2 by subtracting each element of x(n) from 5. Now x(n) and y(n) form a pair and yield their 'first generation sequence', say $x_1(n)$ = 1, 3, 1, 1 , which is obtained by finding the point wise difference between the parent sequences x(n) and y(n). One can easily verify that $x_1(n)$ also satisfies theorem 4.4.1 and forms a dual pair with $y_1(n)$ = 2, 0, 2, 2. This property is termed here as 'regenerative property'. Another important observation follows. $x_1(n)$ forms a dual pair with $y_1(n)$ but this pair does not produce an offspring sequence as it was done by the pair <x(n), y(n)>.This could be easily verified by the fact that the point wise difference between $x_1(n)$ and $y_1(n)$ yields $x_1(n)$ only. Hence for brevity we call the sequences $x_1(n)$ and $y_1(n)$ as 'sterile sequences' and the pair <$x_1(n)$, $y_1(n)$> as a 'sterile pair'. This property exhibits the possibility of further research on generative and sterile pairs.

4.6 Scalar property

Let x(n) be a number sequence and ℓ be a scalar. Then the RT of ℓx(n) will be ℓX(k), where X(k) is the RT of x(n). For example, let us consider a sequence x(n) = 1, 3, 1, 2 and a scalar ℓ of value 2. Now the RT X(k) of x(n) is 7, 3, 1, 1. The RT of ℓx(n) = 2, 6, 2, 4 is 14, 6, 2, 2 which is nothing but ℓX(k).

4.7 Linearity property

In general, RT does not satisfy the linearity property. However, it was observed that for a pair x(n) and y(n) which are number sequences either in the increasing order or in the decreasing order, the linearity property holds. That is, for ℓx(n)+my(n) where ℓ and m are scalars and x(n) and y(n) are two number sequences either in the increasing or decreasing order, the RT will be ℓX(k)+mY(k), where X(k) and Y(k) are respectively the RTs of x(n) and y(n). A characterization theorem is yet to be established for categorizing pairs of sequences which would satisfy linearity property.

4.8 Linear pair forming property

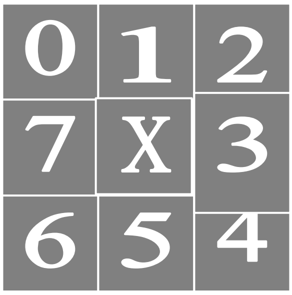
Two sequences x(n) and y(n) are said to form a linear pair when X(k)+Y(k) is the RT of x(n)+y(n), where X(k) and Y(k) are the RTs of x(n) and y(n) respectively. The symbol + denotes the point wise addition of two sequences. As outlined earlier, pairs of sequences consisting of increasing numbers of decreasing numbers only form linear pair. In general arbitrary sequences do not form linear pairs. Consequently RT could be viewed as a nonlinear transform. However, higher order RT spectra do form linear pairs, and this has been identified as a very useful property for pattern recognition

purposes. For instance let us consider two arbitrary sequences x(n) and y(n) given below:
x(n) = 2, 2, 2, 1, 6, 2, 6, 1, 2, 3, 0, 0, 2, 5, 5, 4;
y(n) = 4, 5, 3, 1, 0, 1, 4, 6, 6, 8, 0, 0, 7, 9, 0, 5;
Now the RT of x(n) denoted as X_1(k) is computed as X_1(k) = 43, 7, 7, 5, 19, 7, 7, 3, 15, 1, 1, 1, 9, 1, 3, 3 and the RT of y(n) denoted as Y_1(k) is computed as Y_1(k) = 59, 11, 21, 1, 17, 9, 9, 5, 29, 3, 11, 7, 11, 1, 9, 1. Now, z(n) = x(n) + y(n) is given by the sequence 6, 7, 5, 2, 6, 3, 10, 7, 8, 11, 0, 0, 9, 14, 16, 6, 8, 6, 8, 6, where as X_1(k) + Y_1(k) = 102, 18, 28, 6, 36, 16, 16, 8, 34, 4, 12, 8, 20, 2, 12, 4. Note that Z(k) is not equal to X_1(k) + Y_1(k). Now, the second order RT spectra of X_1(k) and Y_1(k) are respectively computed as X_2(k) = 132, 76, 72, 64, 32, 32, 28, 28, 64, 32, 36, 20, 24, 16, 20, 12 and Y_2(k) = 204, 128, 76, 56, 80, 68, 48, 44, 72, 20, 32, 20, 36, 32, 16, 12. Let z_1(n) be the sequence X_1(k) + Y_1(k). Then Z_1(k) is the RT of and z_1(n) and is computed to be 326, 194, 138, 110, 98, 86, 70, 66, 138, 70, 86, 42, 66, 62, 42, 38. Note that Z_1(k) is not equal to X_2(k) + Y_2(k) = 336, 204, 148, 120, 112, 100, 76, 72, 136, 52, 68, 40, 60, 48, 36, 24. This procedure is repeated for the third order spectra as follows: X_3(k) = 688, 128, 128, 64, 304, 96, 96, 64, 240, 0, 32, 32, 144, 32, 32, 32 and Y_3(k) = 944, 184, 336, 104, 272, 136, 144, 88, 464, 40, 176, 24, 176, 24, 144, 8 are the RTs of X_2(k) and Y_2(k) respectively. The RT of X_2(k) + Y_2(k) is computed to be 1632, 312, 464, 168, 576, 232, 240, 152, 704, 40, 208, 56, 320, 56, 176, 40 which is nothing but X_3(k) + Y_3(k). Thus it is clear from the above example that higher order RT spectra of arbitrary sequences of equal length do form linear pairs, and so the name ‘linear pair forming property’. Linear pair forming property is also called self-organizing property.

5 Pattern Recognition and Processing using Rajan Transform

5.1 Basic philosophy

Spectral domain image processing advocates the manipulation of spectral components in deciding the conditions for extracting various features in a given image. For example, high pass filtering of frequency spectral components allows the extraction of edges in a digital image, where as, low pass filtering of frequency spectral components causes smoothing of edges. Rajan Transform is basically a fast algorithm which yields spectral components of a signal sequence. Thus it is reasonable to surmise that one can as well think of using RT for spectral domain processing of digital images and one may proceed on the following lines. The given digital image is scanned by the neighbourhood window, which is shown in figure 5.1.1.



0	1	2
7	X	3
6	5	4

Figure 5.1.1: Neighbourhood window.

The image pixels corresponding to the cells in the window are labeled as x_0, $x_{1,}$ x_2, x_3, x_4, x_5, x_6 and $x_{7.}$ RT of this sequence is X(k). The CPI is divided by 8, which yields the average value. The number of spectral values in the RT sequence, which are below or above the average value, plays an important role in the processing of an image. Consider a sequence x(n) = 1, 3, 2, 1, 2, 3, 1, 2 and its RT : X(k) = 15, 3, 3, 3, 3, 1, 1, 1.

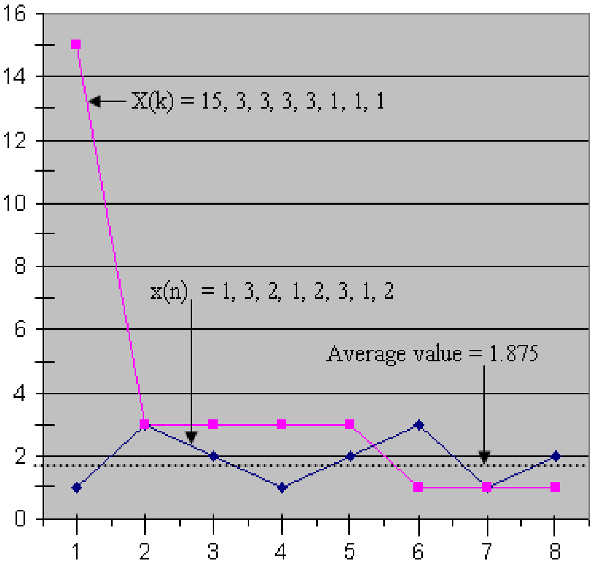


Figure 5.1.2: Pictorial representation of x(n) and X(k).

With reference to figure 5.1.2, the CPI: X_0 is 15 and the average value is X_0 / 8 = 15 / 8 = 1.875. Note that there are three spectral values less than the average value and three are above the average value, excluding X_0. The spectral components 1, 1 and 1 shown in figure 5.1.2 are less than the average value 1.875. The central pixel value of the image is updated based on such facts.

5.2 Sample algorithms for image processing and pattern recognition

In what follows, we discuss RT based fast algorithms for certain traditional image processing operations like contouring and thinning and for pattern recognition operations like corners detection, lines detection and curves detection [3].

5.2.1 Contouring algorithm

The given digital image is scanned by a 3X3 empty window. On each move, the boundary pixels of the 3X3 sub image covered by this window are stored in a temporary array and its RT found. Then the average value T is calculated by dividing the CPI by 8. Now each RT element is compared with T. The number of elements that are less than T is counted. If this count is greater than 6, then the central pixel is substituted with a value 0.

Otherwise, the window moves to the right. This procedure is continued till the entire image is scanned.

The overall effect is that the boundaries of regions that appear to be uniform are retained and the central pixels are erased. One can always choose the value of T from 3 to 7 depending on the requirement. Figure 5.2.1.1 shows the digital image of a pattern and its contour map obtained the RT based algorithm.

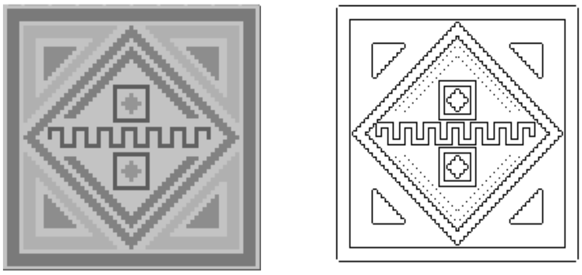


Figure 5.2.1.1: Original image and its contour map.

5.2.2 Thinning algorithm

Thinning is the complementing operation of contouring. The algorithm given for contouring is used in this case also but with the exception that instead of removing the central pixel, the boundary pixels are removed. This operation is otherwise called Onion Peeling. Repeated boundary removal leads to thinned version of the original image. Figure 5.2.2.1 shows the pattern and its thinned version obtained the above RT based thinning algorithm.

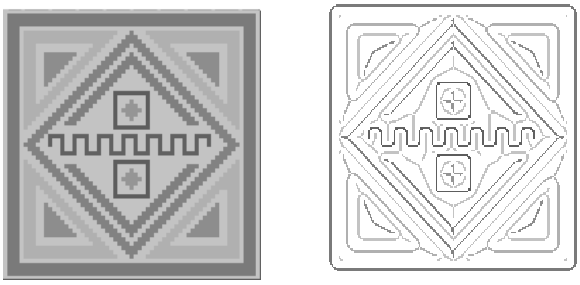


Figure 5.2.2.1: Original image and its thinned version.

5.2.3 Corners detection

On every move of the 3X3 window, the RT sequence, say, G[0],G[1],G[2],G[3],G[4],G[5],G[6],G[7] corresponding to the sequence of the numbered cell values is checked for the following conditions :

Corners due to pairs of lines subtending an angle of 45° or its integral multiples:

(1) The number of RT elements that are less than T must be 4. (2) Alternate RT elements G[0], G[2], G[4], and G[6] should be greater than T.

Corners due to pairs of lines subtending 90° or its integral multiples:

(1) The number of RT elements that are less than T must be 4. (2) The RT elements G[0], G[1], G[4] and G[5] should be greater than T.

The above procedure is repeated till the entire image is scanned. The overall effect is that the resulting image would consist only of corner points. Figure 5.2.3.1 shows the image of the contoured pattern and its processed image consisting of corners only using the above RT based corners detection algorithm.

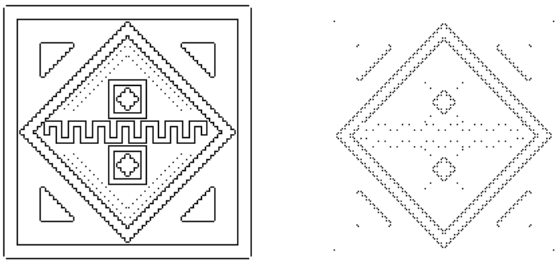


Figure 5.2.3.1: Contoured pattern and image consisting of corners.

5.2.4 Lines detection

On every move of the 3X3 window, the RT of each boundary values sequence is checked for the following condition:

(1) The number of RT elements that are less than T must be 4. (2) The first four RT elements including the CPI should be greater than T.

In such a case, the central pixel has to be a mid point of a line and so is chosen. The overall effect is that the resulting image would consist only of straight lines. Figure 5.2.4.1 shows the image of the contoured pattern and its processed image consisting of lines only using the above RT based lines detection algorithm.

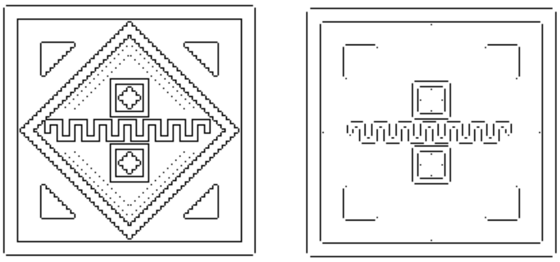


Figure 5.2.4.1: Contoured pattern and image consisting of lines.

5.2.5 Curves detection

A curve is anything other than a straight line. So, the algorithm for detecting curves advocates the invalidity of the conditions for detecting a straight line. Figure 5.2.5.1 shows the image of the contoured pattern and its processed image consisting of curves only using the above RT based curves detection algorithm.

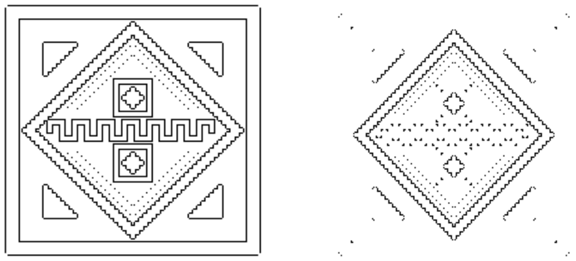


Figure 5.2.5.1: Contoured pattern and image consisting of curves

6 Conclusions

Rajan Transform has been used as a high-speed spectral domain tool to carry out pattern recognition and digital image processing operations. Some of the algorithms meant for pattern recognition using feature extraction have been presented in this paper. Recent investigations show that RT could be one of the best tools that could be used in cryptography. RT based watermarking of text and images and character recognition would yield a repertoire of tools of the future. Efforts are being made to abstract the notion of RT to Symbolic Processing of Signals and Images in the DNA computing paradigm.

7 Acknowledgements

This paper is the result of a fundamental research carried out in Pentagram Research Centre, Hyderabad. We sincerely thank Professor Rajan for his valuable guidance and encouragement to work on the concept introduced by him in the year 1997. Thanks are due to Mrs. G. Chitra, Managing Director of Pentagram Research Centre, Hyderabad, for all administrative and financial support given to our team to carry out this research.

8 References

[1] Rajan, E G, Cellular Logic Array Processing Techniques for High-Throughput Image Processing Systems, Invited paper, SADHANA, Special Issue on Computer Vision, The Indian Academy of Sciences, Vol 18, Part-2, June 1993, pp 279-300.

[2] Rajan, E G, On the Notion of Generalized Rapid Transformation, World multi conference on Systemics, Cybernetics and Informatics, Caracas, Venezuela, July 7 – 11, 1997

[3] Rajan, E.G., Symbolic Computing – Signal and Image Processing, B.S. Publications, Hyderabad, 2003.

[4] Venugopal S, Pattern Recognition for Intelligent Manufacturing Systems using Rajan Transform, MS Thesis, Jawaharlal Nehru Technological University, Hyderabad, 1999.

[5] Rajan E. G., "Symbolic Computing – Signal and Image Processing", Anshan, CBS, UK. 2004.

[6] Rajan, E.G., Ramakrishna Rao, P., Mallikarjuna Rao, G., and Venugopal, S., "Use of Generalised Rajan Transform in Recognising Patterns due to defects in the weld", JOM—8 International Conference on the Joining of Materials, American Welding Society, JOM Institute, Helsingor, Denmark, May 1997.

[7] Rajan, E.G., Prakas Rao, K., Vikram, K., Harish Babu, S., Rammurthy, D., Rajasekhar, "A Generalised Rajan Transform Based Expert System for Character Unduerstanding", World multiconference on Systemics, Cybernetics and Informatics, Caracas, Venezuela, July 7-11, 1997.

[8] Rajan, E.G., Venugopal, S., "The role of RT in the machine vision based factory automation", National conference on Intelligent manufacturing systems, February 6 – 8, 1997, Coimbatore Institute of Technology.

[9] Srinivas Rao, N., " Data Encryption and Decryption Using Generalised Rajan Transform", M. Tech Thesis, department of Electronics and Communication Engineering, Regional Engineering College, Warangal, 1997.

[10] Rajan E. G., "Fast algorithm for detecting volumetric and superficial features in 3-D images", International Conference on Biomedical Engineering, Osmania University, Hyderabad, 1994.

[11] Kishore K, Subramanyam J. V., and Rajan E. G., "Thinning Lattice Gas Automation Model for Solidification Processes, National Conference by ASME, USA, Hilton Hotel, California.

[12] Rajan E. G., "On the notion of a geometric filter and its relevance in the neighborhood processing of digital images in hexagonal grids", 4th International Conference on Control, Automation, Robotics and Vision, Westim Stamford, Singapore 1996.

A Petri-Net Approach to Refining Object Behavioural Specifications

King-Sing Cheung
University of Hong Kong
Pokfulam, Hong Kong
E-mail : ks.cheung@hku.hk

Paul Kai-On Chow
City University of Hong Kong
Tat Chee Avenue, Hong Kong
E-mail : cspchow@cityu.edu.hk

Keywords: Petri net, object-oriented system, object-oriented design, behavioural specification

Received: April 12, 2008

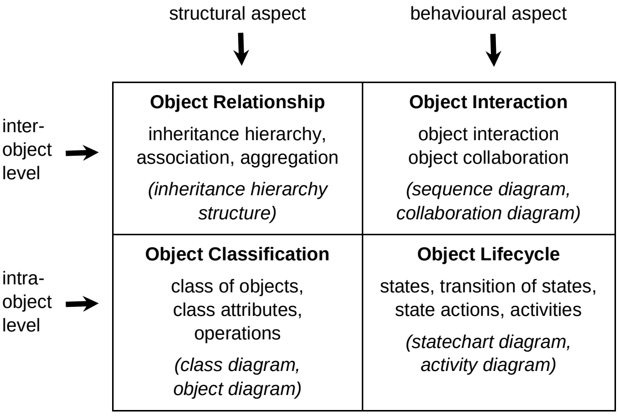
In object-oriented system design, functional requirements are given and expressed as object interaction scenarios whereas implementation is based on classes of objects. One need to derive, from the given object interaction scenarios, object-based behavioural specifications which reflect exactly these object interaction scenarios for implementation purposes. In this paper, a Petri-net-based method is proposed for the refinement. It begins with specifying each object interaction scenario as a labelled Petri net with an AMG-structure. These labelled Petri nets are synthesised into a single integrated net which represents the integrated system. By making use of the special properties of the AMG-structure, the system can be effectively analysed on its liveness, boundedness, reversibility and conservativeness. Duplicate labels are then eliminated by fusing common subnets, so as to attain a uniquely labelled net on which individual object-based behavioural specifications are obtained as projections.

Povzetek: Uporabljen je pristop Petrijevih mrež za objektne specifikacije.

1 Introduction

In the past two decades, object orientation has been an influential discipline in software engineering[1]. According to the principles of object orientation, an object is an entity that encapsulates states and behaviours. A system is considered as a collection of objects which are interacting with others in order to accomplish the system functionalities. It can be abstracted in two aspects (structure and behaviour) and two levels (intra-object and inter-object) as shown in Figure 1 [1, 2, 3, 4, 5, 6, 7, 8, 9]. Structurally, objects with the same attributes are grouped into classes while classes having common attributes are generalised to form an inheritance hierarchy. Objects exhibit different behaviours on interacting with others, thus demonstrating different object interaction scenarios. This paper investigates the behavioural aspect of objects.

In object-oriented system design, the functional requirements of a system are given by the end-users as use cases - the typical cases of how a system can be used [10, 11]. These use cases are elaborated and expressed in terms of object interaction scenarios and specified as UML sequence diagrams and collaboration diagrams [11, 12, 13, 14, 15, 16]. We need to create, from the object interaction scenarios, object-based specifications delineating the behaviours of individual objects for detailed system design and implementation.



	structural aspect	behavioural aspect
inter-object level	**Object Relationship** inheritance hierarchy, association, aggregation *(inheritance hierarchy structure)*	**Object Interaction** object interaction object collaboration *(sequence diagram, collaboration diagram)*
intra-object level	**Object Classification** class of objects, class attributes, operations *(class diagram, object diagram)*	**Object Lifecycle** states, transition of states, state actions, activities *(statechart diagram, activity diagram)*

Figure 1: An object-oriented system by aspects and abstraction levels.

In this refinement process, at least the following problems have to be tackled.

Specification constructs for object interaction scenarios being too primitive. Conventional specification constructs for object interaction scenarios lacks the formality for representing the pre-conditions and post-conditions for each event occurrence. These are however essentially required in deriving the object behavioural specifications, where the conditions, events and their causal relationships need to be explicitly specified.

[1] This paper is an extended version of the authors' paper presented at the REFINE 2006 workshop.

Different abstractions between intra-object lifecycle and inter-object interaction. It is difficult to derive individual object behaviours (within a single object lifecycle) from the object interaction scenarios (among multiple objects) because of the difference in abstraction (intra-object versus inter-object). In the literature of object-oriented system design, there is a lack of systematic approaches to solving this problem satisfactorily.

Difficulty in verifying the correctness of the object behavioural specifications. The object behavioural specifications so derived should be correct in the sense that they reflect exactly the given object interaction scenarios [4, 17, 18, 19, 20]. Without a formal method, one needs to go through all possible object interaction scenarios to ensure correctness of the specifications. The process is time-consuming.

Lack of rigorous methods for analysing the system properties. One major objective in system design is to obtain a live, bounded and reversible system - liveness implies freeness of deadlocks, and boundedness implies absence of capacity overflows, while reversibility refers to recoverability. Without a rigorous analysis method, it is difficult for one to analyse whether the outcome system design is live, bounded and reversible.

In the literature, there are only a few approaches or methods for deriving an object-based behavioural specifications from a given set of use cases or object interaction scenarios. Bordeleau proposed an approach which takes a traceable progression from use cases to the object-based state machines [21, 22]. Dano proposed an approach where the use cases are synthesised into a system design according to some mapping rules [23, 24]. However, these approaches solve only trivial issues. The system design cannot be rigorously analysed on its liveness, boundedness and reversibility. Moreover, they are themselves incomplete and insufficient in the sense that the derived object-based state machines may not reflect exactly the given use cases or object interaction scenarios.

On the other hand, there are approaches or methods which derive a system from a given set of event traces or sequences. Graubmann proposed a method for constructing an elementary net system from a set of event traces [25]. The method is based on the dependence relation between events. A set of possible states and state transitions, which are compatible to the dependence relation, are deduced. Smith proposed a method for constructing a condition-event system from a set of occurrence nets based on the concept of quotient nets [26]. Hiraishi proposed a method for constructing a Petri net from a set of firing sequences [27]. In Hiraishi's method, a language is first identified from the firing sequences. Based on the dependency relation extracted from the language, a safe Petri net is created. Lee also proposed an approach for integration of use cases using constraint-based modular Petri nets [28]. However, without concepts of object-orientation, these approaches and methods cannot be applied to object-oriented system design.

In this paper, based on Petri nets, we propose a method for refining a given set of object interaction scenarios into object-based behavioural specifications, where the above-mentioned problems can be resolved effectively. It involves the following steps :

Step 1. Each object interaction scenario is specified as a labelled Petri net (labelled net) with an AMG-structure (i.e. structurally an augmented marked graph).

Step 2. The labelled nets are synthesised into an integrated net which serves to represent the system. Based on the properties of AMG-structure, the system is analysed.

Step 3. Duplicate labels are eliminated from the integrated net, while preserving the firing sequences (event sequences).

Step 4. Individual object-based specifications are obtained as projections of the integrated net onto the objects.

Figure 2 shows an overview of the proposed method.

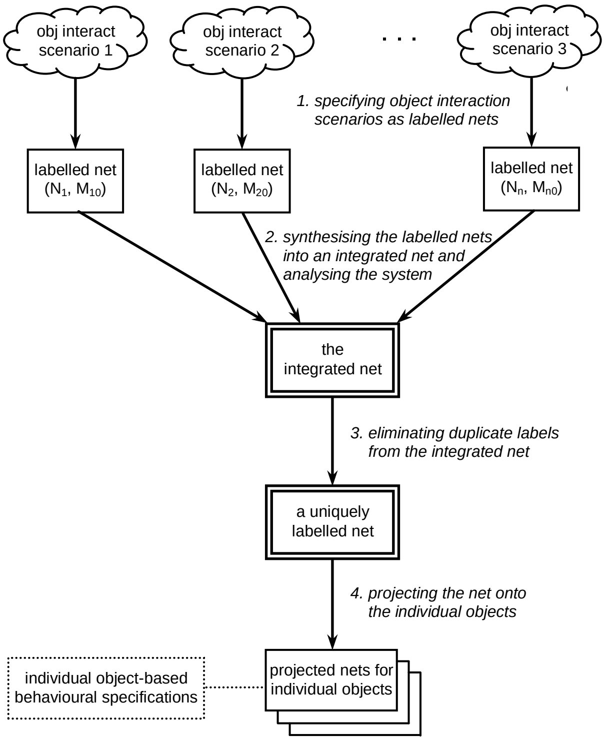

obj interact scenario 1
obj interact scenario 2
. . .
obj interact scenario 3
1. specifying object interaction scenarios as labelled nets
labelled net (N_1, M_{10})
labelled net (N_2, M_{20})
labelled net (N_n, M_{n0})
2. synthesising the labelled nets into an integrated net and analysing the system
the integrated net
3. eliminating duplicate labels from the integrated net
a uniquely labelled net
4. projecting the net onto the individual objects
individual object-based behavioural specifications
projected nets for individual objects

Figure 2: Overview of the proposed method.

Our proposed method offers a number of distinctive features.

Formal specification of object interaction scenarios. The object interaction scenarios are specified as unambiguous and semantically rich labelled nets. The partial orderings of events as well as the causal relationships between events and conditions are explicitly represented.

Effective analysis on the essential system properties. The integrated system possesses an AMG-structure. By making use of the special properties of AMG-structure, the system can be effectively analysed on its liveness, boundedness, reversibility and conservativeness.

Correctness of the derived specifications. Individual object behavioural specifications are rigorously derived from the object interaction scenarios through synthesis and projection. The specifications so obtained reflect exactly the given object interaction scenarios.

Readiness for implementation purposes. In the specifications, every condition or event is uniquely represented so that they can be readily used for implementation purposes.

The rest of this paper is organised as follows. Section 2 provides the preliminaries to be used in this paper. Section 3 introduces the AMG-structure, where augmented marked graphs and their properties are described. In Section 4, we show the formal specification of object interaction scenarios as labelled nets (Step 1 of the proposed method). In Section 5, we focus on synthesising the labelled nets into an integrated system, and analysing the system properties (Step 2 of the proposed method). Section 6 then presents an algorithm for eliminating duplicate labels from the integrated net (Step 3 of the proposed method). In Section 7, we show how individual object-based behavioural specifications are obtained as projections of the integrated net (Step 4 of the proposed method). Section 8 gives a real-life example for illustration. Section 9 concludes this paper.

It should be noted that this paper primarily focus on refinement of object-based behavioural specifications - deriving individual object-based specifications from the object interaction scenarios. The structural aspect of an object-oriented system will not be investigated.

2 Preliminaries

This section provides the preliminaries for readers who are not familiar with Petri nets [29, 30, 31, 32].

A place-transition net (PT-net) is a directed graph consisting of two sorts of nodes called places and transitions, such that no arcs connect two nodes of the same sort. Graphically, a place is denoted by a circle, a transition by a box, and an arc by a directed line. A Petri net is a PT-net with some tokens assigned to its places, and the token distribution over its places is denoted by a marking function.

Definition 2.1. A place-transition net (PT-net) is a 4-tuple $N = \langle P, T, F, W \rangle$, where P is a set of places, T is a set of transitions, $F \subseteq (P \times T) \cup (T \times P)$ is a flow relation and $W : F \rightarrow \{ 1, 2, ... \}$ is a weight function. N is said to be ordinary if and only if the range of W is $\{ 1 \}$. An ordinary PT-net is usually written as $\langle P, T, F \rangle$.

Definition 2.2. Let $N = \langle P, T, F, W \rangle$ be a PT-net. For $x \in (P \cup T)$, ${}^\bullet x = \{ y \mid (y, x) \in F \}$ and $x^\bullet = \{ y \mid (x, y) \in F \}$ are called the pre-set and post-set of x, respectively. For $X = \{ x_1, x_2, \ldots, x_n \} \subseteq (P \cup T)$, ${}^\bullet X = {}^\bullet x_1 \cup {}^\bullet x_2 \cup \ldots \cup {}^\bullet x_n$ and $X^\bullet = x_1^\bullet \cup x_2^\bullet \cup \ldots \cup x_n^\bullet$ are called the pre-set and post-set of X, respectively.

Definition 2.3. For a PT-net $N = \langle P, T, F, W \rangle$, a path is a sequence of nodes $\langle x_1, x_2, ..., x_n \rangle$, where $(x_i, x_{i+1}) \in F$ for i = 1, 2, ..., n-1. A path is said to be elementary if and only if it does not contain the same node more than once.

Definition 2.4. For a PT-net $N = \langle P, T, F, W \rangle$, a cycle is a sequence of places $\langle p_1, p_2, ..., p_n \rangle$ such that $\exists t_1, t_2, ..., t_n \in T : \langle p_1, t_1, p_2, t_2, ..., p_n, t_n \rangle$ forms an elementary path and $(t_n, p_1) \in F$.

Definition 2.5. For a PT-net $N = \langle P, T, F, W \rangle$, a marking is a function $M : P \rightarrow \{ 0, 1, 2, ... \}$ where M(p) is the number of tokens in p. (N, M_0) represents N with an initial marking M_0.

Definition 2.6. For a PT-net $N = \langle P, T, F, W \rangle$, a transition t is said to be enabled at a marking M if and only if $\forall p \in {}^\bullet t : M(p) \geq W(p,t)$. On firing t, M is changed to M' such that $\forall p \in P : M'(p) = M(p) - W(p,t) + W(t,p)$. In notation, $M [N,t\rangle M'$ or $M [t\rangle M'$.

Definition 2.7. For a PT-net (N, M_0), a sequence of transitions $\sigma = \langle t_1, t_2, ..., t_n \rangle$ is called a firing sequence if and only if $M_0 [t_1\rangle ... [t_n\rangle M_n$. In notation, $M_0 [N,\sigma\rangle M_n$ or $M_0 [\sigma\rangle M_n$.

Definition 2.8. For a PT-net (N, M_0), a marking M is said to be reachable if and only if there exists a firing sequence σ such that $M_0 [\sigma\rangle M$. In notation, $M_0 [N,*\rangle M$ or $M_0 [*\rangle M$. $[N, M_0\rangle$ or $[M_0\rangle$ represents the set of all reachable markings of (N, M_0).

Definition 2.9. Let $N = \langle P, T, F, W \rangle$ be a PT-net, where $P = \{ p_1, p_2, ..., p_m \}$ and $T = \{ t_1, t_2, ..., t_n \}$. The incidence matrix of N is an $m \times n$ matrix V whose typical entry $v_{ij} = W(p_i,t_j) - W(t_j,p_i)$ represents the change in number of tokens in p_i after firing t_j once, for i = 1, 2, ..., m and j = 1, 2, ..., n.

Liveness, boundedness, safeness, reversibility and conservativeness are well known properties of Petri nets. Liveness implies deadlock freeness. Boundedness refers to the property that the system is free from any potential capacity overflow. Safeness and conservativeness are two special cases of boundedness. Reversibility refers to the capability of a system of being recovered or reinitialised from any reachable state. In general, liveness, boundedness and reversibility collectively characterise a robust or well-behaved system.

Definition 2.10. For a PT-net (N, M_0), a transition t is said to be live if and only if $\forall M \in [M_0\rangle, \exists M' : M [*\rangle M' [t\rangle$. (N, M_0) is said to be live if and only if every transition is live.

Definition 2.11. For a PT-net (N, M_0), a place p is said to be k-bounded (or bounded) if and only if $\forall M \in [M_0\rangle : M(p) \leq k$, where $k > 0$. (N, M_0) is said to be k-bounded (or bounded) if and only if every place is k-bounded.

Definition 2.12. A PT-net (N, M_0) is said to be safe if and only if every place is 1-bounded.

Definition 2.13. A PT-net (N, M_0) is said to be reversible if and only if $\forall M \in [M_0\rangle : M [*\rangle M_0$.

Definition 2.14. A PT-net (N, M_0) is said to be well-behaved if and only if it is live, bounded and reversible.

Definition 2.15. A PT-net $N = \langle P, T, F, W \rangle$ is said to be conservative if and only if there exists a m-vector $\alpha > 0$ such that $\alpha V = 0$, where $m = | P |$ and V is the incidence matrix of N.

Figure 3 shows a PT-net (N, M_0) which is live, bounded, safe, reversible and conservative.

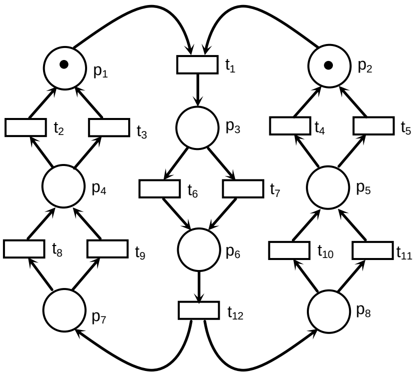


Figure 3. A live, bounded, safe, reversible and conservative PT-net.

3 AMG-structure and its properties

AMG-structure refers to an augmented-marked-graph structure. In the literature, augmented marked graphs are not well known, as compared to other sub-classes of Petri nets such as free-choice nets [33]. However, they possess many special properties pertaining to liveness, boundedness and reversibility. This section introduces augmented marked graphs and their special properties.

Definition 3.1 [34]**.** An augmented marked graph (N, M_0; R) is an ordinary PT-net (N, M_0) with a specific subset of places R, satisfying that : (a) Every place in R is marked by M_0. (b) The net (N', M_0') obtained from (N, M_0; R) by removing the places in R and their associated arcs is a marked graph. (c) For each place r ∈ R, there exist $k_r \geq 1$ pairs of transitions $D_r = \{ \langle t_{s1}, t_{h1} \rangle, \langle t_{s2}, t_{h2} \rangle, ..., \langle t_{skr}, t_{hkr} \rangle \}$, such that $r^\bullet = \{ t_{s1}, t_{s2}, ..., t_{skr} \} \subseteq T$ and $^\bullet r = \{ t_{h1}, t_{h2}, ..., t_{hkr} \} \subseteq T$ and that, for each $\langle t_{si}, t_{hi} \rangle \in D_r$, there exists in N' an elementary path ρ_{ri} connecting t_{si} to t_{hi}. (d) In (N', M_0'), every cycle is marked and no ρ_{ri} is marked.

Figure 4 shows an augmented marked graph (N, M_0; R), where R = { r_1, r_2 }.

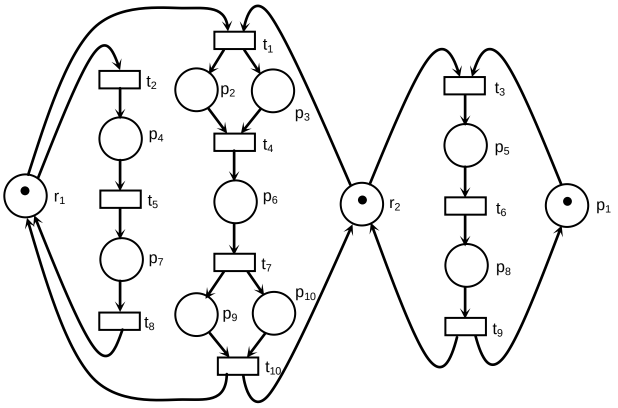


Figure 4. An augmented marked graph.

Definition 3.2. Let (N, M_0) be a PT-net, where R = { r_1, r_2, …, r_k } is the set of marked places such that $| ^\bullet r_i | > 0$ and $| r_i^\bullet | > 0$ for i = 1, 2, …, k. (N, M_0) is said to be of an AMG-structure if and only if (N, M_0; R) is an augmented marked graph.

Definition 3.3. For a PT-net (N, M_0), a set of places S is called a siphon if and only if $^\bullet S \subseteq S^\bullet$. S is said to be minimal if and only if there does not exist any siphon S' in N such that S' ⊂ S. S is said to be empty at a marking M ∈ [M_0⟩ if and only if S contains no places which are marked by M.

Definition 3.4. For a PT-net (N, M_0), a set of places Q is called a trap if and only if $Q^\bullet \subseteq {}^\bullet Q$. Q is said to be maximal if and only if there does not exist any trap Q' in N such that Q ⊂ Q'. Q is said to be marked at a marking M ∈ [M_0⟩ if and only if Q contains at least one place which is marked by M.

Property 3.1 [34]**.** An augmented marked graph is live and reversible if and only if it does not contain any potential deadlock. (Note : A potential deadlock is a siphon which would eventually become empty.)

Definition 3.5. For an augmented marked graph (N, M_0; R), a minimal siphon is called an R-siphon if and only if it contains at least one place in R.

Property 3.2 [35, 36]**.** An augmented marked graph (N, M_0; R) is live and reversible if and only if no R-siphons eventually become empty.

Property 3.3 [34, 35, 36]**.** An augmented marked graph (N, M_0; R) is live and reversible if every R-siphon contains a marked trap.

For the augmented marked graph (N, M_0; R) shown in Figure 4, each R-siphon contains a marked trap. (N, M_0; R) is live and reversible.

Definition 3.6 [37]**.** Suppose an augmented marked graph (N, M_0; R) is transformed into a PT-net (N', M_0') as follows. For each r ∈ R, where $D_r = \{ \langle t_{s1}, t_{h1} \rangle, \langle t_{s2}, t_{h2} \rangle, ..., \langle t_{skr}, t_{hkr} \rangle \}$, replace r with a set of places { q_1, q_2, ..., q_{kr} }, such that $M_0'[q_i] = M_0[r]$ and $q_i^\bullet = \{ t_{si} \}$ and $^\bullet q_i = \{ t_{hi} \}$ for i = 1, 2, ..., k_r. (N', M_0') is called the R-transform of (N, M_0; R).

Property 3.4 [37]**.** Let (N', M_0') be the R-transform of an augmented marked graph (N, M_0; R). (N, M_0; R) is bounded and conservative if and only if every place in (N', M_0') belongs to a cycle.

Figure 5 shows the R-transform (N', M_0') of the augmented marked graph (N, M_0; R) in Figure 4. (N', M_0') is bounded, where every place belongs to a cycle. (N, M_0; R) is bounded and conservative.

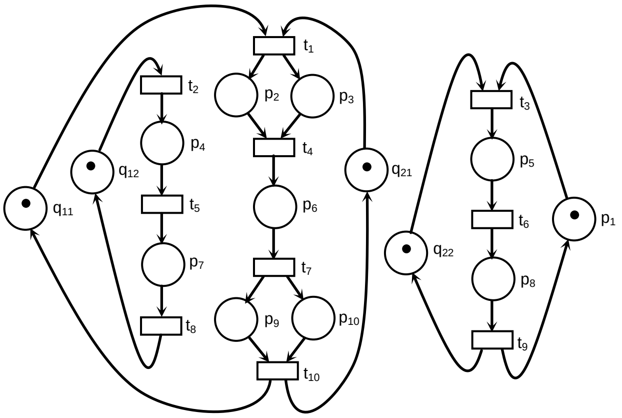


Figure 5. The R-transform of the augmented marked graph in Figure 4.

4 Specifying object interaction scenarios as labelled nets

In this section, we show how an object interaction scenario can be formally specified as a labelled net with an AMG-structure (Step 1 of our proposed method).

A labelled net is a Petri net, where labels are assigned to places and transitions. Usually, places are labelled by conditions to denote specific system substates where the conditions hold, and transitions by events to denote specific occurrences of the events.

Definition 4.1. A labelled Petri net (or labelled net) is a 7-tuple $N = \langle P, T, F, C, E, Lp, Lt \rangle$, where $\langle P, T, F \rangle$ is an ordinary PT-net, C is a set of condition labels, E is a set of event labels, $Lp : P \rightarrow C$ is a function for assigning a condition label to every place, and $Lt : T \rightarrow E$ is a function for assigning an event label to every transition.

Definition 4.2. Let $N = \langle P, T, F, C, E, Lp, Lt \rangle$ be a labelled net. A place p is said to be uniquely labelled in N if and only if $\forall p' \in P : (Lp(p') = Lp(p)) \Rightarrow (p' = p)$. A transition t is said to be uniquely labelled in N if and only if $\forall t' \in T : (Lt(t') = Lt(t)) \Rightarrow (t' = t)$. N is said to be uniquely labelled if and only if all places and transitions are uniquely labelled.

Figure 6 shows a typical labelled net. Places p_3, p_4, p_5, p_6, p_9 and p_{10} are uniquely labelled, whereas p_1, p_2, p_7 and p_8 are not, as for example, condition label c_1 appears in p_1 and p_7, and c_2 in p_2 and p_8. Transitions t_3, t_4 and t_5 are uniquely labelled, whereas t_1, t_2, t_6 and t_7 are not, as for example, event label e_1 appears in t_1 and t_6, and e_2 in t_2 and t_7. Therefore, the labelled net is not uniquely labelled.

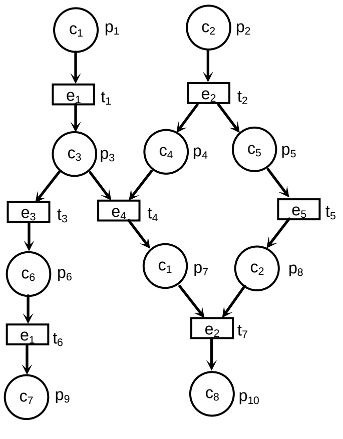


Figure 6. A labelled net which is not uniquely labelled.

For an object interaction scenario specified as a labelled net, the location where an event occurs is represented by a transition and the location of a condition by a place. The semantic meanings of conditions and events are denoted by the labels of the places and transitions respectively. For an event to occur, some conditions must be fulfilled in advance and some afterwards. These pre-conditions and post-conditions are represented by the pre-set and post-set of the transition representing the event.

Step 1 of the proposed method is to specify the given object interaction scenarios as labelled nets with an AMG-structure. Consider an object-oriented system involving two objects, x and y, of classes X and Y respectively. There are three typical interaction scenarios exhibited by x and y, specified as UML sequence diagrams and collaboration diagrams (BRJ99, RJB99) in Figure 7. In conventional UML sequence diagrams and collaboration diagrams, there are no formal notations for denoting the pre-condition and post-condition of each event occurrence in an object interaction scenario. Therefore, for an explicit representation of the causal relationship between events and conditions, appropriate condition labels are appended to these diagrams.

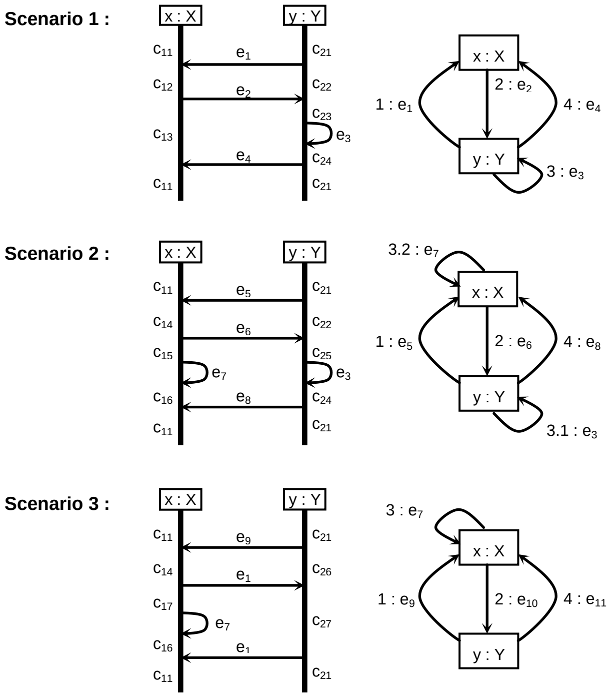
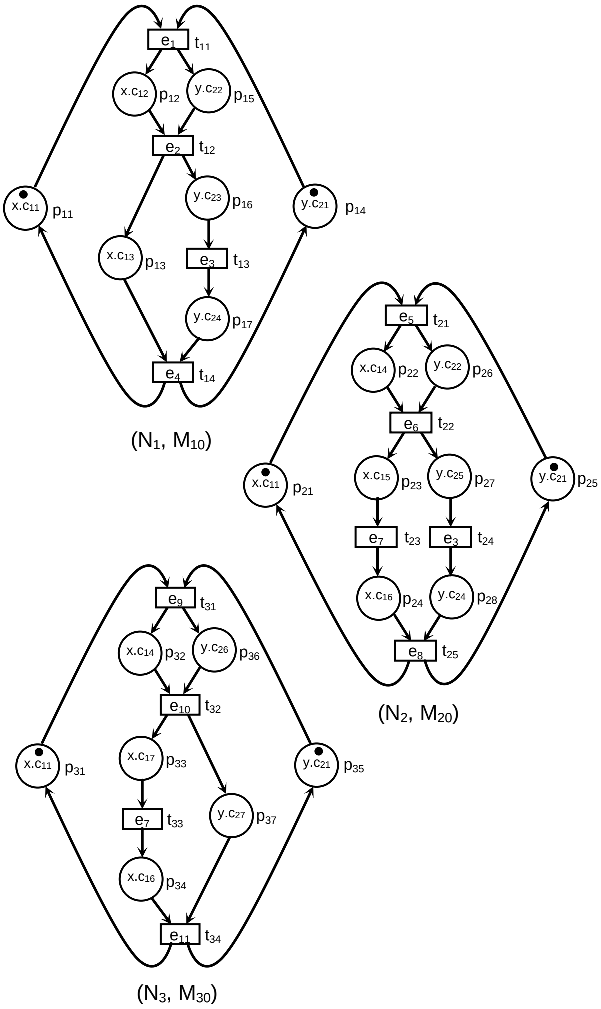


Figure 7. Object interaction scenarios in UML sequence diagrams and collaboration diagram.

Figure 8 shows object interaction Scenarios 1, 2 and 3, specified as labelled nets (N_1, M_{10}), (N_2, M_{20}) and (N_3, M_{30}) respectively. They all are of AMG-structure.

(N_1, M_{10}) is constructed for representing scenario 1 as follows. For each location of a condition, a new place with a proper condition label is created. For example, p_{11} denotes a location of condition c_{11} for object x, so condition label x.c_{11} is assigned to p_{11}. For each event occurrence, a new transition with a proper event label is constructed. For example, t_{11} denotes an occurrence of event e_1, so event label e_1 is assigned to t_{11}. The event occurrence has a pre-condition x.c_{11} and a post-condition x.c_{12}. Hence, $^\bullet t_{11} = \{ p_{11} \}$ and $t_{11}^\bullet = \{ p_{12} \}$. Arcs between p_{11} (pre-condition) and t_{11} and between t_{11} and p_{12} (post-condition) are appended for denoting their causal relationships. The rest locations of conditions and events are created accordingly. Following the same rules, (N_2, M_{20}) and (N_3, M_{30}) are constructed for representing scenarios 2 and 3, respectively.



(N_1, M_{10})

(N_2, M_{20})

(N_3, M_{30})

Figure 8. Labelled nets representing the object interaction scenarios in Figure 7.

5 Synthesising and analysing the integrated system

After specifying the object interaction scenarios as augmented marked graphs (Step 1 of the proposed method), we synthesise these scenarios into an integrated system. In principle, a scenario portrays partial system behaviours of how the objects are interacted in order to perform a specific functionality. These augmented marked graphs are essentially partial system behavioural specifications which are to be synthesised together to form a single coherent whole.

This section describes Step 2 of our proposed method - the synthesis of labelled nets into an integrated net which represents the integrated system, and analysis of the system. The synthesis is based on the authors' earlier work on use-case-driven system design [38]. It is made by fusing those places with refer to the same system initial state or condition. The integrated net so obtained is of AMG-structure, so its liveness, boundedness, reversibility and conservativeness can be effectively analysed by making use of the special properties of augmented marked graphs.

Consider the labelled nets (N_1, M_{10}), (N_2, M_{20}) and (N_3, M_{30}) in Figure 8. Places p_{11} in (N_1, M_{10}), p_{21} in (N_2, M_{20}) and p_{31} in (N_3, M_{30}) refer to the same condition x.c_{11}. Also, places p_{15} in (N_1, M_{10}), p_{24} in (N_2, M_{20}) and p_{34} in (N_3, M_{30}) refer to the same condition y.c_{21}. Hence, p_{11}, p_{21} and p_{31} are fused into one place p_{41}, and p_{15}, p_{24} and p_{34} into p_{42}.

Figure 9 then shows the integrated net (N, M_0) obtained after synthesising (N_1, M_{10}), (N_2, M_{20}) and (N_3, M_{30}). (N, M_0) is of an AMG-structure, meaning that it is structurally an augmented marked graph (N, M_0; R), where R = { p_{41}, p_{42} }.

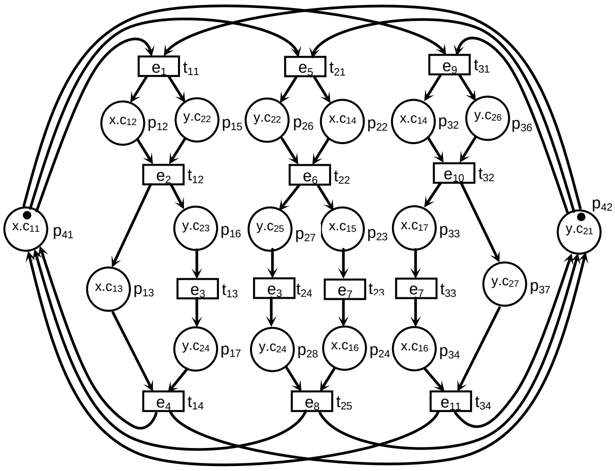


Figure 9. The integrated net obtained by synthesising the labelled nets in Figure 8.

For (N, M_0; R), every R-siphon contains a marked place, and hence, would never become empty. According to Properties 3.2 and 3.3, (N, M_0; R) is live and reversible. Since every place in its R-transform is covered by cycles, according to Property 3.4, (N, M_0; R) is also bounded and conservative. Therefore, it can be concluded that the integrated system is well-behaved.

6 Eliminating duplicate labels from the integrated net

Consolidating the object interaction scenarios, the integrated net obtained from Step 2 of the proposed method serves to represent the system as a coherent integrated whole. In general, this integrated net is not necessarily uniquely labelled. For the integrated net (N, M_0) in Figure 9 for example, places p_{15} and p_{26} have the same condition label y.c_{22}, and transitions t_{13} and t_{24} have the same event label e_3. This reflects the fact that the locations or conditions for occurrence of the same event may be different at different moments within a scenario or among different scenarios. Yet, every condition is eventually implemented as a unique system substate and every event as a unique operation. Therefore, in order for the integrated net to be effectively used for implementation purposes, it need to be uniquely labelled where all the duplicate condition labels and duplicate event labels are eliminated.

The elimination cannot be done by just fusing places with the same condition label, and transitions with the same event label. This is because the resulting net may exhibit firing sequences different from the original ones. In other words, the system behaviours may be distorted. Step 3 of the proposed method is to eliminate all duplicate labels while preserving the original firing sequences (event sequences). This section describes this step in details.

Definition 6.1. Let S be a uniquely labelled subnet of a labelled net N. The pattern of S in N, denoted as Patt(N, S), is a condition-event net with an identical structure and label allocation as S while ignoring the identities of places and transitions of S.

Definition 6.2. Let L_x and L_y be patterns of subnets in a labelled net. $L_x \cup L_y$ and $L_x \cap L_y$ denote the union and intersection of L_x and L_y, respectively. $L_x \setminus L_y$ denotes the displacement of L_x from L_y. L_x and L_y are said to be disjoint if and only if $L_x \cap L_y = \emptyset$.

Definition 6.3. For a labelled net N, a uniquely labelled subnet S is called a common subnet if and only if there exists at least one uniquely labelled subnet S' such that S' ≠ S and Patt(N, S') = Patt(N, S). Let S be a pattern of the common subnets in N. [N, L] = { S | Patt(N, S) = L } represents the group of common subnets having the same pattern L.

Definition 6.4. For a subnet $S = \langle P', T', F' \rangle$ of a PT-net, $Pre(S) = ({}^\bullet P' \setminus T') \cup ({}^\bullet T' \setminus P')$ is called the pre-set of S, $Post(S) = (P'^\bullet \setminus T') \cup (T'^\bullet \setminus P')$ is called the post-set of S, $Head(S) = Pre(S)^\bullet \cap (P' \cup T')$ is called the head of S, and $Tail(S) = {}^\bullet Post(S) \cap (P' \cup T')$ is called the tail of S.

Definition 6.5. A subnet S of a PT-net $N = \langle P, T, F \rangle$ is said to be of PP-type if and only if $Head(S) \subseteq P$ and $Tail(S) \subseteq P$.

Figure 10 shows a uniquely labelled subnet S which is PP-type. Figure 11 shows the pattern of S.

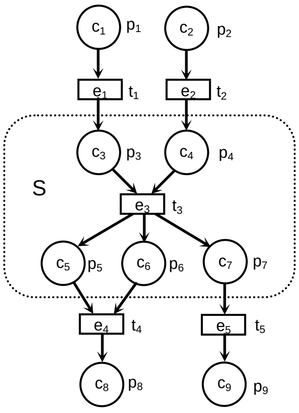


Figure 10. A uniquely labelled subnet S of a labelled net.

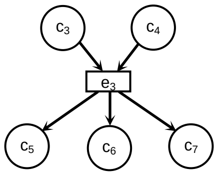


Figure 11. Pattern of S of the labelled net in Figure 10.

We propose to eliminate duplicate labels by fusion of common subnets, as outlined below.

Identify groups of common subnets for fusion. These groups of common subnets need to be maximal and disjoint for two reasons. First, the net obtained after the fusion will become uniquely labelled. Second, the number of groups of common subnets for fusion can be reduced to minimum as they are maximal.

Transformation of common subnets. For preservation of firing sequences, common subnets are transformed before fusion. Based on coloured Petri nets [39], a unique colour is assigned to each common subnet as colour labels of its ingoing and outgoing arcs. A token flowing into the common subnet is coloured according to the colour label of the ingoing arc. Its colour is reset as it flows out via the same colour-labelled outgoing arc. Besides, the subnets are converted to PP-type.

Fusion of transformed common subnets. The transformed common subnets of each group are fused into a single subnet. A uniquely labelled net is ultimately obtained.

The following algorithm formally describes the elimination process. A detailed elaboration of the elimination process can be found in the authors' earlier work [40].

Elimination of Duplicate Labels from a Labelled Net

```
1. Identify maximal disjoint groups of common subnets :
   1.1 Find all possible common subnets from N. Let $\Im = \{ L_1, L_2, ..., L_n \}$ be their patterns.
   1.2 Retain only the maximal patterns : Remove any $L_i$ from $\Im$ if there exists $L_j \in \Im$ such that $L_i$ is a sub-pattern of $L_j$ and $\forall S_i \in [N, L_i], \exists S_j \in [N, L_j]$ : $S_i$ is a subnet of $S_j$.
   1.3 Make the overlapping patterns disjoint : For every $L_i, L_j \in \Im$ such that $L_i \neq L_j$ and $L_i$ and $L_j$ are not disjoint, set $\Im = (\Im - \{ L_i, L_j \}) \cup \{ L_i \cap L_j \} \cup \{ L_i \setminus L_j \} \cup \{ L_j \setminus L_i \}$.
   1.4 Categorise the common subnets of N into groups { [N, $L_i$], $L_i \in \Im$ }.
2. For each group of common subnets [N, $L_i$] :
   2.1 Convert each subnet $S \in [N, L_i]$ if S is not of PP-type :
       2.1.1 For each transition $t_i \in Head(S)$ : (a) Create dummy transition $t_i'$ with unique label $\varepsilon_i$, dummy place $p_i'$ with label $\varphi_i$, and arcs $(t_i', p_i')$ and $(p_i', t_i)$. (b) For each $p \in {}^\bullet t_i$ : Remove arc $(p, t_i)$, and then create arc $(p, t_i')$. (c) Re-define S by including $p_i'$ and $(p_i', t_i)$.
       2.1.2 For each transition $t_j \in Tail(S)$ : (a) Create dummy transition $t_j'$ with unique label $\varepsilon_j$, dummy place $p_j'$ with label $\varphi_j$, and arcs $(t_j, p_j')$ and $(p_j', t_j')$. (b) For each $p \in t_j^\bullet$ : Remove arc $(t_j, p)$, and then create arc $(t_j', p)$. (c) Re-define S by including $p_j'$ and $(t_j, p_j')$.
   2.2 Assign a unique colour label $\kappa$ for each subnet $S \in [N, L_i]$ :
       2.2.1 For each arc $(t_i, p_i)$ where $t_i \in Pre(S)$ and $p_i \in Head(S)$ : Assign colour label $\kappa$ to $(t_i, p_i)$.
       2.2.2 For each arc $(p_j, t_j)$ where $p_j \in Tail(S)$ and $t_j \in Post(S)$ : Assign colour label $\kappa$ to $(p_j, t_j)$.
   2.3 Fuse the common subnets in [N, $L_i$] into one single subnet.
```

We apply the algorithm for eliminating the duplicate labels for the integrated net (N, M_0) in Figure 9. Figure 12 shows the uniquely labelled net (N', M_0') so obtained.

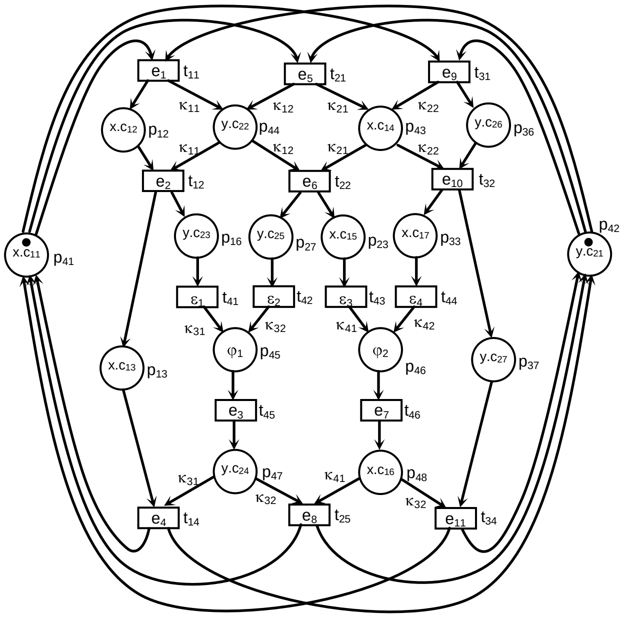


Figure 12. The uniquely labelled net obtained after eliminating duplicate labels from the integrated net in Figure 9.

7 Obtaining object-Based behavioural specifications

In this section, we show Step 4 of our proposed method - to obtain the individual object-based behavioural specifications. These individual object-based behavioural specifications are obtained by projecting the integrated net onto individual objects.

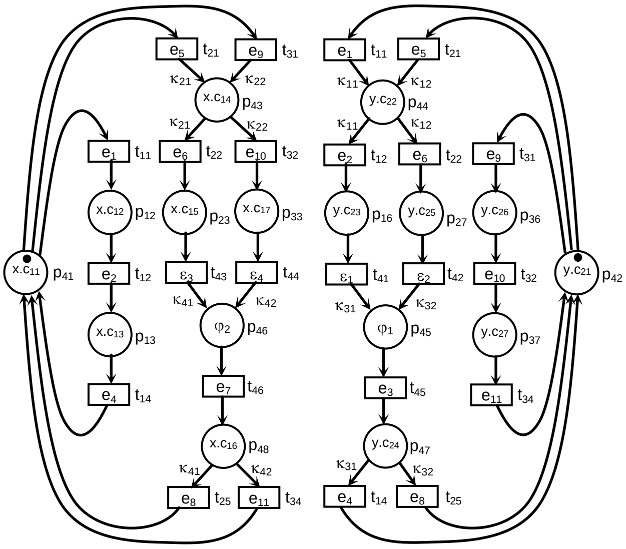
The projection is made by ignoring those places, transitions and arcs which are irrelevant to the object concerned. The projected net so obtained serves as the object behavioural specifications.

Consider the integrated net (N', M_0') in Figure 12. The projection onto object x is obtained as follows. We keep those places with object label x (including dummy places) and those transitions (including dummy transitions) having at least one input place or output place labelled by x, as well as their associated arcs. Similarly, for the projection onto object y, we keep those places with object label y (including dummy places) and those transitions (including dummy transitions) having at least one input place or output place labelled by y, as well as their associated arcs.

Figure 13 shows the projections (N_x, M_{x0}) and (N_y, M_{y0}) obtained by projecting the net (N', M_0') in Figure 12 onto objects x and y, respectively. (N_x, M_{x0}) and (N_y, M_{y0}) are uniquely labelled, simply because (N', M_0') is uniquely labelled. They serve as the behavioural specifications for objects x and y, where conditions and events are uniquely represented.

8 Real-life example

This section presents a real-life example to further illustrate the refinement process.



(N_x M_{x0}) (N_y M_{y0})

Figure 13. The nets obtained by projecting the integrated net in Figure 12 onto objects x and y.

The real-life example is an Office Access Control System. The system is briefly described as follows. It is a system used in a company for controlling staff accesses to its 30+ offices and laboratories. Among these offices and laboratories, some can be accessed by all staff while some others by authorised staff only and/or during specified time periods only. For controlling the staff access, every entrance is implemented with a card-reader, an emergency switch and an electronic lock, all being connected to a centralised server. The server maintains the access privileges and validates every access to the offices/laboratories. There are three typical cases for each request for access.

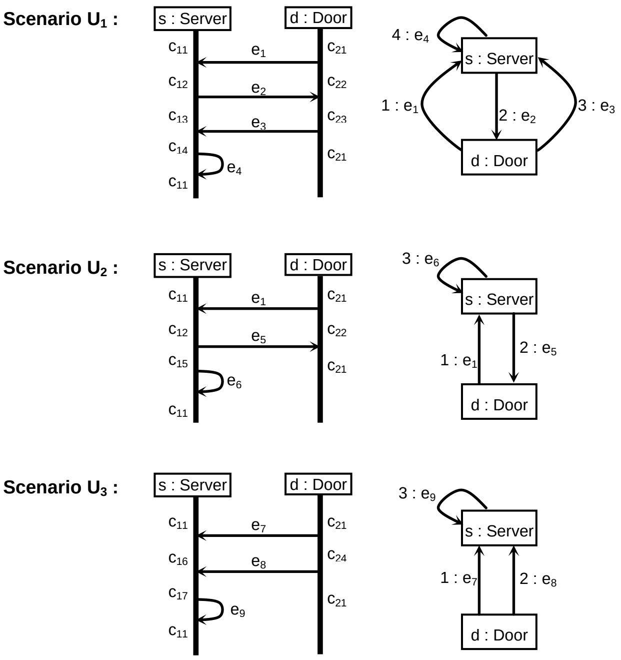
Authorised access (U_1). A staff member wants to access an office/laboratory. He/She presents his/her staff card via a card-reader. Access is granted. The door is unlocked for five seconds and then re-locked.

Unauthorised access (U_2). A staff member wants to access an office/laboratory. He/She presents his/her staff card via a card-reader. Access is not granted. The door is locked.

Emergency access (U_3). A staff member wants to access an office/laboratory for emergency. He/She presses the emergency key. The door is unlocked immediately, until it is reset by a security officer.

From the object-oriented perspectives, the server (s : Server) and doors (d : Door) are objects of the Office Access Control System. They are interacting with each other in order to perform the above system functionalities. There are three object interaction scenarios, corresponding to U_1, U_2 and U_3.

Figure 14 shows these object interaction scenarios specified as UML sequence diagrams and collaboration diagrams, where appropriate condition labels are appended for denoting the pre-conditions and post-conditions for each event occurrence.



Legends for condition labels :

c_{11}	Server is ready
c_{12}	Server is processing access request
c_{13}	Server is waiting for re-lock
c_{14}	Server is writing log (successful access)
c_{15}	Server is writing log (unsuccessful access)
c_{16}	Server is waiting for emergency reset
c_{17}	Server is writing log (emergency access)
c_{21}	Door is locked
c_{22}	Door is waiting for response
c_{23}	Door is unlocked (successful access)
c_{24}	Door is unlocked (emergency access)

Legends for event labels :

e_1	Request for access is received
e_2	Access is granted
e_3	Time expires after access granted
e_4	Successful access is committed
e_5	Access is not granted
e_6	Unsuccessful access is committed
e_7	Request for emergency access is received
e_8	Door is reset to normal
e_9	Emergency access is committed

Figure 14. Object interaction scenarios specified as UML sequence diagrams and collaboration diagrams (the Office Access Control System).

Step 1 of the proposed method is to specify object interaction scenarios as labelled nets. Figure 15 shows the labelled nets (N_1, M_{10}), (N_2, M_{20}) and (N_3, M_{30}) representing the object interaction scenarios for U_1, U_2 and U_3, respectively.

Step 2 of the proposed method is to synthesise the labelled net into an integrated system, and analyse the system on its liveness, boundedness, reversibility and conservativeness. (N_1, M_{10}), (N_2, M_{20}) and (N_3, M_{30}) are synthesised into an integrated net (N, M_0) by fusing those places which refer to the same system initial states or conditions : Places p_{11}, p_{21} and p_{31} are fused into one place p_{41}, and p_{15}, p_{24} and p_{34} into p_{42}. Figure 16 shows the integrated net (N, M_0) so obtained.

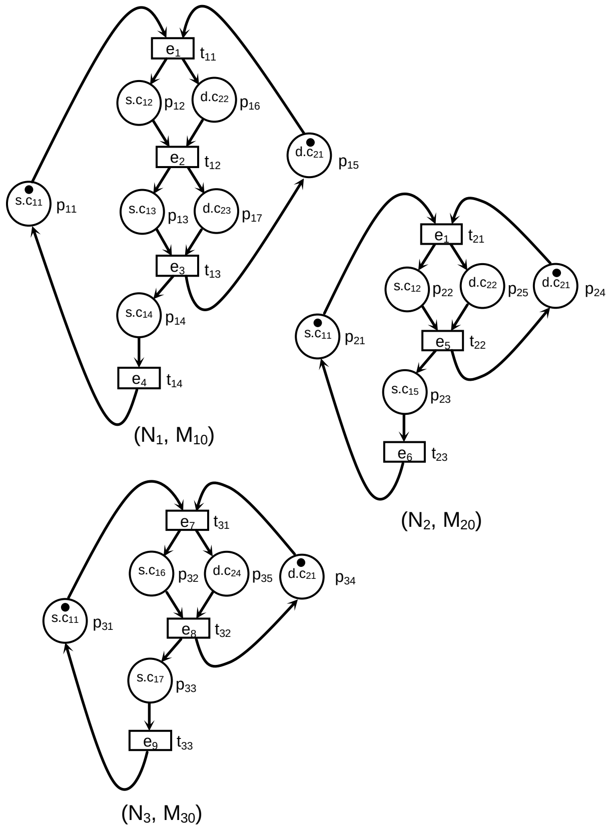


Figure 15. Labelled nets representing the object interaction scenarios in Figure 14.

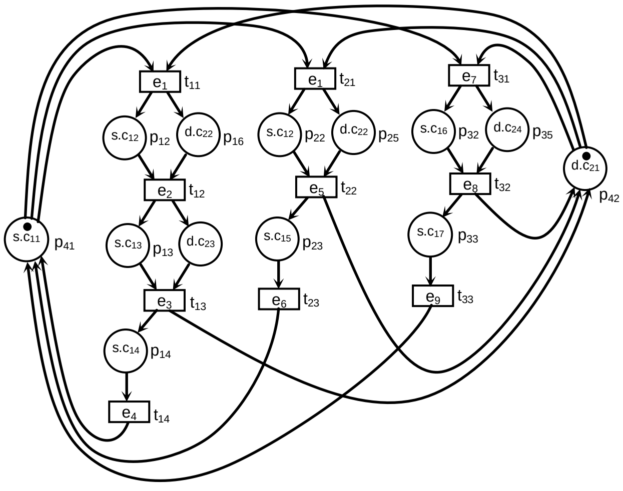


Figure 16. The integrated net obtained by synthesising the labelled nets in Figure 15.

The integrated net (N, M_0) is of an AMG-structure. Let R = { p_{41}, p_{42} }. For (N, M_0; R), every R-siphon contains a marked place and hence would never become empty. According to Properties 3.2 and 3.3, (N, M_0; R) is live and reversible. Since every place in its R-transform is covered by cycles, according to Property 3.4, (N, M_0; R) is also bounded and conservative. Therefore, it may be concluded that the Office Access Control System is well-behaved.

As shown in Figure 16, (N, M_0) is not uniquely labelled as it contains duplicate labels. For example, place p_{12} and p_{22} have the same condition label s.c_{12} and transitions t_{11} and t_{21} have the same event label e_1. Since every condition is eventually implemented as a unique substate and every event as a unique operation, in order for the integrated net to be effectively used for implementation purposes, these duplicate labels must be eliminated.

Step 3 of the proposed method is to eliminate duplicate condition labels and duplicate event labels from the integrated net (N, M_0) through the fusion of common subnets. We perform this elimination process by applying the algorithm described in Section 6. Figure 17 shows the uniquely labelled net (N', M_0') so obtained.

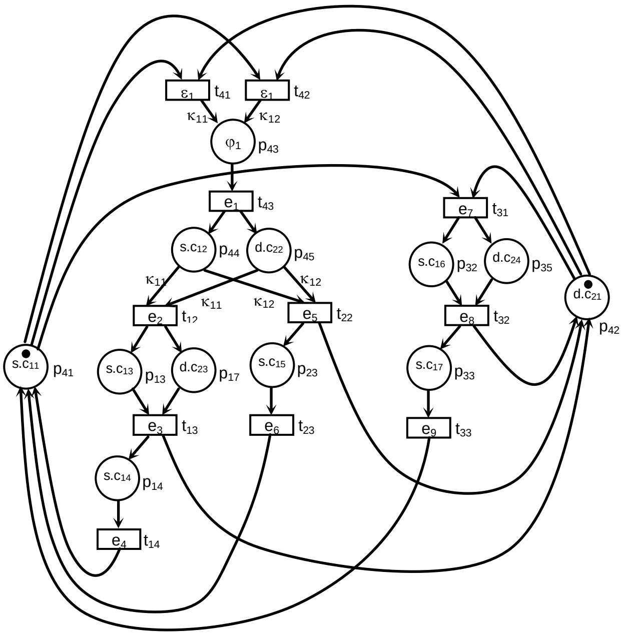
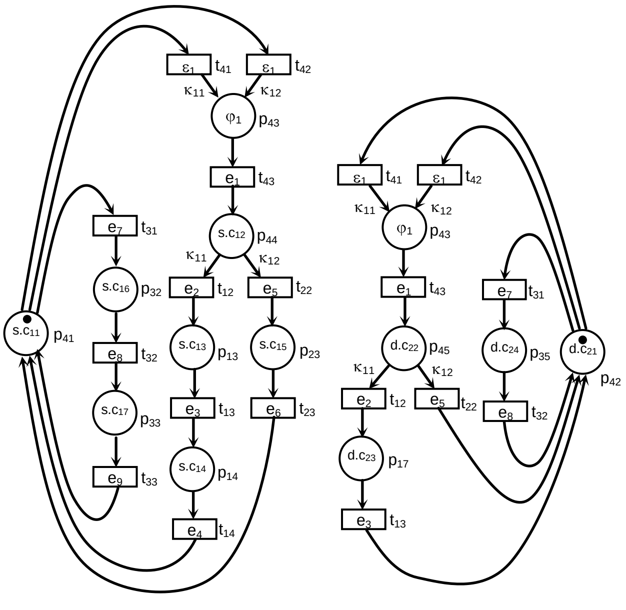


Figure 17. The uniquely labelled net obtained after eliminating duplicate labels from the integrated net in Figure 16.

Step 4 of the proposed method is to obtain the individual object-based behavioural specification as projections of the integrated net onto the objects. The projection is made by ignoring those places, transitions and arcs which are irrelevant to the object concerned.

Consider the integrated net (N', M_0') in Figure 17. For the projection onto object s (the server object), we keep those places with object label s (including dummy places) and those transitions (including dummy transitions) having at least one input place or output place labelled by s, as well as their associated arcs. Similarly, for the projection onto object d (the door object), we keep those places with object label d (including dummy places) and those transitions (including dummy transitions) having at least one input place or output place labelled by d, as well as their associated arcs. Figure 18 shows the projections (N_s, M_{s0}) and (N_d, M_{d0}) obtained by projecting the integrated net (N', M_0') in Figure 17 onto objects s and d, respectively.



(N_s M_{s0}) (N_d, M_{d0})

Figure 18. The nets obtained by projecting the integrated net in Figure 17 onto objects s and d.

As the integrated net (N', M_0') is uniquely labelled, its projections (N_s, M_{s0}) and (N_d, M_{d0}) are also uniquely labelled, where every condition or event is uniquely represented. (N_s, M_{s0}) and (N_d, M_{d0}) then serve as the behavioural specifications for the server (s : Server) and door (d : Door) objects, respectively.

9 Conclusion

One of the most difficult tasks in object-oriented system design is to obtain individual object-based behavioural specifications from a given set of object interaction scenarios. Not only conventional specification constructs for object interaction scenarios are too primitive to represent the partial ordering of events and the causal relationship between the events and conditions, there also involves different abstractions between intra-object lifecycle and inter-object interaction. Moreover, we have to ensure that the derived object-based behavioural specifications reflect exactly the given object interaction scenarios and that the system is well-behaved.

We proposed a Petri-net-based method for refining a given set of object interaction scenarios into individual object-based behavioural specifications. By specifying the object interaction scenarios as labelled nets with an AMG-structure and synthesising them into an integrated net, we analyse the system, based on the special properties of augmented marked graphs. For unique representation of events and conditions, an algorithm is applied to the integrated net to eliminate duplicate condition labels and event labels while preserving the event sequences. Object-based behavioural specifications are then obtained as projections of the integrated nets onto the objects. The whole refinement process has been described, elaborated and illustrated using the Office Access Control System example.

The proposed method offers a number of distinctive features. First, object interaction scenarios are formally specified as labelled nets which are unambiguous and semantically rich for an explicit representation of events, conditions and their causal relationships. Second, object-based behavioural specifications are rigorously derived from the given object interaction scenarios through systematic synthesis and projection. The behavioural specifications so obtained would reflect exactly the given object interaction scenarios. Third, liveness, boundedness, reversibility and conservativeness of the system can be effectively analysed by making use of the special properties of augmented marked graphs. Fourth, every event or condition is uniquely represented in the behavioural specifications so that the specifications can be readily used for implementation purposes.

With a strong theoretical foundation of Petri nets, the proposed method can be effectively used for refining object-based behavioural specifications from a set object interaction scenarios. It resolves a number of problems perplexing the designers of object-oriented systems, such as the lack of formality in specifying object interaction scenarios and the difficulty of ensuring the correctness of object behavioural specifications. The latter is especially important for systems involving shared resources, where erroneous situations such as deadlock and capacity overflow are easily induced. The proposed method can be implemented as tool to support object-oriented system design. By capturing the functional requirements of a system as a set of object interaction scenarios, it helps perform rigorous system synthesis and analysis. The correctness of this refinement can be assured. Moreover, the object-based behavioural specifications so obtained can be readily used for code generation. This inevitably contributes towards a smooth transitions from functional requirements through design to implementation for object-oriented system development.

References

[1] J. Iivari (1995), "Object Orientation as Structural, Functional and Behavioural Modelling : A Comparison of Six Methods for Object-Oriented Analysis", *Information and Software Technology*, Vol. 37, No. 3, pp. 155-163.

[2] K.S. Cheung and K.O. Chow (1996), "Comparison of Object-Oriented Models by Views and Abstraction Constructs", *Proceedings of the International Conference on Intelligent Technologies in Human-Related Sciences*, pp. 335-342, Leon, Spain.

[3] K.O. Chow and S. Yeung (1996), "Behavioural Modelling in Object-Oriented Methodologies", *Information and Software Technology*, Vol. 38, No. 10, pp. 657-666.

[4] K.S. Cheung, K.O. Chow and T.Y. Cheung (1997), "A Feature-Based Approach for Consistent Object-Oriented Requirements Specifications". W.G. Wojtkowshi et al. (eds.), *Systems Development Methods for the Next Century*, pp. 31-38, Plenum Publishing.

[5] K.S. Cheung, K.O. Chow and T.Y. Cheung (1999), "Extending Formal Specification to Object Oriented Models Through Level-View Structured Schemas". J. Chen, J. Lu and B. Meyer (eds.), *Technology of Object Oriented Languages and Systems*, Vol. 31, pp. 118-125, IEEE Computer Society Press.

[6] B. Breu et al. (1998), "Systems, Views and Models of UML". M. Schader and A. Korthaus (eds.), *The Unified Modeling Language : Technical Aspects and Applications*, Physica-Verlag.

[7] G. Booch, J. Rumbaugh and I. Jacobson (1999), *The Unified Modeling Language : User Guide*, Addison-Wesley.

[8] J. Rumbaugh, I. Jacobson and G. Booch (1999), *The Unified Modeling Language : Reference Manual*, Addison-Wesley.

[9] I. Graham et al. (2001), *Object-Oriented Methods : Principles and Practice*, Addison-Wesley.

[10] I. Jacobson et al. (1992), *Object-Oriented Software Engineering : A Use-Case-Driven Approach*, Addison-Wesley.

[11] I. Jacobson, G. Booch and J. Rumbaugh (1999), *The Unified Software Development Process*, Addition Wesley.

[12] G. Schneider and J.P. Winters (1998), *Applying Use Cases*, Addison-Wesley.

[13] P. Kruchten (1999), *The Rational Unified Process : An Introduction*, Addison-Wesley.

[14] D. Rosenberg (1999), *Use Case Driven Object Modeling with UML : A Practical Approach*, Addison-Wesley.

[15] D. Rosenberg and K. Scott (2001), *Applying Use Case Driven Object Modeling with UML*, Addison-Wesley.

[16] J. Arlow and I. Neustadt (2002), *UML and the Unified Process : Practical Object-Oriented Analysis and Design*, Addison-Wesley.

[17] S. Kirani and W.T. Tsai (1994), "Method Sequence Specification and Verification of Classes", *Journal of Object-Oriented Programming*, Vol. 7, No. 6, pp. 28-38.

[18] K.S. Cheung, K.O. Chow and T.Y. Cheung (1998), "Consistency Analysis on Lifecycle Model and Interaction Model". C. Rolland and G. Grosz (eds.), *Object-Oriented Information Systems*, pp. 427-441, Springer.

[19] K.S. Cheung, K.O. Chow and T.Y. Cheung (1998), "Deriving Scenarios of Object Interaction through Petri Nets". J. Chen et al. (eds.), *Technology of Object Oriented Languages and Systems*, Vol. 27, pp. 118-125, IEEE Computer Society Press.

[20] M. Glinz (2000), "A Lightweight Approach to Consistency of Scenarios and Class Models", *Proceedings of the IEEE International Conference on Requirements Engineering*, pp. 49-58, IEEE Computer Society Press.

[21] F. Bordeleau and R.J.A. Buhr (1997), "UCM-ROOM Modelling : From Use Case Maps to Communicating State Machines", *Proceedings of the IEEE International Symposium and Workshop*

on Engineering of Computer-Based Systems, pp. 169-178, IEEE Computer Society Press.

[22] F. Bordeleau, J.P. Corriveau and B. Selic (2000), "A Scenario-Based Approach to Hierarchical State Machine Design", *Proceedings of the International Symposium on Object-Oriented Real-Time Distributed Computing*, pp. 78-85, IEEE Computer Society Press.

[23] B. Dano, H. Briand and F. Barbier (1996), "Progressing Towards Object-Oriented Requirements Specifications Using the Use Case Concept", *Proceedings of the IEEE Symposium and Workshop on Engineering of Computer-Based Systems*, pp. 450-456, IEEE Computer Society Press.

[24] B. Dano, H. Briand and F. Barbier (1997), "An Approach Based on the Concept of Use Case to Produce Dynamic Object-Oriented Specifications", *Proceedings of the IEEE International Symposium on Requirements Engineering*, pp. 54-64, IEEE Computer Society Press.

[25] P. Graubmann (1988), "The Construction of EN Systems from a Given Trace Behaviour", *Advances in Petri Nets*, *Lecture Notes in Computer Science*, Vol. 340, pp. 133-153, Springer-Verlag.

[26] E. Smith (1991), "On Net Systems Generated by Process Foldings", *Advances in Petri Nets*, *Lecture Notes in Computer Science*, Vol. 524, pp. 253-295, Springer-Verlag.

[27] K. Hiraishi (1992), "Construction of a Class of Safe Petri Nets by Presenting Firing Sequences", *Application and Theory of Petri Nets*, *Lecture Notes in Computer Science*, Vol. 616, pp. 244-262, Springer-Verlag.

[28] W.J. Lee, S.D. Cha and Y.R. Kwon (1998), "Integration and Analysis of Use Cases Using Modular Petri Nets in Requirement Engineering", *IEEE Transactions on Software Engineering*, Vol. 24, No. 12, pp. 1115-1130.

[29] J.L. Peterson (1981), *Petri Net Theory and the Modelling of System*, Prentice Hall.

[30] W. Reisig (1985), *Petri Nets : An Introduction*, Springer-Verlag.

[31] T. Murata (1989), "Petri Nets : Properties, Analysis and Applications", *Proceedings of the IEEE*, Vol. 77, No. 4, pp. 541-580.

[32] J. Desel and W. Reisig (1998), "Place Transition Petri Nets", *Lectures on Petri Nets 1 : Basic Models*, *Lecture Notes in Computer Science*, Vol. 1491, pp. 122-173, Springer-Verlag.

[33] J. Desel and J. Esparza (1995), *Free-choice Petri Nets*, Cambridge University Press.

[34] F. Chu and X. Xie (1997), "Deadlock Analysis of Petri Nets Using Siphons and Mathematical Programming", *IEEE Transactions on Robotics and Automation*, Vol. 13, No. 6, pp. 793-804.

[35] K.S. Cheung (2004), "New Characterisations for Live and Reversible Augmented Marked Graphs", *Information Processing Letters*, Vol. 92, No. 5, pp. 239-243.

[36] K.S. Cheung and K.O. Chow (2005), "Cycle Inclusion Property of Augmented Marked Graphs", *Information Processing Letters*, Vol. 94, No. 6, pp. 271-276.

[37] K.S. Cheung (2007), "Liveness and Boundedness of Augmented Marked Graphs", *IMA Journal of Mathematical Control and Information*, Vol. 24, No. 2, pp. 235-244.

[38] K.S. Cheung, T.Y. Cheung and K.O. Chow (2006), "A Petri-Net-Based Synthesis Methodology for Use-Case-Driven System Design", *Journal of Systems and Software*, Vol. 79, No. 6, pp. 772-790.

[39] K. Jensen (1986), "Coloured Petri Nets", *Petri Nets : Central Models and Their Properties*, *Lecture Notes in Computer Science*, Vol. 254, pp. 248-299, Springer-Verlag.

[40] K.S. Cheung and K.O. Chow (2006), "Elimination of Duplicate Labels in Petri-Net-Based System Specification", *Proceedings of the International Conference on Computer and Information Technology*, pp. 932-936, IEEE Computer Society Press.

A Multi-class SVM Classifier Utilizing Binary Decision Tree

Gjorgji Madzarov, Dejan Gjorgjevikj and Ivan Chorbev
Department of Computer Science and Engineering
Faculty of Electrical Engineering and Information Technology
Karpos 2 b.b., 1000 Skopje, Macedonia
E-mail: madzarovg@feit.ukim.edu.mk

Keywords: Support Vector Machine, multi-class classification, clustering, binary decision tree architecture

Received: July 27, 2008

In this paper a novel architecture of Support Vector Machine classifiers utilizing binary decision tree (SVM-BDT) for solving multiclass problems is presented. The hierarchy of binary decision subtasks using SVMs is designed with a clustering algorithm. For consistency between the clustering model and SVM, the clustering model utilizes distance measures at the kernel space, rather than at the input space. The proposed SVM based Binary Decision Tree architecture takes advantage of both the efficient computation of the decision tree architecture and the high classification accuracy of SVMs. The SVM-BDT architecture was designed to provide superior multi-class classification performance. Its performance was measured on samples from MNIST, Pendigit, Optdigit and Statlog databases of handwritten digits and letters. The results of the experiments indicate that while maintaining comparable or offering better accuracy with other SVM based approaches, ensembles of trees (Bagging and Random Forest) and neural network, the training phase of SVM-BDT is faster. During recognition phase, due to its logarithmic complexity, SVM-BDT is much faster than the widely used multi-class SVM methods like "one-against-one" and "one-against-all", for multiclass problems. Furthermore, the experiments showed that the proposed method becomes more favourable as the number of classes in the recognition problem increases.

Povzetek: Predstavljena je metoda gradnje binarnih dreves z uporabo SVM za večrazredne probleme.

1 Introduction

The recent results in pattern recognition have shown that support vector machine (SVM) classifiers often have superior recognition rates in comparison to other classification methods. However, the SVM was originally developed for binary decision problems, and its extension to multi-class problems is not straight-forward. How to effectively extend it for solving multi-class classification problem is still an on-going research issue. The popular methods for applying SVMs to multi-class classification problems usually decompose the multi-class problems into several two-class problems that can be addressed directly using several SVMs.

For the readers' convenience, we introduce the SVM briefly in section 2. A brief introduction to several widely used multi-class classification methods that utilize binary SVMs is given in section 3. The Kernel-based clustering introduced to convert the multi-class problem into SVM-based binary decision-tree architecture is explained in section 4. In section 5, we discuss related works and compare SVM-BDT with other multi-class SVM methods via theoretical analysis and empirical estimation. The experimental results in section 6 are presented to compare the performance of the proposed SVM-BDT with traditional multi-class approaches based on SVM, ensemble of decision trees and neural network. Section 7 gives a conclusion of the paper.

2 Support vector machines for pattern recognition

The support vector machine is originally a binary classification method developed by Vapnik and colleagues at Bell laboratories [1][2], with further algorithm improvements by others [3]. For a binary problem, we have training data points: $\{\mathbf{x}_i, y_i\}$, $i=1,...,l$, $y_i \in \{-1, 1\}$, $\mathbf{x}_i \in \mathbf{R}^d$. Suppose we have some hyperplane which separates the positive from the negative examples (a "separating hyperplane"). The points $\mathbf{x}$ which lie on the hyperplane satisfy $\mathbf{w} \cdot \mathbf{x} + b = 0$, where $\mathbf{w}$ is normal to the hyperplane, $|b|/||\mathbf{w}||$ is the perpendicular distance from the hyperplane to the origin, and $||\mathbf{w}||$ is the Euclidean norm of $\mathbf{w}$. Let d_+ (d_-) be the shortest distance from the separating hyperplane to the closest positive (negative) example. Define the "margin" of a separating hyperplane to be $d_+ + d_-$. For the linearly separable case, the support vector algorithm simply looks for the separating hyperplane with largest margin. This can be formulated as follows: suppose that all the training data satisfy the following constraints:

$$\mathbf{x}_i \cdot \mathbf{w} + b \geq +1 \text{ for } y_i = +1, \qquad (1)$$

$$\mathbf{x}_i \cdot \mathbf{w} + b \leq -1 \text{ for } y_i = -1, \qquad (2)$$

These can be combined into one set of inequalities:

$$y_i (\mathbf{x}_i \cdot \mathbf{w} + b) - 1 \geq 0 \quad \forall i, \qquad (3)$$

Now consider the points for which the equality in Eq. (1) holds (requiring that there exists such a point) is equivalent to choosing a scale for *w* and *b*. These points lie on the hyperplane H_1: $\mathbf{x}_i \cdot \mathbf{w} + b = 1$ with normal **w** and perpendicular distance from the origin $|1\text{-}b|/||\mathbf{w}||$. Similarly, the points for which the equality in Eq. (2) holds lie on the hyperplane H_2: $\mathbf{x}_i \cdot \mathbf{w} + b = -1$, with normal again **w** and perpendicular distance from the origin $|\text{-}1\text{-}b|/||\mathbf{w}||$. Hence $d_+ = d_- = 1/||\mathbf{w}||$ and the margin is simply $2/||\mathbf{w}||$.

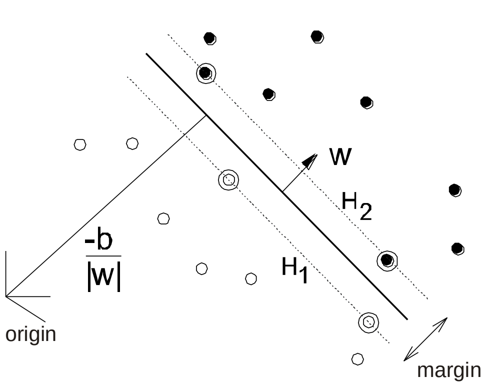


Figure 1 – Linear separating hyperplanes for the separable case. The support vectors are circled.

Note that H_1 and H_2 are parallel (they have the same normal) and that no training points fall between them. Thus we can find the pair of hyperplanes which gives the maximum margin by minimizing $||\mathbf{w}||^2$, subject to constraints (3).

Thus we expect the solution for a typical two dimensional case to have the form shown on Fig. 1. We introduce nonnegative Lagrange multipliers α_i, $i = 1,..., l$, one for each of the inequality constraints (3). Recall that the rule is that for constraints of the form $c_i \geq 0$, the constraint equations are multiplied by *nonnegative* Lagrange multipliers and subtracted from the objective function, to form the Lagrangian. For equality constraints, the Lagrange multipliers are unconstrained. This gives Lagrangian:

$$L_p \equiv \frac{1}{2}||\mathbf{w}||^2 - \sum_{i=1}^{l} \alpha_i y_i (\mathbf{x}_i \cdot \mathbf{w} + b) + \sum_{i=1}^{l} \alpha_i , \qquad (4)$$

We must now minimize L_p with respect to **w**, *b*, and maximize with respect to all α_i at the same time, all subject to the constraints $\alpha_i \geq 0$ (let's call this particular set of constraints C_1). Now this is a convex quadratic programming problem, since the objective function is itself convex, and those points which satisfy the constraints also form a convex set (any linear constraint defines a convex set, and a set of *N* simultaneous linear constraints defines the intersection of *N* convex sets, which is also a convex set). This means that we can equivalently solve the following "dual" problem: *maximize* L_P, subject to the constraints that the gradient of L_P with respect to *w* and *b* vanish, and subject also to the constraints that the $\alpha_i \geq 0$ (let's call that particular set of constraints C_2). This particular dual formulation of the problem is called the Wolfe dual [4]. It has the property that the maximum of L_P, subject to constraints C_2, occurs at the same values of the *w*, *b* and α, as the minimum of L_P, subject to constraints C_1.

Requiring that the gradient of L_P with respect to *w* and *b* vanish gives the conditions:

$$w = \sum_i \alpha_i y_i \mathbf{x}_i , \qquad (5)$$

$$\sum_i \alpha_i y_i = 0 . \qquad (6)$$

Since these are equality constraints in the dual formulation, we can substitute them into Eq. (4) to give

$$L_D \equiv \sum_i \alpha_i - \frac{1}{2}\sum_{i,j}^{l} \alpha_i \alpha_j y_i y_j \mathbf{x}_i \cdot \mathbf{x}_j , \qquad (7)$$

Note that we have now given the Lagrangian different labels (*P* for primal, *D* for dual) to emphasize that the two formulations are different: L_P and L_D arise from the same objective function but with different constraints; and the solution is found by minimizing L_P or by maximizing L_D. Note also that if we formulate the problem with $b = 0$, which amounts to requiring that all hyperplanes contain the origin, the constraint (6) does not appear. This is a mild restriction for high dimensional spaces, since it amounts to reducing the number of degrees of freedom by one.

Support vector training (for the separable, linear case) therefore amounts to maximizing L_D with respect to the α_i, subject to constraints (6) and positivity of the α_i, with solution given by (5). Notice that there is a Lagrange multiplier α_i for every training point. In the solution, those points for which $\alpha_i > 0$ are called "support vectors", and lie on one of the hyperplanes H_1, H_2. All other training points have $\alpha_i = 0$ and lie either on H_1 or H_2 (such that the equality in Eq. (3) holds), or on that side of H_1 or H_2 such that the strict inequality in Eq. (3) holds. For these machines, the support vectors are the critical elements of the training set. They lie closest to the decision boundary; if all other training points were removed (or moved around, but so as not to cross H_1 or H_2), and training was repeated, the same separating hyperplane would be found.

The above algorithm for separable data, when applied to non-separable data, will find no feasible solution: this will be evidenced by the objective function (i.e. the dual Lagrangian) growing arbitrarily large. So how can we extend these ideas to handle non-separable data? We would like to relax the constraints (1) and (2), but only when necessary, that is, we would like to introduce a further cost (i.e. an increase in the primal objective function) for doing so. This can be done by introducing positive slack variables e_i; $i = 1,..., l$, in the constraints, which then become:

$$\mathbf{x}_i \cdot \mathbf{w} + b \geq +1 - e_i \text{ for } y_i = +1, \quad (8)$$

$$\mathbf{x}_i \cdot \mathbf{w} + b \leq -1 + e_i \text{ for } y_i = -1, \quad (9)$$

$$e_i \geq 0 \forall i. \quad (10)$$

Thus, for an error to occur, the corresponding e_i must exceed unity, so $\Sigma_i e_i$ is an upper bound on the number of training errors. Hence a natural way to assign an extra cost for errors is to change the objective function to be minimized from $||\mathbf{w}||^2/2$ to $||\mathbf{w}||^2/2 + C(\Sigma_i e_i)$, where C is a parameter to be chosen by the user, a larger C corresponding to assigning a higher penalty to errors.

How can the above methods be generalized to the case where the decision function (f($\mathbf{x}$) whose sign represents the class assigned to data point $\mathbf{x}$) is not a linear function of the data? First notice that the only way in which the data appears in the training problem, is in the form of dot products, $\mathbf{x}_i \cdot \mathbf{x}_j$. Now suppose we first mapped the data (Figure 2) to some other (possibly even infinite dimensional) Euclidean space H, using a mapping which we will call Φ:

$$\Phi : \mathbf{R}^d \mapsto H, \quad (11)$$

Then of course the training algorithm would only depend on the data through dot products in H, i.e. on functions of the form $\Phi(\mathbf{x}_i) \cdot \Phi(\mathbf{x}_j)$. Now if there were a "kernel function" K such that $K(\mathbf{x}_i, \mathbf{x}_j) = \Phi(\mathbf{x}_i) \cdot \Phi(\mathbf{x}_j)$, we would only need to use K in the training algorithm, and would never need to explicitly even know what Φ is. The kernel function has to satisfy Mercer's condition [1].One example for this function is Gaussian:

$$K(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(-\frac{\left\| \mathbf{x}_i - \mathbf{x}_j \right\|^2}{2\sigma^2} \right), \quad (12)$$

In this particular example, H is infinite dimensional, so it would not be very easy to work with Φ explicitly. However, if one replaces $\mathbf{x}_i \cdot \mathbf{x}_j$ by $K(\mathbf{x}_i, \mathbf{x}_j)$ everywhere in the training algorithm, the algorithm will happily produce a support vector machine which lives in an infinite dimensional space, and furthermore do so in roughly the same amount of time it would take to train on the un-mapped data. All the considerations of the previous sections hold, since we are still doing a linear separation, but in a different space. But how can we use this machine? After all, we need $\mathbf{w}$, and that will live in H. But in test phase an SVM is used by computing dot products of a given test point $\mathbf{x}$ with $\mathbf{w}$, or more specifically by computing the sign of

$$f(\mathbf{x}) \equiv \sum_{i=1}^{N_S} \alpha_i y_i \Phi(\mathbf{s}_i) \cdot \Phi(\mathbf{x}) + b = \sum_{i=1}^{N_S} \alpha_i y_i K(\mathbf{s}_i, \mathbf{x}) + b \quad (13)$$

where the $\mathbf{s}_i$ are the support vectors. So again we can avoid computing $\Phi(\mathbf{x})$ explicitly and use the $K(\mathbf{s}_i, \mathbf{x}) = \Phi(\mathbf{s}_i) \cdot \Phi(\mathbf{x})$ instead.

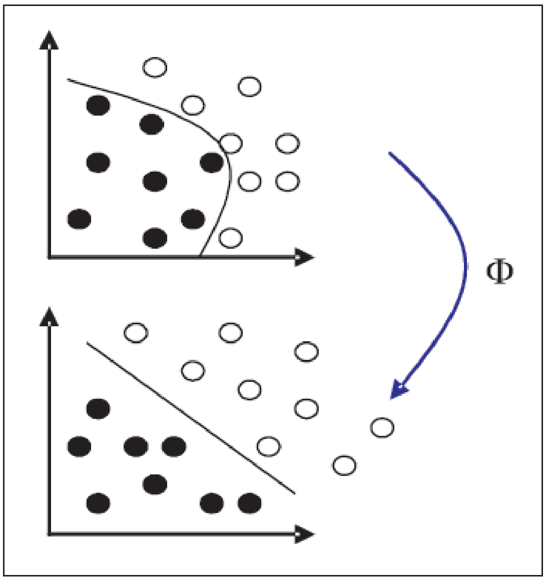


Figure 2 – General principle of SVM: projection of data in an optimal dimensional space.

3 An overview of widely used multi-class SVM classification methods

Although SVMs were originally designed as binary classifiers, approaches that address a multi-class problem as a single "all-together" optimization problem exist [5], but are computationally much more expensive than solving several binary problems.

A variety of techniques for decomposition of the multi-class problem into several binary problems using Support Vector Machines as binary classifiers have been proposed, and several widely used are given in this section.

3.1 One-against-all (OvA)

For the N-class problems (N>2), N two-class SVM classifiers are constructed [6]. The i^{th} SVM is trained while labeling the samples in the i^{th} class as positive examples and all the rest as negative examples. In the recognition phase, a test example is presented to all N SVMs and is labelled according to the maximum output among the N classifiers. The disadvantage of this method is its training complexity, as the number of training samples is large. Each of the N classifiers is trained using all available samples.

3.2 One-against-one (OvO)

This algorithm constructs $N(N\text{-}1)/2$ two-class classifiers, using all the binary pair-wise combinations of the N classes. Each classifier is trained using the samples of the first class as positive examples and the samples of the second class as negative examples. To combine these classifiers, the Max Wins algorithm is adopted. It finds the resultant class by choosing the class voted by the majority of the classifiers [7]. The number of samples used for training of each one of the OvO classifiers is smaller, since only samples from two of all N classes are taken in consideration. The lower number of samples causes smaller nonlinearity, resulting in shorter training times. The disadvantage of this method is that every test sample has to be presented to large number of classifiers $N(N\text{-}1)/2$. This results in slower testing, especially when the number of the classes in the problem is big [8].

3.3 Directed acyclic graph SVM (DAGSVM)

Introduced by Platt [1] the DAGSVM algorithm for training an $N(N\text{-}1)/2$ classifiers is the same as in one-against-one. In the recognition phase, the algorithm depends on a rooted binary directed acyclic graph to make a decision [9]. DAGSVM creates a model for each pair of classes. When one such model, which is able to separate class c_1 from class c_2, classifies a certain test example into class c_1, it does not really vote “for” class c_1, rather it votes “against” class c_2, because the example must lie on the other side of the separating hyperplane than most of the class c_2 samples. Therefore, from that point onwards the algorithm ignores all the models involving the class c_2. This means that after each classification with one of the binary models, one more class can be thrown out as a possible candidate, and after only N-1 steps just one candidate class remains, which therefore becomes the prediction for the current test example. This results in significantly faster testing, while achieving similar recognition rate as One-against-one.

3.4 Binary tree of SVM (BTS)

This method uses multiple SVMs arranged in a binary tree structure [10]. A SVM in each node of the tree is trained using two of the classes. The algorithm then employs probabilistic outputs to measure the similarity between the remaining samples and the two classes used for training. All samples in the node are assigned to the two subnodes derived from the previously selected classes by similarity. This step repeats at every node until each node contains only samples from one class. The main problem that should be considered seriously here is training time, because aside training, one has to test all samples in every node to find out which classes should be assigned to which subnode while building the tree. This may decrease the training performance considerably for huge training datasets.

4 Support vector machines utilizing a binary decision tree

In this paper we propose a binary decision tree architecture that uses SVMs for making the binary decisions in the nodes. The proposed classifier architecture SVM-BDT (Support Vector Machines utilizing Binary Decision Tree), takes advantage of both the efficient computation of the tree architecture and the high classification accuracy of SVMs. Utilizing this architecture, N-1 SVMs needed to be trained for an N class problem, but only at most $\lceil \log_2 N \rceil$ SVMs are required to be consulted to classify a sample. This can lead to a dramatic improvement in recognition speed when addressing problems with big number of classes.

An example of SVM-BDT that solves a 7 - class pattern recognition problem utilizing a binary tree, in which each node makes binary decision using a SVM is shown on Figure 3. The hierarchy of binary decision subtasks should be carefully designed before the training of each SVM classifier.

The recognition of each sample starts at the root of the tree. At each node of the binary tree a decision is being made about the assignment of the input pattern into one of the two possible groups represented by transferring the pattern to the left or to the right sub-tree. Each of these groups may contain multiple classes. This is repeated recursivly downward the tree until the sample reaches a leaf node that represents the class it has been assigned to.

There exist many ways to divide N classes into two groups, and it is critical to have proper grouping for the good performance of SVM-BDT.

For consistency between the clustering model and the way SVM calculates the decision hyperplane, the clustering model utilizes distance measures at the kernel space, rather than at the input space. Because of this, all training samples are mapped into the kernel space with the same kernel function that is to be used in the training phase.

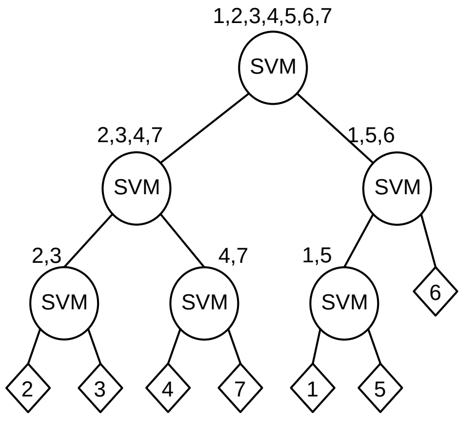


Figure 3: Illustration of SVM-BDT.

The SVM-BDT method that we propose is based on recursively dividing the classes in two disjoint groups in every node of the decision tree and training a SVM that will decide in which of the groups the incoming unknown sample should be assigned. The groups are determined by a clustering algorithm according to their class membership.

Let's take a set of samples x_1, x_2, ..., x_M each one labeled by $y_i \in \{c_1, c_2, ..., c_N\}$ where N is the number of classes. SVM-BDT method starts with dividing the classes in two disjoint groups g_1 and g_2. This is performed by calculating N gravity centres for the N different classes. Then, the two classes that have the biggest Euclidean distance from each other are assigned to each of the two clustering groups. After this, the class with the smallest Euclidean distance from one of the clustering groups is found and assigned to the corresponding group. The gravity center of this group is then recalculated to represent the addition of the samples of the new class to the group. The process continues by finding the next unassigned class that is closest to either of the clustering groups, assigning it to the corresponding group and updating the group's gravity center, until all classes are assigned to one of the two possible groups.

This defines a grouping of all the classes in two disjoint groups of classes. This grouping is then used to train a SVM classifier in the root node of the decision tree, using the samples of the first group as positive examples and the samples of the second group as negative examples. The classes from the first clustering group are being assigned to the first (left) subtree, while the classes of the second clustering group are being assigned to the (right) second subtree. The process continues recursively (dividing each of the groups into two subgroups applying the procedure explained above), until there is only one class per group which defines a leaf in the decision tree.

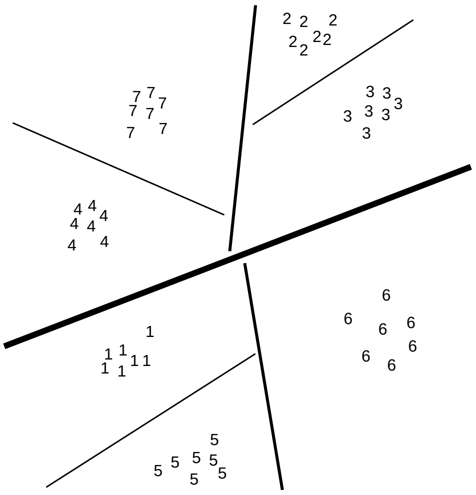


Figure 4: SVM-BDT divisions of the seven classes.

For example, Figure 4 illustrates grouping of 7 classes, while Figure 3 shows the corresponding decision tree of SVMs. After calculating the gravity centers for all classes, the classes c_2 and c_5 are found to be the furthest apart from each other, considering their Euclidean distance and are assigned to group g_1 and g_2 accordingly. The closest to group g_1 is class c_3, so it is assigned to the group g_1, followed by recalculation of the g_1's gravity center. In the next step, class c_1 is the closest to group g_2, so it is assigned to that group and the group's gravity center is recalculated. In the following iteration, class c_7 is assigned to g_1 and class c_6 is assigned to g_2, folowed by recalculating of group's gravity centers. Finally class c_4 is assigned to g_1. This completes the first round of grouping that defines the classes that will be transferred to the left and the right subtree of the root node. The SVM classifier in the root is trained by considering samples from the classes $\{c_2, c_3, c_4, c_7\}$ as positive examples and samples from the classes $\{c_1, c_5, c_6\}$ as negative examples.

The grouping procedure is repeated independently for the classes of the left and the right subtree of the root, which results in grouping c_7 and c_4 in $g_{1,1}$ and c_2 and c_3 in $g_{1,2}$ in the left node of the tree and c_1 and c_5 in $g_{2,1}$ and c_6 in $g_{2,2}$ in the right node of the tree. The concept is repeated for each SVM associated to a node in the taxonomy. This will result in training only N-1 SVMs for solving an N-class problem.

5 Related work and discussion

Various multi-class classification algorithms can be compared by their predictive accuracy and their training and testing times. The training time T for a binary SVM is estimated empirically by a power law [13] stating that $T \approx \alpha M^d$, where M is the number of training samples and α is a proportionality constant. The parameter d is a constant, which depends of the datasets and it is typically in the range [1, 2]. According to this law, the estimated training time for OvA is

$$T_{OvA} \approx N \alpha M^d , \quad (11)$$

where N is the number of classes in the problem.

Without loss of generality, let's assume that each of the N classes has the same number of training samples. Thus, each binary SVM of OvO approach only requires $2M/N$ samples. Therefore, the training time for OvO is:

$$T_{OvO} \approx \alpha \frac{N(N-1)}{2} \left(\frac{2M}{N} \right)^d \approx N^{2-d} \alpha M^d , \quad (12)$$

The training time for DAGSVM is same as OvO.

As for BTS and SVM-BDT, the training time is summed over all the nodes in the $\lceil \log_2 N \rceil$ levels. In the i^{th} level, there are 2^{i-1} nodes and each node uses $2M/N$ for BTS and $M/2^{i-1}$ for SVM-BDT training samples. Hence, the total training time for BTS is:

$$\begin{aligned} T_{BTS} &\approx \sum_{i=1}^{\lceil \log_2(N) \rceil} \alpha 2^{i-1} \left(2 \frac{M}{N} \right)^d \\ &\approx \alpha \left(2 \frac{M}{N} \right)^d \sum_{i=1}^{\lceil \log_2(N) \rceil} 2^{i-1} \approx N^{1-d} \alpha M^d \end{aligned} , \quad (13)$$

and for SVM-BDT is:

$$T_{SVM-BDT} \approx \sum_{i=1}^{\lceil \log_2 (N) \rceil} \alpha 2^{i-1} \left(\frac{M}{2^{i-1}} \right)^d \approx \alpha M^d \text{ , } \quad (14)$$

It must be noted that $T_{SVM\text{-}BDT}$ in our algorithm does not include the time to build the hierarchy structure of the N classes, since it consumes insignificant time compared to the quadratic optimization time that dominates the total SVM training time. On the other hand, in the process of building the tree, BTS requires testing of each trained SVM with all the training samples in order to determine the next step, therefore significantly increasing the total training time.

According to the empirical estimation above, it is evident that the training speed of SVM-BDT is comparable with OvA, OvO, DAGSVM and BTS.

In the testing phase, DAGSVM performs faster than OvO and OvA, since it requires only N-1 binary SVM evaluations. SVM-BDT is even faster than DAGSVM because the depth of the SVM-BDT decision tree is $\lceil \log_2 N \rceil$ in the worst case, which is superior to N-1, especially when N>>2.

While testing, the inner product of the sample's feature vector and all the support vectors of the model are calculated for each sample. The total number of support vectors in the trained model directly contributes to the major part of the evaluation time, which was also confirmed by the experiments.

A multistage SVM (MSVM) for multi-class problem has been proposed by Liu et al. [11]. They use Support Vector Clustering (SVC) [12] to divide the training data into two parts that are used to train a binary SVM. For each partition, the same procedure is recursively repeated until the binary SVM gives an exact label of class. An unsolved problem in MSVM is how to control the SVC to divide the training dataset into exact two parts. However, this procedure is painful and unfeasible, especially for large datasets. The training set from one class could belong to both clusters, resulting in decreased predictive accuracy.

There are different approaches for solving multi-class problems which are not based on SVM. Some of them are presented in the following discussion. However, the experimental results clearly show that their classification accuracy is significantly smaller than the SVM based methods.

Ensemble techniques have received considerable attention within the recent machine learning research [16][17][18][19]. The basic goal is to train a diverse set of classifiers for a single learning problem and to vote or average their predictions. The approach is simple as well as powerful, and the obtained accuracy gains often have solid theoretical foundations [20][20][21]. Averaging the predictions of these classifiers helps to reduce the variance and often increases the reliability of the predictions. There are several techniques for obtaining a diverse set of classifiers. The most common technique is to use subsampling to diversify the training sets as in Bagging [21] and Boosting [20]. Other techniques include the use of different feature subsets for every classifier in the ensemble [23], to exploit the randomness of the base algorithms [24], possibly by artificially randomizing their behavior [25], or to use multiple representations of the domain objects. Finally, classifier diversity can be ensured by modifying the output labels, i.e., by transforming the learning tasks into a collection of related learning tasks that use the same input examples, but different assignments of the class labels. Error-correcting output codes are the most prominent example for this type of ensemble methods [22].

Error-correcting output codes are a popular and powerful class binarization technique. The basic idea is to transform an N-class problem into n binary problems ($n > N$), where each binary problem uses a subset of the classes as the positive class and the remaining classes as a negative class. As a consequence, each original class is encoded as an n-dimensional binary vector, one dimension for each prediction of a binary problem (+1 for positive and −1 for negative). The resulting matrix of the form $\{-1, +1\}$ $N \times n$ is called the coding matrix. New examples are classified by determining the row in the matrix that is closest to the binary vector obtained by submitting the example to the n classifiers. If the binary problems are chosen in a way that maximizes the distance between the class vectors, the reliability of the classification can be significantly increased. Error-correcting output codes can also be easily parallelized, but each subtask requires the total training set.

Similar to binarization, some approaches suggest mapping the original multiple classes into three clsses. A related technique where multi-class problems are mapped to 3-class problems is proposed by Angulo and Catal'a [26]. Like with pairwise classification, they propose generating one training set for each pair of classes. They label the two class values with target values +1 and −1, and additionally, samples of all other classes are labeled to a third class, with a target value of 0. This idea leads to increased size of the training set compared to the binary classification. The mapping into three classes was also used by Kalousis and Theoharis [27] for predicting the most suitable learning algorithm(s) for a given dataset. They trained a nearest-neighbor learner to predict the better algorithm of each pair of learning algorithms. Each of these pairwise problems had three classes: one for each algorithm and a third class named "tie", where both algorithms had similar performances.

Johannes Fürnkranz has investigated the use of round robin binarization (or pair-wise classification) [28] as a technique for handling multi-class problems with separate-and-conquer rule learning algorithms (aka covering algorithms). In particular, round robin binarization helps Ripper [29] outperform C5.0 on multi-class problems, whereas C5.0 outperforms the original version of Ripper on the same problems.

6 Experimental results

In this section, we present the results of our experiments with several multi-class problems. The performance was measured on the problem of recognition of handwritten digits and letters.

Here, we compare the results of the proposed SVM-BDT method with the following methods:

1) one-against-all (OvA);
2) one-against-one (OvO);
3) DAGSVM;
4) BTS;
5) Bagging
6) Random Forests
7) Multilayer Perceptron (MLP, neural network)

The training and testing of the SVMs based methods (OvO, OvA, DAGSVM, BTS and SVM-BDT) was performed using a custom developed application that uses the Torch library [14]. For solving the partial binary classification problems, we used SVMs with Gaussian kernel. In these methods, we had to optimize the values of the kernel parameter σ and penalty C. For parameter optimization we used experimental results. The achieved parameter values for the given datasets are given in Table 1.

Table 1. The optimized values for σ and C for the used datasets.

	MNIST	Pendigit	Optdigit	Statlog
σ	2	60	25	1.1
C	100	100	100	100

We also developed an application that uses the same (Torch) library for the neural network classification. One hidden layer with 25 units was used by the neural network. The number of hidden units was determined experimentally.

The classifications based on ensembles of decision trees [30] (Bagging and Random Forest) was performed by Clus, a popular decision tree learner based on the principles stated by Blockeel et al. [31]. There were 100 models in the ensembles. The pruning method that we used was C4.5. The number of selected features in the Random Forest method was $\lfloor \log_2 M \rfloor$, where M is the number of features in the dataset.

The most important criterion in evaluating the performance of a classifier is usually its recognition rate, but very often the training and testing time of the classifier are equally important.

In our experiments, four different multi-class classification problems were addressed by each of the eight previously mentioned methods. The training and testing time and the recognition performance were recorded for every method.

The first problem was recognition of isolated handwritten digits (10 classes) from the MNIST database. The MNIST database [15] contains grayscale images of isolated handwritten digits. From each digit image, after performing a slant correction, 40 features were extracted. The features are consisted of 10 horizontal, 8 vertical and 22 diagonal projections [25]. The MNIST database contains 60.000 training samples, and 10.000 testing samples.

The second and the third problem are 10 class problems from the UCI Repository [33] of machine learning databases: Optdigit and Pendigit. Pendigit has 16 features, 7494 training samples, and 3498 testing samples. Optdigit has 64 features, 3823 training samples, and 1797 testing samples.

The fourth problem was recognition of isolated handwritten letters – a 26-class problem from the Statlog collection [34]. Statlog-letter contains 15.000 training samples, and 5.000 testing samples, where each sample is represented by 16 features.

The classifiers were trained using all available training samples of the set and were evaluated by recognizing all the test samples from the corresponding set. All tests were performed on a personal computer with an Intel Core2Duo processor at 1.86GHz with the Windows XP operating system.

Tables 2 through 4 show the results of the experiments using 8 different approaches (5 approaches based on SVM, two based on ensembles of decision trees and one neural network) on each of the 4 data sets. The first column of each table describes the classification method. Table 2 gives the prediction error rate of each method applied on each of the datasets. Table 3 and table 4 shows the testing and training time of each algorithm, for the datasets, measured in seconds, respectively.

The results in the tables show that SVM based methods outperform the other approaches, in terms of classification accuracy. In terms of speed, SVM based methods are faster, with different ratios for different datasets. In overall, the SVM based algorithms were significantly better compared to the non SVM based methods.

The results in table 2 show that for all datasets, the one-against-all (OvA) method achieved the lowest error rate. For the MNIST, Pendigit and Optdigit datasets, the other SVM based methods (OvO, DAGSVM, BTS and our method - SVM-BDT) achieved higher, but similar error rates. For the recognition of handwritten letters from the Statlog database, the OvO and DAGSVM methods achieved very similar error rates that were about 1.5% higher than the OvA method. The BTS method showed the lowest error rate of all methods using one-against-one SVMs. Our SVM-BDT method achieved better recognition rate than all the methods using one-against-one SVMs, including BTS. Of the non SVM based methods, the Random Forest method achieved the best recognition accuracy for all datasets. The prediction performance of the MLP method was comparable to the Random Forest method for the 10-class problems, but noticeably worse for the 26-class problem.

The MLP method is the fastest one in terms of training and testing time, which is evident in Table 3 and Table 4. The classification methods based on ensembles of trees were the slowest in the training and the testing phase, especially the Bagging method. Overall, the Random Forest method was more accurate than the other non SVM based methods, while the MLP method was the fastest.

The results in Table 3 show that the DAGSVM method achieved the fastest testing time of all the SVM based methods for the MNIST dataset. For the other datasets, the testing time of DAGSVM is comparable with BTS and SVM-BDT methods and their testing time is noticeably better than the one-against-all (OvA) and one-against-one (OvO) methods. The SVM-BDT method was faster in the recognition phase for the Pendigit dataset and slightly slower than DAGSVM method for the Statlog dataset.

Table 2. The prediction error rate (%) of each method for every dataset

Classifier	MNIST	Pendigit	Optdigit	Statlog
OvA	1.93	1.70	1.17	3.20
OvO	2.43	1.94	1.55	4.72
DAGSVM	2.50	1.97	1.67	4.74
BTS	2.24	1.94	1.51	4.70
SVM-BDT	2.45	1.94	1.61	4.54
R. Forest	3.92	3.72	3.18	4.98
Bagging	4.96	5.38	7.17	8.04
MLP	4.25	3.83	3.84	14.14

Table 3. Testing time of each method for every dataset measured in seconds

Classifier	MNIST	Pendigit	Optdigit	Statlog
OvA	23.56	1.75	1.63	119.50
OvO	26.89	3.63	1.96	160.50
DAGSVM	9.46	0.55	0.68	12.50
BTS	26.89	0.57	0.73	17.20
SVM-BDT	25.33	0.54	0.70	13.10
R. Forest	39.51	3.61	2.76	11.07
Bagging	34.52	2.13	1.70	9.76
MLP	2.12	0.49	0.41	1.10

Table 4. Training time of each method for every dataset measured in seconds

Classifier	MNIST	Pendigit	Optdigit	Statlog
OvA	468.94	4.99	3.94	554.20
OvO	116.96	3.11	2.02	80.90
DAGSVM	116.96	3.11	2.02	80.90
BTS	240.73	5.21	5.65	387.10
SVM-BDT	304.25	1.60	1.59	63.30
R. Forest	542.78	17.08	22.21	50.70
Bagging	3525.31	30.87	49.4	112.75
MLP	45.34	2.20	1.60	10.80

In terms of training speeds, it is evident in Table 4 that among the SVM based methods, SVM-BDT is the fastest one in the training phase. For the three 10-class problems the time needed to train the 10 classifiers for the OvA approach took about 4 times longer than training the 45 classifiers for the OvO and DAGSVM methods. Due to the huge number of training samples in the MNIST dataset (60000), SVM-BDT's training time was longer compared to other one-against-one SVM methods. The huge number of training samples increases the nonlinearity of the hyperplane in the SVM, resulting in an incresed number of support vectors and increased training time. Also, the delay exists only in the first level of the tree, where the entire training dataset is used for training. In the lower levels, the training time of divided subsets is not as significant as the first level's delay.

In the other 10 class problems, our method achieved the shortest training time. For the Statlog dataset, the time needed for training of the 26 one-against-all SVMs was almost 7 times longer than the time for training the 325 one-against-one SVMs. The BTS method is the slowest one in the training phase of the methods using one-against-one SVMs. It must be noted that as the number of classes in the dataset increases, the advantage of SVM-BDT becomes more evident. The SVM-BDT method was the fastest while training, achieving better recognition rate than the methods using one-against-one SVMs. It was only slightly slower in recognition than DAGSVM.

7 Conclusion

A novel architecture of Support Vector Machine classifiers utilizing binary decision tree (SVM-BDT) for solving multiclass problems was presented. The SVM-BDT architecture was designed to provide superior multi-class classification performance, utilizing a decision tree architecture that requires much less computation for deciding a class for an unknown sample. A clustering algorithm that utilizes distance measures at the kernel space is used to convert the multi-class problem into binary decision tree, in which the binary decisions are made by the SVMs. The results of the experiments show that the speed of training and testing are improved, while keeping comparable or offering better recognition rates than the other SVM multi-class methods. The experiments showed that this method becomes more favourable as the number of classes in the recognition problem increases.

References

[1] V. Vapnik. *The Nature of Statistical Learning Theory*, 2nd Ed. Springer, New York, 1999.

[2] C. J. C. Burges. *A tutorial on support vector machine for pattern recognition*. Data Min. Knowl. Disc. 2 (1998) 121.

[3] T. Joachims. Making large scale SVM learning practical. in B. Scholkopf, C. Bruges and A. Smola (eds). *Advances in kernel methods-support vector learning*, MIT Press, Cambridge, MA, 1998.

[4] R. Fletcher. *Practical Methods of Optimization*. 2nd Ed. John Wiley & Sons. Chichester (1987).

[5] J. Weston, C. Watkins. *Multi-class support vector machines*. Proceedings of ESANN99, M. Verleysen, Ed., Brussels, Belgium, 1999.

[6] V. Vapnik. *Statistical Learning Theory*. Wiley, New York, 1998.

[7] J. H. Friedman. Another approach to polychotomous classification. Technical report. Department of Statistics, Stanford University, 1997.

[8] P. Xu, A. K. Chan. Support vector machine for multi-class signal classification with unbalanced samples. *Proceedings of the International Joint Conference on Neural Networks 2003*. Portland, pp.1116-1119, 2003.

[9] Platt, N. Cristianini, J. Shawe-Taylor. Large margin DAGSVM's for multiclass classification. *Advances in Neural Information Processing System*. Vol. 12, pp. 547–553, 2000.

[10] B. Fei, J. Liu. Binary Tree of SVM: A New Fast Multiclass Training and Classification Algorithm. *IEEE Transaction on neural networks*, Vol. 17, No. 3, May 2006.

[11] X. Liu, H. Xing, X. Wang. A multistage support vector machine. *2nd International Conference on Machine Learning and Cybernetics*, pages 1305-1308, 2003.

[12] A. Ben-Hur, D. Horn, H. Siegelmann, V. Vapnik. Support vector clustering. *Journal of Machine Learning Research*, vol. 2:125-137, 2001.

[13] J. Platt. Fast training of support vector machines using sequential minimal optimization. *In Advances in Kernel Methods - Support Vector Learning*. Pages 185-208, Cambridge, MA, 1999. MIT Press.

[14] R. Collobert, S. Bengio, J. Mariéthoz. Torch: a modular machine learning software library. Technical Report IDIAP-RR 02-46, IDIAP, 2002.

[15] __, MNIST, MiniNIST, USA http://yann.lecun.com/exdb/mnist

[16] T. G. Dietterich. Machine learning research: Four current directions. *AI Magazine*, 18(4): 97–136, Winter 1997.

[17] G. Dietterich. Ensemble methods in machine learning. In J. Kittler and F. Roli (eds.) *First International Workshop on Multiple Classifier Systems*, pp. 1–15. Springer-Verlag, 2000a.

[18] D. Opitz and R. Maclin. Popular ensemble methods: An empirical study. *Journal of Artificial Intelligence Research*, 11:169–198, 1999.

[19] E. Bauer and R. Kohavi. An empirical comparison of voting classification algorithms: Bagging, boosting, and variants. *Machine Learning*, 36:105–169, 1999.

[20] Y. Freund and R. E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55(1):119–139, 1997.

[21] L. Breiman. Bagging predictors. *Machine Learning*, 24(2):123–140, 1996.

[22] T. G. Dietterich and G. Bakiri. Solving multiclass learning problems via error-correcting output codes. *Journal of Artificial Intelligence Research*, 2:263–286, 1995.

[23] S. D. Bay. Nearest neighbor classification from multiple feature subsets. *Intelligent Data Analysis*, 3(3):191–209, 1999.

[24] J. F. Kolen and J. B. Pollack. Back propagation is sensitive to initial conditions. *In Advances in Neural Information Processing Systems 3 (NIPS-90)*, pp. 860–867. Morgan Kaufmann, 1991.

[25] T. G. Dietterich. An experimental comparison of three methods for constructing ensembles of decision trees: Bagging, boosting, and randomization. *Machine Learning*, 40(2):139–158, 2000b.

[26] C. Angulo and A. Catal`a. K-SVCR. A multi-class support vector machine. In R. L´opez de M´antaras and E. Plaza (eds.) *Proceedings of the 11th European Conference on Machine Learning (ECML-2000)*, pp. 31–38. Springer-Verlag, 2000.

[27] A. Kalousis and T. Theoharis. Noemon: Design, implementation and performance results of an intelligent assistant for classifier selection. *Intelligent Data Analysis*, 3(5):319–337, 1999.

[28] Johannes Fürnkranz, Round robin classification, *The Journal of Machine Learning Research*, 2, p.721-747, 3/1/2002

[29] W. W. Cohen. Fast effective rule induction. In A. Prieditis and S. Russell (eds.) *Proceedings of the 12th International Conference on Machine Learning* (ML-95), pp. 115–123, Lake Tahoe, CA, 1995. Morgan Kaufmann.

[30] D. Kocev, C. Vens, J. Struyf and S. Dˇzeroski. Ensembles of multi-objective decision trees. *Proceedings of the 18th European Conference on Machine Learning* (pp. 624–631) (2007). Springer.

[31] H. Blockeel, J. Struyf. Efficient Algorithms for Decision Tree Cross-validation. *Journal of Machine Learning Research* 3:621-650, 2002.

[32] D. Gorgevik, D. Cakmakov. An Efficient Three-Stage Classifier for Handwritten Digit Recognition. *Proceedings of 17th Int. Conference on Pattern Recognition*, *ICPR2004*. Vol. 4, pp. 507-510, IEEE Computer Society, Cambridge, UK, 23-26 August 2004.

[33] C. Blake, E. Keogh and C. Merz. UCI Repository of Machine Learning Databases, (1998). Statlog Data Set, http://archive.ics.uci.edu/ml/datasets.html [Online]

[34] Statlog Data Set, http://archive.ics.uci.edu/ml/-datasets/Letter+Recognition [Online]

Churn Prediction Model in Retail Banking Using Fuzzy C-Means Algorithm

Džulijana Popović
Zagrebačka banka d.d., Consumer Finance
Trg bana Josipa Jelačića 10, 10000 Zagreb, Croatia
E-mail: dzulijana.popovic@unicreditgroup.zaba.hr, www.zaba.hr

Bojana Dalbelo Bašić
University of Zagreb, Faculty of Electrical Engineering and Computing
Unska 3, 10000 Zagreb, Croatia
E-mail: bojana.dalbelo@fer.hr

Keywords: churn prediction, fuzzy c-means algorithm, fuzzy transitional condition of the first degree, fuzzy transitional condition of the second degree, distance of k instances fuzzy sum

Received: February 22, 2008

The paper presents model based on fuzzy methods for churn prediction in retail banking. The study was done on the real, anonymised data of 5000 clients of a retail bank. Real data are great strength of the study, as a lot of studies often use old, irrelevant or artificial data. Canonical discriminant analysis was applied to reveal variables that provide maximal separation between clusters of churners and non-churners. Combination of standard deviation, canonical discriminant analysis and k-means clustering results were used for outliers detection. Due to the fuzzy nature of practical customer relationship management problems it was expected, and shown, that fuzzy methods performed better than the classical ones. According to the results of the preliminary data exploration and fuzzy clustering with different values of the input parameters for fuzzy c-means algorithm, the best parameter combination was chosen and applied to training data set. Four different prediction models, called prediction engines, have been developed. The definitions of clients in the fuzzy transitional conditions and the distance of k instances fuzzy sums were introduced. The prediction engine using these sums performed best in churn prediction, applied to both balanced and non-balanced test sets.

Povzetek: Razvita je metoda mehke logike za uporabo v bančništvu.

1 Introduction

Due to intensive competition and saturated markets, companies in all industries realize that their existing clients database is their most valuable asset. Retaining existing clients is the best marketing strategy to survive in industry and a lot of studies showed it is more profitable to keep and satisfy existing clients than to constantly attract new ones [1,4,8,11]. Churn management, as the general concept of identifying those clients most prone to switching to another company, led to development of variety of techniques and models for churn prediction. Next generation of such models has to concentrate on the improved accuracy, robustness and lower implementation costs, as every delay in reaction means increased costs for the company [2].

The aim of this study was to show that the data mining methods based on the fuzzy logic could be successfully applied in the retail banking analysis and, moreover, that the fuzzy c-means clustering performed better than the classical clustering algorithms in the problem of churn prediction.

Although the clustering analysis is in fact an unsupervised learning technique, it can be used as the basis for classification model, if the data set contains the classification variable, what was case in this study.

To our best knowledge this is the first paper considering application of fuzzy clustering in churn prediction for retail banking. Studies of churn prediction in banking are very scarce, and the most of papers used models based on logistic regression, decision trees and neural networks [9,11]. Useful literature review of attrition models can be found in [11]. Some of them [9] reported the percentage of correct predictions varying from 14% to 73%, depending on the proportion of churners in the validation set. The others [3] obtained AUC performance in subscription services varying from 69,4% for overall churn to 90,4% but only for churn caused by financial reasons, which is much easier to predict. Results are not perfectly comparable due to differences in churn moment definitions, data sets sizes or industries, but still can provide valuable subject insight.

2 Fuzzy c-means clustering algorithm

Classical clustering assigns each observation to a single cluster, without information how far or near the observation is from all the other possible decisions. This type of clustering is often called hard or crisp clustering [1,10,12]. Two major classes of crisp clustering methods are hierarchical and optimization (partitive) clustering, with number of different algorithms, used in the study.

Based on the fuzzy set theory, firstly introduced by Zadeh in 1965. [5,6,10] and on the concept of membership functions, the fuzzy clustering methods have been developed. In fuzzy clustering entities are allowed to belong to many clusters with different degrees of membership.

Fuzzy clustering of X into p clusters is characterized by p membership functions μ_j, where

$$\mu_j : X \rightarrow [0,1], j = 1, \dots, p, \qquad (1)$$

$$\sum_{j=1}^{p} \mu_j(x_i) = 1,\ i = 1, 2, \dots, n, \qquad (2)$$

and

$$0 < \sum_{i=1}^{n} \mu_j(x_i) < n,\ j = 1, 2, \dots, p. \qquad (3)$$

Membership functions are based on a distance function, such that membership degrees express proximities of entities to cluster centers (also called *cluster prototypes*).

The most known method of fuzzy clustering is the fuzzy c-means method (FCM), initially proposed by Dunn, generalized by Bezdek [5] and used in this study.

FCM involves two iterative processes: the calculation of cluster centers and the assignment of the observations to these centers using some form of distance. FCM is attempting to minimize a standard loss function

$$l = \sum_{k=1}^{p} \sum_{i=1}^{n} [\mu_k(x_i)]^m \|x_i - c_k\|^2 \qquad (4)$$

from which two fundamental equations necessary to implement FCM are derived [5].

Expression (5) is used to calculate a new cluster center value:

$$c_j = \frac{\sum_i [\mu_j(x_i)]^m x_i}{\sum_i [\mu_j(x_i)]^m} \qquad (5)$$

and expression (6) to calculate the membership in the $j - th$ cluster:

$$\mu_j(x_i) = \frac{\left(\frac{1}{d_{ji}}\right)^{\frac{1}{m-1}}}{\sum_{k=1}^{p} \left(\frac{1}{d_{ki}}\right)^{\frac{1}{m-1}}} \qquad (6)$$

The symbols in the equations (4), (5) and (6) denote:

l is the minimized loss value;
p is the number of fuzzy clusters;
n is the number of observations in the data set;
$\mu_k()$ is a function that returns the membership of x_i in the $k - th$ cluster;
m is the fuzzification parameter;
c_k is the centre of the $k - th$ cluster;
d_{ji} is the distance metric for x_i in cluster c_j;
d_{ki} is the distance metric for x_i in cluster c_k.

3 Churn prediction problem in retail banking and input data set

There is no unique definition of churn problem, but generally, term churn refers to all types of customer attrition whether voluntary or involuntary [1,3]. How to recognize it in practice depends on industry and case. In this study, client is treated as churner if he had at least one product (saving account, credit card, cash loan etc.) at time t_n and had no product at time t_{n+1}, meaning that he cancelled all his products in the period $t_{n+1} - t_n$. If client still holds at least one product at time t_{n+1}, he is considered to be non-churner.

The programs for all research phases, as well as prediction engines, were written in SAS 9.1. [12].

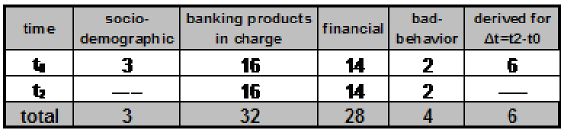
3.1 Input data set

The input data set has 5000 clients, chosen by random sampling from the client population in 2005, aged between 18 and 80 years, preserving the distribution of population according to introduced auxiliary variable which was product level of detection (PLOD). The class imbalance problem [2,9] was solved in the way that precisely 2500 churners and 2500 non-churners entered the final data set, what is in line with results in [2]. Regarding the moment of churn for 2500 churners five sample data sets, with different configuration of churners, have been explored. The analysis showed that the “clear” set, with churners all lost in the same quarter of the year, is best for further clustering. All clients who quitted the relationship with bank in some period, but returned after 6 months or later, were removed from the sample, as the analysis showed they behave similarly to real churners and introduce the noise.

3.2 Variable selection

Not all variables of interest were allowed for study, and the availability of more transactional variables would surely lead to better model performance [11]. This was partially proved through inclusion of derived variables (differences in time, ratios, etc.), what led to the improved accuracy of fuzzy clustering in comparison to clustering results with only original variables. All variables were measured in five equidistant points of time: t_o to t_4. Preliminary clustering analysis showed that, as far as the original variables are statical in their character, including the values of more than two periods leads to more noise than to greater precision. All the combinations of two periods were tested and finally the values in t_0 and t_2 were chosen for the further analysis.

Table 1. gives the description of the 73 variables finally chosen for the further work.



time	socio-demographic	banking products in charge	financial	bad-behavior	derived for Δt=t2-t0
t_0	3	16	14	2	6
t_2	—–	16	14	2	——
total	3	32	28	4	6

Table 1: Variables for the final FCM.

3.3 Canonical discriminant analysis

Canonical discriminant analysis (CDA) finds the linear combinations of the variables that provide maximal separation between clusters [12]. CDA helped in identifying the variables that best describe each of two classes/clusters: churners and non-churners. In some way CDA confirmed that the combination of $t_0 - t_2$ variable values is more adequate for FCM then the other combinations. All the coefficients and corresponding variables have been carefully examined.

3.4 Detection and removal of outliers

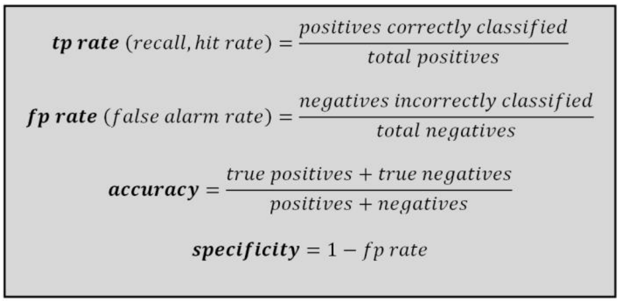
Although the majority of variables is not normally distributed, the standard deviation [10] in combination with CDA and k-means revealed most serious outliers better than other methods.

The most serious outliers were detected for all 73 variables separately and the intersection of those 73 sets was found. For all the outliers the data values have been checked in the data warehouse. The check confirmed that all the data are correct and that the outliers are not the consequence of the errors in database. Top 50 outliers from that intersection were removed from the data set. The outlier removal significantly improved the performance of classical clustering and slightly improved the performance of FCM.

3.5 Results of the hierarchical and crisp k-means algorithm

To prove that fuzzy clustering performs better than the classical methods on the real retail banking data, hierarchical clustering and k-means clustering were done. Applied to all 5000 clients, almost all hierarchical methods, as well as repeated k-means, failed on the same outliers. Most of them separated only one client in the first cluster and all other 4999 were appointed to the second cluster. All the methods were repeated on the data set without top 50 outliers and some of them performed better.

Figure 1 shows the standard measures [7] for comparison of the results in churn prediction, as stated in [2] and Table 2 shows some of the results of classical clustering.



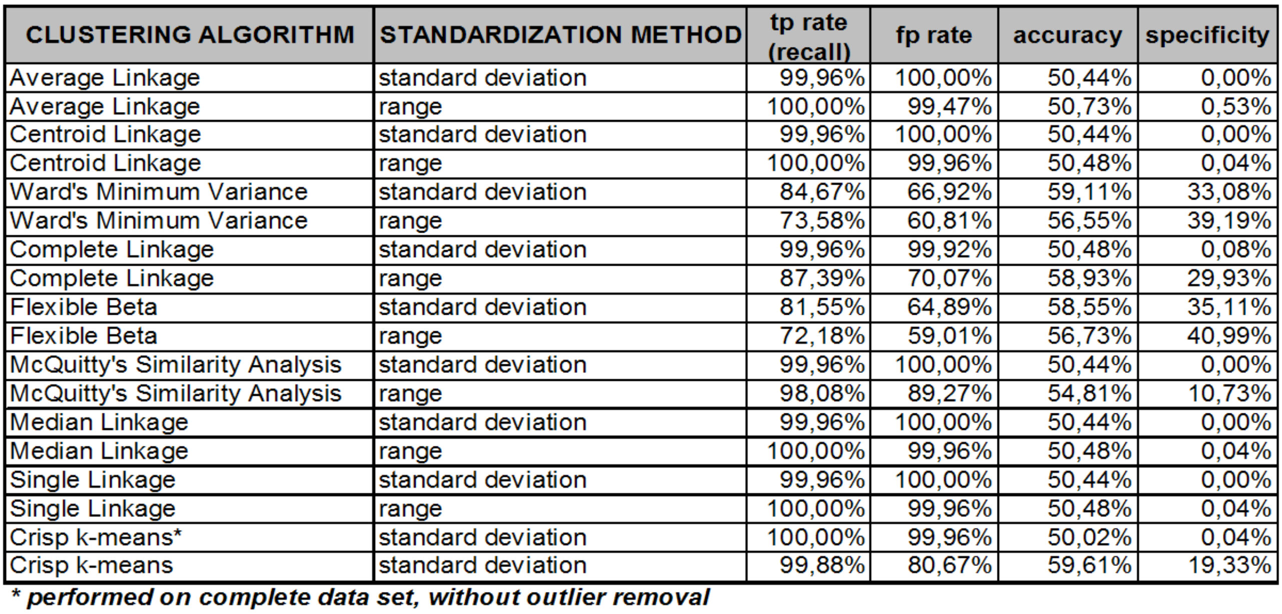
$$\boldsymbol{tp\ rate}\ (recall, hit\ rate) = \frac{positives\ correctly\ classified}{total\ positives}$$

$$\boldsymbol{fp\ rate}\ (false\ alarm\ rate) = \frac{negatives\ incorrectly\ classified}{total\ negatives}$$

$$\boldsymbol{accuracy} = \frac{true\ positives + true\ negatives}{positives + negatives}$$

$$\boldsymbol{specificity} = 1 - fp\ rate$$

Figure 1: Common performance metrics calculated from confusion matrix.



CLUSTERING ALGORITHM	STANDARDIZATION METHOD	tp rate (recall)	fp rate	accuracy	specificity
Average Linkage	standard deviation	99,96%	100,00%	50,44%	0,00%
Average Linkage	range	100,00%	99,47%	50,73%	0,53%
Centroid Linkage	standard deviation	99,96%	100,00%	50,44%	0,00%
Centroid Linkage	range	100,00%	99,96%	50,48%	0,04%
Ward's Minimum Variance	standard deviation	84,67%	66,92%	59,11%	33,08%
Ward's Minimum Variance	range	73,58%	60,81%	56,55%	39,19%
Complete Linkage	standard deviation	99,96%	99,92%	50,48%	0,08%
Complete Linkage	range	87,39%	70,07%	58,93%	29,93%
Flexible Beta	standard deviation	81,55%	64,89%	58,55%	35,11%
Flexible Beta	range	72,18%	59,01%	56,73%	40,99%
McQuitty's Similarity Analysis	standard deviation	99,96%	100,00%	50,44%	0,00%
McQuitty's Similarity Analysis	range	98,08%	89,27%	54,81%	10,73%
Median Linkage	standard deviation	99,96%	100,00%	50,44%	0,00%
Median Linkage	range	100,00%	99,96%	50,48%	0,04%
Single Linkage	standard deviation	99,96%	100,00%	50,44%	0,00%
Single Linkage	range	100,00%	99,96%	50,48%	0,04%
Crisp k-means*	standard deviation	100,00%	99,96%	50,02%	0,04%
Crisp k-means	standard deviation	99,88%	80,67%	59,61%	19,33%

***** performed on complete data set, without outlier removal***

Table 2: Results of preliminary classical clustering.

To get the full comprehension on algorithm performance several measures have to be considered simultaneously. Recall rate of 100% means unsuccessful churners recognition, if comes in combination with specificity under 1%. Losing one client causes greater losses for the bank, then investing in marketing campaign for several clients incorrectly classified as possible churners, which means that costs of false negatives are much higher then costs of false positives. In real clients population there are much less positive then negative instances, so liberal classifiers obtaining high recall rate and acceptable specificity are considered successful in business.

4 Model setup and prediction results

FCM has been repeatedly applied on the complete data set and on the data set without top 50 outliers, with 10 different values of the fuzzification parameter m, and different initial cluster seeds. It performed slightly better without outliers, what means that FCM is very robust against outliers' presence. From application point of view that is very good property of FCM, since it will not always be profitable for the bank to detect and remove outliers, not to mention the fact that these outliers are sometimes the most active and profitable clients and they need to be included in the model development. With crisp k-means it would not be possible, because it performed incredibly poorly with these clients.

Data set was splitted into two parts: training set and test set, in three different ratios. The ratio of 90% of clients in the training set and 10% of clients in test set was chosen. According to the values of the membership functions, the clients in fuzzy transitional conditions (FTC) were detected. For that purpose two new definitions were proposed.

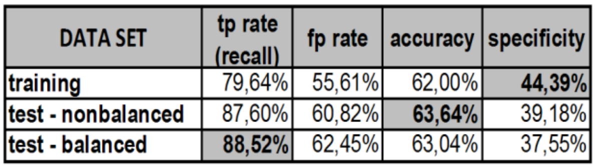
Definition 1. *Let p be the number of clusters in the FCM algorithm. Let us denote $max_{j=1}^{p}\{\mu_j(x_i)\} = \mu_{MAX}^1$ and $max_{j=1}^{p}\{\mu_j(x_i) \setminus \mu_{MAX}^1\} = \mu_{MAX}^2$ for the entity x_i. The entity x_i is said to be in the fuzzy transitional condition of the 1st degree if, for arbitrary small $\varepsilon > 0$, holds that $\mu_{MAX}^1 - \mu_{MAX}^2 < \varepsilon$.*

Definition 2. *Let p be the number of clusters in the FCM algorithm. Let us denote $max_{j=1}^{p}\{\mu_j(x_i)\} = \mu_{MAX}^1$. The entity x_i is said to be in the fuzzy transitional condition of the 2nd degree if, for arbitrary small $\varepsilon > 0$, holds that $\mu_{MAX}^1 - \frac{1}{p} < \varepsilon$.*

Subsets of clients in the FTC of both degrees, and with floating ε values, were further analyzed and the information gained from the fact about their membership values helped in explaining their behavior. Four prediction models were developed, based on the main idea of the distance of the new client from the clients in the training data set. For the predictive purpose in the 4th model, the definition of *distance of k instances* (DOKI) *sums* was introduced.

Definition 3. *Let p be the number of clusters in the FCM algorithm and X be the set of n entities with assigned membership values μ_j, $j = 1, \dots, p$. Distance of k instances sum i.e. $DOKI_j^k(x)$ sum for the new entity x_{n+1} is defined as the sum of membership values $\{\mu_j\}$ in the $j-th$ cluster of the k nearest entities from X, according to distance metric used in FCM.*

Calculation of DOKI sums requires the input parameter k and several different values were applied. Table 3 presents the results of FCM on the training set and prediction engine with DOKI sums applied on balanced and non-balanced test sets. Concept of DOKI sums might seem similar to k nearest neighbors approach, but DOKI sums up values of membership functions and not the pure distances. Recall rate for test sets were even higher then recall rate obtained with FCM on the training set. Improvement in recall was paid in slight decrease in specificity. As mentioned previously, it is more important to hit churners, even if it is paid by hitting some percentage of loyal clients. The cost minimization can be achieved later through more intelligent and multi-level communication channels.



DATA SET	tp rate (recall)	fp rate	accuracy	specificity
training	79,64%	55,61%	62,00%	**44,39%**
test - nonbalanced	87,60%	60,82%	**63,64%**	39,18%
test - balanced	**88,52%**	62,45%	63,04%	37,55%

Table 3: Results of FCM and DOKI prediction model.

5 Conclusions and further work

It is always challenging to deal with real data and business situations, where classical methods can rarely be applied in their simplest theoretical form. The main idea of the study – to prove that fuzzy logic and fuzzy data mining methods can find their place in the reality of retail banking – was completely fulfilled. FCM performed much better than the classical clustering and provided more hidden information about the clients, especially those in fuzzy transitional conditions. Three new definitions were introduced and had the impact on the overall work. Implementation of DOKI sums increased hit rate (recall) by 8,88% in comparison to pure FCM. A lot of work still needs to be done. In the near future every client and every selling opportunity will become important. Methods which require a lot of preprocessing and, above all, removing many outlying clients, will lose the battle with more efficient and robust methods. More accuracy should be obtained through better information exploitation of clients in fuzzy transitional conditions, and not through clients removal. Monitoring clients in FTCs and reacting as they approach to churners could be a way for more intelligent churn management. This requires analysis on larger data sets, including more transactional variables into the model and tuning ε. Model should also include costs of positive and negative misclassifications. Different segments of clients or clients having similar product lines could be modeled on their own, to find empirically best FCM parameters for each segment/product line.

Acknowledgement

The first author's opinions expressed in this paper do not necessarily reflect the official positions of Zagrebačka banka d.d.

This work has been supported by the Ministry of Science, Education and Sports, Republic of Croatia, under the grant No. 036-1300646-1986 and 098-0982560-2563.

References

[1] Berry J.A.M., Linoff S.G. (2004) *Data Mining Techniques For Marketing, Sales, and Customer Relationship Management*, 2nd Ed. Indianapolis: Wiley Publishing, Inc.

[2] Burez J., Van den Poel D. (2008) "Handling class imbalance in customer churn prediction", *Expert Systems with Applications*, In Press, available online 16 May 2008.

[3] Burez J., Van den Poel D. (2008) "Separating financial from commercial customer churn: A modeling step towards resolving the conflict between the sales and credit department", *Expert Systems with Applications* 35 (1-2), pp. 497-514.

[4] Coussement K., Van den Poel D. (2006) "Churn Prediction in Subscription Services: an Application of Support Vector Machines While Comparing Two Parameter-Selection Techniques", *Working Paper 2006/412*, Ghent University.

[5] Cox E. (2005) *Fuzzy Modeling and Genetic Algorithms for Data Mining and Exploration*, San Francisco: Morgan Kaufmann Publishers.

[6] De Oliveira, J.V., Pedrycz W. (editors) (2007) *Advances in Fuzzy Clustering and its Applications*, John Wiley & Sons Ltd.

[7] Fawcett T. (2004) ROC Graphs: Notes and Practical Considerations for Researchers, Netherlands: *Kluwer Academic Publishers*.

[8] Hadden J., Tiwari A., Roy R., Ruta D. (2005) "Computer assisted customer churn management: State-of-the-art and future trends", *Computers & Operations Research* 34, pp. 2902-2917.

[9] Mutanen T., Ahola J., Nousiainen S. (2006) "Customer churn prediction - a case study in retail banking", *ECML/PKDD 2006 Workshop on Practical Data Mining: Applications, Experiences and Challenges*, Berlin.

[10] Theodoridis S., Koutroumbas K. (2003) *Pattern Recognition*, 2nd Ed., San Diego, USA: Academic Press, Elsevier Science.

[11] Van den Poel D., Larivière B. (2004) "Customer attrition analysis for financial services using proportional hazard models", *European Journal of Operational Research* 157 (1), pp. 196-217.

[12] Yeo D. (2005) Applied Clustering Techniques Course Notes, Cary NC, USA: SAS Institute Inc.

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 800 staff, has 600 researchers, about 250 of whom are postgraduates, nearly 400 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 submit an email with the manuscript to one of the editors from the Editorial Board or to the Managing Editor. 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.

QUESTIONNAIRE

□ Send Informatica free of charge

□ Yes, we subscribe

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 sixteen 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 the European computer science and informatics 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 is free of charge for major scientific, educational and governmental institutions. Others should subscribe (see the last page of Informatica).

ORDER FORM – INFORMATICA

Name: ....

Title and Profession (optional): ....

....

Home Address and Telephone (optional): ....

....

Office Address and Telephone (optional): ....

....

E-mail Address (optional): ....

Signature and Date: ....

Informatica WWW:

http://www.informatica.si/

Referees:

Witold Abramowicz, David Abramson, Adel Adi, Kenneth Aizawa, Suad Alagić, Mohamad Alam, Dia Ali, Alan Aliu, Richard Amoroso, John Anderson, Hans-Jurgen Appelrath, Iván Araujo, Vladimir Bajič, Michel Barbeau, Grzegorz Bartoszewicz, Catriel Beeri, Daniel Beech, Fevzi Belli, Simon Beloglavec, Sondes Bennasri, Francesco Bergadano, Istvan Berkeley, Azer Bestavros, Andraž Bežek, Balaji Bharadwaj, Ralph Bisland, Jacek Blazewicz, Laszlo Boeszoermenyi, Damjan Bojadžijev, Jeff Bone, Ivan Bratko, Pavel Brazdil, Bostjan Brumen, Jerzy Brzezinski, Marian Bubak, Davide Bugali, Troy Bull, Sabin Corneliu Buraga, Leslie Burkholder, Frada Burstein, Wojciech Buszkowski, Rajkumar Bvyya, Giacomo Cabri, Netiva Caftori, Particia Carando, Robert Cattral, Jason Ceddia, Ryszard Choras, Wojciech Cellary, Wojciech Chybowski, Andrzej Ciepielewski, Vic Ciesielski, Mel Ó Cinnéide, David Cliff, Maria Cobb, Jean-Pierre Corriveau, Travis Craig, Noel Craske, Matthew Crocker, Tadeusz Czachorski, Milan Češka, Honghua Dai, Bart de Decker, Deborah Dent, Andrej Dobnikar, Sait Dogru, Peter Dolog, Georg Dorfner, Ludoslaw Drelichowski, Matija Drobnič, Maciej Drozdowski, Marek Druzdzel, Marjan Družovec, Jozo Dujmović, Pavol Ďuriš, Amnon Eden, Johann Eder, Hesham El-Rewini, Darrell Ferguson, Warren Fergusson, David Flater, Pierre Flener, Wojciech Fliegner, Vladimir A. Fomichov, Terrence Forgarty, Hans Fraaije, Stan Franklin, Violetta Galant, Hugo de Garis, Eugeniusz Gatnar, Grant Gayed, James Geller, Michael Georgiopolus, Michael Gertz, Jan Goliński, Janusz Gorski, Georg Gottlob, David Green, Herbert Groiss, Jozsef Gyorkos, Marten Haglind, Abdelwahab Hamou-Lhadj, Inman Harvey, Jaak Henno, Marjan Hericko, Henry Hexmoor, Elke Hochmueller, Jack Hodges, John-Paul Hosom, Doug Howe, Rod Howell, Tomáš Hruška, Don Huch, Simone Fischer-Huebner, Zbigniew Huzar, Alexey Ippa, Hannu Jaakkola, Sushil Jajodia, Ryszard Jakubowski, Piotr Jedrzejowicz, A. Milton Jenkins, Eric Johnson, Polina Jordanova, Djani Juričič, Marko Juvancic, Sabhash Kak, Li-Shan Kang, Ivan Kapustøk, Orlando Karam, Roland Kaschek, Jacek Kierzenka, Jan Kniat, Stavros Kokkotos, Fabio Kon, Kevin Korb, Gilad Koren, Andrej Krajnc, Henryk Krawczyk, Ben Kroese, Zbyszko Krolikowski, Benjamin Kuipers, Matjaž Kukar, Aarre Laakso, Sofiane Labidi, Les Labuschagne, Ivan Lah, Phil Laplante, Bud Lawson, Herbert Leitold, Ulrike Leopold-Wildburger, Timothy C. Lethbridge, Joseph Y-T. Leung, Barry Levine, Xuefeng Li, Alexander Linkevich, Raymond Lister, Doug Locke, Peter Lockeman, Vincenzo Loia, Matija Lokar, Jason Lowder, Kim Teng Lua, Ann Macintosh, Bernardo Magnini, Andrzej Małachowski, Peter Marcer, Andrzej Marciniak, Witold Marciszewski, Vladimir Marik, Jacek Martinek, Tomasz Maruszewski, Florian Matthes, Daniel Memmi, Timothy Menzies, Dieter Merkl, Zbigniew Michalewicz, Armin R. Mikler, Gautam Mitra, Roland Mittermeir, Madhav Moganti, Reinhard Moller, Tadeusz Morzy, Daniel Mossé, John Mueller, Jari Multisilta, Hari Narayanan, Jerzy Nawrocki, Rance Necaise, Elzbieta Niedzielska, Marian Niedq'zwiedziński, Jaroslav Nieplocha, Oscar Nierstrasz, Roumen Nikolov, Mark Nissen, Jerzy Nogieć, Stefano Nolfi, Franc Novak, Antoni Nowakowski, Adam Nowicki, Tadeusz Nowicki, Daniel Olejar, Hubert Österle, Wojciech Olejniczak, Jerzy Olszewski, Cherry Owen, Mieczyslaw Owoc, Tadeusz Pankowski, Jens Penberg, William C. Perkins, Warren Persons, Mitja Peruš, Fred Petry, Stephen Pike, Niki Pissinou, Aleksander Pivk, Ullin Place, Peter Planinšec, Gabika Polčicová, Gustav Pomberger, James Pomykalski, Tomas E. Potok, Dimithu Prasanna, Gary Preckshot, Dejan Rakovič, Cveta Razdevšek Pučko, Ke Qiu, Michael Quinn, Gerald Quirchmayer, Vojislav D. Radonjic, Luc de Raedt, Ewaryst Rafajlowicz, Sita Ramakrishnan, Kai Rannenberg, Wolf Rauch, Peter Rechenberg, Felix Redmill, James Edward Ries, David Robertson, Marko Robnik, Colette Rolland, Wilhelm Rossak, Ingrid Russel, A.S.M. Sajeev, Kimmo Salmenjoki, Pierangela Samarati, Bo Sanden, P. G. Sarang, Vivek Sarin, Iztok Savnik, Ichiro Satoh, Walter Schempp, Wolfgang Schreiner, Guenter Schmidt, Heinz Schmidt, Dennis Sewer, Zhongzhi Shi, Mária Smolárová, Carine Souveyet, William Spears, Hartmut Stadtler, Stanislaw Stanek, Olivero Stock, Janusz Stokłosa, Przemysław Stpiczyński, Andrej Stritar, Maciej Stroinski, Leon Strous, Ron Sun, Tomasz Szmuc, Zdzislaw Szyjewski, Jure Šilc, Metod Škarja, Jiři Šlechta, Chew Lim Tan, Zahir Tari, Jurij Tasič, Gheorge Tecuci, Piotr Teczynski, Stephanie Teufel, Ken Tindell, A Min Tjoa, Drago Torkar, Vladimir Tosic, Wieslaw Traczyk, Denis Trček, Roman Trobec, Marek Tudruj, Andrej Ule, Amjad Umar, Andrzej Urbanski, Marko Uršič, Tadeusz Usowicz, Romana Vajde Horvat, Elisabeth Valentine, Kanonkluk Vanapipat, Alexander P. Vazhenin, Jan Verschuren, Zygmunt Vetulani, Olivier de Vel, Didier Vojtisek, Valentino Vranić, Jozef Vyskoc, Eugene Wallingford, Matthew Warren, John Weckert, Michael Weiss, Tatjana Welzer, Lee White, Gerhard Widmer, Stefan Wrobel, Stanislaw Wrycza, Tatyana Yakhno, Janusz Zalewski, Damir Zazula, Yanchun Zhang, Ales Zivkovic, Zonling Zhou, Robert Zorc, Anton P. Železnikar

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, Vožarski pot 12, 1000 Ljubljana, Slovenia.
The subscription rate for 2009 (Volume 33) 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: Dikplast Kregar Ivan s.p., Kotna ulica 5, 3000 Celje.

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):
Robotics Society of Slovenia (Jadran Lenarčič)
Slovene Society for Pattern Recognition (Franjo Pernuš)
Slovenian Artificial Intelligence Society; Cognitive Science Society (Matjaž Gams)
Slovenian Society of Mathematicians, Physicists and Astronomers (Bojan Mohar)
Automatic Control Society of Slovenia (Borut Zupančič)
Slovenian Association of Technical and Natural Sciences / Engineering Academy of Slovenia (Igor Grabec)
ACM Slovenia (Dunja Mladenič)

Informatica is surveyed by: 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

The issuing of the Informatica journal is financially supported by the Ministry of Higher Education, Science and Technology, Trg OF 13, 1000 Ljubljana, Slovenia.

Volume 33 Number 2 May 2009 ISSN 0350-5596

Informatica

An International Journal of Computing and Informatics

An Exquisite Mutual Authentication Scheme with Key Agreement Using Smart Card	C.-H. Liao, H.-C. Chen, C.-T. Wang	**117**
Routing Scalability in Multicore-Based Ad Hoc Networks	A. Marowka	**125**
Similarity Measures for Relational Databases	M. Hajdinjak, A. Bauer	**135**
Robust H_∞ Control of a Doubly Fed Asynchronous Machine	G. Sofiane, Y. Said, S. Moussa	**143**
Balancing Load in a Computational Grid Applying Adaptive, Intelligent Colonies of Ants	M.A. Salehi, H. Deldari, B.M. Dorri	**151**
Improving Part-of-Speech Tagging Accuracy for Croatian by Morphological Analysis	Ž. Agić, Z. Dovedan, M. Tadić	**161**
Applying SD-Tree for Object-Oriented Query Processing	I.E. Shanthi, R. Nadarajan	**169**
Improving Design Pattern Adoption with an Ontology-Based Repository	L. Pavlič, M. Heričko, V. Podgorelec, I. Rozman	**181**
Historical Impulse Response of Return Analysis Shows Information Technology Improves Stock Market Efficiency	W. Leigh, R. Purvis	**191**
Fall Detection and Activity Recognition with Machine Learning	M. Luštrek, B. Kaluža	**197**
Rajan Transform and its Uses in Pattern Recognition	E.N. Mandalapu, E.G. Rajan	**205**
A Petri-Net Approach to Refining Object Behavioural Specifications	K.-S. Cheung, P. K.-O. Chow	**213**
A Multi-Class SVM Classifier Utilizing Binary Decision Tree	G. Madzarov, D. Gjorgjevikj, I. Chorbev	**225**
Churn Prediction Model in Retail Banking Using Fuzzy C-Means Algorithm	D. Popović, B.D. Bašić	**235**

Informatica **33** (2009) Number 2, pp. 117–241