Volume 26 Number 3 November 2002 ISSN 0350-5596

Informatica

An International Journal of Computing and Informatics

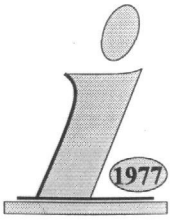
Special Issue:

Cryptology and Network Security

Guest Editors:

Francesco Bergadano, Chuan-Kun Wu

Informatica **26** (2002) Number 3, pp. 245–345



The Slovene Society Informatika, Ljubljana, Slovenia

Informatica

An International Journal of Computing and Informatics

Archive of abstracts may be accessed at USA: http://, Europe: http://ai.ijs.si/informatica, Asia: http://www.comp.nus.edu.sg/ liuh/Informatica/index.html.

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 2002 (Volume 26) is
– USD 80 for institutions,
– USD 40 for individuals, and
– USD 20 for students
Claims for missing issues will be honored free of charge within six months after the publication date of the issue.

LaTeX Tech. Support: Borut Žnidar, Kranj, Slovenia.
Lectorship: Fergus F. Smith, AMIDAS d.o.o., Cankarjevo nabrežje 11, Ljubljana, Slovenia.
Printed by Biro M, d.o.o., Žibertova 1, 1000 Ljubljana, Slovenia.

Orders for subscription may be placed by telephone or fax using any major credit card. Please call Mr. R. Murn, Jožef Stefan Institute: Tel (+386) 1 4773 900, Fax (+386) 1 219 385, or send checks or VISA or Eurocard card number or use the bank account number 900–27620–5159/4 Nova Ljubljanska Banka d.d. Slovenia (LB 50101-678-51841 for domestic subscribers only).

Informatica is published 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: AI and Robotic Abstracts, AI References, ACM Computing Surveys, ACM Digital Library, Applied Science & Techn. Index, COMPENDEX*PLUS, Computer ASAP, Computer Literature Index, Cur. Cont. & Comp. & Math. Sear., Current Mathematical Publications, Cybernetica Newsletter, DBLP Computer Science Bibliography, Engineering Index, INSPEC, Linguistics and Language Behaviour Abstracts, Mathematical Reviews, MathSci, Sociological Abstracts, Uncover, Zentralblatt für Mathematik

The issuing of the Informatica journal is financially supported by the Ministry of Education, Science and Sport, Trg OF 13, 1000 Ljubljana, Slovenia.

Post tax payed at post 1102 Ljubljana. Slovenia taxe Percue.

Special Issue on Cryptology and Network Security

We have come to an age of information explosion. Information is useful in all aspects of our life, however, without proper management, to get the right information from the massive data is not an easy task. Fortunately computer networks, particularly the Internet, enables people to share information worldwide. The recent development of electronic commerce has extended the use of the Internet, and companies can do their business over the Internet. It has brought many conveniences to our everyday life.

In spite of the conveniences that computer networks have brought to us, they also have brought many security problems: poorly designed algorithms, inappropriate use of security systems, inappropriate choice of passwords, inappropriate personnel management, and physical network destruction are all examples that information transmitted over the public networks are under threat. Some of the threats may cause data confidentiality problems, some may cause entity authentication problems, some may cause information integrity problems, and some may cause privacy problems. To eliminate these security problems, traditional legislation is not good enough, even newly added legislations covering electronic criminals are not sufficient enough to release people from having those concerns, because convincing evidence may be hard to get. It is realized that technical prevention and detection will continuously play an essential role in information security.

Techniques for information security have been used for many years. Formal and overwhelming study on this topic started from the mid of the 20th century. It is just in recent years that those techniques are widely used in commercial activities.

The fundamental techniques for information security are from cryptography, a science of designing algorithms for data encryption/decryption, digital signatures and hash functions which can be used in various areas. Cryptanalysis, on the other hand, focuses on finding security flaws of the designed cryptosystems. These fundamental cryptosystems serving as basic building blocks will be used to further design network security protocols, which are real applications to information security.

The needs for information security today is almost everywhere if computer networks is a necessary component, and the study of information security is extensive worldwide. Many journals have included information security as one of their themes, and there are some new journals created in recent years that are particularly devoted to publishing papers on information security. Even though, there are still many good research papers that can hardly find an appropriate literature to publish. It is under this circumstances that we have selected 8 papers from the 2002 International Workshop on Cryptology and Network Security to form this special issue. Hope this excercise will somehow promote information security research.

The 2002 International Workshop on Cryptology and Network Security (CNS02) was held in San Francisco, California, Sep. 26-28, 2002. It was the second time the workshop was organized with the same theme. The workshop accepted 18 papers out of 27 submissions. Each paper was sent to three referees (normally program committee members) for reviewing. Based on the review reports (rating of the papers and confidence of reviewers considered), 8 papers have been selected for publication as a special issue of *Informatica – An International Journal of Computing and Informatics*. These 8 papers cover new techniques in cryptography, cryptanalysis, key management schemes, water marking techniques, and secure web transaction protocols:

A Complete Group Cryptographic System
by Y. Mu and V. Varadharajan

Group cryptography is a relatively new topic driven by the wide use of distributed systems. One of the important applications of group cryptography is group signatures. In a group signature system, any group member can create a signature on behalf of the group, where the validity of the group signature can be verified by the public having the group's public key (or keys). The group signature verifier cannot tell which group member created a particular signature, but in case of dispute, the signer can be identified internally within the group, normally by the group manager.

Digital signatures (including group signatures) are often used together with data encryption. This paper proposes a new approach of integrating data encryption and group signatures together. This is of significance both in theory and practice.

Cryptanalysis of Some Hash Functions Based on Block Ciphers and Codes
by Hongjun Wu, Feng Bao, Robert H. Deng

Cryptosystems are designed to have certain desired security strength. However, many cryptosystems are shown to be less secure than what was initially targeted. Even worse, some crypto algorithms have been broken not long after their publication, some even broken before their formal publication. This is not surprising as there is no theoretical warranty for any cryptosystem designed so far. Therefore, cryptanalysis is an important procedure for any cryptosystem before it is adopted for practical use.

This paper gives a cryptanalysis of some particular hash functions designed using existing block ciphers and codes. Some weakness has been found, which

means that those hash functions are not as secure as they were targeted. The flaws found can be of help in designing similar hash functions, or in design other cryptosystems using the similar techniques.

Analysis of AES S-box with Walsh Spectrum
by Baodian Wei, Dongsu Liu, and Xinmei Wang

Cryptanalysis can be on a whole system, or part of the system, and this paper is the latter case. It is believed that S-box is a core technique in the security of symmetric key block ciphers, although a weak S-box may not necessarily mean the straight forward broken of the cipher. This paper presents a cryptanalysis on the S-box of advanced encryption standard (AES) that was adopted by the US in 2000. The cryptanalysis uses the technique of Walsh spectrum. Some cryptologic properties of the S-box are given in terms of Walsh spectrum representation. These results further supports that the Rijndael algorithm of AES has a good design. The paper also noted that further research needs to be done on how those Walsh spectrum properties can be converted into specific attacks with less effort than brute force attack.

Practical Construction for Multicast Re-keying Schemes

by Chun-yan Bai, Roberta Houston, and Gui-liang Feng

Sometimes a new technique may result in unexpected research results. However, to explore an effective new method is not easy. Coding theory has been an overwhelming research area dominating Information Theory's research that started from Shannon's age. There have been many research results from the combination of error-correcting codes and cryptography. This paper uses Reed-Solomon codes and Algebraic-Geometric codes to construct re-keying schemes in multicast systems. Although more research needs to be done before implementation of the re-keying schemes in order to make them practical for large groups (e.g. in the application of video on demand), the method used in the paper is encouraging and appropriate.

Multisecret Sharing Immune Against Cheating
by Josef Pieprzyk and Xian-Mo Zhang

Secret sharing being a means of secret key management has been proposed for many years. In a secret sharing scheme, each participant has a piece of secret information (called share), and when some of them put their shares together, the original secret can be constructed. With smaller than the required group size, it is computationally infeasible to compute the original secret given their shares. However, in the case where a group of the share holders suppose to be able to recover the secret, if one of the participants is not honest and presents a random number instead of the correct share, then in secret recovery process, if all the others present their correct shares, the original secret cannot be recovered. However, the one who cheated can recover the original secret since the other shares have been given. In this case, although everybody noticed that someone cheated, no one knows who it was, except for the cheater itself. In light the possibility of cheating, cheating resistant secret sharing schemes would be more practical.

This paper gives specific construction methods for multisecret sharing schemes that are immune against cheating. The construction is very technical but the approach is also practical.

Digital Semipublic Watermarking
by Valeri Korzhik, Guillermo Morales-Luna, Dmitry Marakov and Irina Marakova

With the rapid growing of electronic commerce and web publications in recent years, protection of intellectual property of electronic documents becomes more and more subtle. Although there are legal procedures of accusing those who infringe the laws, technical process is still critical to gather the evidence of the infringement. One of the very effective techniques in protecting copyright of electronic documents is digital watermarking, and there has been large amount of study on this topic, particularly on images and multimedia documents. This paper studies the techniques for digital semipublic watermarking, where a new concept of "semipublic" is introduced. Robustness of the watermarking technique is analyzed.

An Improved Anonymous Fingerprinting Scheme
by Bo Yang and Yong Li

Fingerprinting being another technique in copyright protection of electronic documents is also important. It can be used not only for visual documents as watermarking would, but also for other type of electronic documents including softwares. Using cryptographic approach to achieve fingerprinting normally involves large amount of computation. This paper improves one of the existing fingerprinting schemes by reducing the computational complexity significantly. This improvement has both theoretical and practical significances, though computers are becoming faster and faster.

An Efficient Anonymous Fingerprinting Scheme for secure web transaction over Internet
by Changjie Wang, Yumin Wang, Fangguo Zhang

One of the important applications in electronic commerce is intelligent mobile agents. Since mobile

agents are supposed to travel from one host to another over the network, in particular the Internet, and often they are out of control by the originator when they come to a foreign host, security issues associated with mobile agents become a big concern. Letting mobile agents to be communicating with the originator frequently would eliminate the chances of the agents to be tempered by a hostile host, but this would significantly reduce the mobility of the agents, and produces more traffic problems. This paper proposes a new scheme which is claimed to enable secure web transactions over the Internet.

All these papers have variety of contributions to Cryptology and Network Security, which meets the theme of the workshop. Selection of the papers was not based on the area, it was based on the reviewers' comments only.

There are also many other high quality papers which have not been selected for the publication by this special issue of the journal for various reasons.

Continuous support from scholars to the workshop and the Journal is highly appreciated.

Dr. Francesco Bergadano, Dr. Chuan-Kun Wu
Special Issue Editors

Appendix: Referees of the selected papers

Francesco BERGADANO	(Universita di Torino, Italy)
Kefei CHEN	(Shanghai Jiaotong University, China)
Lily CHEN	(Motorola, USA)
Dengguo FENG	(Chinese Academy of Science, China)
Guang GONG	(University of Waterloo, Canada)
Ren-Junn HWANG	(TamKang University, Taiwan)
Shin-Jia HWANG	(TamKang University, Taiwan)
Chi-Sung LAIH	(National Cheng Kung University, Taiwan)
Wei-Bin LEE	(Feng Chia University, Taiwan)
Chin-Laung LEI	(National Taiwan University, Taiwan)
Mitsuru MATSUI	(Mitsubishi Electric, Japan)
Yi MU	(Macquarie University, Australia)
Toshihisa NISHIJIMA	(Hosei University, Japan)
Kouichi SAKURAI	(Kyushu University, Japan)
Chuan-Kun WU	(Australian National University, Australia)
Tzong-Chen WU	(National Taiwan Univ of Science & Technology, Taiwan)
Xinmei WANG	(Xidian University, China)

Group Cryptography: Signature and Encryption

Yi Mu and Vijay Varadharajan
Department of Computing, Macquarie University,
Sydney, Australia
Email: {ymu,vijay}@ics.mq.edu.au

Keywords: public key cryptography, group signature, distributed encryption

Received: May 12, 2001

Traditional group signature schemes reflect only one side of the spectrum of public cryptography, i.e., signing, where the group public key is used for a sole purpose: verification of group signatures. This paper describes a new group cryptographic system that presents a promise for both signing and encryption. That is, besides verifications, a sole group public key can be also used for encryption and the associated decryption can be implemented by any member in the designated group. The proposed system meets all key features for an ideal group cryptography.

1 Introduction

In practice, it may be necessary to construct a cryptographic group that possesses a unique group public key and each of its members possesses a private key associated with the group public key. Any member in the group can sign on behalf of the group and the signatures can be verified using the group public key. The group public key can be also used for encryption where encrypted message can be decrypted by any member in the group. We refer to this kind of systems as public-key group cryptography covering both Group Signature and Distributed Encryption. The concept of group oriented cryptography was actually introduced by Desmedt[7] in 1987, however from the best of our knowledge, there is no a compound scheme that includes both signature and encryption.

The concept of group signatures was initially introduced by Chuam[5] and then several major improvements have been proposed[6, 2, 3, 4], especially for Camenisch's Scheme[4]. All existing group signature schemes cannot be converted to include group oriented encryption. For example, the group signature scheme in [4] is based on the RSA signature scheme[11], where the group certificate of a member is signed by the group manager using his RSA private key. Any member in the group can prove his/her membership without revealing his/her identification. Since the group public key is the group manager's public key whose associated private key is used to sign the group certificates of group members, the message encrypted using the group public key can be decrypted only by the group manager but not group members. It is found that using the RSA based method is infeasible for us to construct a group cryptographic system that allows encryption.

We often refer to a group oriented encryption as distributed encryption. This is because the encrypted message is distributed to the group and all members in the group can decrypt the message. There are two such schemes were independently proposed[1, 9]. Unfortunately, these schemes cannot be converted to include group signatures.

In this paper, we present a new group public-key scheme where the group public key can be used for both encryption and signing. The proposed new group system is based on the ElGamal signature scheme[8]. The certificate of a group member is signed by the group manager using the ElGamal signature scheme. We will see later that our method is naturally associated with the ElGamal encryption scheme.

The rest of this paper is arranged as follows. In Section 2, we give the model of our group cryptographic system. In Section 3, we construct some building blocks that will be later used in the construction of the group signatures and encryptions. In Section 4, we study the details of group signature scheme. In Section 5, we show how to use the group public key to encrypt a message. In Section 6, we propose some potential applications based our scheme. We then conclude the paper in Section 7.

2 Model

Like a normal group signature system, there is a designated group that consists of a manager and a group of members $\mathcal{M}$.

- Group Manager:

  The group manager has a pair of personal private and public key. The group manager's personal public key is also the unique group public key. The group manager's private key is used to sign group certificates of group members. The certificates are associated with group members' private keys. The group public key can be used to encrypt a message and all legitimate group members can decrypt it. The group public key is also used to verify a group signature from any member of the group.

- Group Members:

  A member $M_l \in \mathcal{M}$ can prove to any party that its certificate is indeed signed by the group manager. Every member has a private key known to itself only. The exponential function of its private key is signed by the group manager and the resultant signature token is the group certificate of the member. This private key can be used to sign on the group's behalf and decrypt a message encrypted with the group public key. The signer of a group signature is anonymous to all parties involved except the group manager. There is a revocation algorithm that allows the group manager to find out who is the signer of a group signature.

There are seven major algorithms in our system: (1) The algorithm of key generation that produces a pair of group manager's private and public key as well as the private keys group members. (2) The algorithm of certificate signing that takes an exponential function of a group member's private key and the private key of the group manager as input and produces an ElGamal signature as output. (3) The group signature algorithm that takes a message and the private key of a group member as input and produces a branch of signature knowledge proofs based on the ElGamal algorithm. (4) The verification algorithm of group signatures that takes the signature knowledge proofs as input and produces a boolean value, true or false. (5) The encryption algorithm that takes a message and the group public key as input and produces a ciphertext based on the ElGamal encryption. (6) The decryption algorithm that takes the ciphertext and a group member's private key as input and returns the original message. (7) The revocation algorithm that takes a signature knowledge as input and returns the identity of the corresponding signer.

The proposed group cryptographic system meets the following requirements:

- Robust. The size of the group public key is independent of the number of group members. Therefore, the number of members can be increased without changing the group public key.

- Group member anonymity. Any verifier of a group signature cannot find who is the signer, but he knows that the signature definitely comes from one of members in the designated group.

- Group member untraceability. If a group member has generated two group signatures, the verifier cannot find any connection between them.

- Revocation. Group manager can find who is the signer of a group signature when needed.

- Non-repudiation. Group manager cannot sign on behalf of any group member, therefore the signer cannot deny his action of signing.

3 Preliminary and Building Blocks

In this section, we give the definition of signature knowledge proofs of discrete logarithms. We consider three building blocks. The first two proofs were given by Camenisch[2], but used under our requirements. The third one shows a more efficient version of signature knowledge proof of double discrete logs.

We follow the Camenisch-Stadler setting[4] assuming that there is a cyclic group $G = \langle g \rangle$ of order n. There exists $h \in \mathbb{Z}_n$ such that g^{h^x} for some x exists (x is the smallest value).

3.1 Signature Knowledge Proof of a Discrete Logarithm

The first signature knowledge proof involves only a single discrete log. The signer/prover has a pair of secret and public key (x, y), where $y = g^x$ for $g \in G$ and $x \in \mathbb{Z}_n$. We also assume a collision resistant hash function $H : \{0,1\}^* \rightarrow \{0,1\}^\ell$, where ℓ is the length of a hash value. This signature knowledge proof is equivalent to a Schnorr signature[2, 12].

Definition 1 *A pair* $(c, \sigma) \in \{0,1\}^* \times \mathbb{Z}_n$ *satisfying* $c = H(m||y||g||g^\sigma y^c)$ *is a signature of knowledge of discrete logarithm of the element* $y \in \mathbb{Z}_n^*$ *to base* g *on message* $m \in \{0,1\}^*$ *and is denoted* $\mathsf{SKLOG}[x : y = g^x]$.

This discrete log proof shows that the signer knows the value of the secret key x, i.e. $x = \log_g y$, but reveals no additional information to the verifier. In the signing/proof process, the signer chooses an integer $r \in_R \mathbb{Z}_n$ and computes $c \leftarrow H(m||y||g||g^r)$ and $\sigma = r - cx$. Obviously, the proof can only be generated when knowing the value of x. The verification is done by computing the hash value c and checking the equality $g^r \stackrel{?}{=} g^\sigma y^c$.

3.2 Signature Knowledge Proof of Representation of Discrete Logarithms

We will use Camenisch and Stadler's definition of signature knowledge of representation[4].

Definition 2 *A signature of the knowledge of representation of* y *and* z *to base* g_1 *and* g_2 *(in* G*) on message* m *is denoted as follows*

$$\mathsf{SKREP}[(\alpha_1, \alpha_2) : y = g_1^{\alpha_1} \wedge z = g_1^{\alpha_1} g_2^{\alpha_2}]$$

The signature consists of a triplet $(c, s_1, s_2) \in \{0,1\}^\ell \times \mathbb{Z}_n^2$ *satisfying*

$$c = H(m||g_1||g_2||y||y^c g_1^{s_2}||z^c g_1^{s_1} g_2^{s_2})$$

for $s_1 = r_1 - c\alpha_1$ *and* $s_2 = r_2 - c\alpha_2$. □

This is used for proving the discrete logarithm of y to the base g_1 and a representation of z to the base g_1 and g_2, and

that the g_2-part of this representation equals the discrete logarithm of y to the base g_1. The signing procedures are as follows:

- Compute $a = g_1^{r_2}, a_1 = g_1^{r_1}$ and $a_2 = g_2^{r_2}$ for $r_1, r_2 \in_R \mathbb{Z}_n$.
- Compute $c = H(m||g_1||g_2||y||g_1^{r_2}||g_1^{r_1}g_2^{r_2})$.
- Compute s_1 and s_2, which can be done by the signer/prover only.

To verify the signature knowledge, the verifier needs to compute

$$c = H(m||g_1||g_2||y||a||a_1a_2)$$

and verify: $a \stackrel{?}{=} y^c g_1^{s_2}$ and $a_1a_2 \stackrel{?}{=} z^c g_1^{s_1} g_2^{s_2}$.

In our system, signature knowledge proofs require untraceability, i.e., a proof should have no any traceable link with the previous proof. This can be done by simply using a different base for every proof, i.e. replacing g with g^τ, where τ is a secret random number known to the prover only. It can be easily applied to SKLOG.

3.3 Signature Knowledge Proof of a Double Discrete Logarithm

We also need to use the knowledge proof of double discrete logarithms[13, 4]. The proof algorithm is based on a cut-and-choose method. Here, we give a much more efficient method without using the cut-and-choose.

Let x be the secret key and $z = g^y$, where $y = h^x$ that is then given to the verifier.

Definition 3 *A pair* $(c, \sigma) \in \{0,1\} \times \mathbb{Z}_n$ *satisfying* $c = H(m||g||h||h^r||(g^{h^\sigma})^{h^{cr}})$ *is a signature of knowledge of double discrete logarithms of the element* y *to bases* g *and* h *on message* $m \in \{0,1\}^*$ *and is denoted by* SKLOGLOG$[x : z = g^{h^x}]$.

In the process of a proof, the signer

- picks an integer r at random,
- computes the hash value $c = H(m||g||h||h^r||z)$,
- computes $\sigma = x - rc$.

To verify the signature proof, you compute the hash value $c = H(m||g||h||h^r||z)$ and check the equality $z \stackrel{?}{=} (g^{h^\sigma})^{h^{cr}}$.

It is clear that the signer cannot cheat in a proof, since the hash value c ensures that the signer cannot change the value of h^r.

3.4 Security of the ElGamal Signature Scheme

Let $k \in_R \mathbb{Z}_n$ be a secret key of a signer and $z = g^k$ be its corresponding public key. To sign a message m, the signer selects an integer $\gamma \in_R \mathbb{Z}_n$, computes his signing commitment $a = g^\gamma$, and then $C = \gamma^{-1}(m - ak) \bmod n$. The ElGamal signature consists of the triplet (a, C, m). To verify the signature, you check whether or not the following equality holds: $z^a a^C = g^m$.

The original ElGamal signature scheme is existentially forgeable[10]. It suffers from the so-call one-parameter and two-parameter forgeries. The problem can be fixed by using a cryptographic hash function, but it is not suitable for our case. Fortunately, in our system, the message m is an exponent; therefore those forgeries do not apply to our system.

Lemma 1 *Given the tuple* (z, a, C, m) *and* m *is an exponent, it is computationally infeasible to find* a' $(\neq a)$, C' $(\neq C)$, m' $(\neq m)$ *in probabilistic polynomial time such that the verification equality still holds:* $z^{a'} a'^{C'} = g^{m'}$.

The proof is easy once we understand the forgeries. The reader is referred to Ref. [10] for the detail.

4 Construction of a Complete Group Cryptographic System

In our scheme, Alice signs Bob's group certificate using ElGamal signature scheme rather than the RSA. This eliminates the need of using e-th root knowledge proof of discrete logarithms that is quite inefficient and enables a compound scheme of signature and encryption.

4.1 Setup

The group public key consists of a doublet, (z_1, z_2), where $z_1 = g^{k_1}$ and $z_2 = g^{k_2}$. k_1 and k_2 are the corresponding secret keys known to Alice only. The signing key for Alice is $k = k_1 + k_2$ and her corresponding signature verification key is $z = z_1 z_2$.

Each member of the group is issued with a group certificate signed by the unique group manager, Alice. Suppose Bob is a member of the designated group. To obtain his group certificate, Bob selects an integer x and computes $y = h^x \bmod n$ as his group key, where x is the secret known to himself only and y is sent to Alice. Alice then signs $y - 1$ using the ElGamal signature scheme to form Bob's group certificate, $C = \gamma^{-1}(y - 1 - ak) \bmod n$, where

- γ is a large random number known to Alice only. Note that γ is unique to Bob.
- $a = g^\gamma$ is Alice's signing commitment.
- y and C are known to Alice and Bob but not others.

Bob's group key y along with a are used by Alice to construct Bob's decryption key $d = g^y z_1^a$, which also satisfies $d = g^{\eta_1} z_2^{\eta_2}/a$ for suitable η_1 and η_2, where $y + k_1 a \neq \eta_1 + k_2\eta_2 - \gamma$ under modulo n and the difference between

them must divide n. Note that it is easy for Alice to compute d, since she has the values of k_1 and k_2, but computing d by any one else is equivalent to solving a discrete logarithm problem.

Lemma 2 *The setup of group member's decryption key is secure and collusion-resistant.*

Proof: This is because it is impossible for group members to find k_1 and k_2 from available information. For Bob, by using his information, he can establish an equation intending to find k_1 and k_2: $y + k_1 a = \eta_1 + k_2\eta_2 - \gamma + \lambda$, where λ is the product of an integer and a factor of n. Noting that γ is unknown to Bob and is unique to Bob, the number of equations to solve k_1 and k_2 by collusion is always less than unknown variables. □

At the end of setup process, Bob obtains his signed group certificate C along with the tuple $(x, y, a, d, \eta_1, \eta_2)$ as his secret keys (including C), where x is Bob's signing key known to Bob only. Therefore, No one else including Alice can sign for Bob.

4.2 Signing Protocol

To sign a message $m \in \mathbb{Z}_n$, Bob does the following:

- Selects $\theta \in_R \mathbb{Z}_n$ as the blind factor used for the current signing to make the signature knowledge untraceable.
  - Blinds the base, $\tilde{g} = g^{\theta}$.
  - Blinds the group public key, $\tilde{z} = z^{\theta}$.
  - Blinds the Alice's signing commitment, $\tilde{a} = a^{\theta}$.
- Computes $u_1 = \tilde{z}^a$, $u_2 = \tilde{a}^C$, $u_3 = \tilde{g}^{h^x}$.
- Prepares the proof of equality: $\tilde{g}u_1u_2 = u_3$, which is equivalent to an ElGamal signature verification, provided that the following proofs are true.
  - Proves $u_1(a) \Leftrightarrow u_2(a)$, which denotes that a of u_1 divides $\tilde{a}$ of u_2.
    - Computes $u_1^{u_2} = \tilde{z}^{\tilde{a}^{C+\theta^{-1}}}$.
    - Prepares the proof: $S_0 \stackrel{\text{def}}{=} \mathsf{SKLOGLOG}[\delta : \tilde{z}^{\tilde{a}^{\delta}}]$ for $\delta = C + \theta^{-1}$.
  - Prepares the following signature knowledge proofs:
    - $S_1 \stackrel{\text{def}}{=} \mathsf{SKREP}[(\theta, a, C) : z^{\theta} \wedge g^{\theta}\tilde{z}^a\tilde{a}^C](m)$. In this proof, Bob proves his knowledge in the following discrete logarithms: $\theta = \log_g \tilde{g}$, $\log_g \tilde{g} = \log_z \tilde{z}$, $a = \log_{\tilde{z}} \tilde{z}^a$, and $C = \log_{\tilde{a}} \tilde{a}^C$
    - $S_2 \stackrel{\text{def}}{=} \mathsf{SKLOGLOG}[x : u_3](m)$

  These proofs can be executed only by a legitimate group member - in our case, Bob, because only he knows x and his certificate, $y - 1$, which has been signed by Alice. In the SKREP verification, $g^{\theta}, \tilde{z}^a, \tilde{a}^C$ should be known to the verifier. This differs from the original proof, but maintains the same functionality (in fact this proof is stronger).

The signature tuple consists of $(m, g, \tilde{g}, \tilde{a}, \tilde{z}, u_1, u_2, u_3, S_0, S_1, S_2)$. The verification protocol is as follows:

- Verification:
  - Compute: $u_1^{u_2}$.
  - Verify: $u_1(a) \Leftrightarrow u_2(a)$.
  - Verify: S_0, S_1, S_2.
- The verifier verifies the equalities, $\tilde{g}u_1u_2 \stackrel{?}{=} u_3$.

Completeness: The verification of S_1 shows that the same blind parameter and correct bases have been used on the left side of the verification equation. The verification of S_2 shows that Bob has the knowledge of some values (a and C) on the left side of the equation, whereas he does not know that Bob has used the correct value of a. S_0 can be verified by the verifier, showing that Bob indeed knows the value of $x = \log_{\tilde{g}}(\log_h u_3)$.

Soundness: Bob cannot cheat in the signing process. Bob can only prove that $y - 1$ has been signed by Alice, but not anything else. The lemma below supports this point.

Lemma 3 *There exists no a variant of y (say, y') such that both y and y' satisfy equation $\tilde{g}u_1u_2 = u_3$.*

Proof: Two cases should be addressed. First, let ϵ be an integer in $\mathbb{Z}_n$ such that $\tilde{g}^{\epsilon}u_1^{\epsilon}u_2^{\epsilon} = u_3^{\epsilon}$. Since the base $\tilde{g}$ (in this case, $\tilde{g}^{\epsilon}$) is public, the base for proof S_0 must be $\tilde{g}^{\epsilon}$. Second, let θ be the product of two integers, θ_1 and θ_2. Let us see if it is possible to make a new function y' by combining y with θ_2 and using θ_1 as the blind parameter. The verification equation now looks like this: $\tilde{g}\tilde{z}^{\theta_2 a}\tilde{a}^{\theta_2 C} = \tilde{g}^{\theta_2(y-1)+1}$, where the tilde represents A^{θ_1} for the base A. However, due to the discrete log problem, it is impossible to solve the equation $\theta_2(y-1)+1 = y'$ for $y' = h^{x'} \bmod n$ to find x'. □

The proof of the lemma above actually implies that the blind fact θ does not impact the correct verification of Alice's signature on $y - 1$. That is, the verification equality, $\tilde{g}u_1u_2 = u_3$, is equivalent to the normal ElGamal verification, $gz^a a^C = g^y$. This property is the exact one we want.

$\tilde{z}^a\tilde{a}^C$ along with S_0 forms a complete ElGamal verification that ensures the correctness of the signature knowledge proof.

Lemma 4 *Proving $u_1(a) \Leftrightarrow u_2(a)$ ensures that u_1 and u_2 are correctly constructed from the ElGamal verification the signature knowledge proof is unique.*

Proof: It is obvious that $\tilde{z}^a\tilde{a}^C$ is an original ElGamal verification if we write it in the form $\tilde{z}^a a^{C'}$ for $C' = \theta C$. In fact, since we are ensured by S_1 and $\tilde{g}$ in the verification equation $\tilde{g}\tilde{z}^a\tilde{a}^C = \tilde{g}^y$ that a common blind factor θ has been used, $\tilde{z}^a\tilde{a}^C$ is of the exact ElGamal verification.

To make the equality $\tilde{g}\tilde{z}^a\tilde{a}^C = \tilde{g}^y$ and discrete log proofs S_0, S_1, S_2, one has to use one of two ways: (1) having correct values of Alice's signature C, a, y and x, and (2) having the value of Alice's secret key k. There is no a third method that constructs a legitimate signature for Bob. Assume that Eve is a cheater who does not have C and a issued by Alice. It is easy to see that it is impossible for her to construct a correct equality. In fact, it is easy to see that Eve cannot find suitable A and B, which lead to correctly constructing S_0, in the forged verification equation, $\tilde{z}^B A^{C'} = \tilde{g}^D$, where C' is not a function of k. Although Eve can easily find $A = (\tilde{g}^D/\tilde{z}^B)^{C'^{-1}}$, to product the proof knowledge S_0, Eve has to find out X from $BA = A^X$. This is a discrete log problem, since A is a function of B. □

We would like to stress the following properties that have been mentioned in Section 2. Only Bob can sign the message, no one else including Alice can do so on his behalf. This is because only Bob has the value of his private key x. The signature knowledge proofs are not linked to any other proofs, since blind factor θ, which is used only once for a particular signature, is introduced to make the proof data different from other signature knowledge proofs. The group certificates of members are based on the ElGamal signature scheme. Since every member has a different signature commitment, a, using for the associated certificate, according to the proven security of ElGamal signature scheme, it is impossible for any group members to collude in order to find the group manager's administration key, k.

4.3 Revocation of a Signature

Having shown that the proposed group signatures are untraceable and unlinkable, we now show that as the group manager Alice can find out who actually signed a group signature. This is because Alice knows group keys of her members. To find the signer (say, Bob), she checks the equality, $\tilde{g}^y \stackrel{?}{=} u_3$, by trying a member's group key y picked from her database till finding a match. However, this method is not efficient in a large group.

Luckily, our scheme can be easily modified to suit a large group without making any substantial change in the signing procedure. To sign a message, Bob computes $\lambda = y(\theta - 1)$ and encrypts g^y using λ to obtain the ciphertext $\delta_1 \leftarrow g^y g^\lambda$ along with $\delta_2 \leftarrow z^\lambda$. Note that $g^y g^\lambda = g^{\theta y} \equiv u_3$.

Bob must prove that g^y is indeed encrypted correctly by using the discrete log knowledge proof of representations:

$$\mathsf{SKREP}[(y, \lambda) : z^\lambda \wedge g^y g^\lambda]$$

Note that since θ is used once only, and λ as an encryption parameter is used once as well. This makes the signature unlinkable and untraceable.

Alice can decrypt δ_1 to obtain g^y by removing g^λ, i.e., $g^y = \delta_1/\delta_2^{k^{-1}}$. This therefore reveals the identity of the signer.

4.4 Distributed Encryption

The public key doublet, (z_1, z_2), can be used for encryption and an encrypted message can be decrypted by any legitimate member in the group using its decryption key. Bob's decryption key d is constructed using his signing key, $d = g^y z_1^a$ or $d = g^{\eta_1} z_2^{\eta_2}/a$.

The encryption steps are as follows:

- Select a number, $\tau \in_R \mathbb{Z}_n$.
- Compute the ciphertext, $\omega_1 = m z_1^\tau$, along with $\omega_2 = z_2{}^\tau$ and $\mu = g^\tau$.

Bob decrypts the ciphertext by:

- Computing $\omega_1^a \mu^y = m^a g^{\tau(ak_1+y)} = m^a d^\tau$ and $\omega_2^{\eta_2} \mu^{\eta_1}/a = g^{\tau(\eta_2 k_2 + \eta_1)} = d^\tau$.
- Removing d^τ and a from $m^a d^\tau$ to obtain m.

Any one can encrypt a message using the group public key, but only a legitimate member in the designated group can decrypt and obtain the message. The Completeness of the protocol is obvious. If the message is correctly encrypted, it can be decrypted by any member in the group. The soundness of the protocol lies in the security of decryption keys. In the Setup section, we have proved that the setup of decryption keys is secure and collusion-resistant.

4.5 Conclusion

We have proposed a novel group cryptography that includes both signing and distributed encryption. >From the best of our knowledge, our system is the first such system ever proposed. The novel group cryptography has some potential applications to electronic commerce. For example, in a bank a group of bank staff should be able to sign or issue the bank files such as billing electronic letters on the bank's behalf, while at the meanwhile they should be also able to receive and handle electronic payments encrypted using the unique bank or group public key.

References

[1] D. Boneh and M. Franklin (1999) An efficient public key traitor tracing scheme, *Adances in cryptology - CRYPTO '99, Lecture Notes in Computer Secience 1666*, Springer Verlag, Berlin, pp. 338–353.

[2] J. Camenisch (1997) Efficient and generalized group signatures, *Adances in cryptology - EUROCRYPT'97, Lecture Notes in Computer Secience 1233*, Springer-Verlag, Berlin, pp. 465–479.

[3] J. Camenisch and M. Michels (1998) A group signature scheme with improved efficiency, *Advances in Cryptology–ASIACRYPT'98*, LNCS 1514, Springer-Verlag, Berlin, pp. 160–174.

[4] J. Camenisch and M. Stadler (1997) Efficient group signature schemes for large groups, *Advances in Cryptology, Proc. CRYPTO 97*, LNCS 1296, Springer-Verlag, Berlin, pp. 410–424.

[5] D. Chaum and E. van Heijst (1991) Group signatures. *Advances in Cryptology, Proc. EUROCRYPT 91,* LNCS 547, Springer-Verlag, Berlin, pp. 257–265.

[6] L. Chen and T. P. Pedersen (1994) New group signature schemes. *Adances in cryptology - EUROCRYPT'94, Lecture Notes in Computer Secience 950,* Springer-Verlag, Berlin, pp. 171–181.

[7] Y. Desmedt (1987) Society and group oriented cryptography: A new concept, *Advances in Cryptology, Proc. CRYPTO 87,* LNCS 293, Springer-Verlag, Berlin, pp. 120–127.

[8] T. ElGamal (1985) A public-key cryptosystem and a signature scheme based on discrete logarithms, *Advances in Cryptology, Proc. CRYPTO 84,* LNCS 196, Springer-Verlag, Berlin, pp. 10–18.

[9] Y. Mu, V. Varadharajan, and K. Q. Nguyen (1999) Delegated decryption, *Procedings of Cryptography and Coding, Lecture Notes in Computer Science,* Springer Verlag, Berlin, pp. 258–269.

[10] D. Pointcheval and J. Stern (2000) Security arguments for digital signatures and blind signatures. *Cryptology,* 13(3) pp. 361–396,

[11] R. Rivest, A. Shamir, and L. Adelman (1978) A method for obtaining digital signatures and public-key cryptography, *Communications of the ACM,* 21(2) pp. 120–126.

[12] C. P. Schnorr (1990) Efficient identification and signatures for smart cards, *Adances in cryptology - CRYPTO'89, Lecture Notes in Computer Secience 435,* Springer-Verlag, Berlin, pages 239–251.

[13] M. Stadler (1996) Publicly verifiable secret sharing, *Advances in Cryptology, Proc. EUROCRYPT 96,* LNCS 1070, Springer-Verlag, Berlin, pp. 190–199.

Cryptanalysis of Some Hash Functions Based on Block Ciphers and Codes

Hongjun Wu, Feng Bao and Robert H. Deng
Institute for Infocomm Research
21 Heng Mui Keng Terrace, Singapore 119613
{hongjun, baofeng, deng}@i2r.a-star.edu.sg

Keywords: cryptanalysis, hash function, block cipher, codes

Received: June 2, 2002

At PKC 2000, Inoue and Sakurai proposed some methods to design hash functions from block ciphers and codes (block codes and convolutional codes). They claimed that their hash functions are secure: $2^{(d-1)m/2}$ encryptions are necessary to find a collision, where d and m are the minimal distance of the code and the block size of block cipher, respectively. However, we show in this paper that a collision could be found with about $\alpha \cdot 2^m$ encryptions, where α is a small number.

1 Introduction

A hash function is a one-way function that maps an arbitrary-length message into a fixed-length quantity. It plays important roles in integrity protection, authentication and signature generation. There are two basic requirements on a secure hash function, namely, the collision resistance property (that it is hard to find two messages with the same hash result) and the preimage resistance property (that it is hard to find a message corresponding to a given hash result).

There are two approaches to design hash functions. One approach is to design customised hash functions, such as the MD4 [10], MD5 [11] and SHA-1 [3]. Another approach is to design hash functions based on some existing secure crypto algorithms (most likely, some block ciphers). A number of hash functions based on block ciphers have been proposed. The construction of single-block-length hash functions (the size of the hash result is the same as the block size of the block cipher) has been studied extensively and a synthetic study on this issue was done by Preneel, Govaerts and Vandewalle [9]. The drawback of this design is that for block cipher with small block size m, the hash function is not secure since a collision could be found with only about $2^{m/2}$ operations due to the birthday attack. The multiple-block-length hash functions are thus needed. But it is not easy to design a secure and efficient multiple-block-length hash function. Some proposed multiple-block-length hash functions are proved to be insecure: the MDC-4 [1] were broken by Knudsen and Preneel [8]; the Parallel-DM [4] was broken by Knudsen and Lai [6]. Some multiple-block-length hash functions remain secure but with relatively poor performance [7, 8]. Inoue and Sakurai recently proposed some multiple-block-length hash functions and claimed that their hash functions are secure and efficient [5]. However, as will be shown in this paper, their hash functions are not as secure as they claimed.

We believe that with the emergence of large block size block ciphers, such as the Rijndael with flexible block size up to 256 bits [2], the design of secure and efficient block cipher based hash functions could eventually be simplified.

This paper is organised as follows. Section 2 gives an introduction to Inoue and Sakurai's hash functions. The cryptanalysis of Inoue and Sakurai's block code based hash functions and convolutional code based hash functions are given in Section 3 and Section 4, respectively. Section 5 concludes this paper.

2 Inoue and Sakurai's Hash Functions

Most of the hash functions are in iterated form, i.e., they are based on the iterative computations of an easily computable compression function $h(\cdot,\cdot)$. The function $h(\cdot,\cdot)$ compresses two binary sequences of respective lengths m and an l into a binary sequence of length m. The message M is denoted as t l-bit blocks, $M = (M_1, M_2, \cdots, M_t)$ (In case the length of M is not multiple of l, padding is used). The hash result $H = H_t$ is obtained by computing iteratively

$$H_i = h(H_{i-1}, M_i) \qquad i = 1, 2, \cdots, t,$$

where H_0 is a pre-specified initial value.

The compression function of a block cipher based hash function is constructed from a block cipher. In the rest of this paper, we consider only the block cipher with equal key length and block size since Inoue and Sakurai's constructions are based on this kind of block cipher. To construct a compression function $h(\cdot,\cdot)$ with m-bit output from a block cipher with m-bit block size, the following Davies-Mayer function could be used:

$$h(M_i, H_{i-1}) = E_{M_i}(H_{i-1}) \oplus H_{i-1}$$

where both M_i and H_i are m-bit binary sequences. To construct a hash function with $n \cdot m$-bit ($n > 1$) hash result from a block cipher with m-bit block size, the following multiple Davies-Meyer function could be used.

Definition 1 (Multiple Davies-Meyer Function). Let $E_K(\cdot)$ be a block cipher with m-bit block size and key length. Let $h_1, h_2, \cdots, h_n$ be different instantiations of the Davies-Meyer function, i.e., $h_i(X_i, Y_i) = E_{X_i}(Y_i) \oplus Y_i$, obtained by fixing $\lceil \log_2 n \rceil$ key bits to different values. The M_i and H_i are expanded by an affine mapping to the n pairs (X_i, Y_i). The output is the concatenation of the outputs of the function $h_1, h_2, \cdots, h_n$.

To construct a compression function h based on the multiple Davies-Meyer function, the main task is to design the affine mapping. In Knudsen and Preneel's work [7, 8], non-binary codes are used. Recently Inoue and Sakurai introduced the block codes as well as the convolutional codes in designing the affine mapping [5]. In Inoue and Sakurai's affine mapping, the input are k m-bit message blocks (M_i) and n m-bit blocks of the previous iteration output H_{i-1} but only the message blocks M_i are encoded. Their design are given in the following two subsections.

2.1 Construction with Block Codes

The input to the affine mapping are the k m-bit message blocks (M_i) and n m-bit blocks of previous hashed value H_{i-1}. The k-block M_i is encoded into n m-bit blocks, denoted as C_i, with the use of an (n, k, d) error correction code. From H_{i-1} and C_i, form n input pairs to those n different Davies-Meyer functions.

A typical example in [5] is given below:

Example 1. Consider the (7,4,3) Hamming code with the following generator matrix:

$$G = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 \end{bmatrix}$$

Let $M_i = (M_{i,1}, M_{i,2}, M_{i,3}, M_{i,4})$, where $M_{i,j}$ is an m-bit block. M_i is encoded as $C_i = (C_{i,1}, C_{i,2}, \cdots, C_{i,7})$, where $C_{i,j} = M_{i,j}$ for $j = 1, 2, 3, 4$, and

$$C_{i,5} = M_{i,1} \oplus M_{i,2} \oplus M_{i,3}$$

$$C_{i,6} = M_{i,2} \oplus M_{i,3} \oplus M_{i,4}$$

$$C_{i,7} = M_{i,1} \oplus M_{i,2} \oplus M_{i,4}$$

Let $H_{i-1} = (H_{i-1,1}, H_{i-1,2}, \cdots, H_{i-1,7})$ and $H_i = (H_{i,1}, H_{i,2}, \cdots, H_{i,7})$. H_i is obtained as follows:

$$H_{i,j} = h_j(H_{i-1,j}, C_{i,j}) \qquad j = 1, 2, \cdots, 7$$

The other block codes with higher error correction capability could also be used. It was claimed in [5] that for a hash function constructed from an (n, k, d) code in this way, about $2^{(d-1)m/2}$ encryptions are required to find a collision.

2.2 Construction with Convolutional Codes

Let the convolutional code be with parameters $(n, k, d; N)$, where n, k, d and N denote the output size, input size, minimum distance and constraint length, respectively. The length of a message is padded to multiple of km bits. Then $(N-1)km$ zero bits are padded to the message. The message after padding is denoted as $(M_1, M_2, \cdots, M_t)$, where each M_i is a km-bit block. The input to the compression function $h(\cdot, \cdot)$ are $(M_{i-N+1}, \cdots, M_i)$ and the n-block H_{i-1}. The $(M_{i-N+1}, \cdots, M_i)$ is encoded into an n-block code word, $C_i = (C_{i,1}, C_{i,2}, \cdots C_{i,n})$ with the use of an $(n, k, d; N)$ convolutional code. From H_{i-1} and C_i, form n input pairs to the n different Davies-Meyer functions. The final hash result is the concatenation of the last N iterations' outputs $(H_{t-N+1}, \cdots, H_{t-1}, H_t)$.

We give a simple example below to illustrate the operation of this kind of hash function.

Example 2. Consider a simple (2,1,3;3) binary convolutional code. For each output, the following generator matrix is used:

$$G = \begin{bmatrix} 1 & 1 \\ 1 & 1 \\ 0 & 1 \\ 1 & 1 \end{bmatrix}$$

$(M_{i-3}, M_{i-2}, M_{i-1}, M_i)$ is encoded into $C_i = (C_{i,1}, C_{i,2})$ where

$$C_{i,1} = M_{i-3} \oplus M_{i-2} \oplus M_i$$
$$C_{i,2} = M_{i-3} \oplus M_{i-2} \oplus M_{i-1} \oplus M_i$$

H_i is obtained as follows:

$$H_{i,j} = h_j(H_{i-1,j}, C_{i,j}) \qquad j = 1, 2$$

The concatenation of the outputs of the last three iterations are the final hash result.

It was claimed in [5] that for a hash function constructed from an $(n, k, d; N)$ convolutional code in this way, about $2^{(d-1)m/2}$ encryptions are needed to find a collision.

3 Attack on the Hash Functions Based on Block Codes

Inoue and Sakurai claimed that about $2^{(d-1)m/2}$ encryptions are needed to find a collision of their hash functions based on block codes [5]. In this section, we give an attack to find a collision with only about $d \cdot 2^m$ encryptions. The flaw in Inoue and Sakusai's design is that the differences introduced into the n-block hash output do not affect one

another in the following iterations, i.e., there is no difference propagation. It allows us to deal with those differences separately.

Our attack is based on the following assumption:

Assumption 1. Consider a Davies-Mayer function $h(X,Y) = E_X(Y) \oplus Y$, where $E_k(\cdot)$ is a block cipher with m-bit block size and m-bit key length. For two randomly chosen Y and Y' with $Y \neq Y'$, an X satisfying $h(X,Y) = h(X,Y')$ exits with probability 0.368.

The following attack is to find two different messages M and M' satisfying $Hash(M) = Hash(M')$, where the hash function is based on an (n,k,d) block code.

1. Find a k-block $\triangle M_1$ so that $(n-d)$ blocks of its n-block code word are with value zero. The positions of those non-zero blocks are denoted as $E = (e_1, e_2, \cdots, e_d)$, where $1 \leq e_i \leq n$.
2. Randomly choose M_1 and let $M_1' = M_1 \oplus \triangle M_1$. Then $H_1' \neq H_1$ and they differ at d blocks indicated by the set E.
3. In this step, we try to find $M_2 = M_2'$ that eliminates the difference at the e_1th block of the hash output, i.e., we aim at achieving $H_{2,e_1} = H'_{2,e_1}$
   (a) Find an m-bit c such that $h_{e_1}(H_{1,e_1}, c) = h_{e_1}(H'_{1,e_1}, c)$, i.e., $H_{2,e_1} = H'_{2,e_1}$. In case such a collision could not be found, go to step 2, select a different M_1 and repeat the attack.
   (b) Choose M_2 such the e_1th block of its code word is c, i.e., $C_{2,e_1} = c$.
   (c) Let $M_2' = M_2$. Clearly, $C'_{2,e_1} = C_{2,e_1} = c$. Thus $H_{2,e_1} = H'_{2,e_1}$, i.e., M_2 and M_2' are the desired message blocks.
4. Similar to step 2, find $M_3' = M_3$ so that the difference at the e_2th block of the output is eliminated. Repeat the similar attacks until all those d differences in the hash output are eliminated.
5. The M and M' (with only $M_1 \neq M_1'$) constructed above are with the same hash result.

We note that according to Assumption 1, the Step 3 in the attack could succeed with about 2^m encryptions. Thus the total amount of computations required in this attack is about $d \cdot 2^m$.

4 Attack on the Hash Functions Based on Convolutional Codes

Inoue and Sakurai claimed that about $2^{(d-1)m/2}$ encryptions are needed to find a collision of their hash functions based on convolutional codes [5]. In this section, we give an attack to find such a collision with only about $n \cdot 2^m$ encryptions. Similar to the attack in Section 3, our attack in this section is to eliminate the differences in the hash output separately.

The following attack is to find two different messages M and M' satisfying $Hash(M) = Hash(M')$, where the hash function is based on an $(n,k,d;N)$ convolutional code.

1. Randomly select two k-block M_1 and M_1' with $M_1 \neq M_1'$.
2. Randomly set the values of M_i $(2 \leq i \leq N)$. Let $M_i' = M_i$ for $i = 2, 3, \cdots, N+1$, i.e., at the beginning of the $(N+2)$th iteration, the message blocks in the memory are the same for M and M'.
3. At the $(N+2)$th iteration, the only difference we need to consider is that $H_{N+1} \neq H'_{N+1}$. In this step, we try to eliminate the difference between the first blocks of H_{N+2} and H'_{N+2}.
   (a) Find an m-bit c such that $h_1(H_{N+1,1}, c) = h_1(H'_{N+1,1}, c)$. In case such collision could not be found, go to step 2, select a different M_{N+1} and repeat the attack.
   (b) Choose M_{N+2} such that the first block of the $(N+2)$th code word is c, i.e., $C_{N+2,1} = c$.
   (c) Let $M'_{N+2} = M_{N+2}$. Then $C'_{N+2,1} = C'_{N+2,1} = c$ since $M_i = M_i'$ for $i = 2, 3, \cdots, N+2$. Thus $H_{N+2,1} = H'_{N+2,1}$. The M_{N+2} and M'_{N+2} are the desired message blocks.
4. Carry out the similar attack to eliminate the remaining differences at those $(n-1)$ blocks of the hash output in the next $(n-1)$ iterations.
5. At the end of the $(N+n+1)$th iteration, $H_{N+n+1} = H'_{N+n+1}$. And there is no difference in the memory. Let $M'_{N+n+1+i} = M_{N+n+1+i}$ for $i = 1, 2, \cdots, N-1$.
6. From the $(2N+n)$th iteration on, any final hash outputs (the concatenation of the last N hash outputs) would be the same.

We note that according to Assumption 1, the Step 3 in the attack could succeed with about 2^m encryptions. Thus the total amount of computations required in this attack is about $n \cdot 2^m$.

In addition, we point out here that the claim in [5] that "this (the convolutional codes based hash function) reduces the circuit complexity of the hardware-implementation $1/N$ times in terms of the number of Davies-Mayers's module function than that based on block error correcting codes" is misleading. The hash function, no matter it is based on block codes or convolutional codes, could be implemented with only one Davies-Mayer's module function

in the hardware implementation. The reason is that those Davies-Mayer functions are all based on the same block cipher.

5 Conclusions

In this paper, we show that only about $\alpha \cdot 2^m$ encryptions are needed to find a collision of Inoue and Sakurai's hash functions based on the block codes and convolutional codes. The claim that a collision could only be found with about $2^{(d-1)m/2}$ encryptions is incorrect.

References

[1] B.O. Brachtl, D. Coppersmith, M.M. Hyden, S.M. Matyas, C.H. Meyer, J. Oseas, S. Pilpel, M. Schilling, "Data Authentication Using Modification Detection Codes Based on a Public One Way Encryption Function", U.S. Patent Number 4,908,861, March 13, 1990.

[2] J. Daeman, and V. Rijmen, "AES Proposal: Rijndael", available at *http://www. nist.gov/aes* .

[3] FIPS PUB 180-1, "Secure Hash Standard", Federal Information Processing Standard (FIPS), Publication 180-1, National Institute of Standards and Technology, US Department of Commerce, Washington D.C., April 17, 1995.

[4] W. Hohl, X. Lai, T. Meier, and C. Waldvogel, "Security of Iterated Hash Functions Based on Block Ciphers", in *Advances in Cryptology - Crypto93*, LNCS 773, Springer-Verlag, pp. 379-390, 1994.

[5] T. Inoue, and K. Sakurai, "Making Hash Functions from Block Ciphers Secure and Efficient by Using Convolutional Codes", in *Public Key Cryptography - PKC 2000*, LNCS 1751, Springer-Verlag, pp. 391-404, 2000.

[6] L.R. Knudsen, and X. Lai, "New Attacks on All Double Block Length Hash Functions of Hash Rate 1, including the Parallel-DM", in *Advances in Cryptology - Eurocrypt'94*, LNCS 959, Springer-Verlag, pp. 410-418, 1995.

[7] L.R. Knudsen, and B. Preneel, "Hash Functions Based on Block Ciphers and Quaternary Codes", in *Advances in Cryptology - Asiacrypto'96*, LNCS 1163, Springer-Verlag, pp. 77-90, 1996.

[8] L.R. Knudsen, and B. Preneel, "Fast and Secure Hasing Based on Codes", in *Advances in Cryptology - Crypto'97*, LNCS 1294, Springer-Verlag, pp. 485-498, 1997.

[9] B. Preneel, R. Govaerts, J. Vandewalle, "Hash Functions Based on Block Ciphers: A Synthetic Approach", in *Advances in Cryptology - Crypto'93*, LNCS 773, Springer-Verlag, pp. 368-378, 1994.

[10] R.L. Rivest, "The MD4 Message Digest Algorithm", in *Advances in Cryptology - Crypto'90*, LNCS 537, Springer-Verlag, pp. 303-331, 1991.

[11] R.L. Rivest, "The MD5 Message Digest Algorithm", *Request for Comments (RFC) 1321*, Internet Activities Board, Internet Privacy Task Force, April 1992.

Analysis of AES S-box with Walsh Spectrum

Baodian Wei, Dongsu Liu and Xinmei Wang, Member, IEEE
National Key Lab. of Integrated Service Networks, Xidian Univ., Xi'an 710071, China
Phone: +86 29 8201012
weibaodian@hotmail.com

Keywords: AES, S-box, Boolean function, Walsh spectrum

Received: May 20, 2002

We investigate the security of the AES S-box in the view of Boolean function with the technique of Walsh spectrum. The cryptographic characteristics of AES such as linearity, nonlinearity, strict avalanche, propagation and correlation immunity are revealed.

1 Introduction

Substitution box(S-box)[1,2] is a core component of most of the modern block ciphers and often servers as the only nonlinear component of the algorithms. In other words, the nonlinearity and cryptographic security of the cryptosystems depends heavily on the strength of the S-box. It is the same case for the algorithm Rijndael[3] which is the Advanced Encryption Standard(AES) of USA. When Joan Danmen and Vincent Rijmen proposed this algorithm, they focused upon the resistance to the differential cryptanalysis[4] and the linear cryptanalysis[5]. We once verified this by working out the $2^8 \times 2^8$ differential matrices and the linear matrices of AES and found that AES does show significant performance as the designers acclaimed! But what things are if we view the S-box of AES as a set of Boolean functions? We discuss this issue in this work with Walsh spectrum[6,7] as a mathematical tool. The cryptographic properties of AES such as linearity, nonlinearity, strict avalanche criteria (SAC),propagation criteria(PC) and correlation immunity (CI) are analysed and some new conclusions are drawn for the first time.

The rest of the paper is organized as follows. Section 2 is dedicated to the introduction of the structure of AES S-box and the eight truth tables of Boolean functions are given. Section 3 summarizes the Walsh spectrum theory and its applications in the analysis of cryptosystems. Theory of Walsh spectrum is employed in Section 4 to analyze the AES Boolean functions and some new remarks are made. We conclude the paper in Section 5.

2 AES S-box

The 8×8 S-box of AES performs a nonlinear transformation called byte substitution[3] which operates independently on each of the State bytes. The substitution is invertible and is constructed by the composition of two transformations as described as below.

(1)View the State bytes as elements of field $GF(2^8)$ and take the multiplicative inverse under the irreducible polynomial $m(x) = x^8 + x^4 + x^3 + x + 1$ with the element zero mapped onto itself.
(2)An affine transformation is applied on the result of (1). The fixed matrix and the fixed vector over $GF(2)$ in the affine transformation are shown in Figure 1(with y_0 the most significant bit).

$$\begin{bmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \\ y_6 \\ y_7 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \end{bmatrix}$$

Fig.1 Affine transformation of AES ByteSub

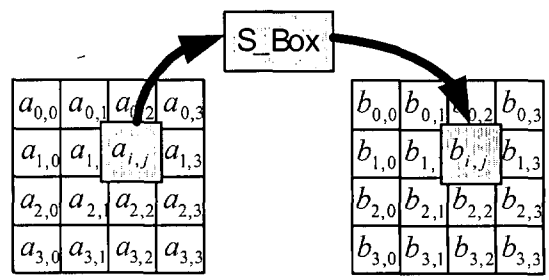


Fig.2 AES ByteSub

Fig.2 illustrates the overall effect of the byte substitution on the State. In practical realization, the whole transformation is expressed in the form of a table and is worked out in advance for the purpose of less reduplicate calculation and faster speed. The ByteSub table is in fact a permutation over $GF(2^8)$. Since the S-box is viewed in

this paper as a set of Boolean functions, we derive the eight truth tables of the AES S-box as shown in Table 1 directly from the ByteSub table. The truth tables help to compute the corresponding Walsh spectrum.

The first digit of each line in the Table 1 is the function value of the input '00000000' and the last one of '1111 1111'.

3 Walsh Spectrum

Walsh transformation is an important tool to analyse the properties of Boolean functions such as balancedness, linear structures, the best linear approximation, nonlinearity, Bentness, SAC, PC and CI. Here we summarize the concept of Walsh spectrum and its application in the analysis of cryptographic properties mentioned above.

Definition 1[6,7] The Walsh transformation of n-tuple Boolean function $f(x)$ is defined as

$$S_f(w) = 2^{-n} \sum_{x=0}^{2^n-1} (-1)^{w \bullet x} f(x),$$

where $w \in GF(2^n)$, $w \bullet x = w_0 x_0 + \cdots + w_{n-1} x_{n-1}$. $S_f(w)$ is also called the type- Ⅰ Walsh spectrum of $f(x)$.

In some literatures, $S_f(w)$ takes a lightly different form with the coefficient 2^{-n} absent, so does the next definition $S_{(f)}(w)$. However the different forms do not bring any contradiction or inconvenience to the applications of Walsh transformation.

Definition 2[6,7]

$$S_{(f)}(w) = 2^{-n} \sum_{x=0}^{2^n-1} (-1)^{w \bullet x} (-1)^{f(x)} = 2^{-n} \sum_{x=0}^{2^n-1} (-1)^{w \bullet x + f(x)}$$

is defined as the type- Ⅱ Walsh spectrum of n-tuple function $f(x)$.

By the definition, $S_{(f)}(w) = \frac{1}{2^n}(|\{x \in GF(2^n) \mid f(x) = w \bullet x\}| - |\{x \in GF(2^n) \mid f(x) \neq w \bullet x\}|) = \frac{1}{2^n}(2|\{x \in GF(2^n) \mid f(x) = w \bullet x\}| - 2^n) = \frac{1}{2^n}(2^n - |\{x \in GF(2^n) \mid f(x) \neq w \bullet x\}|)$. That is to say that type- Ⅱ Walsh spectrum shows how much $f(x)$ is close to the linear function $w \bullet x$. The idea is described in the following theorem.

Theorem 1[7] If $w, x \in GF(2^n)$ and $f(x)$ is a n-tuple Boolean function, then

$$P(f(x) = w \bullet x) = \frac{1 + S_{(f)}(w)}{2},$$

$$P(f(x) = w \bullet x + 1) = P(f(x) \neq w \bullet x) = \frac{1 - S_{(f)}(w)}{2},$$

where $P(\bullet)$ is the probability.

By Theorem 1, it is easy to find the best linear approximation of $f(x)$. Let $|S_{(f)}(w')| = \underset{w}{Max} |S_{(f)}(w)|$ and w'' is an arbitrary value different from w'. If $S_{(f)}(w') > 0$, then $P(f(x) = w' \bullet x) = \frac{1 + S_{(f)}(w')}{2} > \frac{1 + S_{(f)}(w'')}{2} = P(f(x) = w'' \bullet x) > \frac{1 - S_{(f)}(w'')}{2} = P(f(x) = w'' \bullet x + 1)$. It indicates that $w' \bullet x$ is the best linear approximation of $f(x)$. If $S_{(f)}(w') < 0$, then $P(f(x) = w' \bullet x + 1) = \frac{1 - S_{(f)}(w')}{2} > \frac{1 - S_{(f)}(w'')}{2} = P(f(x) = w'' \bullet x + 1) > \frac{1 + S_{(f)}(w'')}{2} = P(f(x) = w'' \bullet x)$. It implies that $w' \bullet x + 1$ is the best linear approximation of $f(x)$.

Two types of Walsh spectrum both have some relation with the properties of Boolean functions as summarized in Theorem 2.

Theorem 2[6,7] For the n-tuple Boolean function $f(x)$,

(1) $f(x)$ is balanced if and only if $S_f(0) = \frac{1}{2}$ or $S_{(f)}(0) = 0$.

(2) Identify two kinds of linear structure set $U_f^{(0)}$ and $U_f^{(1)}$ with $\{\alpha \mid f(x) + f(x+\alpha) = 0\}$ and $\{\alpha \mid f(x) + f(x+\alpha) = 1\}$ respectively. We have that $\alpha \in U_f^{(0)}$ if and only if for any $w \in GF(2^n)$ and $w \bullet \alpha = 1$, $S_{(f)}(w) = 0$. We also have that $\alpha \in U_f^{(1)}$ if and only if for any $w \in GF(2^n)$ and $w \bullet \alpha = 0$, $S_{(f)}(w) = 0$.

(3) Nonlinearity of $f(x)$, $N_f = 2^{n-1}(1 - \underset{w \in GF^n(2)}{Max} |S_{(f)}(w)|)$ or $N_f = 2^{n-1}(1 - Max\{2 \underset{w \in GF^n(2)/\{0\}}{Max} |S_{(f)}(w)|, |1 - 2 \times S_f(0)|\})$.

(4) $f(x)$ is a Bent function if and only if for any $w \in GF(2^n)$, $S_{(f)}(w) = \pm 2^{-\frac{n}{2}}$ or $S_f(w) = \begin{cases} \pm 2^{-\frac{n}{2}-1}, w \neq 0 \\ \pm 2^{-\frac{n}{2}} - 1/2, w = 0 \end{cases}$.

(5) $f(x)$ satisfies SAC if and only if for any $\alpha \in GF(2^n)$ and $W_H(\alpha) = 1$, $\sum_{w \in F_2^n} S_{(f)}^2(w)(-1)^{w \bullet \alpha} = 0$.

(6) $f(x)$ satisfies PC for $\alpha \in GF(2^n) \setminus \{0\}$ if and only if $\sum_{\omega \in GF^n(2)} S_{(f)}^2(w)(-1)^{\alpha \bullet \omega} = 0$ or $\sum_{\omega \in GF^n(2)} S_f^2(w)(-1)^{\alpha \bullet \omega} = S_f(0) - \frac{1}{4}$.

(7) $f(x)$ satisfies PC of degree k if and only if for any $\alpha \in GF^n(2)$ and $1 \leq W(\alpha) \leq k$, $\sum_{\omega \in GF^n(2)} S_{(f)}^2(w)(-1)^{\alpha \bullet \omega} = 0$ or $\sum_{\omega \in GF^n(2)} S_f^2(w)(-1)^{\alpha \bullet \omega} = S_f(0) - \frac{1}{4}$.

(8) $f(x)$ satisfies PC of degree k and order m if and only if for any $1 \le i_1 < \cdots < i_k \le n$, $\alpha \in GF(2^k), 1 \le W(\alpha) \le k$,

$$\sum_{w_2 \in GF^{n-k}(2)} \Big(\sum_{w_1 \in GF^k(2)} S_{(f)}^{(i_1,\cdots,i_k)}(w_1, w_2)(-1)^{a \bullet w_1} \Big)^2 (-1)^{\alpha \bullet \alpha w} = 0$$

or

$$\sum_{w_2 \in GF^{n-k}(2)} \Big(\sum_{w_1 \in GF^k(2)} S_f^{(i_1,\cdots,i_k)}(w_1, w_2)(-1)^{a \bullet w_1} \Big)^2 (-1)^{\alpha \bullet w_2} =$$

$$\sum_{w_1 \in GF^k(2)} S_f^{(i_1,\cdots,i_k)}(w_1, 0)(-1)^{a \bullet w_1} - \frac{1}{4}$$,where $S_{(f)}^{(i_1,\cdots,i_k)}(w_1, w_2) = S_{(f)}(w)$, and the k bits in the positions $i_1, \cdots, i_k$ of w are the bits of w_1 and the other bits are the bits of w_2.

(9) $f(x)$ satisfies CI of order m if and only if for any $w \in GF^n(2)$ and $1 \le W(w) \le m$, $S_{(f)}(w) = 0$ or $S_f(w) = 0$.

4 Application in the Analysis of AES

As discussed in the previous section, Walsh spectrum is quite powerful in the analysis of Boolean functions of block ciphers. We have found here its application in the analysis of AES. For convenience, we compute the type-II Walsh spectrum (enlarged 256 times) of all eight Boolean functions of the AES S-box and list them in Table 2 where the first digit of each line is $256 * S_{f_i}(0)$ and the last one $256 * S_{f_i}(255)$. Fig.3. visually presents the spectrum of f_4. For simplicity, no type-I Walsh spectrum is listed any more. In fact, it is very easy to derive the type-I Walsh spectrum directly from the type-II one as illustrated in Theorem 3.

Theorem 3[7] $S_f(w) = \begin{cases} -\frac{1}{2} S_{(f)}(w), w \ne 0 \\ \frac{1 - S_{(f)}(w)}{2}, w = 0 \end{cases}$.

Table 2 256 times of type-II Walsh spectrum of AES Boolean functions

Functions	type-II Walsh spectrum
f0	0,24,4,12,-16,16,12,-20,4,-20,-8,0,-12,-28,16,16,-24,0,-20,4,24,8,4,4,28,28,-16,-24,12,-20,-8,-8,12,4,16,24,-4,-4,-8,-8,16,-24,4,-4,0,-32,-4,12,-12,-4,-8,0,4,4,16,0,-8,0,12,-28,-24,-24,-12,20,8,-16,12,4,24,-24,4,20,-20,-12,16,24,-20,-4,8,-24,-24,16,12,20,-8,-8,-12,4,12,-12,-16,-24,12,-20,-8,-8,-12,12,24,0,-12,-12,0,-32,-24,16,-20,-12,24,8,20,20,-28,12,8,-16,4,-28,0,-16,-24,0,12,-12,24,8,4,-12,-24,16,12,4,-8,24,-28,-28,-28,12,8,16,4,4,-32,-16,-16,-24,4,28,-16,16,-4,12,-4,20,-16,-8,12,-20,8,8,4,-4,-8,-32,-12,4,16,0,-16,-8,4,-20,16,16,-4,-20,12,4,-16,8,12,-20,8,-8,24,0,-4,-12,-24,24,-12,4,-16,-24,20,28,-32,-16,28,-20,28,4,16,24,12,12,8,-8,0,8,20,-4,0,-16,-4,-4,-20,4,-16,-8,-20,-20,-24,-24,12,4,0,8,12,-4,24,-24,-8,16,28,-12,24,-24,4,4,12,4,16,8,-4,28,-24,24,8,0,28,4,-8,24,4,4
f1	0,0,16,16,4,20,12,-4,4,-4,-20,4,8,0,24,-16,-4,-4,12,-4,-16,0,24,-8,-24,0,0,8,12,4,12,-12,-20,-4,-4,-20,24,-24,0,16,8,16,-16,-8,4,-4,-12,-20,-24,24,8,8,20,4,12,-20,12,-12,-12,-20,-8,-16,8,16,-8,-8,-16,16,-12,-12,20,-12,-20,-28,12,4,-8,-32,16,24,12,12,4,-12,8,-24,8,24,-24,0,-8,0,4,12,12,-28,4,-28,12,12,24,8,8,24,16,-24,-32,-8,4,28,12,4,-8,-8,16,0,-4,-20,-4,28,12,4,28,4,0,-8,-24,-16,-24,-16,0,8,-20,20,12,-12,-12,4,-12,-28,-8,24,-16,16,-12,-20,12,20,24,16,24,-32,-24,-24,-8,8,-4,12,4,4,4,-20,-4,4,-16,8,-16,-24,24,-24,-8,8,20,4,-20,-4,-32,24,-24,16,-4,4,12,4,28,28,28,12,-8,24,16,0,0,-8,16,-24,28,4,4,-20,-20,12,-12,20,-8,-8,-24,-24,4,-20,-12,-20,16,-24,-8,0,24,-24,16,-16,-28,-12,4,-28,-4,4,28,4,-16,24,8,-16,-16,0,-24,-8,-28,4,-28,4,-16,8,0,-24,4,-4,12,-12,12,12,4,20,-16,-32,16,16
f2	0,28,4,24,12,-24,16,-12,8,12,28,-8,20,24,-8,-28,-12,0,16,20,24,-12,-12,24,-12,8,-16,-4,24,-4,-28,16,-4,0,24,4,-16,20,28,24,12,-8,24,12,-16,-4,-4,0,-16,-12,-12,0,-4,-16,16,-4,-24,-12,-20,0,4,0,8,-4,-24,-28,-28,-8,-12,16,-16,-12,-8,-4,20,16,4,8,-16,-4,-12,16,8,-4,8,-12,-4,16,-20,16,-16,-20,0,20,-12,-16,4,24,-24,-28,-8,12,12,-8,28,-8,32,4,0,-4,-28,24,-16,-12,-4,8,12,0,-24,20,0,12,-4,16,-20,8,-24,-4,32,20,-12,0,4,-24,-24,-12,0,12,-12,24,-12,16,8,12,-12,-24,16,-4,-16,4,-4,24,-4,8,-24,12,-24,-12,20,8,28,-24,24,12,24,-12,20,8,20,-8,-16,-4,-16,4,-4,-24,32,-4,20,24,20,-16,-24,28,16,4,20,16,-28,-8,-8,4,-16,4,12,-8,4,8,-16,28,-8,-28,4,8,-4,24,-24,-20,-4,0,0,-4,-8,-4,12,24,-4,-8,32,-12,8,4,12,-16,-4,-8,16,20,8,20,-4,0,-4,16,32,-4,8,-20,-4,24,24,20,4,8,12,-24,24,-20,0,-12,-20,8,-12,8,-16,28
f3	0,24,12,-12,12,28,24,-8,0,16,-12,-12,-12,-4,24,16,-4,4,-16,24,-8,24,-4,-4,-12,-12,-16,16,-24,0,4,-4,-24,-8,-12,-12,-4,20,8,-16,16,8,-12,-4,-20,-20,0,-16,4,-28,-8,-8,-8,32,-4,-28,-12,-4,0,-24,0,16,-20,-4,32,0,4,-12,12,-12,16,8,-24,16,4,-4,-4,12,-24,8,28,12,24,8,-8,-16,4,-4,-4,20,-16,-24,16,16,-28,4,-16,-8,4,28,-28,4,24,-24,0,0,-4,12,-4,-12,8,-16,-4,20,8,0,16,32,-20,-4,20,4,-24,24,0,8,-12,28,28,4,-24,16,8,-8,20,-28,-12,20,24,24,8,-16,-4,-28,0,24,20,28,12,12,-16,0,-16,0,-4,-4,-12,-20,0,8,-28,-28,16,-16,-8,16,4,-4,-12,-4,8,16,16,-16,20,-12,24,-24,12,12,28,4,0,24,-16,8,12,20,12,-4,8,-24,20,-12,-24,-24,0,24,-12,12,-12,-4,-16,24,8,-24,-12,20,-24,24,-12,20,20,-4,-16,-24,32,24,20,28,4,20,-8,-8,4,12,8,-16,-8,8,-4,-20,-4,-4,-8,24,-8,32,4,12,-8,16,20,-4,-4,-4,8,24,24,8,12,-20,-12,-20,8,-16
f4	0,16,-4,20,24,8,4,12,24,-24,-4,-12,0,0,4,-4,-16,8,-12,-12,-24,0,12,-4,-8,-16,20,20,-16,24,-4,28,4,-12,32,-8,-28,20,16,-8,-12,4,-24,0,20,20,-24,0,12,-12,-16,0,12,-12,16,16,12,-12,-8,-24,28,20,24,8,12,12,0,-8,28,-20,32,24,-4,12,8,16,-20,-4,24,16,-28,-20,0,0,20,-20,-16,16,4,-4,-24,24,-28,-4,-8,24,-8,8,-4,-28,32,0,20,-4,16,16,28,-12,-24,8,20,-4,8,16,20,20,-16,-24,-4,-4,-16,8,-12,4,-8,-16,12,12,-16,-24,-12,-28,16,24,-12,-28,-16,24,-4,-4,16,-8,12,-4,8,-24,4,-4,-24,24,20,-20,24,-24,12,20,-24,24,-4,-12,28,20,-16,0,-12,-4,8,24,-12,28,32,0,-4,4,24,8,-4,12,8,-16,-12,-12,-16,24,4,4,24,16,-4,28,16,-8,-4,4,-24,-8,4,12,-16,16,-12,-4,-8,8,-20,4,32,16,-20,-4,-16,8,-12,-28,8,-16,-12,4,0,24,-4,28,-24,0,0,-8,-20,12,-16,8,-4,12,0,-8,20,-12,-16,-8,-12,-12,24,8,-4,-12,8,24,-4,4,8,-8,4,-4,-24,8,-28,-4
f5	0,-8,16,8,-32,24,16,-24,12,4,-20,4,4,-4,-12,12,-20,4,4,12,-12,12,-4,4,-8,-16,-16,24,-24,0,0,-24,0,0,-8,-8,-16,-16,24,-8,-12,4,12,-4,28,12,20,-28,-12,4,4,4,28,-20,12,12,24,24,24,8,-24,-24,-8,8,-16,-24,16,-24,0,-8,16,8,28,4,-20,-12,-12,-4,4,-20,-20,-12,4,-4,-12,28,-20,4,-8,16,-16,24,24,16,0,-24,16,16,-8,-8,16,16,-8,24,4,4,-20,-20,12,12,4,4,4,4,20,-28,-20,12,12,-4,8,-24,8,24,-24,8,8,-8,-24,16,8,0,24,16,8,-16,20,-4,20,12,12,4,12,20,-28,-4,-20,4,-4,4,4,12,-16,8,8,0,0,-24,8,-16,-8,-24,16,-16,24,-8,16,0,-4,28,-28,-12,4,-12,-4,28,12,-4,-4,12,4,4,-28,4,0,-32,-16,16,-16,-32,0,16,-16,-24,-16,-8,0,8,0,-8,-20,20,-20,4,-12,12,4,12,-20,-28,-28,-4,-12,-4,-4,-28,-24,16,-16,-8,24,16,0,-8,-32,0,24,8,-32,-16,-24,-24,-12,-12,28,12,-20,28,-12,20,4,-12,4,20,12,-20,12,12,-24,-8,-24,24,-8,24,8,-24
f6	0,8,-32,-24,4,-20,-4,-12,4,4,12,-4,-8,24,-24,-24,0,-8,-16,-8,4,12,12,20,4,4,-20,12,24,8,-24,8,-16,-24,0,8,4,-20,-4,4,-12,4,28,-20,-8,24,24,-24,16,-8,16,24,4,-20,12,4,4,20,12,12,8,24,-24,-8,-24,0,24,-16,-4,20,4,12,-4,-20,4,4,-16,0,0,-16,-8,-16,24,0,28,-28,-12,-4,-4,-4,4,-28,0,16,-16,16,-16,-8,0,-8,20,-20,12,4,-28,-28,-4,-4,-24,-8,24,-8,-32,8,-32,-24,-12,28,28,-12,-12,4,-20,12,-24,24,-8,-24,-8,16,24,16,12,4,4,12,-20,12,-12,4,-16,-16,0,0,0,-8,-16,24,-12,-4,28,-28,-12,4,28,12,24,24,-24,-8,-24,16,-8,16,28,-12,-12,-20,-20,-4,-12,4,16,-16,16,0,16,-8,16,-8,4,-20,12,4,4,-28,-20,-4,-24,8,8,8,16,8,16,8,20,12,12,20,-28,4,-4,12,8,8,-24,8,-24,16,-8,16,-4,-12,4,28,12,12,4,4,-32,-16,-32,0,-24,-16,8,0,-20,4,-12,12,-20,-4,-12,-28,16,-16,16,0,0,24,-16,-24,-12,12,-20,20,4,20,12,12,-8,-24,24,8
f7	0,24,-4,12,-4,-12,-24,24,8,16,-12,-12,-4,-12,-24,-8,16,-8,-4,-4,-4,-28,-24,-24,8,-16,20,-12,-4,-12,-8,8,24,16,4,-12,20,-4,0,0,16,-24,28,12,4,28,0,-16,16,8,-20,-4,28,4,8,-24,-8,32,-28,-28,12,-12,24,24,8,-16,-12,-12,-12,12,16,0,8,16,4,-28,12,-12,-24,8,24,-16,-28,20,4,12,16,-16,8,-16,20,-12,-4,4,24,-8,-16,-8,-20,-20,-4,4,-8,24,-16,8,12,-4,-12,-4,32,0,24,0,20,-12,4,12,16,-16,-24,16,-12,20,-4,-12,8,-8,-8,32,20,4,-4,4,8,24,0,8,-4,-20,-4,20,-8,24,24,0,20,4,12,20,8,24,-16,-24,-4,-4,-20,-12,-24,24,0,-24,-20,-4,4,-4,16,16,8,0,-28,4,4,-4,-16,16,8,0,20,-12,28,4,24,8,32,24,12,12,-4,20,-24,8,-16,-24,-4,-4,20,-4,16,0,0,-24,12,28,-4,-28,-24,24,-16,8,12,12,20,-4,-16,0,-16,8,-4,-4,28,20,-8,-8,-8,-16,-12,-12,28,-12,-8,-8,24,-16,4,-28,-12,-4,16,32,16,-8,-4,12,20,28,-16,0,0,24,12,-20,-20,-12,24,-24

We can verify the data in Table 2 with the Plancheral formula $\sum_{w=0}^{2^n-1} S_f^2(w) = S_f(0) = 2^n W_H(f)$ or the Parseval formula $\sum_{w=0}^{2^n-1} S_{(f)}^2(w) = 1$.

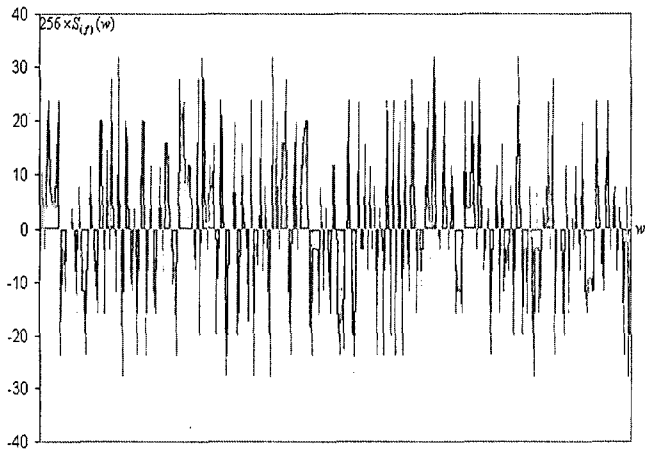


Fig. 3. 256 times of type-II Walsh spectrum of f_4

By Theorem 2 and data in Table 2, we draw some new conclusions about AES Boolean function as follows.

(1) The best linear approximations corresponding to the eight Boolean functions are $x_5+x_3+x_2+x_0+1$, $x_6+x_3+x_2+x_0+1$, $x_6+x_5+x_3+x_1$, $x_5+x_4+x_2+x_0$, x_5+1, x_2+1, x_1+1 and $x_5+x_4+x_3+x_0$ respectively.

(2)Eight Boolean functions are all balanced.

(3)No any non-zero linear structure exists in the eight Boolean functions (The zero vector is a linear structure of any function).

(4)Nonlinearity of eight Boolean functions are all 112 and none of them is Bent function.

(5)None of the eight Boolean functions satisfies SAC of any order or PC of any degree or any order.

(6) None of the eight Boolean functions is correlation immune.

(7)For 32 inputs, i.e. x = 11,33,35,39,43,51,90,93,106,122, 134,139,142,157,164,165,170,178,192,193,194,197,217, 129,230,232,237,240,243,249,250,255, f_0 satisfies PC. Similar conclusions about $f_1 \sim f_7$ can be obtained and are omitted here.

5 Conclusion

From the view of Boolean function and with the tool of Walsh spectrum, we have studied several cryptographic properties of AES S-box. We found that AES Boolean functions satisfied the cryptosystem design requirement such as balancedness, high nonlinearity and few linear structures. This indicates that AES has good nonlinear features which guarantee its cryptographic strength to some extent. On the other hand, none of the AES Boolean functions satisfies SAC of any order or PC of any degree or any order, which we think may contribute to the cryptanalysis of AES. Further work is worthwhile in this point.

Moreover, the concepts of general autocorrelation[8] we proposed in another paper and that of autocorrelation will simplify the analysis of SAC, PC and its order.

References

[1] NBS, Federal Information Processing Standard Publication 46: Data Encryption Standard(DES), National Bureau of Standards, Gainthersburg, MD, 1997

[2] E.F.Brickell,J.H.Moore and M.R.Purtill, Structure in the S-boxes of the DES(extended abstract), in Advances in Cryptology--Crypto'86, p. 3-8,Springer-Verlag,1987

[3] Joan Daemen and Vincent Rijmen, AES proposal: Rijndael, First Advanced Encryption Standard Candidate Conference, N.I.S.T., 1998.

[4] Eli Biham and Adi Shamir, Differential Cryptanalysis of DES-like Cryptosystems, Springer-Verlag, New York, 1993

[5] Mitsuru Matsui, Linear cryptanalysis method for DES cipher, in Advances in Cryptology: Eurocrypt'93, p. 386- 397,Springer-Verlag,1994

[6] Feng Dengguo, Spectrum theory and its applications in cryptology , Science Press, Beijing PRC,2000

[7] Wen Qiaoyan,Niu Xinxin and Yang Yixian, Boolean functions in modern cryptology, Science Press, Beijing RPC, 2000

[8] Baodian Wei, Dongsu Liu and Xinmei Wang, "The general autocorrelation and its application", in the proceedings of First International Symposium on Cyber Worlds, November 6-8, 2002, Tokyo,Japan, p. 267-273.

Practical Construction for Multicast Re-keying Schemes Using R-S Code and A-G Code

Chun-Yan Bai, Roberta Houston and Gui-Liang Feng
The Center for Advanced Computer Studies
University of Louisiana at Lafayette
Lafayette, LA 70504, USA
Email: cxb7146, rah1231, glf@cacs.louisiana.edu

Keywords: Multicast, Re-keying, Reed-Solomon code, Hash function, KDP

Received: April 11, 2002

Multicast Re-keying means the establishment of a new session key for the new subgroup in the multicast system. Practical construction methods for multicast re-keying scheme using Reed-Solomon codes and Algebraic-Geometric codes are presented in this paper with examples to show the detailed constructions. The constructions require no computational assumptions. The storage complexity for group members(Group Controller and other users) and the transmission complexity for the schemes have been reduced to $O(log(n))$ at the same time.

1 Introduction

With the rapid development of networks, the need for high bandwidth, very dynamic and secure group(multicast) communications is increasingly evident in a wide variety of commercial, govermnent, and Internet communities such as video-on-demand, multi-party teleconferencing, stock quote distribution and updating software. Specifically, the security in the multicast communication is the necessity for multiple users who share the same security attributes and communication requirements to securely communicate with each other using a common group session key.

The general goal of secure group communication is to dynamically transmit a message encrypted by the new *session key* over a broadcast channel shared by an exponential number $n = 2^m$ of users so that *all but* some specified small coalition of k excluded users can decipher the message, even if these excluded users collude with each other in an arbitrary manner. This is what we call the *broadcast exclusion* problem(also known as the *blacklisting* problem). The establishment of a new *session key* for the new subgroup is called the *Re-keying* of the system.

In the multicast communication system, the group is dynamic, which means that at different time, different subgroups of the initial group is authorized to receive the multicast message because of those dynamically joining and leaving group members. So the secure communication in multicast environment is much more challenging than traditional point-to-point communication and raises numerous new security problems. Examples are the forward secrecy and backward secrecy guarantee. A protocol provides *perfect backward secrecy* if a member joining the group at time t does not gain any information about the content of messages communicated at times $t' < t$. A protocol provides *perfect forward secrecy* if a member leaving the group at time t does not gain any information about the content of messages communicated at time $t' > t$.

Member-joining is easy to handle by just encrypting the new session key with the old session key which is decryptable by all old members and sending the new session key individually to each new member encrypted by their own secret keys. So we just focus on the member-leaving case and assume that there is a group controller(GC) who knows all the system keys in this paper.

The initial study on the secure multicast communication can be traced back to the early 90's [1]. And a lot of works had followed [2,3,4,5,6,7,8,9]. All in all, the work can be divided into two major groups, one of which [2,3,4,5] uses the concept of *key tree structure* to set up the new session key based on the Diffie-Hellman key agreement. In [2], Wallner proposed a scheme which requires only $O(n) = O(n+(n-1))$ keys for GC, $O(log^n) = O(d+1)$ keys for each user and have at most $O(log^n) = O(kd-1)$ transmissions overhead per single eviction. The requirement is further improved in[3,4,5] which greatly reduces the transmission and storage complexity of re-keying schemes. Another stronger property of the tree structured scheme is that it allows the number of excluded users k to be arbitrary, rather than fixed in advance. But some balanced tree structure based schemes have the disadvantage of not providing collusion prevention.

The other group makes use of the *broadcast encryption* idea proposed by Fiat and Naor[6]. The broadcast encryption scheme enables the GC to communicate data secretly to dynamically changing authorized users while preventing any coalition of users to learn anything about the data. Other studies on broadcast encryption schemes can be found in [7,8] and [9]. In [7] , the concept of the Perfect Hash Family(PHF) is reviewed and proved to be useful for the secure new session key distribution. The possibility of

using the error correcting code to construct such a scheme is given there without providing any practical and detailed construction. Hartono *et.al.* [8] borrows the idea of Key Distribution Pattern (KDP), based on which the broadcast encryption scheme that can remove up to t users from a group of n users and is secure against collusion of t malicious users can be set up. How to use the error correcting codes to construct such a KDP is not discussed. In [9], Poovendran and Baras show that by assigning probabilities to member revocations, the optimality, correctness and the system requirements of some of the schemes in [6,7,8] can be systematically studied using information theoretic concepts and also show that the optimal average number of keys per member in a secure multicast scheme is related to the entropy of the member revocation event, thus provides a way for us to inspect each scheme from the theory point of view.

2 Related Work Review

Assume that there is a set of users U, a group controller GC and a set K of Key Encrypting Keys (KEK) that is generated and stored by the GC. *Session keys* are used for group member communication. A user u_i will have a subset of Key Encrypting Keys, $K(u_i) \subseteq K$. KEKs are used to update the SK in the event of membership change due to any of the following reasons: (a) a new member admission, (b) expiration of the SK, (c) member compromise, (d) voluntary leave, and (e) member revocation. We only consider the last case, member revocation in this paper.

The secure group communication requires KEKs to securely distribute the updated SK. If every member has an individual public key, for a group consisting of n members, the SK update will involve $O(n)$ encryptions by the GC. The linear increase of the required number of encryptions in group size is not suitable for very large scale applications common in Internet, due to the amount of computational burden on the GC.

Next, we will review two scalable re-keying schemes which can reduce the number of encryptions.

2.1 Re-keying Scheme Based on PHF

A re-keying scheme called OR scheme in [7] specifies an algorithm by which the GC produces a common session key $k^{U\setminus W}$ for the group $U \setminus W$ without letting those users in W to know the new session key, where $W \subseteq U$. The scheme is as follows:

1. *Key initialization* : The GC generates and stores a set K of KEKs and securely gives u_i the set of his KEKs $K(u_i) \subseteq K$.

2. *Broadcast* : To remove a set of users W from U, the GC randomly chooses a session key $k^{U\setminus W}$ and encrypts it with those keys not belonging to W, then broadcasts the encrypted messages to all the users. That is, the GC broadcasts

$$\{E_k(k^{U\setminus W}) |\ k \in K, k \notin K(W), K(W) = \cup_{j\in W} K(u_j)\}.$$

3. *Decryption:* Each user $u_i \in U \setminus W$ uses one of his own KEKs $k \in K(u_i)$ to decrypt $E_k(k^{U\setminus W})$ and obtain the new session key $k^{U\setminus W}$.

We review the concept of PHF here for the completeness of this paper. Let n and m be integers such that $2 \leq m \leq n$, $A = \{1, 2, ...n\}$ and $B = \{1, 2, ...m\}$ be two sets. A *hash function* is a function h from A to B $h : A \to B$. We say a hash function $h : A \to B$ is perfect on a subset $X \subseteq A$ if h is injective when restricted on X. Let w be an integer such that $2 \leq w \leq m$,and let $H \subseteq \{h : A \to B\}$. H is called an (n, m, w) *perfect hash family*(PHF) if for any $X \subseteq A$ with $|X| = w$ there exists at least one element $h \in H$ such that h is perfect on X.

It is proven in [7] that if there exists a $PHF(N, n, m, w)$, then there exists a re-keying scheme in which the number of KEKs for each user and the GC are N and Nm respectively and the number of broadcast transmissions to remove up to w users is less than $(m-1)N$.

It is also proven in [7] that an (N, n, d, m) erasure code gives rise to a $PHF(N, n, m, w)$ as long as $N > \binom{w}{2}(N-d)$, thus can be used for the construction of the above re-keying scheme. Such a scheme can prevent w users from colluding. The performance of the re-keying scheme based on PHF is determined by the parameter N when w and m are fixed, which should be minimized to reduce the storage and transmission complexity. But the author didn't mention any details on which kind of error correcting code should be used and how it is used for the construction.

2.2 Re-keying Scheme Based on KDP

In [8], H. Kurnio reviewed the concept of Key distribution Patterns(KDP).

Let $X = \{x_1, x_2, ..., x_n\}$ and $B = \{B_1, B_2, ..., B_N\}$ be a family of subsets of X. The pair (X, B) is called an (n, N, t)-key distribution pattern if

$$|(B_i \cap B_j) \setminus \cup_{k=1}^{t} B_{s_k}| \geq 1$$

for any $(t+2)$-subset $\{i, j, s_1, ..., s_t\}$ of $\{1, 2, ..., N\}$.

With the idea of KDP, the author presented a theorem to show the existence of a multicast re-keying scheme with dynamic controller based on KDP. But how to effectively construct KDP is still an open problem.

Inspired by the work from [7] and [8], we look at the problem of multicast re-keying from the error-correcting codes point of view in this paper. In order to achieve constructions with feasible storage that do not require computational assumptions, we make an improvement on the constraints that must be satisfied to construct the broadcast encryption scheme in[7,8] by avoiding the requirement of

being PHF and KDP. Based on the OR model mentioned above and assumed a system with GC, we give two practical construction of schemes based on Reed-Solomon codes and avoid any computational assumptions. Conditions underlining the constructions are also given together with examples to show the detail constructions.

Kumar *et.al.* [10] also consider the blacklisting problem through error-correcting codes, but their method is quite different from ours.

3 Multicast Re-keying Scheme Based on R-S Code

3.1 Background on Code

Let $GF(q)$ be a finite field.

Definition 3.1 (Linear Code) *An $[m,k,d]$ linear code is a k-dimensional subspace $V_{m,k}$ of m-dimensional linear space V_m over GF_q, where the minimum Hamming distance between any pair of elements is d.*

Reed-Solomon code is an important kind of linear block BCH codes which had been widely used in such areas as space communication systems, spread-spectrum communication systems and computer storage systems.

Definition 3.2 (Reed − Solomon Code) *Let $x_1,...,x_m \in GF(q)$ be distinct and $k > 0$. The $(m,k)_q$ Reed-Solomon code is given by the subspace $\{(f(x_1),...,f(x_m))|f \in GF_{q,k}\}$, where $GF_{q,k}$ denote the set of polynomials on $GF(q)$ of degree less than k.*

R-S code is a Maximum Distance Separable (MDS) code, which means that the error-correcting capability of the R-S code can reach the Singleton bound. The R-S code has the property that the $(m,k)_q$ R-S code is an $[m,k,m-k+1]_q$ linear code and it requires that $m \leq q$.

3.2 R-S Code Based Construction

3.2.1 First Construction

Theorem 3.1 *Let (N,k,d) be a Reed-Solomon code over $GF(q)$, where N is the length of the codewords, k is the length of the information bits and d is the minimum distance of the code. The number of the codewords $n = q^k$. Let W be a subset of $\{1,2,...,n\}$ with $|W| = w$. Then such an error-correcting code can be used to construct a multicast encryption scheme as long as it satisfies that*

$$N > w*(N-d).$$

Proof: Let T be the set of codewords of a (N,k,d) code, $|T| = n$. We write each element of T as $(c_{i1},c_{i2},...,c_{iN})$ with $c_{ij} \in \{1,2,...,q\}$, where $1 \leq i \leq n, 1 \leq j \leq N$ and n is the number of codewords. For each j we define a function h_j from $A = \{1,...,n\}$ to $B = \{1,...,q\}$ by $h_j(i) = c_{ij}$ and let $H = \{h_j|j = 1,...N\}$.

In the key initialization phase, The GC generates and stores a set of Nq keys defined as $K = \{k_{(h,b)} \mid h \in H, b \in B\}$. For a user $u_i, 1 \leq i \leq n$, GC secretly gives u_i the set of N Key Encryption Keys $K(u_i) = \{k_{(h,h(i))}|h \in H\}$.

In the broadcast stage of removing a set of users W from $U, |W| \leq w$, the GC randomly select a new session key and encrypt it with those KEKs that do not belong to W, then broadcast the encrypted messages to all the users. So those users that have been removed can not use their own KEKs to decrypt and obtain the new session key.

As to the decryption phase, we need to prove that any user u_i that does not belong to W has at least one key to decrypt and obtain the new session key.

Let $W = \{u_{i1},...,u_{iw}\}$. Since the minimum distance of the code is d, for any given pair of elements $x_1, x_2 \in U$, there are at most $N-d$ functions from H such that the values of these $N-d$ functions evaluated on x_1 and x_2 are the same. For any user $u_i \notin W$, it has at most $N-d$ functions that is the same as u_{i1}, at most $N-d$ same functions as u_{i2}, ... and at most $N-d$ same functions as u_{iw}. The worst case is that the same $N-d$ functions that u_i has with u_{i1} is different from those $N-d$ functions that u_i has with u_{i2}, which is different from those $N-d$ functions that u_i has with u_{i3}, ... That is, all the w (N-d) functions are different. So we conclude that if $N > w*(N-d)$, then u_i has at least one function that is different from all those functions belong to W. That is, there exists a function $h_\alpha \in H$ such that $\{h_\alpha(j)|j = i_1,i_2,...,i_w,i\}$ are all distinct. It follows that $k_{(h_\alpha,h_\alpha(i))}$ is in $K(u_i) \subseteq K(U \setminus W)$, so u_i can decrypt the encrypted message and obtain the new session key $k^{U\setminus W}$. □

The theorem holds for any set L of members who wants to leave the original group as long as $|L| \leq w$.

Example 3.1 *Take the $(N,k,d) = (4,2,3)$ RS code over finite field $GF(4) = GF(2^2) = \{0,1,\alpha,\alpha^2\}$. The primitive element α is the root of $x^2+x+1 = 0$. From theorem 3.1 we know that, if*

$$N - w(N-d) > 0,$$

then there exists a broadcast encryption scheme based on such a RS code, which means that $w < \frac{N}{N-d} = \frac{4}{4-3} = 4$.

Since $k = 2$, the information sequence is

$$\overline{m} = (m_1, m_2).$$

The codewords, that is KEKs for all users corresponding to all possible information sequence is shown in Table 1.

3.2.2 Discussion

1. From [7], it is also known that any (N,k,d) error correcting code gives rise to a PHF(N,n,m,w) which is proven to be effective to set up the multicast encryption scheme.

	m_2	m_1	h(0)	h(1)	$h(\alpha)$	$h(\alpha^2)$
u_1	0	0	0	0	0	0
u_2	0	1	1	1	1	1
u_3	0	α	α	α	α	α
u_4	0	α^2	α^2	α^2	α^2	α^2
u_5	1	0	0	1	α	α^2
u_6	1	1	1	0	α^2	α
u_7	1	α	α	α^2	0	1
u_8	1	α^2	α^2	α	1	0
u_9	α	0	0	α	α^2	1
u_{10}	α	1	1	α^2	α	0
u_{11}	α	α	α	0	1	α^2
u_{12}	α	α^2	α^2	1	0	α
u_{13}	α^2	0	0	α^2	1	α
u_{14}	α^2	1	1	α	0	α^2
u_{15}	α^2	α	α	1	α^2	0
u_{16}	α^2	α^2	α^2	0	α	1

Table 1: User KEKs constructed from (4,2,3) R-S code

There, it requires that the minimum distance of the error-correcting code d' satisfies that

$$d' > \frac{\left(\binom{w}{2} - 1\right) N}{\binom{w}{2}}.$$

That is,the minimum d' that satisfies the above inequality is

$$d' = \left\lfloor \frac{\left(\binom{w}{2} - 1\right) N}{\binom{w}{2}} \right\rfloor + 1.$$

While here, from the above theorem, since

$$N > w * (N - d),$$

the minimum distance d has to satisfy that

$$d > \left(1 - \frac{1}{w}\right) N.$$

That is, the minimum d that satisfies the above inequality is

$$d = \left\lfloor \frac{w-1}{w} N \right\rfloor + 1.$$

Because $d \leq d'$, the requirement for constructing the broadcast encrypting scheme using R-S code had been reduced since it is more easier to find such a RS code, which allows us to increase the length of the information bit k when the code length is fixed and further more to reduce the requirement for the bandwidth.

2. For any $[n, k, d]_q$ R-S code over finite field F_q, when $N = q$, $k = log_q n$, where n is the number of codewords,

$$d = N - k + 1 = q - log_q n + 1,$$

then from

$$N < w * (N - d),$$

we get

$$w < \frac{N}{N-d} = \frac{q}{log_q n - 1}.$$

Example 3.2 *Take an* $[8, 3, 6]_8$ *R-S code over finite field*

$$GF(8) = \{0, 1, \alpha, \alpha^2, \alpha^3, \alpha^4, \alpha^5, \alpha^6\}$$

as an example. Since $q = 8, N = q = 8, k = 3, d = N - k + 1 = 8 - 3 + 1 = 6$, *then the number of codewords* $n = 8^3 = 512$ *and*

$$w < \frac{N}{N-d} = \frac{q}{log_q n - 1} = \frac{8}{3-1} = 4.$$

So the $[8, 3, 6]_8$ *R-S code over finite field* $GF(8)$ *can be used to construct the secure broadcast scheme as long as the number of members who want to quit is less than or equal to 3 where* $N > w(N - d) \leftrightarrow 8 > 3(8 - 6)$.

3.2.3 Extension of the Scheme

In order to improve the communication efficiency, the OR scheme we discussed can be slightly modified with erasure code such that the bandwidth used by the GC for broadcasting information can be reduced.

An $[n, k, m]$ erasure code is a special class of error-correcting codes that allow recovery of a message if part of its messages($\leq (n - m)$) are damaged or erased during the transmission. An erasure code can be constructed using Reed-Solomon code over finite field $GF(q)$. The decoding procedure uses k pairs of $(e_i, p_v(e_i))$ to recover the original k information messages, where e_i is one of the field element over $GF(q)$ and $p(x) = v_0 + v_1(x) + ... + v_{k-1}(x^{k-1})$ is the polynomial for the R-S encoding.

The broadcast encryption scheme described in Theorem 3.1 can be modified as follows. In the broadcast phase, before broadcasting the new session key $k^{U \setminus W}$, an encoding procedure is first applied to the new session key. The new session key $k^{U \setminus W}$ was divided into t pieces $k^{U \setminus W} = (k_1^{U \setminus W}, k_2^{U \setminus W}, ..., k_t^{U \setminus W})$, then encodes them using $[Nm, t, \alpha]$ erasure code to obtain the codeword $C(k^{U \setminus W}) = (c_1, c_2, ..., c_{Nm})$. The GC uses all the KEKs that do not belong to the users of W to encrypt the corresponding components of $C(k^{U \setminus W})$ and broadcasts the encrypted messages to all the users. That is, the GC broadcasts

$$\{E_{k_i}(c_i) \mid k_i \in K, k_i \notin K(W)\}.$$

As long as each non-excluded user has at least α keys that can decrypt α messages of $\{E_{k_i}(c_i)\}$, he can then apply the erasure code to obtain the new session key. While for those users in W, same as before, they can not find the session keys.

Theorem 3.2 *The above scheme works as long as the following inequality holds:*

$$N - w(N - d) \geq \alpha.$$

For an $[n, k, m]$ erasure code over $GF(q)$, we expect k to be as large as possible in order to minimize the extra bandwidth n/k for the transmission. Actually, the basic scheme we discussed in Theorem 3.1 is a special case of using $[n, 1, 1]$ erasure code for the construction.

Example 3.3 *Consider the same example as in Example 3.1: the RS code $(N, k, d) = (4, 2, 3)$ over finite field $GF(4) = GF(2^2) = \{0, 1, \alpha, \alpha^2\}$. The primitive element α is the root of $x^2 + x + 1 = 0$. The KEKs for all users corresponding to all possible information sequence is shown in Table 1.*

For the above scheme to work, it needs that

$$N - w(N - d) \geq \alpha,$$

that is,

$$4 - w(4 - 3) \geq \alpha,$$

that is,

$$4 - w \geq \alpha.$$

For $\alpha = 1$, w can be 1,2 or 3. For $\alpha = 2$, w can be 1 or 2. And for $\alpha = 3$, w can only be 1. We take $w = 2$ and $\alpha = 2$ as an example.

We divide the new session key $k^{U\setminus W}$ into two parts $k^{U\setminus W} = (k_0^{U\setminus W}, k_1^{U\setminus W})$, then encodes $k^{U\setminus W}$ using a [16,2,2] erasure code to obtain a codeword

$$C(k^{U\setminus W}) = (c_1(0), c_2(1), c_3(\alpha), ..., c_{16}(\alpha^{14})),$$

where

$$c_i(e_i) = p(e_i), e_i \in GF(16)$$

and

$$p(x) = k_0 + k_1 x.$$

Suppose any two users $W = \{u_{s_1}, u_{s_2}\}, |W| = 2$ want to leave the group, the GC uses all the KEKs that do not belong to this two users to encrypt all these 16 pieces of encoded keys and broadcasts the encrypted messages to all the users. Then each user that is not in W has at least 2 keys to decrypt 2 messages, thus can recover the original new session key.

3.3 Second R-S Code Based Construction

In [8], Hartono proposed a broadcast encryption scheme based on the KDP(Key Distribution Pattern) which can be used for dynamic GC. If the GC is fixed in the system and is trustable, then the condition for the scheme can be improved to make it work for general case.

3.3.1 Scheme Description

Theorem 3.3 *Let $X = \{x_1, x_2, ...x_{n^*}\}$ be a set of KEKs, $U = \{U_1, U_2, ..., U_n\}$ be the set of users' KEKs, which is a family of subset of X, that is, for $\forall i$, $U_i \subseteq X$. Let $W = \{U_{s_1}, U_{s_2}, ...U_{s_w}\}$ be a subset of U with $|W| = w$. If for $\forall i$, it satisfies that:*

$$|U_i \setminus \cup_{k=1}^{w} U_{s_k}| \geq 1,$$

Then a broadcast encryption scheme can be constructed which can remove up to w users from a group of n users.

In this scheme, The GC generates and stores a set X of KEKs in the key initialization phase, and sends each user u_i a subset $U_i \subseteq U$ of X as the user's KEKs. When a set of users W want to quit from the group, the GC selects a new session key $k^{U\setminus W}$ and encrypts the session key with all KEKs except those belong to users in W, that is, GC broadcasts $\{E_{k_r}(k^{U\setminus W}) \mid k_r \in X \setminus (U_{s_1} \cup U_{s_2} ... \cup U_{s_w})\}$. So, those users in W can not decrypt the encrypted message. While, since for $\forall u_i$ that $U_i \in X \setminus W$,

$$|U_i \setminus \cup_{k=1}^{w} U_{s_k}| \geq 1,$$

it has at least one key that does not belong to W, so it can decrypt $E_{k_r}(k^{U\setminus W})$ and obtain the new session key $k^{U\setminus W}$.

>From the theorem we know that for any given w and n, we should make n^* as small as possible. Same, for any given w and n^*, we hope n to be as large as possible.

Next, we will show how to use Reed-Solomon code to construct the KEK set X and U and how the scheme works.

3.3.2 R-S Code Based Construction

We take the R-S code (N, k, d) over the finite field $GF(q) = \{0, 1, \alpha, ...\alpha^{q-2}\}$, The number of users $n = q^k$. The RS codeword $\bar{c}$ of length N is generated from k information symbols taken from the finite field $GF(q)$ through polynomial

$$h(x) = m_0 + m_1 x + m_2 x^2 + ...$$
$$+ m_{k-2} x^{k-2} + m_{k-1} x^{k-1},$$

where

$$\overline{m} = (m_0, m_1, ...m_{k-1})$$

and

$$\bar{c} = (c_0, c_1, ..., c_{q-1})$$
$$= (h(0), h(1), h(\alpha), h(\alpha^2), ..., h(\alpha^{q-2})).$$

For each user u_i, the KEK set that corresponds to the k information symbols

$$\overline{m_i} = (m_{i1}, m_{i2}, ...m_{ik})$$

is

$$U_i = \{(h_i(0), h_i(1), h_i(\alpha), ..., h_i(\alpha^{q-2}))\},$$

where $|U_i| = q = N$. So, the KEK set for all users is

$$U = \cup_{i=1}^{n} U_i.$$

The total KEK set X for GC is

$$X = \{X_i,\ i = 1, 2, ..., n^*\},$$

where $n^* = N * q = q^2$, and

$$X_i = \{(h, \beta) \mid h \in \{h(0), h(\alpha), ..., h(\alpha^{q-1})\},$$

$$\beta \in GF(q).\}$$

Next, we will use an example to show the exact procedure on the RS-code construction of the scheme.

Example 3.4 *Take the same example as in Example 3.1: that is the* $(N, k, d) = (4, 2, 3)$ *RS code over finite field* $GF(4) = GF(2^2) = \{0, 1, \alpha, \alpha^2\}$. *The primitive element* α *is the root of* $x^2 + x + 1 = 0$. *The KEKs for all users corresponding to all possible information sequence is shown in Table 1. After extending the users' KEKs by use of the way shown in section 3.3.2, we obtain the KEKs set* $X = \{X_i, i = 1, 2, ..., 16\}$ *as shown in Table 2.*

>*From Table 2 we can see that,*

$$U_1 = \{x_1, x_5, x_9, x_{13}\}$$

$$= \{(h_1, 0), (h_2, 0), (h_3, 0), (h_4, 0)\}$$

$$U_2 = \{x_2, x_6, x_{10}, x_{14}\}$$

$$= \{(h_1, 1), (h_2, 1), (h_3, 1), (h_4, 1)\}$$

$$U_3 = \{x_3, x_7, x_{11}, x_{15}\}$$

$$= \{(h_1, \alpha), (h_2, \alpha), (h_3, \alpha), (h_4, \alpha)\}$$

$$U_4 = \{x_4, x_8, x_{12}, x_{16}\}$$

$$= \{(h_1, \alpha^2), (h_2, \alpha^2), (h_3, \alpha^2), (h_4, \alpha^2)\}$$

$$U_5 = \{x_1, x_6, x_{11}, x_{16}\}$$

$$= \{(h_1, 0), (h_2, 1), (h_3, \alpha), (h_4, \alpha^2)\}$$

...

$$U_{16} = \{x_4, x_5, x_{11}, x_{14}\}$$

$$= \{(h_1, \alpha^2), (h_2, 0), (h_3, \alpha), (h_4, 1)\}.$$

All the KEKs hold by the GC is given by:

$$U = \cup_{i=1}^{16} U_i.$$

Suppose there are $w = 2$ *users who want to quit from the group* $\{u_1, u_2, ..., u_{16}\}$, *say users* $W = \{u_7, u_8\}$, *we can check that for each user* $u_i \notin W$,

$$|U_i \setminus \cup_{k=1}^{w} U_{s_k}| \geq 1.$$

For example,

$$|U_1 \setminus \cup_{k=1}^{2} u_{s_k}| = |\{x_1, x_5\}| = 2 \geq 1.$$

So such a set of KEKs can be used to implement the broadcast encryption scheme.

4 Construction of the Scheme Using A-G Code

Since the R-S code over $GF(q)$ requires the length of the codewords $N \leq q$, we can not make the codeword longer than q. Using Algebraic-geometric code(A-G code), the scheme can be extended to the case when codeword length $N > q$. Next we will show an example on how to use A-G code to construct the OR model for the multicast re-keying scheme.

4.1 A-G Code

For those who are interested in more details about A-G code, please refer to the paper [11] and [12].

4.2 Example of A-G Code Based Multicast Re-keying Scheme

Let us consider the Hermitian code over $GF(4) = GF(2^2)$ with $k = 2$. The Hermitian curve over $GF(2^2)$ is $x^3 + y^2 + y = 0$. The curve has rational points:

$$\{(0,0), (0,1), (1,\alpha), (1,\alpha^2),$$

$$(\alpha,\alpha), (\alpha,\alpha^2), (\alpha^2,\alpha), (\alpha^2,\alpha^2)\}$$

$$=^{\triangle} \{((x_1,y_1), (x_2,y_2), (x_3,y_3), (x_4,y_4),$$

$$(x_5,y_5), (x_6,y_6), (x_7,y_7), (x_8,y_8)\}.$$

Let code polynomial be $c(x) = m_0 + m_1 x$, it has 16 codewords:

$$\overline{c} = (c(x_1,y_1), c(x_2,y_2), c(x_3,y_3), c(x_4,y_4),$$

$$c(x_5,y_5), c(x_6,y_6), c(x_7,y_7), c(x_8,y_8))$$

All the codewords, that is, the KEKs set are shown in Table 3.

In this example, $c(x)$ has at most 2 zero points, $d = 8 - 2 = 6$. since $q = 4, N = 8, d = 6$, the number of users $n = q^2 = 16$, the number of keys is $N * q = 8 * 4 = 32$.

For the OR multicast re-keying scheme to work, it requires that

$$N > w(N - d),$$

that is,

$$8 > w(8 - 6),$$

so w can be 2 or 3. Since w can be 3, from $N - w(N - d) = \alpha$, we know that α can be 2.

5 Conclusions

In this paper, two practical constructions for Multicast Re-keying Schemes using Reed-Solomon Codes are given with examples to show the detailed construction procedure. Because it has many properties we expected, RS code provides us a practical way to construct the multicast re-keying

	h_1	h_1	h_1	h_1	h_2	h_2	h_2	h_2	h_3	h_3	h_3	h_3	h_4	h_4	h_4	h_4
	0	1	α	α^2	0	1	α	α^2	0	1	α	α^2	0	1	α	α^2
	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}	x_{11}	x_{12}	x_{13}	x_{14}	x_{15}	x_{16}
u_1	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0
u_2	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0
u_3	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0
u_4	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1
u_5	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0	1
u_6	0	1	0	0	1	0	0	0	0	0	0	1	0	0	1	0
u_7	0	0	1	0	0	0	0	1	1	0	0	0	0	1	0	0
u_8	0	0	0	1	0	0	1	0	0	1	0	0	1	0	0	0
u_9	1	0	0	0	0	0	1	0	0	0	0	1	0	1	0	0
u_{10}	0	1	0	0	0	0	0	1	0	0	1	0	1	0	0	0
u_{11}	0	0	1	0	1	0	0	0	0	1	0	0	0	0	0	1
u_{12}	0	0	0	1	0	1	0	0	1	0	0	0	0	0	1	0
u_{13}	1	0	0	0	0	0	0	1	0	1	0	0	0	0	1	0
u_{14}	0	1	0	0	0	0	1	0	1	0	0	0	0	0	0	1
u_{15}	0	0	1	0	0	1	0	0	0	0	0	1	1	0	0	0
u_{16}	0	0	0	1	1	0	0	0	0	0	1	0	0	1	0	0

Table 2: Construction of Re-keying scheme with (4,2,3) R-S code.

	m_2	m_1	h(0)	h(1)	$h(\alpha)$	$h(\alpha^2)$				
u_1	0	0	0	0	0	0	0	0	0	0
u_2	0	1	0	0	1	1	α	α	α^2	α^2
u_3	0	α	0	0	α	α	α^2	α^2	1	1
u_4	0	α^2	0	0	α^2	α^2	1	1	α	α
u_5	1	0	1	1	1	1	1	1	1	1
u_6	1	1	1	1	0	0	α^2	α^2	α	α
u_7	1	α	1	1	α^2	α^2	α	α	0	0
u_8	1	α^2	1	1	α	α	0	0	α^2	α^2
u_9	α	0	α	α	α	α	α	α	α	α
u_{10}	α	1	α	α	α^2	α^2	0	0	1	1
u_{11}	α	α	α	α	0	0	1	1	α^2	α^2
u_{12}	α	α^2	α	α	1	1	α^2	α^2	0	0
u_{13}	α^2	0	α^2	α^2	α^2	α^2	α^2	α^2	α	α
u_{14}	α^2	1	α^2	α^2	α	α	1	1	0	0
u_{15}	α^2	α	α^2	α^2	1	1	0	0	α	α
u_{16}	α^2	α^2	α^2	α^2	0	0	α	α	1	1

Table 3: User KEKs constructed from (8,2,6) A-G code.

scheme efficiently. The storage complexity and transmission complexity have been reduced at the same time, which is another advantage of the method proposed in this paper.

Because this paper is only an initial work for using RS-code in constructing the re-keying scheme, a lot of work is being done and will be done in the future such as how to improve the communication transmission efficiency by encoding the new session key with error correcting code first, how to deal with multiuser and multistage leaving from the group, how to handle when a new user is joining, but the members in the group has reached the maximum, how to apply AG code instead of RS code in the construction to improve the performance, how to make the GC be a group member also, how to extend these two schemes to apply for the distributed environment and so on.

References

[1] S.Berkovits. How to Broadcast a Secret. *Advances in Cryptology - EUROCRYPT'91, Lecture Notes in Computer Science 547*, p535-541, 1991;

[2] D.M.Wallner, E.J.Harder and R.C.Agee. Key Management for Multicast: Issues and Architectures. *Intenet Draft*, September, 1999;

[3] C.K.Wong, M.Gouda an dS.S.Lam, Secure Group Communication using Key graphs. *Proceedings of SIGCOMM'98*, p68-79, 1998;

[4] A.Perrig, Efficient Collaborative Key Management Protocols for Secure Autonomous Group Communication. *CrypTec'99*, Hongkong, December, 1999;

[5] I.Chang, R.Engel, *et.al.*, Key Management for Secure Internet Multicast using Boolean Fnction Minimization Techniques. *INFOCOM'99*,September, 1999;

[6] A.Fiat and M.Naor, Broadcast Encryption. *Advances in Cryptology - CRYPT'92, Lecture Notes in Computer Science*, vol.773, p481-491, 1993;

[7] S.Naini R. and H.Wang, New Constructions for Multicast Re-Keying Schemes using Perfect Hash Families. *7th ACM Conference on Computer and Communication Security*, ACM Press,p228-234,2000;

[8] H.Harnio, R.S.Naini, W.Susilo and H. Wang, Key Management for Secure Multicast with Dynamic Controller. *Information Security and Privacy, 5th Australasian Conference, ACISP'00, Lecture Notes in Computer Science 1841*, p178-190,2000; September, 1998;

[9] R.Poovendran and J.S.Baras, An Information Theoretic Analysis of Rooted-Tree Based Secure Multicast Key Distribution Schemes. *IEEE Transactions on Information Theory*, May, 1999;

[10] R.Kumar, S.Rajagophlan and A.Sahai, Coding constructions for blacklisting problems without computational assumptions. *Advances in Cryptology - CRYPTO'99*, Lecture Notes in Computer Science, p609-623, 1999;

[11] G.L. Feng and T.R.N. Rao, Decoding Algebraic Geometric Codes up to the Design Minimum Distance. *IEEE Transaction on Information Theory*, vol.1, No.1, p37-45, Jan. 1993;

[12] G.L. Feng, V.K. Wei, T.R.N. Rao, and K.K. Tzeng, Simplified Understanding and Efficient Decoding of a Class of Algebraic-Geometric Codes. *IEEE Transaction on Information Theory*, vol.40, No.4, p981-1002, Jul. 1994;

Multisecret Sharing Immune Against Cheating

Josef Pieprzyk and Xian-Mo Zhang
Centre for Advanced Computing – Algorithms and Cryptography
Department of Computing
Macquarie University
Sydney, NSW 2109, AUSTRALIA
E-mail: josef, xianmo@ics.mq.edu.au

Keywords: Secret Sharing, Multisecret Secret Sharing, Cheating Immune Secret Sharing

Received: May 19, 2002

Cheating in multisecret sharing is considered. Multisecret sharing is defined by a mapping F: $GF(p^t)^n \longrightarrow GF(p^t)^m$ that provides a generic model. In this model, we propose nonlinear multisecret sharing that is immune against cheaters. Two cheating strategies are considered. In the first one, all cheaters always submit their invalid shares and they collectively know their own valid shares. In the second one, some cheaters may submit their valid shares while again sharing their knowledge about their valid shares. The combiner (or recovery algorithm) interacts with shareholders by collecting shares from them and distributing the recovered secrets back to active participants. Two different scenarios are considered when the combiner recreates all secrets (this is simultaneous recovery) or gradually (so called sequential recovery). Probabilities of successful cheating are derived and constructions for cheating immune multisecret sharing are given.

1 Introduction

Cheating prevention in secret sharing and group oriented cryptography becomes one of the central security issues. Roughly saying, secret sharing is cheater-immune if a cheater is not better off than a participant who follows the protocol honestly. Tompa and Woll [8] demonstrated how Shamir secret sharing can be subject to cheating so after the recovery phase, honest participants are left with an invalid secret while cheaters are able to compute the valid one.

The problem of cheating prevention was investigated in [6]. It was shown that secret sharing can be constructed in such a way that cheaters after revealing an invalid secret by the combiner (or recovery algorithm), are getting no information about the valid secret. In a sense, the knowledge about the valid secret of honest and dishonest participants is the same with an obvious exception that cheaters know that the recovered secret is invalid while the honest ones will learn about this fact later when the recovered secret fails to trigger the intended action.

Multisecret sharing was probably first discussed in [5]. General formulation of the problem for the case when m different secrets are shared among participants with a single access structure was studied in [1]. Some further works can be found in [3, 4, 2].

Clearly, cheating participants in multisecret sharing schemes have more possibilities to deviate during the reconstruction of secrets depending on how the combiner who collects shares is working and also how the secrets are reconstructed. To make our considerations explicit we assume a simple multisecret sharing model in which every n participants can recover the secrets. The combiner who reconstructs the secrets can return all secrets (parallel reconstruction) or secret by secret (sequential reconstruction) where secrets are recreated in a some publicly known order. From now on we assume that the combiner is implemented in such a way that it accepts shares and after getting the appropriate number of them, reveals the reconstructed secret to all active participants (shares are never revealed by the combiner). This of course, does not restrict dishonest participants who may reveal their shares to each other.

2 Notations

Multisecret sharing is defined by a mapping F: $GF(p^t)^n \longrightarrow GF(p^t)^m$. F is called the defining mapping (or distribution rule) and is publicly accessible. Each vector $\alpha \in GF(p^t)^n$ determines a collection of n shares held by n participants and the vector $F(\alpha) \in GF(p^t)^m$ specifies a collection of m secrets. In particular, when $m = 1$, the defining mapping becomes the defining function that was studied in [6].

We consider two basic cheating strategies that are possible for dishonest participants to undertake. The strategies characterise the way the combiner works:

- simultaneous recovery of all secrets – in this case dishonest participants can modify all their shares or perhaps, they can collectively decide that some portion of their valid shares will be submitted to the combiner (those two scenarios will be considered). Note that the knowledge of cheaters is restricted to the shares they hold,

• sequential recovery of secrets – again dishonest participants submit a collection of invalid shares to the combiner who returns a single secret. The recovery process of a single secret is independent as dishonest participants can deliver a different collection of invalid shares for each recovery. Observe that the knowledge of cheaters changes after each recovery as they obtain a secret returned by the combiner.

We assume that the combiner is honest and returns the secret corresponding to the submitted shares and after the recovery it "forgets" all shares and secrets.

Let $GF(p^t)$ denote a finite field with p^t elements where p is a prime number and t is a positive integer. We write $GF(p^t)^n$ to denote the vector space of n tuples of elements from $GF(p^t)$. Then each vector $\alpha \in GF(p^t)^n$ can be expressed as $\alpha = (a_1, \ldots, a_n)$ where $a_1, \ldots, a_n \in GF(p^t)$. The *Hamming weight* of a vector $\alpha \in GF(p^t)^n$, denoted by $HW(\alpha)$, is the number of nonzero coordinates of α.

We consider a mapping F: $GF(p^t)^n \longrightarrow GF(p^t)^m$ written either $F(x)$ or $F(x_1, \ldots, x_n)$ where $x = (x_1, \ldots, x_n)$ and each $x_j \in GF(p^t)$. F is said to be *regular* if $F(x)$ takes each vector in $GF(p^t)^m$ precisely $p^{t(n-m)}$ times while x goes through each vector in $GF(p^t)^n$ once. A regular mapping $GF(p^t)^n \longrightarrow GF(p^t)^m$ exists only when $n \geq m$. A mapping f: $GF(p^t)^n \longrightarrow GF(p^t)$ is called a *function* on $GF(p^t)^n$. A regular function f is also called a *balanced* function. A mapping F: $GF(p^t)^n \longrightarrow GF(p^t)^m$ can be expressed as $F = (f_1, \ldots, f_m)$ or $F(x) = (f_1(x), \ldots, f_m(x))$, where each coordinate f_j is a function on $GF(p^t)^n$ and $x \in GF(p^t)^n$.

Let $x = (x_1, \ldots, x_n)$ and $\delta = (\delta_1, \ldots, \delta_n)$ be two vectors in $GF(p^t)^n$. Define a vector $x_\delta^+ \in GF(p^t)^n$, whose j-th coordinate is x_j if $\delta_j \neq 0$, or 0 if $\delta_j = 0$. In addition, we define a vector $x_\delta^- \in GF(p^t)^n$, whose j-th coordinate is 0 if $\delta_j \neq 0$, or x_j if $\delta_j = 0$. Clearly $(\beta + \gamma)_\delta^+ = \beta_\delta^+ + \gamma_\delta^+$, $(\beta + \gamma)_\delta^- = \beta_\delta^- + \gamma_\delta^-$ and $\beta_\delta^+ + \beta_\delta^- = \beta$ hold for any $\beta, \gamma \in GF(p^t)^n$, also $\delta_\delta^+ = \delta$, $\delta_\delta^- = 0$. Let $\tau = (\tau_1, \ldots, \tau_n)$ and $\delta = (\delta_1, \ldots, \delta_n)$ be two vectors in $GF(p^t)^n$. We write $\tau \preceq \delta$ to denote the property that if $\tau_j \neq 0$ then $\delta_j \neq 0$. In addition, we write $\tau \prec \delta$ to denote the property that $\tau \preceq \delta$ and $HW(\tau) < HW(\delta)$. Clearly if $\tau \preceq \delta$ then $\tau + \delta \preceq \delta$. In particular, if $\delta' \preceq \delta$ and $HW(\delta') = HW(\delta)$ we write $\delta \bowtie \delta'$. It is easy to verify that $\delta \bowtie \delta' \Longleftrightarrow \delta' \preceq \delta$ and $\delta \preceq \delta' \Longleftrightarrow$ both $x_\delta^+ = x_{\delta'}^+$ and $x_\delta^- = x_{\delta'}^-$ hold for any $x \in GF(p^t)^n$, where $\Longleftrightarrow$ denotes "if and only if".

3 Simultaneous Recovery of Secrets with Cheaters Using Invalid Shares only

3.1 Probability of Successful Cheating

In this work we use a mapping F: $GF(p^t)^n \longrightarrow GF(p^t)^m$ that defines a multisecret sharing. Let δ be a nonzero vector in $GF(p^t)^n$, $\tau \preceq \delta$ and $\mu \in GF(p^t)^m$. F can be equivalently represented in the form of table $\mathcal{T}$ with rows containing $(\alpha, F(\alpha))$. Set $R_F(\delta, \tau, \mu) = \{x_\delta^- \mid F(x_\delta^- + \tau) = \mu\}$. We also simply write $R_F(\delta, \tau, \mu)$ as $R(\delta, \tau, \mu)$ if no confusion occurs. The following statement can be formulated.

Lemma 1 *Let δ be a nonzero vector in $GF(p^t)^n$, $\tau \in GF(p^t)^n$, $\tau \preceq \delta$, and $\mu \in GF(p^t)^m$. Then for any given mapping F: $GF(p^t)^n \longrightarrow GF(p^t)^m$, (i) $R(\delta, \tau, \mu) = R(\delta', \tau, \mu)$ if $\delta \bowtie \delta'$, (ii) $R(\delta, \alpha_\delta^+, \mu) = R(\delta, \gamma_\delta^+, \mu)$ for any $\alpha, \gamma \in GF(p^t)^n$ with $\alpha_\delta^+ = \gamma_\delta^+$, (iii) there exists some $\mu \in GF(p^t)^m$ such that $R(\delta, \tau, \mu) \neq \emptyset$, where $\emptyset$ denotes the empty set.*

Given a mapping F: $GF(p^t)^n \longrightarrow GF(p^t)^m$. We introduce the following notations:

• Let $\alpha \in GF(p^t)^n$ be the sequence of n shares held by the group $\mathcal{P} = \{P_1, \ldots, P_n\}$ of n participants and the multisecret $\mu = F(\alpha)$.

• The collection of cheaters is determined by the sequence $\delta = (\delta_1, \delta_2, \ldots, \delta_n)$ where P_i is a cheater $\Longleftrightarrow$ δ_i is nonzero.

• At the pooling time, the cheaters submit their shares. It is assumed that the cheaters always submit invalid shares. The honest participants always submit their valid shares. We consider the vector $\alpha + \delta$. From the properties of α_δ^+ and α_δ^-, we can write that $\alpha + \delta = \alpha_\delta^- + \alpha_\delta^+ + \delta$. Thus the combiner obtains $\alpha + \delta$ that splits into two parts: α_δ^- – the part submitted by honest participants, and $\alpha_\delta^+ + \delta$ – the part submitted by the cheaters. The combiner (or recovery algorithm) returns an invalid multisecret $\mu^* = F(\alpha + \delta)$. Note that the cheaters always change their shares. We assume that there exists at least one cheater, in other words, δ is nonzero or $HW(\delta) > 0$.

• α_δ^+ determines valid shares held by the cheaters. The set $R(\delta, \alpha_\delta^+, \mu)$, or $\{x_\delta^- | F(x_\delta^- + \alpha_\delta^+) = \mu\}$, determines a collection of rows of $\mathcal{T}$ with the correct multisecret μ and valid shares held by the cheaters.

• The set $R(\delta, \alpha_\delta^+ + \delta, \mu^*)$, or $\{x_\delta^- | F(x_\delta^- + \alpha_\delta^+ + \delta) = \mu^*\}$, represents the view of the cheaters after getting back μ^* from the combiner.

In this work the *cheating* means the action of cheaters by submitting incorrect shares, and *successful cheating* means the case that the cheaters not only submit incorrect shares but also guess the correct secret.

The mapping F is called the *defining mapping* as it determines the multisecret sharing. The nonzero vector $\delta = (\delta_1, \ldots, \delta_n)$ is called a *cheating vector*, α is called an *original vector*. The value of $\rho_{\delta,\alpha} = \#(R(\delta, \alpha_\delta^+ + \delta, \mu^*) \cap R(\delta, \alpha_\delta^+, \mu))/\#R(\delta, \alpha_\delta^+ + \delta, \mu^*)$, expresses the probability of successful cheating with respect to δ and α, where $\#X$ denotes the number of elements in the set X. As an original vector α is always in $R(\delta, \alpha_\delta^+ + \delta, \mu^*) \cap R(\delta, \alpha_\delta^+, \mu)$, the probability of successful cheating always satisfies $\rho_{\delta,\alpha} > 0$. Clearly the number of cheaters is equal to $HW(\delta)$.

Theorem 1 *Given a multisecret sharing scheme with its defining mapping* F: $GF(p^t)^n \longrightarrow GF(p^t)^m$. *Let* $\delta \in GF(p^t)^n$ *with* $0 < HW(\delta) < n$ *be a cheating vector and* α *be an original vector in* $GF(p^t)^n$. *If* $\rho_{\delta,\alpha} < p^{-tm}$ *then there exists a vector* $\gamma \in GF(p^t)^n$ *such that* $\rho_{\delta,\gamma} > p^{-tm}$.

Proof Let $F(\alpha) = \mu$ and $F(\alpha + \delta) = \mu^*$. By definition, $R(\delta, \alpha_\delta^+, \mu) = \{x_\delta^- | F(x_\delta^- + \alpha_\delta^+) = \mu\}$ and $R(\delta, \alpha_\delta^+ + \delta, \mu^*) = \{x_\delta^- | F(x_\delta^- + \alpha_\delta^+ + \delta) = \mu^*\}$. We partition $R(\delta, \alpha_\delta^+ + \delta, \mu^*)$ into p^{tm} parts: $R(\delta, \alpha_\delta^+ + \delta, \mu^*) = \cup_{\lambda \in GF(p^t)^m} Q_\lambda$ where $Q_\lambda = R(\delta, \alpha_\delta^+ + \delta, \mu^*) \cap R(\delta, \alpha_\delta^+, \lambda + \mu)$. Clearly

$$\#R(\delta, \alpha_\delta^+ + \delta, \mu^*) = \sum_{\lambda \in GF(p^t)^m} \#Q_\lambda \tag{1}$$

Note that $R(\delta, \alpha_\delta^+ + \delta, \lambda^*) \cap R(\delta, \alpha_\delta^+, \lambda) = Q_0$. Therefore

$$\begin{aligned} \rho_{\delta,\alpha} &= \#(R(\delta, \alpha_\delta^+ + \delta, \mu^*) \cap R(\delta, \alpha_\delta^+, \mu)) \\ &\quad / \#R(\delta, \alpha_\delta^+ + \delta, \mu^*) \\ &= \#Q_0 / \#R(\delta, \alpha_\delta^+ + \delta, \mu^*) \end{aligned} \tag{2}$$

Since $\rho_{\delta,\alpha} < p^{-tm}$, from (2), $\#Q_0 / \#R(\delta, \alpha_\delta^+ + \delta, \mu^*) < p^{-tm}$. It follows that

$$\#Q_0 < p^{-tm} \#R(\delta, \alpha_\delta^+ + \delta, \mu^*) \tag{3}$$

From (1) and (3), we know that $\sum_{\lambda \in GF(p^t)^m, \lambda \neq 0} \#Q_\lambda > (1 - p^{-tm}) \#R(\delta, \alpha_\delta^+ + \delta, \mu^*)$. Thus there exists some $\lambda' \in GF(p^t)^m$ with $\lambda' \neq 0$ such that $\#Q_{\lambda'} > p^{-tm} \#R(\delta, \alpha_\delta^+ + \delta, \mu^*)$. By definition, $Q_{\lambda'} = \{x_\delta^- | F(x_\delta^- + \alpha_\delta^+ + \delta) = \mu^*, F(x_\delta^- + \alpha_\delta^+) = \lambda' + \mu\}$. Then there exists a vector $\beta_\delta^- \in Q_\nu$ and then $F(\beta_\delta^- + \alpha_\delta^+ + \delta) = \mu^*$, $F(\beta_\delta^- + \alpha_\delta^+) = \lambda' + \mu$. Set $\gamma = \beta_\delta^- + \alpha_\delta^+$. Thus $F(\gamma + \delta) = \mu^*$ and $F(\gamma) = \lambda' + \mu$. Clearly $\gamma_\delta^+ = \alpha_\delta^+$ and $\gamma_\delta^- = \beta_\delta^-$. Next we choose γ as an original vector. Due to $R(\delta, \gamma_\delta^+ + \delta, \mu^*) = \{x_\delta^- | F(x_\delta^- + \gamma_\delta^+ + \delta) = \mu^*\}$, $R(\delta, \gamma_\delta^+, \lambda' + \mu) = \{x_\delta^- | F(x_\delta^- + \gamma_\delta^+) = \lambda' + \mu\}$ and $\gamma_\delta^+ = \alpha_\delta^+$, we know that $R(\delta, \gamma_\delta^+ + \delta, \mu^*) \cap R(\delta, \gamma_\delta^+, \lambda' + \mu) = Q_{\lambda'}$ and $\rho_{\delta,\gamma} = \#(R(\delta, \gamma_\delta^+ + \delta, \mu^*) \cap R(\delta, \gamma_\delta^+, \lambda' + \mu)) / \#R(\delta, \gamma_\delta^+ + \delta, \mu^*) = \#Q_{\lambda'} / \#R(\delta, \gamma_\delta^+ + \delta, \mu^*) = \#Q_{\lambda'} / \#R(\delta, \alpha_\delta^+ + \delta, \mu^*) > p^{-tm}$.

Corollary 1 *Given a multisecret sharing scheme with its defining mapping* F: $GF(p^t)^n \longrightarrow GF(p^t)^m$. *Then* $\max\{\rho_{\delta,\alpha} | \alpha \in GF(p^t)^n\} \geq p^{-tm}$ *for any fixed nonzero vector* $\delta \in GF(p^t)^n$.

3.2 k-Cheating Immune Multisecret Sharing

Given a multisecret sharing with its defining mapping F on $GF(p^t)^n$. For a fixed nonzero $\delta \in GF(p^t)^n$, due to Theorem 1, it is desirable that $\rho_{\delta,\alpha} = p^{-tm}$ holds for every $\alpha \in GF(p^t)^n$. A multisecret sharing is said to be *k-cheating immune* if $\rho_{\delta,\alpha} = p^{-tm}$ holds for every $\delta \in GF(p^t)^n$ with $1 \leq HW(\delta) \leq k$ and every $\alpha \in GF(p^t)^n$.

Theorem 2 *Given a multisecret sharing with its defining mapping* F: $GF(p^t)^n \longrightarrow GF(p^t)^m$. *Then the multisecret sharing is k-cheating immune $\Longleftrightarrow$ for any integer l with $1 \leq l \leq k$, any $\delta \in GF(p^t)^n$ with $HW(\delta) = l$, any $\tau \preceq \delta$ and any $\mu, \nu \in GF(p^t)^m$, the following conditions hold simultaneously: (i) $\#R(\delta, \tau, \nu) = p^{t(n-l-m)}$, (ii) $\#(R(\delta, \tau, \nu) \cap R(\delta, \tau + \delta, \mu)) = p^{t(n-l-2m)}$.*

The proof is given in the Appendix.

Theorem 3 *Given a multisecret sharing with its defining mapping* F: $GF(p^t)^n \longrightarrow GF(p^t)^m$. *Then the following statements are equivalent: (i) the multisecret sharing is k-cheating immune, (ii) for any integer l with $1 \leq l \leq k$, any $\delta \in GF(p^t)^n$ with $HW(\delta) = l$, any $\tau \preceq \delta$ and any $\mu, \nu \in GF(p^t)^m$, we have $\#(R(\delta, \tau, \nu) \cap R(\delta, \tau + \delta, \mu)) = p^{t(n-l-2m)}$, (iii) for such l, δ, τ, μ and ν mentioned in (ii), the system of equations:* $\begin{cases} F(x_\delta^- + \tau + \delta) = \mu \\ F(x_\delta^- + \tau) = \nu \end{cases}$ *has precisely $p^{t(n-l-2m)}$ solutions on x_δ^-.*

Proof Clearly (ii) $\Longleftrightarrow$ (iii). Due to Theorem 2, (i) $\Longrightarrow$ (ii). To complete the proof, we only need prove that (ii) $\Longrightarrow$ (i). Assume that (ii) holds. Thus $\#(R(\delta, \tau, \nu) \cap R(\delta, \tau + \delta, \mu)) = p^{t(n-l-2m)}$ for every $\mu, \nu \in GF(p^t)^m$. Note that $R(\delta, \tau, \nu) = \cup_{\mu \in GF(p^t)^m} R(\delta, \tau, \nu) \cap R(\delta, \tau + \delta, \mu)$ and then $\#R(\delta, \tau, \nu) = \sum_{\mu \in GF(p^t)^m} \#(R(\delta, \tau, \nu) \cap R(\delta, \tau + \delta, \mu))$. This proves that $\#R(\delta, \tau, \nu) = p^{t(n-l-m)}$. Using Theorem 2, we have proved that (i) holds.

Corollary 2 *Given a multisecret sharing with its defining mapping* F: $GF(p^t)^n \longrightarrow GF(p^t)^m$. *If the multisecret sharing is k-cheating immune, then (i) $n \geq 2m + k$, (ii) F is regular.*

Proof Let $l = k$ in Theorem 2, we have $\#(R(\delta, \tau, \nu) \cap R(\delta, \tau + \delta, \mu)) = p^{t(n-k-2m)} \geq 1$. This proves that $n \geq 2m + k$. Again from Theorem 2 we have $\#R(\delta, \tau, \nu) = p^{t(n-k-m)}$, for any $\tau \preceq \delta$ and any $\nu \in GF(p^t)^m$. This means that for each fixed τ with $\tau \preceq \delta$, the mapping $F(x_\delta^- + \tau)$ is regular. Thus the mapping F is regular.

Due to Corollary 2, a k-cheating immune multisecret sharing, defined by a mapping F: $GF(p^t)^n \longrightarrow GF(p^t)^m$, exists only when $n \geq 2m + k$.

4 Simultaneous Recovery of Secrets with Cheaters using Valid and Invalid Shares

4.1 Probability of Successful Cheating

Given a mapping F: $GF(p^t)^n \longrightarrow GF(p^t)^m$. We introduce the following notations. As before we can see F as a table $\mathcal{T}$ with rows containing $(\delta, F(\delta))$. We also assume that the combiner returns all secrets (or the multisecret for short).

• Let $\alpha \in GF(p^t)^n$ be the sequence of shares held by the group $\mathcal{P} = \{P_1, \ldots, P_n\}$ of n participants and the secret $\mu = F(\alpha)$.

• The collection of cheaters is determined by the sequence $\delta = (\delta_1, \delta_2, \ldots, \delta_n)$ where P_i is a cheater $\Longleftrightarrow$ if $\delta_i \neq 0$.

• At the pooling time, the cheaters submit their shares. This time it is assumed that cheaters may submit a mixture of valid and invalid shares. The honest participants always submit their valid shares. The collection of cheaters who submit invalid shares is determined by the sequence $\tau = (\tau_1, \ldots, \tau_n)$ where $\tau_j = 0 \Longleftrightarrow P_j$ is honest or P_j is a cheater who submits a valid share, in other words, $\tau_j \neq 0 \Longleftrightarrow P_j$ is a cheater who submits an invalid share. Clearly $\tau \preceq \delta$. We assume that there exists at least one cheater who submits invalid share, in other words, we only consider the case that τ is nonzero or $HW(\tau) > 0$. We consider the vector $\alpha + \tau$. Due to the properties of operations α_δ^+ and α_δ^-, we can write $\alpha + \tau = \alpha_\delta^- + \alpha_\delta^+ + \tau$. The combiner obtains $\alpha + \tau$ that splits into two parts: α_δ^- – the part submitted by honest participants and $\alpha_\delta^+ + \tau$ the part submitted by cheaters. The combiner returns an invalid multisecret $\mu^* = F(\alpha + \tau)$.

• $R(\delta, \alpha_\delta^+ + \tau, \mu^*)$, or $\{x_\delta^- | F(x_\delta^- + \alpha_\delta^+ + \tau) = \mu^*\}$, where α_δ^+ determines valid shares held by the cheaters, represents the view of the cheater after getting back μ^* from the combiner.

• The set $R(\delta, \alpha_\delta^+, \mu)$, or $\{x_\delta^- | f(x_\delta^- + \alpha_\delta^+) = \mu\}$, determines a collection of rows of $\mathcal{T}$ with the correct multisecret μ and valid shares held by the cheaters.

As mentioned in Section 3, the cheating means the action of cheaters by submitting incorrect shares, and successful cheating means the case when cheaters are able to guess the correct secret.

In generalised model of cheating, τ is used to determine how to cheat while δ is only used to determine which participants are dishonest, therefore we can define δ as a $(0,1)$-vector in $GF(p^t)^n$. However, in basic model of cheating, δ is not only used to determine which participants are dishonest but also used to determine how to cheat, thus δ has a more general form.

The mapping F is called the *defining mapping* of multisecret sharing. We assume that the combiner returns multisecrets (all secrets). The nonzero vector $\delta = (\delta_1, \ldots, \delta_n)$ is called a *cheating vector*, the nonzero vector $\tau \preceq \delta$ is called an *active cheating vector*, α is called an *original vector*. The value of $\rho_{\delta,\tau,\alpha} = \#(R(\delta, \alpha_\delta^+ + \tau, \mu^*) \cap R(\delta, \alpha_\delta^+, \mu))/\#R(\delta, \alpha_\delta^+ + \tau, \mu^*)$ expresses the probability of successful cheating with respect to δ, τ and α. As an original vector α is always in $R(\delta, \alpha_\delta^+ + \tau, \mu^*) \cap R(\delta, \alpha_\delta^+, \mu)$, the probability of successful cheating always satisfies $\rho_{\delta,\tau,\alpha} > 0$. Clearly the number of cheaters is equal to $HW(\delta)$ and the number of active cheaters is equal to $HW(\tau)$. In particular, if $\tau = \delta$, we regain basic model of cheating.

4.2 Strictly k-cheating Immune Multisecret Sharing

By using the same arguments as in the proof of Theorem 1, we can state:

Theorem 4 *Given a multisecret sharing with its defining mapping F: $GF(p^t)^n \longrightarrow GF(p^t)^m$. Let $\delta \in GF(p^t)^n$; $0 < HW(\delta) < n$, be a cheating vector, let $\tau \preceq \delta$; $\tau \neq 0$, be an active cheating vector, and let $\alpha \in GF(p^t)^n$ be an original vector (representing valid shares). If $\rho_{\delta,\tau,\alpha} < p^{-tm}$ then there exists a vector $\gamma \in GF(p^t)^n$ such that $\rho_{\delta,\tau,\gamma} > p^{-tm}$.*

Corollary 3 *Given a multisecret sharing with its defining mapping F: $GF(p^t)^n \longrightarrow GF(p^t)^m$. Then $\max\{\rho_{\delta,\tau,\alpha} \mid \alpha \in GF(p^t)^n\} \geq p^{-tm}$ for any fixed δ and τ with $\tau \preceq \delta$ and $\tau \neq 0$.*

For the same reason mentioned in Section 3.2, we introduce the concept of k-cheating immunity. Given a secret sharing with its defining mapping F on $GF(p^t)^n$. Let k be an integer with $1 \leq k \leq n-1$. The secret sharing is said to be *strictly k-cheating immune* if the probability of successful cheating satisfies $\rho_{\delta,\tau,\alpha} = p^{-tm}$ for every $\delta \in GF(p^t)^n$ and any $\tau \preceq \delta$ with $1 \leq HW(\tau) \leq HW(\delta) \leq k$ and every $\alpha \in GF(p^t)^n$. The following theorem establishes a relationship between the two models of cheating immunity.

Theorem 5 *Given a multisecret sharing with its defining mapping F: $GF(p^t)^n \longrightarrow GF(p^t)^m$. Then the multisecret sharing is strictly k-cheating immune $\Longleftrightarrow$ for any integer r with $0 \leq r \leq k-1$, any subset $\{j_1, \ldots, j_r\}$ of $\{1, \ldots, n\}$ and any $a_1, \ldots, a_r \in GF(p^t)$, the mapping $F(x_1, \ldots, x_n)|_{x_{j_1}=a_1, \ldots, x_{j_r}=a_r}$, with the variables $x_{i_1}, \ldots, x_{i_{n-r}}$, where $\{i_1, \ldots, i_{n-r}\} \cup \{j_1, \ldots, j_r\} = \{1, \ldots, n\}$, is the defining mapping of a $(k-r)$-cheating immune secret sharing.*

Proof Assume that the multisecret sharing is strictly k-cheating immune (model in Section 4.1). Denote $F(x_1, \ldots, x_n)|_{x_{j_1}=a_1, \ldots, x_{j_r}=a_r}$ by G. Then G is a mapping: $GF(p^t)^{n-r} \longrightarrow GF(p^t)^m$. Comparing the model in Section 3.1 with the model in Section 4.1, we know that G is the defining mapping of $(k-r)$-cheating immune secret sharing (model in Section 3.1). This proves the necessity. By definition, we can prove the sufficiency by inverting the above reasoning.

5 Secret Sharing versus Multisecret Sharing

We regard $GF(p^{tm})$ as a simple extension of $GF(p^t)$ and then there exists an element $\epsilon \in GF(p^{tm})$ such that each element in $GF(p^{tm})$ can be uniquely expressed as $b_1 + b_2\epsilon + \cdots + b_m\epsilon^{m-1}$ where each $b_j \in GF(p^t)$. Let f be a function on $GF(p^{tm})^n$, i.e., a mapping:

$GF(p^{tm})^n \longrightarrow GF(p^{tm})$, and ψ be a nonzero linear mapping: $GF(p^{tm}) \longrightarrow GF(p^t)$. From f and ψ, we now define a mapping $F_{f,\psi}$: $GF(p^t)^n \longrightarrow GF(p^t)^m$ such that $F_{f,\psi}(a_1, \ldots a_n) = (b_1, \ldots, b_m)$, where each $a_j, b_i \in GF(p^t)$, $\Longleftrightarrow f(c_1, \ldots c_n) = c$, where $a_j = \psi(c_j)$, $j = 1, \ldots, n$, and $c = b_1 + b_2\epsilon + \cdots + b_m\epsilon^{m-1}$.

Theorem 6 *Given a secret sharing with its defining function f on $GF(p^{tm})^n$. Let ψ be a nonzero linear mapping from $GF(p^{tm})$ to $GF(p^t)$. If the secret sharing is k-cheating immune then the mapping $F_{f,\psi}$: $GF(p^t)^n \longrightarrow GF(p^t)^m$ is the defining mapping a k-cheating immune multisecret sharing.*

Proof Let δ be any vector in $GF(p^t)^n$ with $HW(\delta) = l$, where $1 \leq l \leq k$, and τ be any vector in $GF(p^t)^n$ with $\tau \preceq \delta$. Consider the system of equations:

$$\begin{cases} F_{f,\psi}(x_\delta^- + \tau + \delta) = (a_1, \ldots, a_m) \\ F_{f,\psi}(x_\delta^- + \tau) = (b_1, \ldots, b_m) \end{cases} \tag{4}$$

where each $a_j, b_j \in GF(p^t)$, and

$$\begin{cases} f(x_\delta^- + \tau + \delta) = a_1 + a_2\epsilon + \cdots + a_m\epsilon^{m-1} \\ f(x_\delta^- + \tau) = b_1 + b_2\epsilon + \cdots + b_m\epsilon^{m-1} \end{cases} \tag{5}$$

Due to Theorem 3, Equations (5) have precisely $p^{tm(n-l-2)}$ solutions. Note that for each element $a \in GF(p^t)$, there precisely exist $p^{t(m-1)}$ elements $c \in GF(p^{tm})$ such that $\psi(c) = a$. Therefore for each vector $(a_1, \ldots, a_{n-l}) \in GF(p^t)^{n-l}$, there precisely exist $p^{t(m-1)(n-l)}$ vectors $(c_1, \ldots, c_{n-l}) \in GF(p^{tm})^{n-l}$ such that $(\psi(c_1), \ldots, \psi(c_{n-l})) = (a_1, \ldots, a_{n-l})$. Summarising the above, we know that Equations (4) have precisely $p^{tm(n-l-2)}/p^{t(m-1)(n-l)} = p^{t(n-l-2m)}$ solutions. Due to Theorem 3, we have proved that the mapping $F_{f,\psi}$: $GF(p^t)^n \longrightarrow GF(p^t)^m$ is the defining mapping a k-cheating immune multisecret sharing.

Combining Theorems 5 and 6, we can prove the following statement.

Corollary 4 *Given secret sharing with its defining function f on $GF(p^{tm})^n$. Let ψ be a nonzero linear mapping from $GF(p^{tm})$ to $GF(p^t)$. If the secret sharing is strictly k-cheating immune then the mapping $F_{f,\psi}$: $GF(p^t)^n \longrightarrow GF(p^t)^m$ is the defining mapping of a strictly k-cheating immune multisecret sharing.*

The construction of a k-cheating immune secret sharing defined by a function has been studied [6]. Therefore applying Theorems 6 and 4, we can construct a k-cheating immune secret sharing defined by a mapping from a k-cheating immune secret sharing defined by a function. Using Corollary 4 and construction given in [6], we can obtain strictly cheating immune secret sharing defined by a mapping. The constructions will be shown in Examples 1 and 2.

Theorem 7 *Let F be a mapping: $GF(p^t)^n \longrightarrow GF(p^t)^m$. Write $F(x) = (f_1(x), \ldots, f_m(x))$ where each f_j is a function on $GF(p^t)^n$ and $x \in GF(p^t)^n$. Let s be an integer with $1 \leq s \leq m$ and $\{j_1, \ldots, j_s\} \subseteq \{1, \ldots, n\}$. Define a mapping H: $GF(p^t)^n \longrightarrow GF(p^t)^s$ such that $H(x) = (f_{j_1}(x), \ldots, f_{j_s}(x))$. If F is the defining mapping of a k-cheating immune multisecret sharing, so is H.*

Proof Without loss of generality, we only prove the theorem in the special case that $j_1 = 1, \ldots, j_s = s$. Consider the system of equations:

$$\begin{cases} H(x_\delta^- + \tau + \delta) = \omega \\ H(x_\delta^- + \tau) = \sigma \end{cases} \tag{6}$$

where $\omega, \sigma \in GF(p^t)^s$.

Since F is the defining mapping of a k-cheating immune multisecret sharing, due to Theorem 3, for any integer l with $1 \leq l \leq k$, any $\delta \in GF(p^t)^n$ with $HW(\delta) = l$, any $\tau \preceq \delta$ and any $\mu, \nu \in GF(p^t)^m$, the system of equations: $\begin{cases} F(x_\delta^- + \tau + \delta) = \mu \\ F(x_\delta^- + \tau) = \nu \end{cases}$ has precisely $p^{t(n-l-2m)}$ solutions on x_δ^-. On the other hand, there precisely exist $p^{t(m-s)}$ vectors $\mu \in GF(p^t)^m$ satisfying $\mu = (\omega, y)$ where $y \in GF(p^t)^{m-s}$ and there precisely exist $p^{t(m-s)}$ vectors $\nu \in GF(p^t)^m$ satisfying $\nu = (\sigma, z)$ where $z \in GF(p^t)^{m-s}$. It is easy to see that the system of equations (6) has precisely $p^{t(n-l-2m)} \cdot p^{2t(m-s)} = p^{t(n-l-2s)}$ solutions on x_δ^-. Applying Theorem 3 to H, we have proved that H is the defining mapping of a k-cheating immune multisecret sharing.

The above theorem can be rephrased to the following statement.

Theorem 8 *Let F be a mapping: $GF(p^t)^n \longrightarrow GF(p^t)^m$ such that $F(x) = (f_1(x), \ldots, f_m(x))$ where each f_j is a function on $GF(p^t)^n$ and $x \in GF(p^t)^n$. If F is the defining mapping of a k-cheating immune multisecret sharing then each f_j is the defining function of a k-cheating immune secret sharing.*

Applying Theorem 5 to Theorem 8, we obtain

Corollary 5 *Let F be a mapping: $GF(p^t)^n \longrightarrow GF(p^t)^m$ such that $F(x) = (f_1(x), \ldots, f_m(x))$ where each f_j is a function on $GF(p^t)^n$ and $x \in GF(p^t)^n$. If F is the defining mapping of a strictly k-cheating immune multisecret sharing then each f_j is the defining function of a strictly k-cheating immune sharing.*

Theorem 9 *Let F be a mapping: $GF(p^t)^n \longrightarrow GF(p^t)^m$ and B is a nonsingular $m \times m$ matrix over $GF(p^t)$. Define another mapping G: $GF(p^t)^n \longrightarrow GF(p^t)^m$ such that $G(x) = (F(x))B$. If F is the defining mapping of a k-cheating immune multisecret sharing, so is G.*

Proof For any integer l with $1 \leq l \leq k$, any $\delta \in GF(p^t)^n$ with $HW(\delta) = l$, any $\tau \preceq \delta$ and any $\mu, \nu \in GF(p^t)^m$, consider the system of equations: $\begin{cases} G(x_\delta^- + \tau + \delta) = \mu \\ G(x_\delta^- + \tau) = \nu \end{cases}$ that is equivalent to

$$\begin{cases} F(x_\delta^- + \tau + \delta) = \mu B^{-1} \\ F(x_\delta^- + \tau) = \nu B^{-1} \end{cases} \tag{7}$$

Since F is the defining mapping of a k-cheating immune multisecret sharing, due to Theorem 3, (7) has precisely $p^{t(n-l-2m)}$ solutions on x_δ^-. Therefore G has precisely $p^{t(n-l-2m)}$ solutions on x_δ^-. Again using Theorem 3, G is also the defining mapping of a k-cheating immune multisecret sharing.

Theorem 10 *Let F be a mapping: $GF(p^t)^n \longrightarrow GF(p^t)^m$ such that $F(x) = (f_1(x), \ldots, f_m(x))$ where each f_j is a function on $GF(p^t)^n$ and $x \in GF(p^t)^n$. If F is the defining mapping of a k-cheating immune multisecret sharing then any nonzero linear combination of $f_1, \ldots, f_m$, i.e., $b_1 f_1 + \cdots + b_m f_m$ where each $b_j \in GF(p^t)$ and $(b_1, \ldots, b_m) \neq (0, \ldots, 0)$, is the defining function of a k-cheating immune secret sharing.*

Proof Let $(b_1, \ldots, b_m)$ be a nonzero vector in $GF(p^t)^m$. Let B be a nonsingular $m \times m$ matrix over $GF(p^t)$, whose first column is $(b_1, \ldots, b_m)^T$ where X^T denote the transpose of the matrix X. Set $G(x) = (g_1(x), \ldots, g_m(x)) = (f_1(x), \ldots, f_m(x))B$. Using Theorem 9, we know that G is the defining mapping of a k-cheating immune multisecret sharing. Applying Theorem 8 to G, we know that g_1 is the defining function of a k-cheating immune secret sharing. Since $g_1 = b_1 f_1 + \cdots + b_m f_m$, we have proved that $b_1 f_1 + \cdots + b_m f_m$ is the defining function of a k-cheating immune secret sharing.

Applying Theorem 5 to Theorem 10, we obtain

Corollary 6 *Let F be a mapping: $GF(p^t)^n \longrightarrow GF(p^t)^m$ such that $F(x) = (f_1(x), \ldots, f_m(x))$ where each f_j is a function on $GF(p^t)^n$ and $x \in GF(p^t)^n$. If F is the defining mapping of a strictly k-cheating immune multisecret sharing then any nonzero linear combination of $f_1, \ldots, f_m$ is the defining function of a strictly k-cheating immune secret sharing.*

Due to Theorem 10 (Corollary 6), we have $f_1, \ldots, f_m$ that are m coordinate functions of the mapping F. Denote the set of all the nonzero linear combinations of $f_1, \ldots, f_m$, by $\Omega = \{g_1, \ldots, g_{p^{tm}-1}\}$. Then each g_j is the defining function of a (strictly) k-cheating immune secret sharing. Clearly $g_j \pm g_i \in \Omega$ for $j \neq i$, and thus $g_j \pm g_i$ is the defining function of a (strictly) k-cheating immune secret sharing. Due to Corollary 2, $g_j \pm g_i$ is balanced. This means that any f_j does not give any information on any other f_i with $i \neq j$ as balance means unbiased for every element in $GF(p^t)$. Note that $\Omega \cup \{0\}$, where 0 denotes the zero function on $GF(p^t)^n$ form an m-dimensional linear space over $GF(p^t)$.

6 Constructions

The following two examples indicate how to construct a multisecret sharing mentioned in Theorem 10 and Corollary 6.

Example 1 *Define a function χ_{2k+1} on $GF(p^{tm})^{2k+1}$ by $\chi_{2k+1}(x_1, \ldots, x_{2k+1}) = x_1x_2 + x_2x_3 + \cdots + x_{2k}x_{2k+1} + x_{2k+1}x_1$ and then define a function χ_{4k+2} on $GF(p^{tm})^{4k+2}$ by $\chi_{4k+2}(x_1, \ldots, x_{4k+2}) = \chi_{2k+1}(x_1, \ldots, x_{2k+1}) + \chi_{2k+1}(x_{2k+2}, \ldots, x_{4k+2})$.*

Let k and s be positive integers with $s \geq k+1$, $n_1, \ldots, n_s = 4k+1$ or $4k+2$, and $n = n_1 + \cdots + n_s$. Define a function on $GF(p^{tm})^n$ such as $f(x) = \chi_{n_1}(y) + \cdots + \chi_{n_s}(z)$ where $x = (y, \ldots, z)$, $y \in GF(p^{tm})^{n_1}, \ldots, z \in GF(p^{tm})^{n_s}$, and $\chi_{n_1}, \ldots, \chi_{n_s}$ have disjoint variables mutually. From [6], the secret sharing with the defining function f is k-cheating immune. Let ψ be a nonzero linear mapping: $GF(p^{tm}) \longrightarrow GF(p^t)$. Due to Theorem 6, the mapping $F_{f,\psi}$: $GF(p^t)^n \longrightarrow GF(p^t)^m$ is the defining mapping a k-cheating immune multisecret sharing.

Example 2 *Let $\lambda_{n,p}$ be a function on $GF(p^{tm})^n$ ($n \geq 2p^2 + p$) defined by $\lambda_{n,p}(x_1, \ldots, x_n) = x_1 + \sum_{j=1}^{n}(x_j x_{[j+1]_{(n)}} + x_j x_{[j+2]_{(n)}} + \cdots + x_j x_{[j+p]_{(n)}})$ where $[i]_{(n)}$ denotes the integer j such that $1 \leq j \leq n$ and $j \equiv i \bmod n$ (we replace i by $[i]_{(n)}$ as i is possibly greater than n). Let s be an integer with $s \geq 2p$, $n_1, \ldots, n_s = 2p^2 + p$ or $2p^2 + p + 1$, and $n = n_1 + \cdots + n_s$. Define a function on $GF(p^{tm})^n$ such as $f(x) = \lambda_{n_1,p}(y) + \cdots + \lambda_{n_s,p}(z)$ where $x = (y, \ldots, z)$, $y \in GF(p^{tm})^{n_1}, \ldots, z \in GF(p^{tm})^{n_s}$, and $\lambda_{n_i,p}, \ldots, \lambda_{n_j,p}$ have disjoint variables if $i \neq j$. From [6], the secret sharing with the defining function f is strictly p-cheating immune. Let ψ be a nonzero linear mapping: $GF(p^{tm}) \longrightarrow GF(p^t)$. Due to Corollary 4, the mapping $F_{f,\psi}$: $GF(p^t)^n \longrightarrow GF(p^t)^m$ is the defining mapping a strictly p-cheating immune multisecret sharing.*

7 Sequential Recovery of Secrets

Given a multisecret sharing with its defining mapping F: $GF(p^t)^n \longrightarrow GF(p^t)^m$ such that $F = (f_1, \ldots, f_m)$ where each f_j is a function on $GF(p^t)^n$. In the multisecret sharing schemes mentioned in Sections 3 and 4, we stipulate that the multisecret as a vector $(b_1, \ldots, b_m)$ that determines m secrets. We assume that those secrets are recovered simultaneously.

In this section, we consider a scenario where the combiner will recover single secrets (instead of the multisecret) in a some order (perhaps imposed by the participants) and return the recovered secrets to active participants. We will study two basic cheating strategies

- cheaters use the same cheating vector and the same original vector for all recoveries – this case is equivalent to the simultaneous recovery of secrets,
- cheaters modify their cheating vectors depending on the returned secrets.

Assume that the participants sequentially perform m secret sharing schemes with defining functions $f_1, \ldots, f_m$ by taking cheating vectors $\delta_1, \ldots, \delta_m \in GF(p^t)^n$ and original vectors. $\beta_1, \ldots, \beta_m \in GF(p^t)^n$ and original vectors.

Since the secret of each secret sharing defined by f_j is independent to the secret of secret sharing defined by f_i if $i \neq j$, the probability of successful cheating is identical with $\rho_{\delta_1,\beta_1} \cdots \rho_{\delta_m,\beta_m}$, where ρ_{δ_i,β_i} denotes the probability of successful cheating of the secret sharing defined by f_i with respect to the cheating vector δ_i and the original vector β_i. We notice that each ρ_{δ_i,β_i} can be calculated by the definition of probability of successful cheating. Obviously the probability of successful cheating of sequentially recovery is invariable under a permutation on the order of participants. In particular, $F = (f_1, \ldots, f_m)$ is the mapping of a k-th immune secret sharing, then using Theorem 10, we conclude that $\rho_{\delta_1,\beta_1} = \cdots = \rho_{\delta_m,\beta_m} = p^{-t}$. Therefore the probability of successful cheating is identical with p^{-tm}.

As for the model that cheaters submit a mixture of valid and invalid shares, using the same arguments, we conclude that the probability of successful cheating is identical with $\rho_{\delta_1,\beta_1} \cdots \rho_{\delta_m,\beta_m}$. In particular, $F = (f_1, \ldots, f_m)$ is the mapping of a k-th immune secret sharing, then using Theorem 10, we conclude that $\rho_{\delta_1,\beta_1} = \cdots = \rho_{\delta_m,\beta_m} = p^{-t}$. Therefore the probability of successful cheating is identical with p^{-tm}.

8 Conclusions

We define a multisecret sharing by its defining mapping F: $GF(p^t)^n \longrightarrow GF(p^t)^m$. For n participants, each vector $\alpha \in GF(p^t)^n$ is a share and the vector $F(\alpha) \in GF(p^t)^m$ is the multisecret corresponding to the share α. It has been proven that the probability of recovery of correct multisecret by cheaters is can be made as small as p^{-tm}. Clearly, cheaters who are interested in getting a single secret may get it with the probability no smaller than p^{-t}. In a sense each recovered invalid secret provides no information about the valid secret but also gives no help in gaining information about other secrets.

We have investigated two models of k-cheating immune secret sharing defined by a mapping. A relationship between defining mappings and functions has been examined and constructions of k-cheating immune multisecret sharing have been given. We have shown how k-cheating immune multisecret sharing relates to a linear space defined over k-cheating immune secret sharing schemes. We have also demonstrated that k-cheating immunity guarantees that multisecret sharing can be used for simultaneous and sequential recovery without any impact on the probability of guessing of valid secrets by cheaters.

Acknowledgement

The work was partially supported by Australian Research Council grant A00103078.

References

[1] C. Blundo, A. De Santis, and U. Vaccaro (1993), Efficient sharing of many secrets, In K.W. Wagner P. Enjalbert, A. Finkel, editor, *Proceedings of STACS93, 10th Symposium on Theoretical Aspects of Computer Science*, Springer, LNCS No. 665, pp. 692–703.

[2] Carlo Blundo, Alfredo De Santis, Giovanni Di Crescenzo, Antonio Giorgio Gaggia (1984), and Ugo Vaccaro, Multi-secret sharing schemes, In *Advances in Cryptology - CRYPTO'94*, LNCS No. 839, Springer-Verlag, pp. 150–163.

[3] Wen-Ai Jackson, Keith M. Martin, and Christine M. O'Keefe (1994), Multisecret threshold schemes, In Douglas R. Stinson, editor, *CRYPTO93*, Springer, LNCS No. 773, pp. 126–135.

[4] Wen-Ai Jackson, Keith M. Martin, and Christine M. O'Keefe (1994), On sharing many secrets, In J. Pieprzyk and R. Safavi-Naini, editors, *ASIACRYPT'94*, Springer, LNCS No. 917, pp. 42–54.

[5] E.D. Karnin, J.W. Greene, and M.E. Hellman (1983), On secret sharing systems, *IEEE Transactions on Information Theory*, IT-29: pp. 35–41.

[6] J. Pieprzyk and X. M. Zhang (2001), Cheating prevention in immune secret sharing over $GF(p^t)$, In *Indocrypt 2001*, LNCS No. 2247, Springer-Verlag, pp. 79–91.

[7] A. Shamir (1979), How to share a secret, *Communications of the ACM*, 22: pp. 612–613.

[8] Martin Tompa and Heather Woll (1988), How to share a secret with cheaters, *Journal of Cryptology* 1: pp. 133–138.

Appendix: Proof of Theorem 2

Proof Assume that the secret sharing is k-cheating immune. Choose δ as a cheating vector and any vector $\alpha \in GF(p^t)^n$ as an original vector. Due to Lemma 1, there exist $\mu', \nu' \in GF(p^t)^m$ such that $R(\delta, \alpha_\delta^+ + \delta, \mu') \neq \emptyset$ and $R(\delta, \alpha_\delta^+, \nu') \neq \emptyset$. Note that $R(\delta, \alpha_\delta^+ + \delta, \mu')$ can be partitioned into p^t parts:

$$R(\delta, \alpha_\delta^+ + \delta, \mu') = \bigcup_{\nu \in GF(p^t)^m} R(\delta, \alpha_\delta^+ + \delta, \mu') \cap R(\delta, \alpha_\delta^+, \nu) \quad (8)$$

Assume that $R(\delta, \alpha_\delta^+ + \delta, \mu') \cap R(\delta, \alpha_\delta^+, \nu) \neq \emptyset$ for some $\nu \in GF(p^t)^m$. Then there exists a vector $\beta_\delta^- \in R(\delta, \alpha_\delta^+ + \delta, \mu') \cap R(\delta, \alpha_\delta^+, \nu)$. Set $\gamma = \beta_\delta^- + \alpha_\delta^+$. Since the secret sharing is k-cheating immune, $\#(R(\delta, \gamma_\delta^+ + \delta, \mu') \cap$

$R(\delta, \gamma_\delta^+, \nu))/\#R(\delta, \gamma_\delta^+ + \delta, \mu') = \rho_{\delta,\gamma} = p^{-tm}$, where $\gamma_\delta^+ = \alpha_\delta^+$. Thus

$$\#R(\delta, \alpha_\delta^+ + \delta, \mu') = p^{tm}\#(R(\delta, \alpha_\delta^+ + \delta, \mu') \cap R(\delta, \alpha_\delta^+, \nu)) \quad (9)$$

whenever $R(\delta, \alpha_\delta^+ + \delta, \mu') \cap R(\delta, \alpha_\delta^+, \nu) \neq \emptyset$. From (8),

$$\#R(\delta, \alpha_\delta^+ + \delta, \mu') = \sum_{\nu \in GF(p^t)^m} \#(R(\delta, \alpha_\delta^+ + \delta, \mu') \cap R(\delta, \alpha_\delta^+, \nu)) \quad (10)$$

Combing (9) and (10), we know that $R(\delta, \alpha_\delta^+ + \delta, \mu') \cap R(\delta, \alpha_\delta^+, \nu) \neq \emptyset$ for every $\nu \in GF(p^t)^m$ and thus

$$\#(R(\delta, \alpha_\delta^+ + \delta, \mu') \cap R(\delta, \alpha_\delta^+, \nu)) = p^{-tm}\#R(\delta, \alpha_\delta^+ + \delta, \mu') \quad (11)$$

for every $\nu \in GF(p^t)^m$. Replacing α, δ, by $\alpha+\delta, (p-1)\delta$ respectively, due to the same arguments for (11), we have

$$\#(R((p-1)\delta, \alpha_\delta^+ + p\delta, \nu') \cap R((p-1)\delta, \alpha_\delta^+ + \delta, \mu)) = p^{-tm}\#R((p-1)\delta, \alpha_\delta^+ + p\delta, \nu')$$

for every $\mu \in GF(p^t)^m$. Since the characteristic of the finite field $GF(p^t)$ is p, $pe = 0$ for every $e \in GF(p^t)$. It follows that $\#(R((p-1)\delta, \alpha_\delta^+, \nu') \cap R((p-1)\delta, \alpha_\delta^+ + \delta, \mu')) = p^{-tm}\#R((p-1)\delta, \alpha_\delta^+, \nu')$ for every $\mu \in GF(p^t)^m$. Using Lemma 1, we obtain

$$\#(R(\delta, \alpha_\delta^+, \nu') \cap R(\delta, \alpha_\delta^+ + \delta, \mu)) = p^{-tm}\#R(\delta, \alpha_\delta^+, \nu') \quad (12)$$

for every $\mu \in GF(p^t)^m$. Recall that $R(\delta, \alpha_\delta^+ + \delta, \mu') \neq \emptyset$ and $R(\delta, \alpha_\delta^+, \nu') \neq \emptyset$. Therefore (11) and (12) imply that $R(\delta, \alpha_\delta^+, \nu) \neq \emptyset$ and $R(\delta, \alpha_\delta^+ + \delta, \mu) \neq \emptyset$ for every $\mu, \nu \in GF(p^t)^m$. Due to the same reasoning for (11) and (12), we have

$$\#(R(\delta, \alpha_\delta^+ + \delta, \mu) \cap R(\delta, \alpha_\delta^+, \nu)) = p^{-tm}\#R(\delta, \alpha_\delta^+ + \delta, \mu) \quad (13)$$

and

$$\#(R(\delta, \alpha_\delta^+, \nu) \cap R(\delta, \alpha_\delta^+ + \delta, \mu)) = p^{-tm}\#R(\delta, \alpha_\delta^+, \nu) \quad (14)$$

for every $\mu, \nu \in GF(p^t)^m$. Comparing (14) with (13), we conclude that $\#R(\delta, \alpha_\delta^+ + \delta, \mu) = \#R(\delta, \alpha_\delta^+, \nu)$ for every $\mu, \nu \in GF(p^t)^m$. Therefore both $\#R(\delta, \alpha_\delta^+ + \delta, \mu)$ and $\#R(\delta, \alpha_\delta^+, \nu)$ are constant. Note that $\sum_{\nu \in GF(p^t)^m} \#R(\delta, \alpha_\delta^+, \nu) = p^{t(n-l)}$. We have proved that

$$\#R(\delta, \alpha_\delta^+, \nu) = p^{t(n-l-m)} \quad (15)$$

for any $\nu \in GF(p^t)^m$. From (15) and (14), we have proved that

$$\#(R(\delta, \alpha_\delta^+ + \delta, \mu) \cap R(\delta, \alpha_\delta^+, \nu)) = p^{t(n-l-2m)} \quad (16)$$

for every $\mu, \nu \in GF(p^t)^m$. For any $\tau \preceq \delta$, choose $\alpha \in GF(p^t)^n$ such that $\alpha_\delta^+ = \tau$. Due to (15) and (16), both conditions (i) and (ii) hold.

Conversely assume the defining mapping F satisfies conditions (i) and (ii). Choose any $\delta \in GF(p^t)^n$ with $HW(\delta) = l$, where $1 \leq l \leq k$, as a cheating vector and any α as an original vector. Set $F(\alpha) = \mu$ and $F(\alpha + \delta) = \mu^*$. By definition, $\rho_{\delta,\alpha} = \#(R(\delta, \alpha_\delta^+ + \delta, \mu^*) \cap R(\delta, \alpha_\delta^+, \mu))/\#R(\delta, \alpha_\delta^+ + \delta, \mu^*)$. Due to conditions (i) and (ii), $\rho_{\delta,\alpha} = p^{-tm}$. Thus we have proved that the secret sharing is k-cheating immune.

Digital Semipublic Watermarking

Valery Korzhik
Telecommunication Security Department
State University of Telecommunications
St. Petersburg, Russia
E-mail: ibts@ns.lanck.ru

Guillermo Morales-Luna[1]
Computer Science Section
CINVESTAV-IPN, Mexico
E-mail: gmorales@cs.cinvestav.mx

Dmitry Marakov and Irina Marakova
Institute of Radio Electronics and Telecommunication Systems
Odessa National Polytechnic University, Ukraine
E-mail: n0mad@all.odessa.ua

Keywords: information security, copyright protection, information hiding, watermarking

Received: June 13, 2002

The theory still remains uncompleted. There have been some advances in asymptotic theory concerning capacity of WM as well as in some practical proposals, mainly where WM is applied without any theory. Most of times, references are made to public watermarking, in which the watermark detector does not require any secret key at all. In this paper, our contribution is a WM that we name semipublic watermarking. *This is a scenario in which any ordinary user is enabled to detect WM without any secret key, if the message is not subject of attack trying to remove its WM. Moreover, at any time, the owner of WM is able to detect WM even after such an attack. We propose two methods of semipublic WM and we calculate the probabilities P_m, of missing WM (when WM is present, indeed), and P_{fa} of WM* false alarm *(when WM is absent but its presence is supposed). The first method guarantees that both P_m and P_{fa} converge to zero as the number N of WM elements increases, only if the distortion constraints are very strong. In second method also P_m, P_{fa} converge to zero as N increases, for any distortion constraints, but it is required the so called* property of quadratic compensation*: In order to compensate an m times increase of the distortion constraint "signal-to-noise ratio", the number at WM elements should increase m^2 times. Although we consider just additive noise attacks, our results are useful since they provide lower bounds for WM reliability for any optimal attack with given distortion constraints.*

1 Introduction

Information hiding (IH) is a rather new area of *information security*. The goal of an IH designer is to hide even the fact that messages do exist in some ostensible harmless host data (the so called *cover message* (CM)). *Watermarking* (WM) is one of IH applications: The CM should be combined with some other information, like owner identification, for instance. Then identification code is permanently embedded in the data and should remain present within the data after any transformations initiated by the attacker attempting to remove the WM (or to disable it of being detected by the owner) keeping an acceptable quality of CM.

We consider in the current paper only one type of attack - additive noise attack. The reason of such a hard restriction is the opportunity to estimate theoretical lower bounds for the probabilities of WM-*missing* and WM-*false alarm* as functions of the WM system main parameters. An additional point of consideration into this model is the fact that a lower bound can provide the most optimistic results regarding the WM application in comparison with other types of more sophisticated attacks.

Two main types of WM are known [1, 2, 4, 6, 7]. In the *private* version both the encoder and the decoder use *secret key*. In the *public* (*blind*) version no suplementary information at all is available to the decoder. Since the term *semiprivate* has been introduced already for another scenario we propose the term *semipublic* for this version of WM. In this case, it is assumed that each user is able to extract WM out of the watermarked message without any secret key, provided the absence of an attacker trying to re-

move WM. At the same time, the WM designer, possessing the secret key, is able to detect WM even after an attack if it satisfies some given distortion constraints.

2 Performance Evaluations of Semipublic WM

2.1 Semipublic WM Based on Double Sequence

In this case the *stegomessage* (SM) (or in other words - the watermarked message) is formed as

$$S(n) = C(n) + w(n), \qquad n = 1, 2, \ldots, N \tag{1}$$

where $C(n)$ is the cover message, $w(n) = \Delta(n)e(n)$, $\Delta(n)$ is a nonnegative-real valued sequence, $e(n)$ is a sequence of signs $+1, -1$ and N is the number of WM elements, e.g. pixels of image CM.

The key point of our approach is to consider $\Delta(n)$ and $e(n)$ as *randomly chosen* sequences (although they should be fixed later for a particular system). Different WM designers can select their WM sequences randomly. We are going to estimate the *average* WM reliability over all possible choices.

We consider only the case when CM is not used by the WM-encoder to form $\Delta(n)$ and $e(n)$. From the decoding process point of view, we assume that it is impossible for any ordinary user to use the CM (since otherwise the WM can be removed trivially) but it is possible for the WM owner to use the CM in order to detect WM after an attack. We consider just the particular case of WM when each SM either contains or does not contain WM. Hence any decoder of WM has to decide just one of two possibilities: Is there or not a WM in the presented SM? (*Zero-bit watermark system* [3]).

In the semipublic version of WM, the sequence $e(n)$ is assumed to be known for every user, whereas the sequence $\Delta(n)$ is kept secret. The sequence $e(n)$ takes independent and equally distributed samples of signs ± 1 and $\Delta(n)$ takes also independent and uniformly distributed nonnegative samples within an interval $(0, D)$, where $D > 0$ is a fixed positive number. These sequences are statistically mutually independent. The cover message $C(n)$ is described by a random discrete zero-mean process with variance σ_C^2. The *distortion constraint* after watermarking is given as a *signal-to-noise ratio*

$$\eta_w = \frac{\text{var}\,(C(n))}{\text{var}\,(\Delta(n)e(n))} = \frac{3\sigma_C^2}{D^2} \tag{2}$$

We propose for each ordinary user to detect WM as follows. Calculate the value

$$\Lambda = \sum_{n=1}^{N} S(n)e(n) \tag{3}$$

and compare it with some given threshold λ. If $\Lambda \geq \lambda$, then a WM has been detected, otherwise it has not. (It is easy to show that such detector is optimum if $C(n)$ is a Gaussian sequence with independent samples.)

Let us derive the formulas for the probability of WM-*missing* (P_m) (when detector is unable to find WM in the watermarked SM) and the probability of WM-*false alarm* (P_{fa}) (when detector claims the presence of WM, while in fact it is absent in that message at all).

If WM is present in SM then we have

$$\Lambda = \sum_{n=1}^{N} \left(C(n) + \Delta(n)e(n)\right) e(n) =: \Lambda_1 \tag{4}$$

and if it is not present in SM then

$$\Lambda = \sum_{n=1}^{N} C(n)e(n) =: \Lambda_0 \tag{5}$$

Since $\Delta(n)$ and $e(n)$ are independent identically distributed (i.i.d.) random sequences we get for large N as a consequence of the *Central Limit Theorem* [9] that both Λ_0 and Λ_1 are Gaussian random variables independent on the $C(n)$-probability distribution (this means that the detector based on the correlation sum Λ is indeed an asymptotically *robust* detector). Using the fact that Λ_0 and Λ_1 are Gaussian variables we can derive quite easily the formulas for the sought probabilities P_m and P_{fa}

$$P_m = 1 - Q\left(\frac{\lambda - E(\Lambda_1)}{\sqrt{\text{var}\,\Lambda_1}}\right) \tag{6}$$

$$P_{fa} = Q\left(\frac{\lambda - E(\Lambda_0)}{\sqrt{\text{var}\,\Lambda_0}}\right) \tag{7}$$

where

$$Q(x) = \frac{1}{\sqrt{2\pi}} \int_x^{+\infty} e^{-\frac{t^2}{2}}\, dt.$$

It is easy to see that

$$E(\Lambda_0) = 0 \quad , \quad E(\Lambda_1) = N \cdot E(\Delta(n)) = N\frac{D}{2}$$
$$\text{var}\,\Lambda_0 = N \cdot \sigma_C^2 = \text{var}\,\Lambda_1 \tag{8}$$

Substituting (8) into (6) and (7), and taking into account the relation (2) we get

$$P_m = 1 - Q\left((\lambda' - 1)\sqrt{\frac{3N}{4\eta_w}}\right) \tag{9}$$

$$P_{fa} = Q\left(\lambda'\sqrt{\frac{3N}{4\eta_w}}\right) \tag{10}$$

where $\lambda' = \frac{2\lambda}{N\Delta}$. Comparing formulas (9), (10) with the formulas for P_m and P_{fa} corresponding to the case of private WM in which CM is unknown by both encoder and decoder [5], we can find just medium *degradation coefficient* by increasing the WM length N.

Now, let us consider an additive noise attack against WM detection, performed by ordinary users. The decision about the presence or absence of WM is taken after comparing with a given threshold λ, the value

$$\Lambda' = \sum_{n=1}^{N} S'(n)e(n) \tag{11}$$

where $S'(n) = S(n) + \epsilon(n)$, for an *attacking additive noise* $\epsilon(n)$. Since the attacker knows the sequence $e(n)$, for $n = 1, \ldots, N$, (as indeed does any other ordinary user) he is able to detect the presence or absence of WM in the CM with a high reliability. Hence he can create an additive noise sequence $\epsilon(n)$, for $n = 1, \ldots, N$, as

$$\epsilon(n) = \begin{cases} \sigma e(n) & \text{if WM is absent} \\ 0 & \text{otherwise} \end{cases}$$

Thus, the following expectations of Λ' result, for ordinary users trying to detect WM after the attack:

$$E(\Lambda') = \begin{cases} E(\Lambda'_0) = N \cdot \sigma & \text{if WM is absent} \\ E(\Lambda'_1) = \dfrac{N \cdot D}{2} & \text{otherwise} \end{cases} \tag{12}$$

Using the distortion constraint after attack

$$\eta_a = \frac{\operatorname{var} C(n)}{\operatorname{var} \epsilon(n)} = \frac{\sigma_C^2}{\sigma^2}$$

we get

$$\sigma = \sqrt{\frac{\sigma_C^2}{\eta_a}} \tag{13}$$

On the other hand, the distortion constraint after watermarking (2) gives the relation

$$D = \sqrt{\frac{3\sigma_C^2}{\eta_w}} \tag{14}$$

We can replace η_a in (13) by η_w since the attack occurs only in the absence of WM. This allows to represent the equations (12) as follows

$$E(\Lambda'_0) = N\sigma = N\sqrt{\frac{\sigma_C^2}{\eta_w}} \tag{15}$$

$$E(\Lambda'_1) = N\frac{D}{2} = N\sqrt{\frac{3\sigma_C^2}{4\eta_w}} \tag{16}$$

Since (15) is always larger than (16), the attacker can cause false alarm of WM with a high probability for any threshold λ chosen by ordinary users. This fact *compromises* completely the WM system as a *public* one but in our case it is a *semipublic* WM system where WM detection *after attack* is just possible for the WM owner since he is the only person knowing the secret sequence $\Delta(n)$.

Let us consider the case when the detector (as the WM owner in his role of detection) knows both sequences $e(n)$ and $\Delta(n)$. Then he compares with a given threshold λ the value

$$\Lambda'' = \sum_{n=1}^{N} S'(n)e(n)\Delta(n) \tag{17}$$

Since the attacker knows with a high probability whether WM is presented or not in SM he can create the following optimal noises:

$$\epsilon(n) = \begin{cases} \epsilon_0(n) := \sigma e(n) & \text{if WM is absent} \\ \epsilon_1(n) := -\Delta'(n)e(n) & \text{otherwise} \end{cases} \tag{18}$$

where σ is a nonnegative parameter, $\Delta'(n)$ is a nonnegative i. i. d. random sequence, uniform within an interval $(0, D')$. In the case of WM absence, we get from (17) and the first case in (18)

$$\begin{aligned} E(\Lambda'') &= E(\Lambda''_0) \\ &= \sigma N E(\Delta(n)) \\ &= \frac{\sigma N D}{2} \\ &= N\frac{\sigma_C^2}{\eta_w}\sqrt{\frac{3}{4}} \end{aligned} \tag{19}$$

Otherwise, if WM is present we get from from (17) and the second case in (18)

$$\begin{aligned} E(\Lambda'') &= E(\Lambda''_1) \\ &= N\left[E(\Delta(n)^2) - E(\Delta(n))E(\Delta'(n))\right] \\ &= N\left(\frac{D^2}{3} - \frac{DD'}{4}\right) \\ &= N\left(\frac{\sigma_C^2}{\eta_w} - \sqrt{\frac{3\sigma_C^2}{16}}D'\right) \end{aligned} \tag{20}$$

A comparison between (19) and (20) shows that $E(\Lambda''_0) < E(\Lambda''_1)$ (and as a consequence that WM can be detected by the WM owner after an attack for large N) if and only if D' is very small. It means in turn that such semipublic WM system works fine whenever the distortion constraint after the attack is very strong, that is $\eta_a \approx \eta_w$, where η_a is the signal-to-noise ratio after the attack, and it can be determined as follows

$$\eta_a = \frac{\operatorname{var} C(n)}{\operatorname{var}(w(n) + \epsilon(n))} \tag{21}$$

Due to the fact that condition $\eta_a \approx \eta_w$ is not interesting enough in practice we throw away this method and consider instead in the next section another semipublic WM system based on a periodic sequence.

2.2 Semipublic WM based on a Periodic Sequence

Let us define the SM as follows

$$S(n) = C(n) + w(n), \qquad n = 1, 2, \ldots, 2N_0 \tag{22}$$

where $w(n) = \alpha\pi(n)$ and $N_0 = \frac{N}{2}$. The sequence $\pi(n)$ is an i.i.d. random sequence whose entries are signs ± 1, periodic with two periods of length N_0 each. (In practice it is not necessary to use all N elements. We can take $N_0 < \frac{N}{2}$ and select elements of both periods within N elements in such a way to distinguish the WM of different owners. Another reason for do not selecting consecutive elements in each period of $\pi(n)$ is to provide an interleaving of the CM elements).

If a periodic WM has been used in the SM but the sequence $\pi(n)$ is the secret sequence of the WM owner then each ordinary user can calculate the value

$$\Lambda = \sum_{n=1}^{N_0} S(n)S\left(n + N_0\right) \tag{23}$$

and compare it with a given threshold λ. Here, we are assuming that for the semiprivate WM there has not been an attack on the SM in the case of WM detection by ordinary users, hence $S'(n) = S(n)$.

Then in the case of WM presence

$$\begin{aligned}\Lambda' &= \Lambda_1' \\ &= \sum_{n=1}^{N_0} \left(C(n) + w(n)\right)\left(C\left(n + N_0\right) + w(n)\right) \\ &= \sum_{n=1}^{N_0} C(n)C\left(n + N_0\right) \\ &\quad + \sum_{n=1}^{N_0} w(n)\left(C(n) + C\left(n + N_0\right)\right) \\ &\quad + \alpha^2 N \end{aligned} \tag{24}$$

In the case of WM absence

$$\Lambda = \Lambda_0 = \sum_{n=1}^{N_0} C(n)C\left(n + N_0\right) \tag{25}$$

Let us arrange elements of periods with enough interleaving. There are two Gaussian distributions resulting from both Λ_0 and Λ_1, consequently the formulas (6), (7) can be used to find the probabilities P_m and P_{fa}. It is easy to see that

$$E(\Lambda_0) = 0 \quad , \quad E(\Lambda_1) = \alpha^2 N \tag{26}$$

$$\begin{aligned}\mathrm{var}(\Lambda_0) &= NE(C(n)^2)E\left(C\left(n + N_0\right)^2\right) \\ &= N\sigma_C^4. \end{aligned} \tag{27}$$

Assuming that $C(n)$ and $C\left(n + N_0\right)$ are uncorrelated random variables, after interleaving, we get

$$\mathrm{var}(\Lambda_1) = N\left(\sigma_C^4 + 2\alpha^2\sigma_C^2\right) \tag{28}$$

Substituting (26)-(28) in (6)-(7) we obtain

$$P_m = 1 - Q\left(\frac{\lambda - \alpha^2 N}{\sqrt{N\left(\sigma_C^4 + 2\alpha^2\sigma_C^2\right)}}\right) \tag{29}$$

$$P_{fa} = Q\left(\frac{\lambda}{\sqrt{N\sigma_C^4}}\right) \tag{30}$$

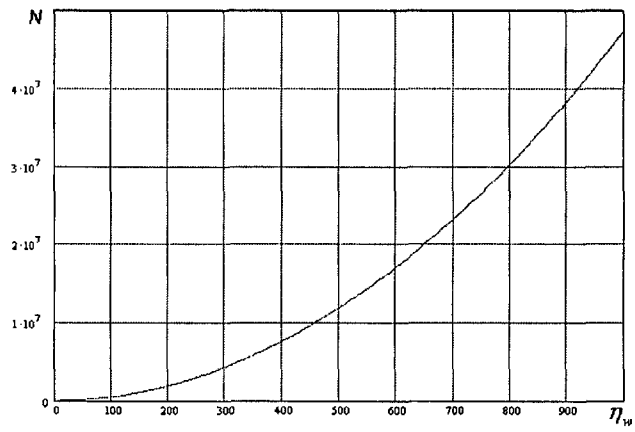


Figure 1: The number N of WM bits required for any ordinary user to provide $P_m = P_{fa} = 10^{-4}$ versus the distortion constraint η_w.

The distortion constraint after WM is given as the signal-to-noise-ratio

$$\eta_w = \frac{\mathrm{var}\left(C(n)\right)}{\mathrm{var}\, w(n)} = \frac{\sigma_C^2}{\alpha^2} \tag{31}$$

which, together with (29)-(30), gives

$$\begin{aligned} P_m &= 1 - Q\left((\lambda' - 1)\sqrt{\frac{N}{\eta_w^2 + 2\eta_w}}\right) \\ &\approx 1 - Q\left((\lambda' - 1)\sqrt{\frac{N}{\eta_w^2}}\right) \end{aligned} \tag{32}$$

$$P_{fa} = Q\left(\lambda'\sqrt{\frac{N}{\eta_w^2}}\right) \tag{33}$$

where $\lambda' = \frac{\lambda_0}{\alpha^2 N}$.

A comparison of (9), (10) with (32), (33) shows a *dramatic degradation* in the case when a periodic sequence is used for WM.

In fact, in order to "compensate" an increasing of η_w, say by m times, it is necessary to increase N also by m times for the WM method when using double sequences (see eqs. (9), (10)), while it is necessary to increase N by m^2 times for the WM method when using a periodic sequence.

This situation is illustrated in Fig. 1, where the number of WM bits required to provide $P_m = P_{fa} = 10^{-4}$ is presented versus η_w. (It is worth to remark that a similar effect of *"quadratic compensation"* is known in telecommunications *spread spectrum systems* in the case when it is necessary to use a noncoherent receiver [8].) This quadratic compensation is a pay off for the use of the semipublic method that works (as we show in the sequel) for any distortion constraints after WM (η_w) and after attack (η_a) in contrast to the previous method that works only when $\eta_a \approx \eta_w$.

In order to show that this method is actually WM semipublic instead of WM public we remark that an attacker may always disable the WM by creating the noise sequences as follows:

- For the case of WM absence:

$$\epsilon(n) = \begin{cases} \sigma\tilde{\pi}(n) & , \quad n \leq N_0 \\ \sigma\tilde{\pi}(n-N_0) & , \quad N_0+1 \leq n \leq 2N_0 \end{cases}$$

- For the case of WM presence:

$$\epsilon(n) = \begin{cases} \sigma\tilde{\pi}(n) & , \quad n \leq N_0 \\ -\sigma\tilde{\pi}(n-N_0) & , \quad N_0+1 \leq n \leq 2N_0 \end{cases}$$

where $\tilde{\pi}(n)$ is any sequence of signs ± 1, and σ satisfies the distortion constraint after the attack.

Let us consider the case when a detector (as the WM owner in his detectionrole) knows not only the position of the two periods but he also knows the sequence $\pi(n)$ as the secret key. Then the detector is able to calculate

$$\Lambda = \sum_{n=1}^{2N_0} S'(n)\pi(n) \tag{34}$$

where $\pi(n+N_0) = \pi(n)$, $n = 1, 2, \ldots, N_0$, and

$$\begin{aligned} S'(n) &= S(n) + \epsilon(n) \\ S(n) &= C(n) + w(n) \\ w(n+N_0) &= w(n) \end{aligned}$$

Since an attacker does not know the secret key sequence $\pi(n)$, the optimum additive noise sequence can be any zero mean i.i.d, sequence $\epsilon(n)$, $n = 1, 2, \ldots, N_0$, with variance σ^2 and $\epsilon(n+N_0) = \epsilon(n)$, $n = 1, 2, \ldots, N_0$. (We will see in the sequel that last condition increases the variance Λ and as a consequence it decreases the reliability of WM detection.)

We may assume that for large N the value Λ is Gaussian due to the Central Limit Theorem, hence the formulas (6), (7) remain valid if we substitute for Λ_1 the value Λ given at (34), when WM is transmitted, and for Λ_0 the value Λ given at (34), when WM is absent. It is easy to see that

$$E(\Lambda_0) = 0 \quad , \quad E(\Lambda_1) = 2\alpha N_0 \tag{35}$$

$$\text{var}(\Lambda_0) = 2N_0\left(\sigma_C^2 + 2\sigma^2\right) \tag{36}$$

We note that if an attacker uses independent sequences $\epsilon(n)$ and $\epsilon(n+N_0)$, then it would result in

$$\text{var}(\Lambda_0) = 2N_0\left(\sigma_C^2 + \sigma^2\right) = \text{var}(\Lambda_1) \tag{37}$$

which is less than (36). Hence the chosen noise attack sequence $\epsilon(n+N_0) = \epsilon(n)$ is more convenient for an attacker than the independent sequences.

The distortion constraints after watermarking and after WM followed by an attack can be presented, respectively, as follows

$$\eta_w = \frac{\text{var}\, C(n)}{\text{var}\, w(n)} = \frac{\sigma_C^2}{\alpha^2} \tag{38}$$

$$\eta_a = \frac{\text{var}\, C(n)}{\text{var}\,(w(n)+\epsilon(n))} = \frac{\sigma_C^2}{\alpha^2+\sigma^2} \tag{39}$$

Substituting (35)-(37) into (6), (7) and taking into account (38), (39) we get eventually

$$P_m \approx 1 - Q\left((\lambda'-1)\sqrt{\frac{N\eta_a}{\eta_w\eta_a + 2(\eta_w-\eta_a)}}\right) \tag{40}$$

$$P_{fa} = Q\left(\lambda'\sqrt{\frac{N\eta_a}{\eta_w\eta_a + 2(\eta_w-\eta_a)}}\right) \tag{41}$$

where $N = 2N_0$ and $\lambda' = \frac{\lambda}{\alpha N}$. Since it is inherent in WM application that $\eta_w > \eta_a >> 1$, we get from (40), (41)

$$P_m \approx 1 - Q\left((\lambda'-1)\sqrt{\frac{N}{\eta_w}}\right) \tag{42}$$

$$P_{fa} = Q\left(\lambda'\sqrt{\frac{N}{\eta_w}}\right) \tag{43}$$

This case is illustrated in Fig. 2, where the number of WM bits required to provide $P_m = P_{fa} = 10^{-4}$ is presented versus η_w.

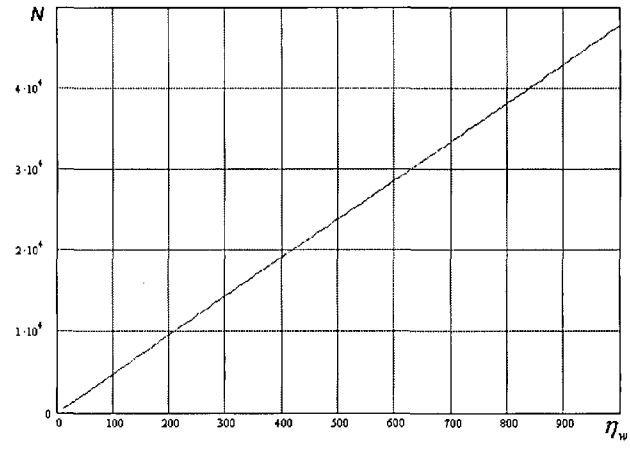


Figure 2: The number N of WM bits required for the WM owner to provide $P_m = P_{fa} = 10^{-4}$ versus the distortion constraint η_w.

We can see from (40), (41) that in contrast to the first method based on the use of a double sequence, the strong restriction $\eta_a \approx \eta_w$, is no longer necessary here and any high reliability of watermarking can be provided for any η_a, η_w at the cost of increasing N. And, in contrast to WM detection by ordinary users (see (32), (33)) we get from (40), (41) a *linear compensation* for WM owners who posses the secret key sequence $\pi(n)$. Moreover, those owners can significantly improve WM detection if the CM is

known by the WM decoder and it is considered as part of the secret key.

In fact, in this case the owner's detector is able to calculate

$$\Lambda = \sum_{n=1}^{2N_0} (S'(n) - C(n))\,\pi(n) \tag{44}$$

and compare it with some threshold λ. The variable $\Lambda = \Lambda_0$ in the case of WM absence and the variable $\Lambda = \Lambda_1$ in the case of WM presence are, as before, Gaussian variables. It is easy to show that

$$\begin{aligned} E(\Lambda_0) = 0 \quad &, \quad E(\Lambda_1) = 2\alpha N_0 \\ \operatorname{var}\Lambda_0 = \; 4N_0\sigma^2 \; &= \operatorname{var}\Lambda_1 \end{aligned} \tag{45}$$

Substituting (45) into (6), (7) and taking into account (38), (39) we get

$$P_m = 1 - Q\left((\lambda' - 1)\sqrt{\frac{N_0}{(\eta - 1)}}\right) \tag{46}$$

$$P_{fa} = Q\left(\lambda'\sqrt{\frac{N_0}{(\eta - 1)}}\right) \tag{47}$$

where $\lambda' = \frac{\lambda}{\alpha N}$, $\eta = \frac{\eta_w}{\eta_a}$.

A comparison of (46), (47) with (40), (41) shows that the case in which CM is used by the owner as decoder is more favorable for WM detection. Let us confirm this fact by means of the following:

Example. If $\eta_w = 120$, $\eta_a = 100$, then the probabilities P_m and P_{fa} are the same for both cases (in which CM is known and is unknown by the detector) if and only if the number N of CM elements used for WM in the second case is 300 times greater than this number as required for the first case. ■

3 Conclusion and Open Problems

In this paper we have proposed two constructions of semipublic watermarking. Such schemes enable WM detection for any ordinary user whenever the SM has not been subject of a previous attack attempting to remove the WM or to disable it. However these schemes enable WM detection even after an attack if some distortion constraints are satisfied but it is possible just by the WM owner only.

We have derived the formulas for the probabilities of WM-missing and WM-false alarm as functions of distortion constraints and the length of WM. These probabilities approach to zero, as the length of WM approaches to infinity, but only under some strong distortion constraints for the first method and without any restrictions for the second method. We have shown also which significant gain in the length of WM can be achieved if CM is used at the detection performed by the WM owner. We emphasized also a "quadratic compensation behavior" of the second semipublic scheme: in order to compensate an m times increase of the distortion constraint given by the signal-to-noise-ratio, it is necessary an m^2 times increase of the WM length. Thus, an interesting open problem is to propose a WM semipublic scheme having linear compensation for any distortion constraints. Perhaps a possible solution to the last problem can be provided by using the knowledge of the CM at the encoder side to form WM as it has been considered in [5]. There is also a more interesting open problem: how to design an effective WM pure public scheme and present its performance evaluations.

It is worth to note that there exists a trivial construction of a semipublic WM-scheme in which two independent additive WM sequences are used to watermark CM. Namely, let

$$S(n) = C(n) + w_s(n) + w_p(n) \tag{48}$$

where $w_s(n)$ is a secret sequence, known only by the WM-owner, and $w_p(n)$ is a public sequence known by all users. But in this case two main defects arise from scheme (48): The first one is that, using the knowledge of the public watermak $w_p(n)$, it can be removed completely from SM without any distortion of CM. The second is that $w_p(n)$ produces an additional corruption of CM.

The WM's presented in this paper should transmit only one bit of information: "there is or there is not a WM in the presented message". This problem statement can be extended to the case of multiple bit transmission. It is just the case of *fingerprinting* when WM can contain either the name of the WM owner, data and/or the name of person who bought this CM legally from the owner. We foresee that there will not be any problem in such extension and we are going to present its solution in the immediate future. The consideration of additive noise attack differs from our model essentially just by the practical situation condions. But as we have already mentioned in the Introduction of this paper, it provides an useful result for practical WM application because it determines lower bounds for WM reliability. Hence, any practical WM system will require more elements for the WM than our bound. In the near future we are going to consider more general types of attacks on WM systems such as combinations of linear filtering and additive noise attack.

References

[1] R. E. Anderson. Information hiding: First international workshop. *Lecture Notes in Computer Science*, 1174, 1996.

[2] D. E. Aucsmith. Information hiding: Second international workshop. *Lecture Notes in Computer Science*, 1525, 1998.

[3] I. J. Cox, M. L. Miller, and J. A. Bloom. *Digital Watermarking*. Morgan Kaufman Publishers, 2002.

[4] S. Katzenbeiser and F. Petitcolas. *Information Hiding*. Artech House Inc., 2000.

[5] V. Korjik, G. Morales-Luna, D. Marakov, and I. Marakova. A performance evaluation of digital private watermark under an additive noise attack condition. In *the forthcoming "VII Spanish Meeting on Cryptology and Information Security" 2002*, 2002.

[6] I. S. Moskowitz. Information hiding: Fourth international workshop. *Lecture Notes in Computer Science*, 2137, 2001.

[7] P. Moulin and J. O'Sullivan. Information theoretic analysis of information hiding. In *Proc. IEEE Symp. on IT'1998*. IEEE, 1998.

[8] R. Peterson, R. Ziemer, and D. Borth. *Introduction to Spread Spectrum Communications*. Prentice Hall, 1993.

[9] J. Proakis. *Digital Communications, Fourth Edition*. Mc Graw Hill, 2001.

An Efficient Anonymous Fingerprinting Scheme

Yong Li, Bo Yang and Xiang Hua
National Key Lab. of Integrated Service Networks, Xidian Univ., Xi'an 710071, China
Phone: +86 29 8202525
yonglee2001@163.com

Keywords: Anonymous Fingerprinting, Digital Fingerprinting, Digital Watermarking, TTP, STPC

Received: May 13, 2002

An anonymous fingerprinting scheme was presented by Domingo, which can identify the redistributors without the help of registration authority. However, this scheme on average requires N/2 exponential operations when the merchant identifies the redistributors, where N is the number of public keys in the directory. Chanjoo Chung proposed a more efficient scheme than Domingo's, but there is a fatal weakness in the proposed registration protocol of his scheme. This weakness causes that any impersonator can personate an honest buyer with the probability of 1/2. In this paper, we give an efficient scheme that requires only one exponential operation, which improves the whole scheme's efficiency. Security analysis is also given at last.*

1 Introduction

Protection of intellectual property in digital form has been a subject of research for many years and led to the development of various techniques. Fingerprinting schemes are an important class of these techniques. They are cryptographic methods applied to deter people from redistributing a data item by enabling an original merchant to trace a copy back to its original buyer. Dishonest buyers who redistribute the data item illegally are called traitors. The identifying information, called fingerprint, is embedded into copies of the original data item. The underlying watermarking techniques should guarantee that the embedded fingerprints are imperceptible and resistant to data manipulation as long as a traitor only uses one copy.

The first enhancement is the addition of collusion tolerance [1,2], i.e., resistance even if traitors compare up to a certain number of different copies. But both the merchant and the buyer know the fingerprinted copy in a symmetrical scheme, which means that the merchant's previous knowledge of the fingerprinted copies would not be used as proof of redistribution to a third party. A second addition [3,4,5] is asymmetry, whereby only the buyer knows the fingerprinted copy. The drawback of this scheme is that any merchant knows the buyer's identity even if the buyer is honest. The third addition [6] is anonymity where the buyer can stay anonymous in purchasing a fingerprinted data item. Only when some buyers redistribute the data item, the identity is revealed. We adapt the meaning of anonymity as the original definition in [6], i.e., any coalition of merchants, central parties and other buyers should not be able to distinguish purchases of the remaining buyers. A weak form can easily be achieved by using any asymmetric fingerprinting scheme under a certified pseudonym instead of a real identity. But on finding a fingerprinted copy, the merchant needs the help from a registration authority to identify the redistributors. In [7], Domingo introduced a new anonymous fingerprinting scheme, which identifies the redistributors without the help of registration authority. However, this scheme on average requires $N/2$ exponential operations when the merchant identifies the redistributors, where N is the number of public keys in a directory. In [8], Chanjoo Chung proposed more efficient scheme than Domingo's scheme, but there is a fatal weakness in the proposed registration protocol of his scheme, which leads that any impersonator can personate an honest buyer with the probability of 1/2. We will give detailed explanation about this point in Section 3.1. In this paper, we introduce an efficient scheme that requires only one exponential operation when one identifies the redistributors, which improves the whole scheme's efficiency. In addition, it is shown that nobody can personate an honest buyer.

The remainder of the paper is organized as follows: Section 2 shows our anonymous fingerprinting scheme. Section 3 describes the performance and security of our scheme. Conclusion is given in Section 4.

2 Our Scheme

System setup

Let p be a large prime such that $q=(p-1)/2$ is also a prime. Let Z_p^* be the multiplicative group modulo p, and let g be a generator of Z_p^* such that computing

* The work is supported by the National Natural Science Foundation of China (No. 69972034) and Foundation of National Laboratory For Secure Communication of China, grand no. 99JS06.3.1.DZ0112

discrete logarithms to the base g is difficult. Assume that both the buyer B and the registration authority R have ElGamal-type public-key/private-key pairs. The buyer's private key is x_B and his public key is $y_B = g^{x_B} \bmod p$. The registration authority R is ***Trusted third party*** (***TTP***) and uses its private key to issue certificates which can be verified by using it's public key. All the public keys are assumed to be known and certified.

The whole scheme is constructed as follows:

Protocol 1 (Registration)

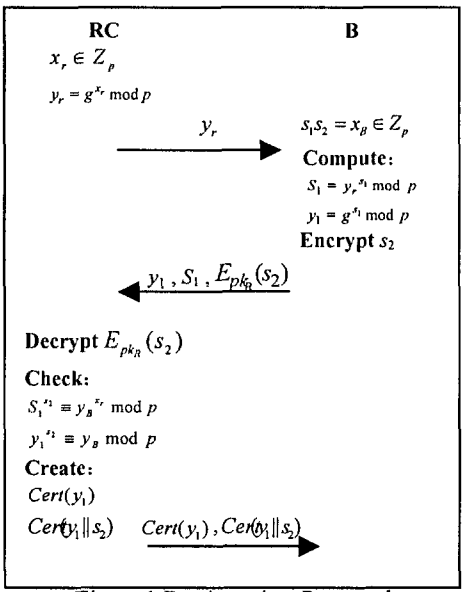


Figure1.Registration Protocol

(i) R chooses a random secret nonce $x_r \in Z_p$ and sends $y_r = g^{x_r} \bmod p$ to B.

(ii) B chooses secret random numbers s_1 and s_2 in Z_p such that $s_1 s_2 = x_B \in Z_p$, and computes $S_1 = y_r^{s_1} \bmod p$ and $y_1 = g^{s_1} \bmod p$, encrypts s_2 using the registration authority's public key pk_R into $E_{pk_R}(s_2)$, and then sends S_1, y_1 and $E_{pk_R}(s_2)$ to R. The buyer B convinces R with zero-knowledge the possession of s_1. In fingerprinting, y_1 will act as a pseudonym and an anonymous public key of the buyer B.

(iii) The registration authority R decrypts the value $E_{pk_R}(s_2)$ using his private key sk_R and checks that $S_1^{s_2} = y_B^{x_r} \bmod p$ and $y_1^{s_2} = y_B \bmod p$ hold. If verification is successful, R returns to B the certificates $Cert(y_1)$ and $Cert(y_1 \| s_2)$.

By going through the above registration procedure for several times, a buyer can obtain several different pseudonyms y_1.

Protocol 2 (Fingerprinting)

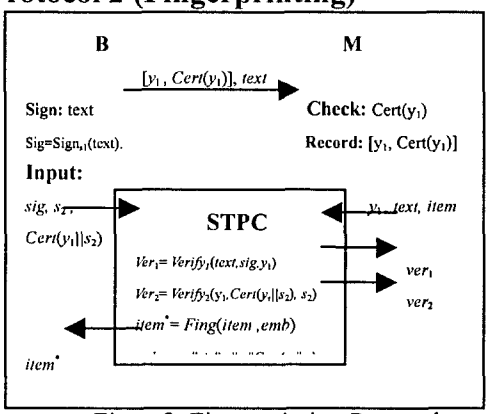


Figure2. Fingerprinting Protocol

(i) The buyer B sends $[y_1, Cert(y_1)]$ and *text* to the merchant M, where *text* is a string identifying the purchase. B computes an ElGamal signature *sig* on *text* using the private key s_1, *sig* is not sent to M.

Signature process is as follows:

(a) Selects a secure random number $k \in Z_p^*$.

(b) Computes $H(text)$ ($H(\cdot)$ is a Hash function)

$$r = g^k \bmod p$$

$$s = (H(text) - s_1 r)k^{-1} \bmod (p-1).$$

(c) Signature is $sig = (text, r, s)$.

(ii) M verifies the certificate on y_1 and records $[y_1 \| Cert(y_1)]$.

(iii) B and M enter a secure two-party computation [9]. M's inputs are *text*, y_1 and *item*, where *item* is the original information to be fingerprinted. The inputs of B are *sig*, s_2 and $Cert(y_1 \| s_2)$.

The computations performed are:

(a) $ver_1 = Verify_1(text, sig, y_1)$. The signature *sig* on *text* is verified using the public key y_1. The output ver_1 is a Boolean variable that can only be seen by the merchant M which is true if and only if the signature verification succeeds.

(b) $ver_2 = Verify_2(y_1, Cert(y_1 \| s_2), s_2)$. Firstly, the certificate $Cert(y_1 \| s_2)$ is verified. Secondly, it is verified that the value of y_1 in the certificate $Cert(y_1 \| s_2)$ is the same as the value y_1 input by M. Thirdly, it is checked that the value of s_2 in the certificate $Cert(y_1 \| s_2)$ is the same as the value s_2 input by B. The output ver_2 is a Boolean variable only seen by the merchant M which is true if and only if the aforementioned checks succeed.

(c) $item^* = Fing(item, emb)$. A classical fingerprinting algorithm is used to embed *emb* into the original information *item*, where

$$emb = text \| sig \| y_1 \| s_2 \| Cert(y_1 \| s_2) \quad (2)$$

The fingerprinted information $item^*$ is obtained as out and is only seen by the buyer B.

In the above two-party computation, M obtains outputs first and, unless ver_1 and ver_2 are both true, B cannot obtain the output $item^*$.

Protocol 3 (Identification)
On finding a redistributed copy, M extracts emb. The extracted information contains the values specified by equ.2 and is combined by M with the purchase record $[y_1, Cert(y_1)]$ to construct a redistribution proof:
(i) The signature sig on the $text$ is verified using y_1.
(ii) It is checked that the value y_1 is the same as the value y_1 in the certificates $Cert(y_1)$ and $Cert(y_1 \| s_2)$. Since it is part of the certificates, y_1 cannot be altered.
(iii) Finally, to identify a buyer, the merchant M computes $y_1^{s_2} = y_B \bmod p$. The dishonest buyer B has been identified. Note that, since s_2 is certified, it cannot be forged by the merchant M to unjustly accuse a buyer.

Protocol 4 (Disputation protocol)
This protocol is performed only when the merchant M shows the redistribution proof to an arbiter. The merchant M sends the proof (purchase record $[y_1, Cert(y_1)]$ and the extracted information emb) to the arbiter. The arbiter verifies the proof. He first verifies the certificate $Cert(y_1)$ using the registration authority's public key, then checks if $y_1^{s_2} = y_B \bmod p$ hold. If above three verifications in the identification protocols are valid, then the owner of the public key y_B is accused. If the registration record is necessary to prove guilty of buyer, the arbiter asks registration record to the registration authority.

3 Security analysis

3.1 The Security weakness of Chanjoo Chung's scheme

Chanjoo Chung [8] gave a more efficient scheme than Domingo's, but there is a fatal weakness in his registration protocol. We review this registration protocol. The buyer B chooses two secret random numbers s_1 and s_2 in Z_p such that $s_1 s_2 = x_B \in Z_p$, computes $y_1 = g^{s_1} \bmod p$, encrypts s_2 using the registration authority's public key pk_R, sends y_1 and $E_{pk_R}(s_2)$ to the registration authority R. R decrypts $E_{pk_R}(s_2)$ with his private key sk_R and checks if $y_B = y_1^{s_2} \bmod p$. If this is hold, R returns to B the certificate. In the above procedure, R verifies if a buyer is the owner of y_B by only checking whether $y_B = y_1^{s_2} \bmod p$. However, a malicious buyer can personate the owner of y_B by selecting t and x from Z_p such that $y_B = t^x \bmod p$. We can construct such values t and x. Based on the Fermat theorem and the Quadratic Residue theorem, we have $y_B = y_B^{(p-1)/2} y_B^{(p+1)/2} \bmod p$, where $y_B^{(p-1)/2} \equiv 1 \bmod p$ if y_B is a quadratic residue modulo p. The probability of y_B being a quadratic residue is 1/2. If y_B is a quadratic residue, we can factor $(p+1)/2$ into the multiplication of x_1 and x_2 with high probability, i.e., $(p+1)/2 = x_1 x_2$. Let $t = y_B^{x_1} \bmod p$ and $x = x_2$, we gain t and x satisfying $y_B = t^x \bmod p$, which means we can personate the buyer B.

3.2 The performance and security of our scheme

Performance We now discuss the performance and security of our scheme. Firstly, in the registration protocol, we replace $s_1 + s_2 = x_B \in Z_p$ in Domingo's scheme with $s_1 \cdot s_2 = x_B \in Z_p$. In view of security, the length of key should be two times longer than the one in Domingo's scheme. But computation complexity isn't increased since x_B does not participate in the computation when the scheme is performed. Secondly, in order to transmit s_2, we require the encryption and decryption on s_2, which adds 2 exponential operations in our registration protocol than Domingo's, but pass numbers and transmitted parameters are both decreased by one. Finally, when the merchant identifies the redistributor, exponential operations in our scheme are required 1 time, but averagely required $N/2$ in Domingo's scheme, where N is the number of public key in directory. Generally, the number of public key N is very large, which means that much computation quantity is needed. In order to accurately illuminate our scheme's efficiency, we contrast our scheme with Domingo's scheme according to the computation quantity of view. This result is as follows (Table 1):

Scheme / Object	Domingo's Scheme	Our Scheme
Exponential Operations	14+N/2 (Average)	13 (Any case)
Multiply Operations	8	6
Transmitted Parameters	22	18

Table 1. Comparison of performances

Security Since our scheme is the same problem based on computing discrete logarithm as Domingo's scheme, security is not decreased than previous scheme. The following two results show the security of our scheme.
Proposition 1 (Buyer's registration security) Registration protocol provides any impersonator cannot personate an honest buyer.
Proof: In registration, if some impersonator wants to personate an honest buyer B, he must find the values y_1', S_1' and s_2' which are related in the same way as y_1, S_1 and s_2, i.e. $y_B = y_1^{s_2'} \bmod p$ and $y_B^{x_r} = S_1^{s_2'} \bmod p$. Hence the impersonator can compute the discrete logarithm $x_r = \log_g y_r \bmod p$. If the impersonator is feasible, so is computing discrete logarithm problem. In general, discrete logarithm problem is hard, so any

impersonator doesn't make x_r such that $y_r = g^{x_r} \bmod p$.

Proposition 2 (Buyer's anonymity) An honest buyer who follows the fingerprinting protocol will not be identified if computing discrete logarithm is hard and secure two-party computation is feasible.

Proof: In the fingerprinting protocol, the merchant M knows $[y_1, Cert(y_1)]$ and his outputs of the secure two-party computation that ver_1 and ver_2. Finding the public key y_B would require knowledge of s_2. However, if the secure two-party computation is feasible, the merchant will not attain any knowledge of s_2.

4 Conclusion

We presented an efficient anonymous fingerprinting scheme. It not only protects the electronic data copyright and hides the buyer's identity, but also possesses higher efficiency than Domingo's scheme. Since our scheme is based on discrete logarithm, its security is the same as Domingo's scheme. With the development of electronic commerce, more efficient scheme is necessary. The efficient of scheme will play an important role in defending against the piracy of many software and multimedia contents.

5 Reference

[1] G. R. Blakeley, Catherine Meadows, George B. Purdy, "Fingerprinting Long Forgiving Messages". Crypto'85, LNCS 218, Spinger-Verlag, Berlin 1986, 180-189.

[2] Boneh. D and Shaw. J, "Collusion-Secure Fingerprinting for Digital Data". Advances in Cryptology, Proceedings of CRYPT-O'95, vol. 963 of LNCS, Springer-Verlag,Berlin 1995, pp. 452-465.

[3] Pfitzmann. B and M. Schunter, "Asymmetric Fingerprinting". Advances in Cryptology, Proceedings of EUROCRYPT'96 v-ol. 1070 of LNCS. Springer-Verlag, 1996,pp.84-95.

[4] Pfitzmann. B and M. Waidner, "Asymmetric Fingerprinting for Larger Collusions".4th ACM conference on Compuetr and Communication Security, Zurith, April 1977, 151-160.

[5] Ingrid Biehl, Bernd Meyer, "Protocols forCollusion–Secure Asymmetric Fingerprinting". STACS97, LNCS 1200, Spinger-Verlag, Berlin 1997, 399-412

[6] Pfitzmann. B and M. Waidner, "Anonym-ous Fingerprinting". Advances in Crypto-logy, Proceedings of EUROCRYPT'97. v-ol. 1233 of LNCS. Springer-Verlag,1997, pp.88-102.

[7] J. Domingo-Ferrer, "Anonymous Fingerprinting of Electronic Information with Automatic Identification of Redistributors". Electronics Letters 43/13, 1998, pp 1303-1304.

[8] Chanjoo Chung. Seungbok Choi, Youngchul Choi, Dongho Won. "Efficient Anonymous Fingerprinting of Electronic Information with Improved Automatic Identification of Redistributors". In PKC2000,pp 221-234.

[9] Chaum D., Damgaard. I. B., and Van De Graaf. J. "Multiparty Computation Ensuring Privacy of Each Party's Input and Correctness of the Result". Advances in Cryptology, Proceedings of CRYPTO '87, vol. 293 of Lecture Notes in Computer Science, Springer-Verlag, 1987, pp.87-119.

[10] Chaum D. Evertse J. H. and Van De Graaf. J. "An Improved Protocol for demonstrating Possession of Discrete Logarithm and Some Generalizations". Advances in Crypto-Eurocryt'87. LNCS 304, Spinger-Verlag, Berlin 1988, 127-141.

An Anonymous Mobile Agents Scheme for Secure Web Transaction over the Internet

Changjie Wang and Yumin Wang
National Key Lab. On ISN, Xidian University, Xi'an 710071, P.R.China
cjwang@ee.cityu.edu.hk, ymwang@xidian.edu.cn
AND
Fangguo Zhang
CAIS Lab. ICU (Information and communications Univ.), Korea.
zhfg@icu.ac.kr

Keywords: Mobile Agent, Digital Signature, Anonymity and Electronic Commerce

Received: May 7, 2002

A major problem of mobile agents is their apparent inability to authenticate transactions in hostile environments. In this paper, we propose a new secure anonymous mobile agent scheme for the prevention of agent tempering without compromising the mobility or autonomy of the agent. In our scheme, a mobile agent can produce valid signature on Website's bid (means that transact a contact with the Web site) on behalf of its customer, without leaking the customer's really private key. In addition, the anonymity of the customer is also achieved when its agent transacting with the Websites. Furthermore, the customer who issued a malicious agent or deny the transaction can be identified and revealed by Agent Management Center (AMC). Therefore, our scheme is practical in the future electronic commerce over Internet.

1 Introduction

Mobile agents are autonomous software entities that are able to migrate across different execution. In contrast to traditional communication mechanisms like client-server system, the main feature of mobile agent system are mobility and autonomy, which make permanent connections unnecessary; thus mobile agents are suitable for providing low-bandwidth connections and asynchronous communication [1,2,3]. Furthermore, they provide better support for heterogeneous environments such as World Wide Web.

The characteristics of mobile agents make them ideal for electronic commerce applications in open networks. Besides providing a very flexible approach for information gathering on special products、service and assets available from the several company servers they visit, they can effectively take over the different aspects of the electronic commercial transaction, from price settlement to paying and delivery of goods purchased. However, mobile agents are vulnerable to several attacks [4] and in particular to attacks by malicious hosts. Without any protection scheme, a visited site may read an agent's data and thus collect information about the agent's owner, e.g. its services, customers, service strategies, collected data, etc. Furthermore, In order to contact transaction with Websites, and sign on a contract on behalf of its' owner, the agent software may take the private key of its owner when roaming among Websites. In such case, the malicious host may steal and abuse the private key of the customer easily.

In this paper, we propose a secure mobile agent scheme for Web transaction over Internet, where both the securities of the Websites and the mobile agent software are considered. In addition, the anonymity of the customer is achieved during the transactions, which is a new security property of our scheme contrast with existing mobile agent systems.

Following is the organization of the paper. We firstly present the previous work in Section 2. A secure mobile agent scheme and the implementing protocols for Web transaction are proposed in Section 3, in which allow a mobile agent to conduct a transaction with the Website without being abused. In Section 4, we analyze the security of our scheme and finally the conclusion.

2 Previous Work

By far, many mobile agent systems have been proposed [5,6,7], most of which have considered the security problem. The most security concern of current mobile agent systems is how to protect Web server from attacking by the malicious agent software. The familiar security measure is based on the rules that the Web server should verify the identity of the agent (i.e. verify the customer's signature on the agent software or verify the signature signed by agent on behalf of the customer) before authorizing the access of the agent software. However, the security of the agent is seldom considered. Usually, the protection of mobile agent software from malicious host is considered more difficult than that for Websites. Until quite recently there was a general belief that mobile agent vulnerability could be prevented only with hardware solutions. Such solutions, as described in [2], are always based on an extra trusted and tamper-resistant hardware in system. However, the above belief has been shown misleading in [8], in which Sander and Tschudin propose the use of encrypted functions to solve the security problem under the conditions without extra hardware. As described in [8], the user encrypts a function s attached to the agent, which is then executed by the Website, without the host having access to the s. We give an example to show the above solution: Suppose that a customer wishes to send a mobile agent to purchase some goods from an electronic shop over Internet. The agent can authenticate the transaction only if it is able to use a signature function s (i.e. knowing the secret key of the customer) of the customer. However, the agent is executed by a potentially malicious Web server. To protect the signature function s, the customer encrypts it with a function f to obtain: $f_{signed} = s\ (f(.)\ )$ and embeds the pair of functions $(f(.),f_{signed}(.))$ into the agent executable code. On migration, the Web server executes the pair $(f(.),f_{signed}(.))$ on input x (which is linked to the server's bid) to obtain the so-called undetachable signature pair: $f(x) = m$ and $f_{signed} = s\ (f(x)\ ) = s(m)$. In this way, the pair of functions $(f(.),f_{signed}(.))$ enables the agent to create signature of the customer on message of the Web server, without revealing the signature function s (i.e. the secret key of customer). Usually, the function f is linked to the constraints (include details of the required product) of the customer so that the Web server cannot use the pair $(f(.),f_{signed}(.))$ to sign arbitrary messages. Although above solution is effective to protect the customer's secret key, the realization of the secure undetachable signature scheme remains open problem. The main difficulty is how to construct the pair of functions $(f(.),f_{signed}(.))$ so that it is hard to get s by decomposing the encrypted function $f_{signed}(.)$.

In [9],Panayiotis Kotzanikolaou, Mike Burmester and Vassilios Chrissikopoulos firstly propose and implement a undetachable signature scheme based on RSA and exponential functions. In their scheme, $f(.) = h^{(.)} mod\ n$ and $f_{signed}(.) = k^{(.)} mod\ n$, where $k = h^d\ mod\ n$ is the customer's RSA signature of h. Observe that $f_{signed}(.)$ is the encryption of the RSA signature function $s(.) = (.)^d\ mod\ n$ of customer, that is:

$f_{signed}(.) = s(f(.)) = s(h^{(.)}) = (h^{(.)})^d = (h^d)^{(.)} = k(.)$. Thus, on migration the server executes the agent on the input x (which is linked to the server's bid) to obtain the undetachable RSA signature (m, z), where $m = f(x) = h^x\ mod\ n$ and

$z = f_{signed}(x) = k^x\ mod\ n = (h^d)^x\ mod\ n = (h^x)^d = m^d\ mod\ n = s(m)$.

Above scheme is an effective implementing of undetachable signature. However, the scheme is not very practical when applied to electronic commerce due to the less consideration of users' anonymity. It is well known that the anonymity of users is an important consideration in electronic commerce. For example, anonymous electronic payment scheme is suggested for E-commerce, which is designed for the protection of users' privacy. That is, the bank should not know the form details of the e-cash withdrew by the customers, and the electronic shop should not know the identity of the customer in an electronic transaction. More discussion about the anonymity of electronic payment can be referred to elsewhere [10][11]. Such anonymity property of the users will has to be broken if an agent system is applied for electronic transaction, without considering the anonymity of the agent. That is, the shop server may reveal the identity of users in an electronic transaction easily just by verifying the identity of the agent, though an anonymous electronic payment system may be adopted.

3 A Secure and Anonymous Mobile Agent Scheme

In this Section, we propose a secure and anonymous mobile agent scheme for electronic transaction, using a digital signature algorithm with pair of one-off keys. In our scheme, not only the customer's private key but also the anonymity of the user can be protected. We first introduce the model of our scheme, and then propose the implementing protocols. An elliptic cure based digital signature with pair of one-off keys is also proposed in order to protect the anonymity and private key of user.

Model

There are three parts in the model of our scheme, named "User Agent Home (*UAH*)", "Agent Management Center (*AMC*)" and "Websites" respectively, as shown in Fig.1.

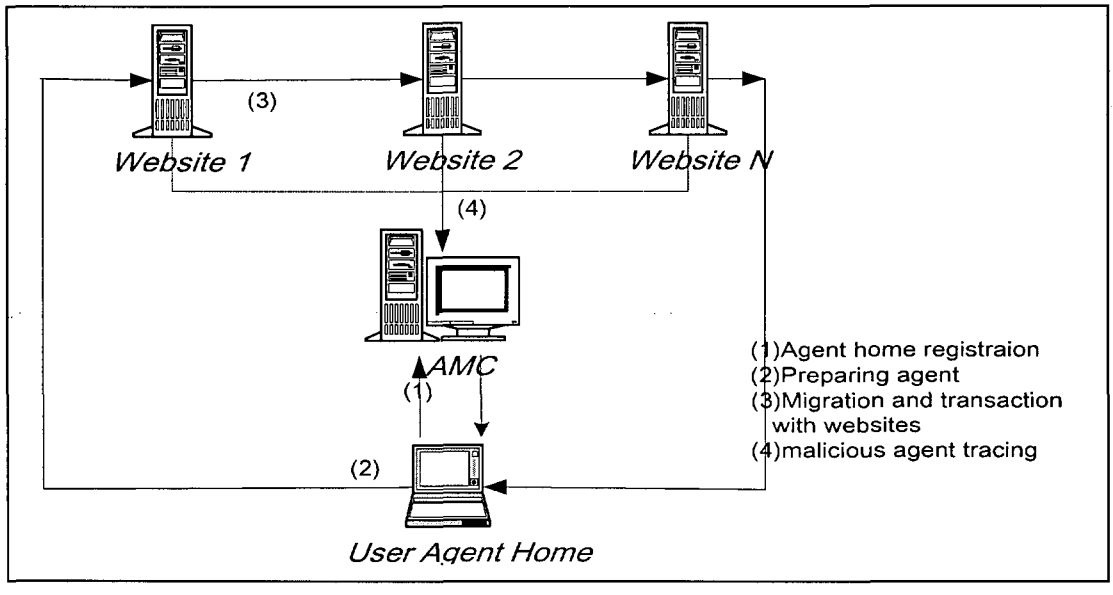


Fig.1 Model of our scheme

The working flow is illustrated at the down right corner of the Fig.1, and the explanations of the participators in the model is below:

User Agent Home (*UAH*): In our system, an agent control platform (called "User Agent Home") should be installed on each User's computer, who is in charge of the preparing, issuing and receiving of the User's mobile agent.

Agent Management Center (*AMC*): We introduce a new role (i.e. ***AMC***) in our model, whose duty is dealing with the registration of customer and the tracing of malicious agent.

Websites: We use "Websites" to denote all kinds of Web severs (may be electronic shop) over the Internet.

Implementing protocols

In the following, we present the detailed implementing protocols of our scheme:

Settings. We use ECC-based signature scheme settings. Let p be a large prime , a, $b \in GF(p)$ satisfy $4a^3+27b^2 \neq 0$. The elliptic curve $E_{(a,b)}(GF(p))$ is defined to be the set of points $(X,Y) \in GF(p) \times GF(p)$ satisfying equation $Y^2 = X^3+aX+b$ and a special point O (called infinity). These points form an Abelian group. G is an element of $E_{(a,b)}(GF(p))$ with order q, which is a prime at least 160 bits long. $R_x(A)$ denotes the *x-coordinate* of point A. For more details of elliptic curves, one can be refer to [12,13]. Let H be a one-way hash function, $H:\{0,1\}^* \rightarrow \{0,1\}^k \ (k \approx 160)$. Denoted $(\bullet \| \bullet)$ by the concatenation of two strings.

In our scheme, AMC first selects $X_{AMC} \in GF(q)^*$ randomly as his secret key, then calculate his public key $P_{AMC}=X_{AMC}G=(X_P, Y_P)$. Finally, AMC publishes the following public parameters to all Users and Websites: ($E_{(a,b)}(GF(p))$, G, q, P_{AMC}, H).

User registration. In our scheme, each User should first register with ***AMC*** before accessing the service of mobile agents system. The protocol for registration (between ***UAH*** and ***AMC***) is shown in Fig.2 (in the following, ID_X denotes the identifier of X).

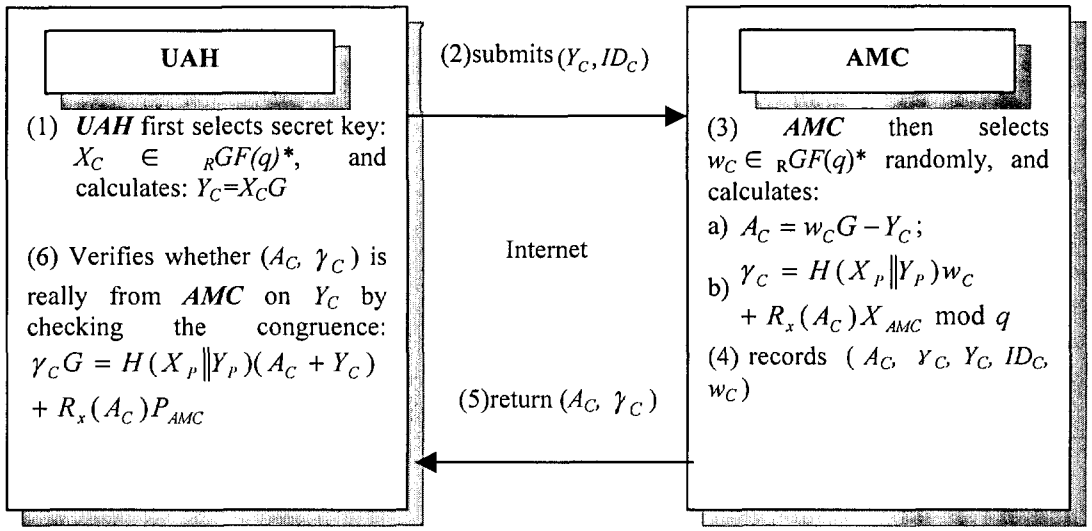


Fig.2 Protocols for User registration

At the end of above protocol, *AMC* records the information of the user's identity (Y_C, ID_C) while knowing nothing about the private key of user. In addition, a registered *UAH* obtains a certificate (A_C, γ_C) from *AMC*. Here, Y_C and X_C are the long-term pair of keys of the User, and (A_C, γ_C) is the long-term certificate of the user.

Preparing agent. Before issuing mobile agent according to the User's request, ***UAH*** should generate a pair of one-off keys and a virtual certificate attached to the agent as the part of its executable code. This procedure is described below:

Firstly, ***UAH*** select secret key $x_C \in {}_RGF(q)^*$ randomly, and the corresponding public key $y_C=x_CG$. Then calculate: $T=\psi G$, ($\psi \in {}_RGF(q)^*$ and ψ should be selected such that $R_x(T)=t \neq 0 \bmod q$),

$\alpha=A_C+T$, $\beta=t\alpha$, $z=tR_x(A_C) \bmod q$

$\lambda= \gamma_C t+(\psi t - X_C t + x_C H(M))H(X_P|Y_P) \bmod q$, here, M is a message includes all static part of agent code (such as the customer's request, routing table and the time-stamp). Thus, the ***UAH*** has produced a pair of one-off keys (x_C,y_C), with a virtual certificate *VCert*: $\{y_C, \alpha, \beta, \lambda, z\}$, for the mobile agent. Then ***UAH*** attaches the above data to the mobile agent code as shown in Fig.3, and issues the agent out.

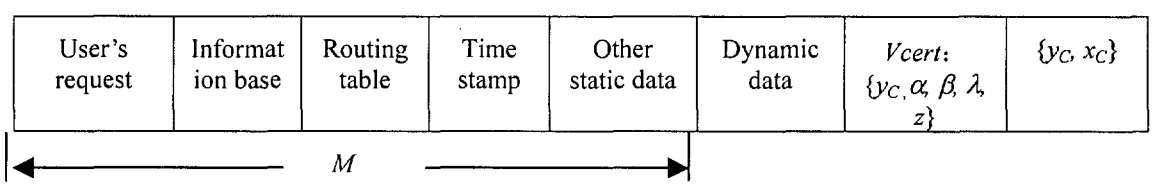


Fig.3 The structure of an issued mobile agent

Executing agent. On migration, the Websites first verify the validity of the agent and then execute the agent on input $m = H(ID_W, W\_bid)$ (here, *W_bid* is the Website's bid according to the User's request), to obtain an one-off signature *Sig*:$\{K,r\}$of m. The detailed protocol is described below:

Parameters	Input	Output
$\{y_C, x_C\}$: pair of one-off keys of User *Verct* $\{y_C, \alpha, \beta, \lambda, z\}$: one-off certificate of User M: all part of static code of agent (constraints)	Website's bid: $m = H(ID_W, W\_bid)$	One-off Signature on m: *Sig*:$\{K,r\}$

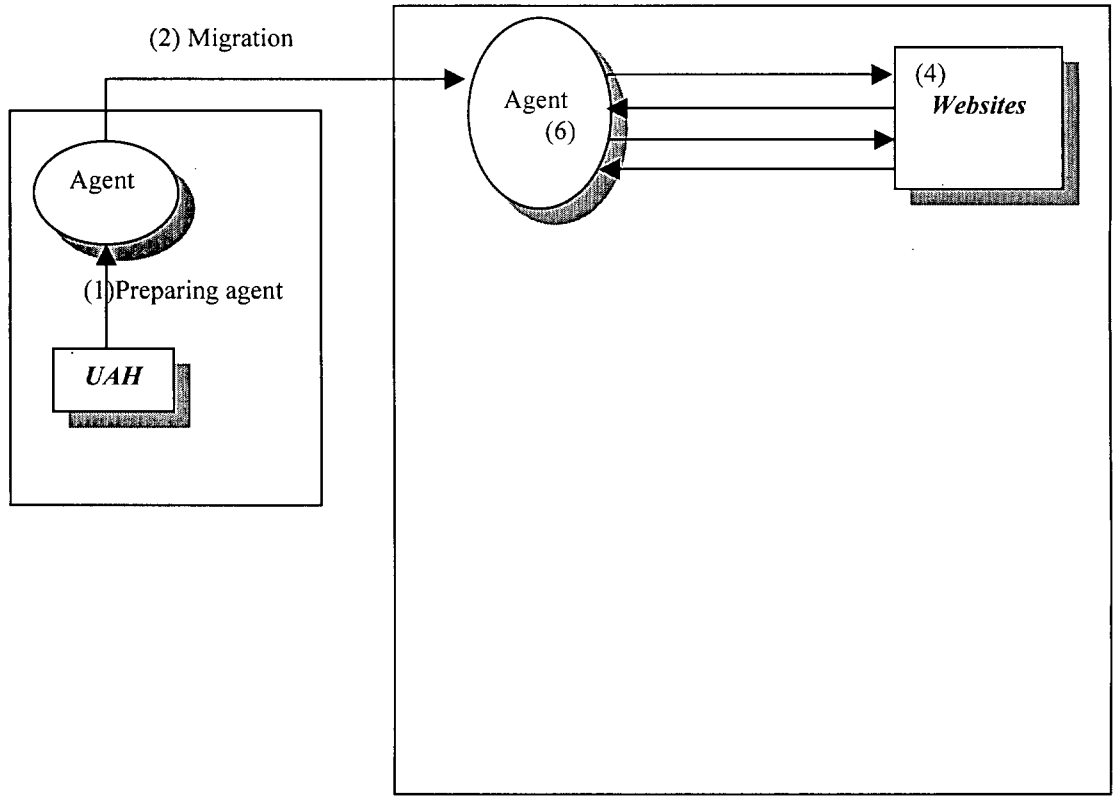


Fig.4 Protocols for executing agent

We give some explanations of above protocols.

In step (4), Websites should verify the validity of the agent by checking the congruency:

$\lambda G = H(X_P||Y_P)(\beta+y_C H(M)) + zP_{AMC}$ [1] before authorizing the mobile agent corresponding access to the Website's resource. Here, successful verification means that the mobile agent is really issued by a registered ***UAH*** and does not been modified while roaming over Internet. In this way, the Websites can resist the attack from a malicious agent.

In step (6), the agent generates a one-off signature on the *W_bid* (the bid from the Website) if the bid satisfies the Users' request. Otherwise, the agent just migrates to the next Website for another negotiation. In this way, the agent can sign on a contrast (i.e. the *W_bid* from Website) on behalf of its' owner without leaking the real private key of the customer. Note that, *M* (includes the request of User) is a constraint to prevent any malicious Websites from abusing the one-off keys to sign on a bogus bid. More security considerations will be discussed in next section.

In step (9), message $Sig_W(ID_W, W\_bid)$ means that the Website is required to sign on (ID_W, W_bid) to bind itself to the bid to avoid the impersonation attacks on the Website.

After the transaction, the agent obtains a message of Website's bid, and the Website records the data (*Vcert*, *Sig*, *M*), which is helpful in tracing the User when necessary. Note that even if a Website abuses the one-off private key to produce a signature of the User for a bogus bid, the signature will be invalid due to that the content of the signature may disagree with the constraint *M*, which includes the User's request. In fact, if the Website is not willing to bid for a transaction, then it forwards the agent to another Website.

Agent identification. When necessary, the Websites can submit the *VCert* and *Sig* of an agent to ***AMC*** to reveal the identity of the mobile agent (i.e. its Owner). Here, we use "necessary" to denote the situation when the mobile agent is considered to be malicious or the User want to deny the transaction signed by its' agent.

When receiving *VCert*, *Sig* and *M*, ***AMC*** first verifies whether the User of the agent has been registered by checking the congruence:

$\lambda G = H(X_P||Y_P)(\beta+y_C H(M)) + zP_{AMC}$.

Then for each record (A_i, Y_i, Y, ID_i, w_i) in ***AMC***'s database, ***AMC*** checks whether the following congruence holds: $\beta \stackrel{?}{=} (z/R_x(A_i))\alpha$ [2]. It can be proved that the above congruence has a unique solution $(A_C, Y_C, Y_C, ID_C, w_C)$. In such way, the identity of the mobile agent (i.e. its Owner) can be identified by AMC. It seems that this

[1] That is: $H(X_P||Y_P)(\beta+y_C H(M)) + zP_{AMC}$
$= H(X_P||Y_P)(t\alpha+x_C GH(M)) + tR_x(A_C)X_{AMC}\, G$
$= (H(X_P||Y_P)(tw_c - tX_C + t\psi + x_C H(M)) + tR_x(A_C)X_{AMC})G$
$= (t(H(X_P||Y_P)w_C + R_x(A_C)X_{AMC}) + H(X_P||Y_P)(t\psi - tX_C + x_C H(M)))G$
$= (t\gamma_C + (t\psi - tX_C + x_C H(M))\, H(X_P||Y_P))G = \lambda G$

[2] That is: $(z/R_x(A_C)) = (tR_x(A_C)/R_x(A_C))\,\alpha = t\alpha = \beta$

approach is not efficient enough if there are many Users in one ***AMC*** database, and we will make more improvements in the future work.

4 Security analysis of our scheme

Our scheme is mostly based on a digital signature with pair of one-off keys and its security is depended on elliptic curve discrete logarithm problem (ECDLP). As it is well known that to solve the ECDLP over finite field is open problem, we presume that the attacker do not have the ability to solve the ECDLP. In the following, we make a security analysis of our scheme:

(1) One important security property of our scheme is that the mobile agent can produce valid signature on Website's bid on behalf of its customer, without leaking the customer's long-term private key. Instead of the long-term private key, in our scheme, a pair of one-off keys and a *Vcert* is attached to each agent to make signature. Clearly, it is easy for a malicious Website to construct a signature on a bogus *W_bid* for a given $\{y_C, x_C\}$. The problem for a malicious Website is to construct a new *Verct'*: $\{y_C, \alpha, \beta, \lambda, z\}$ of the user which will include modified constraint *M'* of the user. As proposed in our protocols, if above attack is possible, then one can also solve the ECDLP.

(2) Ensuring users' anonymity while using the mobile agent system is another new security property of our scheme, which is not achieved in other existing mobile agent system.

In some case, the Websites may deal with many agents from the same user for different goals (such as for enquiry of different goods). If these agents are signed with the unaltered private key of Users, the Websites will identify the customer of these mobile agents easily just by comparing that whether both of the signed agents can be verified by the same public key (i.e. the same public key certificate). In our scheme, the ***UAH*** produces a pair of one-off keys and a virtual certificate attached to each issuing mobile agent. Even the Web server has transacted with many agents issued by the same user, the Website cannot identify the user because that the agents are attached with different pair of keys and *Vcert*.

In order to identify the user by recuperating his really certificate $\{A_C, \gamma_C\}$ from the virtual certificate *VCert*: $\{y_C, \alpha, \beta, \lambda, z\}$, what a Websites have to do is shown below:

First, solve the t from the equation: $\beta = t\alpha$ *, therefore get the T; accordingly, solve the A_C as: $A_C = \alpha - T$;

Second, solve the X_{AMC} from the equation: $P_{AMC} = X_{AMC}G$ *; solve the x_C from the equation: $y_C = x_C G$ * and solve the ψ from the equation: $T = \psi G$. Then solve the variables: γ_C, X_C, w_C from the simultaneous equations:

$$\begin{cases} A_C = w_C G - X_C G \\ \gamma_C = H(X_P \| Y_P) w_C - R_x(A_C) X_{AMC} \\ \lambda = \gamma_C t + (\psi t - X_C t + x_C H(M)) H(X_P \| Y_P) \end{cases}$$

To make a successfully attack as shown above, a Website has to solve the ECDLP at three places, marked out with *. As our presumption above, the ECDLP is difficulty to be solved. Therefore, the anonymity of the agent is achieved.

In addition, ***AMC*** can repeal the anonymity of agent's User when necessary. However, such revocation is under constraints and can be carried out by AMC only when an agent is regard as malicious. In this way, any User also cannot deny what its agent has transacted with the Website.

5 Conclusion

Current research on software-based active prevention contradicts the general belief that mobile code vulnerability could be actively prevented only with hardware-based solutions. In this paper, we propose a practical scheme for mobile agents to conduct private and binding transactions in a hostile environment, by using cryptographic technique.

As a conclusion, we summarize the main features of our scheme below:

- *Efficiency*: The implementation of our scheme is most based on a signature scheme of ECC, which is more effective due to that the key length and computation amount of ECC-based signature scheme are less than that of other similar scheme, such as RSA.

- *Secrecy*: The mobile agent can contact a transaction with the Websites by making a signature *Sig*, while nothing of the User's secret key is leaked. In addition, the Websites is able to bind the User to the transaction via *Sig* and *Vcert*, which link the constraints of the agent *M* and the bid of the Website *W_bid*.

- *Anonymity*: The mobile agent issued by a registered User can make an anonymous transaction with Websites. The Websites can only verify the validity of the agent to resist the malicious agent, while knowing nothing about the User's identity.

- *Fairness*: A dishonest User who issued a malicious agent or denied the transaction contacted by its agent can be traced and revealed by ***AMC***. Note that the work mode of ***AMC*** is off-line. This means that ***AMC*** will not involved the transaction only when a disputation is happened.

- *Unforgeability*: A malicious Website can use a pair of one-off keys $\{x_C, y_C\}$ and the *Vcert* $\{y_C, \alpha, \beta, \lambda, z\}$ to generate a forgeable signature *Sig*(*K'*, *r'*) of a message *m'* that includes a bogus bid *W_bid'*. But such a signature will be invalid due to that the pair of keys $\{x_C, y_C\}$ and the *Vcert* $\{y_C, \alpha, \beta, \lambda, z\}$ are constrained by a unique *M*, and the *W_bid'* is non-compatible with the customer's request in constraint *M*.

6 Acknowledgment

This work is supported by the National Nature Science Foundation of China under the reference number 60073052.

Reference:

[1]. Danny B. Lange and Mitsuru Oshima. "*Seven Good Reasons for Mobile Agents*", Communications of ACM, P.88-89, 1999 Mar.

[2]. Chess, D., Grosof, B., Harrison, C., Levine, D., Parris, C. and Tsudik, G. *"Itinerant Agents for Mobile Computing"*, Technical Report, RC 20010 (1995) IBM T.J. Watson Research Center, NY

[3]. Kotzanikolaou Panayiotis, Katsirelos George and Chrissikopoulos Vassilios. *"Mobile Agents for Secure Electronic Transactions"*. Recent Advances in Signal Processing and Communications, World Scientific and Engineering Society Press (1999) 363-368.

[4]. Wayne Jansen, Tom Karygiannis. *"NIST Special Publication 800-19 – Mobile Agent Security"*. Technical Report MD 20899, National Institute of Standards and echnology Computer Security Division Gaithersburg, 1999.

[5]. IBM, Inc. IBM Aglets Documentation web page. Available at URL http:// aglets.trl.ibm.co.jp/documentation.html, 1998.

[6] "*Concordia-Java Mobile Agent Technology*". Available at http://www.meitca.com/HSL/Projects/Concordia/

[7]. Robert S.Gray. "*Agent Tcl: A flexible and secure mobile-agent system.*" In *Proceedings of the Fourth Annual Tcl/Tk Workshop (TCL'96)*, Jul 1996

[8]. Sander Tomas and Tschudin Christian F. *"Protecting Mobile Agents Against Malicious Hosts"*. Mobile Agent Security, LNCS Vol.1419, Springer-Verlag, (1998) 44-60.

[9]. Panayiotis Kotzanikolaou , Mike Burmester and Vassilios Chrissikopoulos. *"Secure Transactions with Mobile Agents in Hostile Environments"*. Information Security and Privacy. Proceedings of the 5th Australasian Conference, ACISP 2000. LNCS Vol. 1841, Springer-Verlag, pp. 289-297, 2000.

[10]. J.Claessens, B.Preneel and J.Vandewalle, *Anonymity controlled electronic payment systems,* Proceedings of the 20th symposium on information theory in the benclux, Haasrode, Belgium,May 27-28 1999, pp.109-116.

[11]. D. Chaum, A. Fiat, and M. Naor, "Untraceable electronic cash", in Proceedings of Crypto'88, LNCS, Springer-Verlag, pp. 319-327, 1990.

[12]. V.S.Miller, "*Use of Elliptic Curve in Cryptography",* In Advances in Cryptology—CRYPTO'85(Santa Barbara,Calif.,1985),LNCS 218(1986), pp.417-426, Spring-Verlag.

[13]. N.Koblitz, "*Elliptic Curve Cryptosystems",* Mathematics of Computation, 48(1987), pp.203-209.

Compiling and Using the IJS-ELAN Parallel Corpus

Tomaž Erjavec
Department of Intelligent Systems
Jožef Stefan Institute
Jamova 39
SI-1000 Ljubljana
Slovenia
tomaz.erjavec@ijs.si, http://nl.ijs.si/et/

Keywords: natural language processing, corpus annotation, multilinguality, lexicon extraction

Received: July 10, 2002

With increasing amounts of text being available in electronic form, it is becoming relatively easy to obtain digital texts together with their translations. The paper presents the processing steps necessary to compile such texts into parallel corpora, an extremely useful language resource. Parallel corpora can be used as a translation aid for second-language learners, for translators and lexicographers, or as a data-source for various language technology tools. We present our work in this direction, which is characterised by the use of open standards for text annotation, the use of publicly available third-party tools and wide availability of the produced resources. Explained is the corpus annotation chain involving normalisation, tokenisation, segmentation, alignment, word-class syntactic tagging, and lemmatisation. Two exploitation results over our annotated corpora are also presented, namely a Web concordancer and the extraction of bi-lingual lexica.

1 Introduction

With more and more text being available in electronic form, it is becoming easy to obtain large quantities of digital texts and to process them computationally. If a collection of such texts is chosen according to specific criteria and is consistently and correctly marked-up [19], it is said to be a text corpus. Such corpora can be used for a variety of different purposes [17], from empirically grounded linguistic studies, lexicography and language teaching, to providing datasets for language technology programs for terminology extraction, word-sense disambiguation, etc.

Collected and uniformly encoded collections of texts are already quite useful, but it is the addition of (linguistic) markup that makes corpora a prime resource for language exploration. As will be seen, we view the process of compiling a corpus as one of annotation accrual: starting from a plain text, we successively add mark-up, thereby enriching the information contained in the corpus. This markup is typically automatically produced, but can be hand validated and, in a cyclic process, can serve for inductive programs to learn better models of the language with which to annotate subsequent generations of corpora. The added annotation enables the people and software using the corpus to employ extra levels of abstraction, leading to better exploitation results.

If monolingual corpora are already useful for a variety of purposes, it is multilingual corpora that open the way for empirical study of the translation process. Especially valuable are so called parallel corpora, i.e., corpora consisting of texts together with their translation into one or many languages. They can be used directly as translation aids for humans or can provide data for the automatic induction translation resources (lexica) and software (machine translation).

In this paper we explore the process of compilation and exploitation of such parallel corpora, grounding the discussion in our experience with two annotated parallel corpora: the 7-language MULTEXT-East corpus [6, 9], which contains the novel "1984" by G. Orwell (100,000 words per language) and has had its annotation manually validated; and the larger (500,000 words per language) automatically annotated Slovene-English IJS-ELAN corpus [8, 10].

Our work is characterised by the use of open standards for text annotation and the use of publicly available third-party tools. We claim that it is better to invest labour into producing high-quality annotated corpora than in trying to build from scratch tools for such annotation. Unlike local and idiosyncratic software, linguistic resources encoded in a standard manner will be sooner useful to other research groups. Such largesse aside, there also exist more and more (statistical or symbolic) machine learning programs that are able to induce language models from pre-annotated corpora. They are typically more robust and, with large enough training sets, might even perform better than hand crafted systems.

The rest of this paper is structured as follows: Section 2 introduces standards for corpus annotation, which are then used in the examples in the remainder of the paper; Section 3 enumerates the basic (pre-linguistic) processing steps in-

volved in the compilation of a corpus; Section 4 details the more complex word-level syntactic annotation, which is performed by a trainable algorithm; Section 5 turns to the exploitation of corpora and gives two examples: an on-line concordancer, and an experiment in bi-lingual lexicon extraction; Section 6 gives conclusions and directions for further research.

2 Encoding Standards

While the question of the encoding format for corpora and other language resources might seem incidental to the main task of producing and exploiting the corpora, it has long been known that the proliferation of data formats and annotation schemes, many of which are proprietary and poorly documented, is a significant bottleneck for resource sharing, re-use and longevity. There have been therefore a number of attempts to standardise the encoding of various language resources, among them corpora.

All such standardisation efforts have as their basis the ISO Standard Generalized Markup Language, SGML, or, more recently, the W3C Extensible Markup Language, XML [26], a simplified form of SGML meant primarily for interchange of data on the Web. SGML and XML are metalanguages, that is, a means of formally describing a language, in this case, a markup language. They do thus not directly define a particular set of tags but rather enable the mechanisms for defining such sets for particular purposes, e.g., for the encoding of corpora.

The best known and widely used set of conventions for encoding a wide variety of texts, among them corpora, are the SGML-based Text Encoding Initiative Guidelines (TEI), the most recent version of which is also XML compliant [20]. The TEI consist of the formal part, which is a set of SGML/XML Document Type Definition fragments, and the documentation, which explains the rationale behind the elements available in these fragments, as well as giving overall information about the structure of the TEI. The DTD fragments are combined to suit an individual project, and, if necessary, also extended or modified. We have used parametrisations of TEI for both the MULTEXT-East corpus as well as for the IJS-ELAN corpus. TEI encoded examples from these corpora will be used in the rest of this paper.

3 Pre-processing the Corpus

In this section we deal with the basic processing steps involved in normalising and marking-up the corpus, in order to make it minimally useful for exploitation and to prepare it for further annotation. The steps we outline below usually proceed in sequence, and can be, for the most part, performed automatically although — given the unconstrained nature of texts — are likely to be less than 100% accurate. While further development of tools and associated resources lowers the error rate, manual validation might still be necessary for high-quality corpora.

The texts constituting a corpus can be collected from a variety of sources and so usually come in a range of formats. The first step in corpus preparation is therefore invariably the normalisation of the texts into a common format. Usually custom filters — written in pattern matching languages such as Perl — are employed to, on the one hand, normalise the character sets of the documents and, on the other, to remove and convert the formatting of the originals.

3.1 Character sets

As far as character sets go the corpus compliers have a few options at their disposal. One possibility is to use — at least for European languages — an 8-bit encoding in the corpus, preferably a standard one, e.g., ISO 8859 Latin 2 for encoding Slovene texts. While the advantage is that the corpus texts are immediately readable — given that we have installed the appropriate fonts — the disadvantage is that a number of processing applications do not handle well 8 bit characters and, more importantly, that it is impossible to mix languages that use different character sets; this is, of course, a special concern in multilingual corpora.

Until a few years ago the standard solution involved translating the non-ASCII characters of the original texts into ISO-mandated SGML entities. SGML (and XML) entities are descriptive names, somewhat like macros, which the application then substitutes for their values. In our case, the names are of the characters in question; so, for example, the Slovene letter č is written as the entity `č` (small c with a caron), where ampersand starts an entity and semicolon ends it. Such entities for characters are defined in public entity sets, for the case of `č` in `ISO 8879:1986//ENTITIES Added Latin 2//EN`, i.e., the added entity set for encoding the Latin alphabets of Eastern European languages. Similar entity sets exist for Western European languages (Latin 1), for Greek, Russian and non-Russian Cyrillic, as well as for mathematical relations, publishing symbols, etc. The advantage of using entities is their robustness: because they are encoded in (7 bit) ASCII characters they are portable, and can be read on any platform. For the application, the SGML processor then translates them into the desired encoding via their definitions.

With the advent of XML, this solution has somewhat lost its currency. While entities are supported in XML, the default character set of XML is Unicode, which, because a character can be encoded in two bytes, is sufficient to accommodate most of the characters of the world's scripts; so, this is in fact the only solution for Easter language scripts, such as Kanji. But while using Unicode makes for a theoretically good solution, practice still lags behind, with many applications not yet being Unicode-aware. In our corpora we have therefore chosen to use SGML entities for the representation of non-ASCII characters.

3.2 Markup of gross document structure

Various encodings of input texts also encode the structure of the documents in vastly different and often inconsistent manners. This structure includes such things as divisions of the text, headers and titles, paragraphs, tables, footnotes, emphasised text, etc. In general it is a very hard task to correctly and completely transform this structure into descriptive TEI markup. For many natural language processing applications this might not even be necessary, as the task here isn't to preserve the layout of the text, but only the information that is relevant to correctly classify the text and to enable linguistic processing. To illustrate a case of relatively detailed structure markup, we give, in Figure 1, a TEI encoded example from the MULTEXT-East corpus.

```
<text lang="en" id="Oen.">
 <body>
  <div id="Oen.1" type="part">
   <head>First part</head>
   <div id="Oen.1.1" type="chapter">
    <head>I</head>
    <p id="Oen.1.1.1">
It was a bright cold day in April, and the
clocks were striking thirteen. Winston Smith,
his chin nuzzled into his breast in an effort
to escape the vile wind, slipped quickly
through the glass doors of Victory Mansions,
though not quickly enough to prevent a swirl
of gritty dust from entering along with him.
    </p>
    ...
```

Figure 1: Structure markup in TEI

3.3 Header Information

A corpus is typically composed of a large number of individual texts. For analysing the corpus or for choosing a particular subset out of it, it is vital to include information about the texts into the corpus. Of course, the corpus as a unit must also be documented. The TEI provides a header element, `<teiHeader>` expressly meant to capture such meta-data. The TEI header contains detailed information about the file itself, the source of its text, its encoding, and revision history.

The information in a text or corpus header can be — depending on the number of texts and the regularity of provenance – either inserted manually or automatically. Given that the corpora discussed in this paper have relatively few components, their headers have been entered manually via an SGML/XML aware text editor.

3.4 Tokenisation and segmentation

The next step in corpus preparation already involves basic linguistic analysis, namely isolating the linguistic units of the text, i.e., words and sentences. The identification of words — and punctuation marks — is usually referred to as tokenisation, while determining sentence boundaries goes by the name of segmentation.

On the face of it, determining what is a word and what a sentences might seem trivial. But correctly performing these tasks is fraught with complexities [12] and the rules to perform them are, furthermore, language dependent. So, while sentences end with full stops, not every full stop ends a sentence, as with, e.g., *Mr.*; and if some abbreviations will never end sentences, e.g., *e.g.*, other almost invariably will, e.g., *etc.* Correct tokenisation is complex as well; punctuation marks can sometimes be part of a word, as is the case with abbreviations and, say, Web addresses. Some domains, e.g., biomedicine have "words" with an especially complex internal structure, e.g., *Ca(2+)-ATPase.*

In the process of tokenisation various types of words and punctuation symbols must be recognised and this information can be retained in the markup, as it can be potentially useful for further processing. In Figure 2 we give an example of a segmented and tokenised text from the IJS-ELAN corpus, where the `type` attribute expresses such information on words.

```
<seg id="ecmr.en.17">
  <w>Euromoney</w><w type="rsplit">'s</w>
  <w>assessment</w> <w>of</w> <w>economic</w>
  <w>changes</w> <w>in</w> <w>Slovenia</w>
  <w>has</w> <w>been</w> <w>downgraded</w>
  <c type="open">(</c><w>page</w>
  <w type="dig">6</w><c type="close">)</c>
  <c>.</c>
</seg>
```

Figure 2: Segmentation and tokenisation in TEI

While it is possible to write a tokeniser and segmenter using a general purpose computer language there also exist freely available tools for this purpose. We have extensively used the MULTEXT tools [3], which, however, no longer seem to be maintained.

Fortunately, other good choices exist, e.g., the text tokeniser tool, LT TTT [13], which is freely distributed for academic purposes as binaries for Sun/Solaris. LT TTT is based on XML and incorporates a general purpose cascaded transducer which processes an input stream deterministically and rewrites it according to a set of rules provided in a grammar file, typically to add mark-up information. With LT TTT come grammars to segment English texts into paragraphs, segment paragraphs into words, recognise numerical expressions, mark-up money, date and time expressions in newspaper texts, and mark-up bibliographical information in academic texts. These grammars are accompanied by detailed documentation which allows altering the grammars to suit particular needs or develop new rule sets. While we have already successfully used LT TTT for processing English texts, the localisation of its grammars to Slovene remains still to be done.

3.5 Sentence Alignment

An extremely useful processing step involving parallel corpora is the alignment of their sentences. Such alignment would be trivial if one sentence were always translated into exactly one sentence. But while this may hold for certain legal texts, translators — either due to personal preference, or because different languages tend towards different sentence lengths — often merge or split sentences, or even omit portions of the text. This makes sentence alignment a more challenging task.

High-quality sentence alignment is still an open research question and many methods have been proposed [23] involving the utilisation of e.g., bilingual lexicons and document structure. Still, surprisingly good results can be achieved with a language-independent and knowledge poor method, first discussed in [11]. The alignment algorithm here makes use of the simple assumption that longer sentences tend to be translated into longer sentences, and shorter into shorter. So, if we come across, e.g., two short sentences in the original, but one long one in the translation, chances are, the two have been merged. Hence the input to this aligner is only the lengths of the respective sentences in characters and the program, with an algorithm known as dynamic time warping, finds the best fit for the alignments, assuming the valid possibilities are 1-2, 0-1, 1-1, 1-0, 2-1, and 2-2.

Several public implementations of this algorithm exist; we have used the so called Vanilla aligner [4], implemented in C and freely available in source code.

The quality of the automatic alignment is heavily dependent on the manner of translation but, in any case, is seldom perfect. For our corpora we have manually validated the alignments via a cyclic process, with initial errors of alignment corrected and the text then being automatically re-aligned.

The end result is the sentence aligned text; the alignment information might then be encoded in one of several ways. One possibility is to encode the alignments in separate documents, where only pairs of references to sentence IDs are stored. Figure 3 gives a hypothetical Slovene-English alignment span illustrating the syntax and types (one, many, zero) of the alignment links. The first link encodes an 1-1 alignment, the second a 2-1 and the third an 1-0 alignment.

```
<link xtargets="Osl.1.1 ; Oen.1.1"/>
<link xtargets="Osl.1.2 Osl.1.3 ; Oenl.1.2"/>
<link xtargets="Osl.1.4 ; "/>
```

Figure 3: Example of stand-off bilingual alignment

4 Word-class syntactic tagging

It is well known that the addition of word-level syntactic tags adds significantly to the value of a corpus [22]. Knowing the part-of-speech and other morphosyntactic features, such as number, gender and case, helps to lexically determine the word and serves as a basis for further syntactic or semantic processing. In a parallel corpus such annotations can also act as guides for automatic bilingual lexicon extraction or example based machine translation.

The flip side of morphosyntactic tagging is lemmatisation, i.e., annotating the words in the corpus with their lemmas or base forms. Such a normalisation of the word-forms is useful in concordancing the corpus and in identifying translation equivalents. While lemmatisation for English is relatively simple (although wolves and oxen complicate matters) it is a more difficult task in Slovene, which is a heavily inflecting language.

Manual annotation is extremely expensive, so corpora are typically tagged and lemmatised automatically. Below we explain our work on the IJS-ELAN corpus, where we used the statistical tagger TnT which had been trained on the MULTEXT-East parallel corpus, and initial results improved in various ways.

4.1 The TnT tagger

Trainable word-class syntactic taggers have reached the level of maturity where many models and implementations exist, with several being robust and available free of charge. Prior to committing ourselves to a particular implementation, we conducted an evaluation (on Slovene data) of a number of available taggers [7]. The results show that the trigram-based TnT tagger [1] is the best choice considering accuracy (also on unknown words) as well as efficiency. TnT is freely available under a research license, as an executable for various platforms.

The tagger first needs to be trained on an annotated corpus; the training stage produces a table with tag tri- bi- and uni-grams and a lexicon with the word forms followed by their tag ambiguity classes, i.e., the list of possible tags, together with their frequencies. Using these two resources, and possibly a backup lexicon, tagging is then performed on unannotated data.

4.2 The training corpus

The greatest bottleneck in the induction of a quality tagging model for Slovene is lack of training data. The only available hand-validated tagged corpus is the Slovene part of the MULTEXT-East corpus, which is annotated with validated context disambiguated morphosyntactic descriptions and lemmas.

These morphosyntactic descriptions (MSDs) for Slovene — and six other languages, English among them — were developed in the MULTEXT-East project [6, 9]. The MSDs are structured and more detailed than is commonly the case for English part-of-speech tags; they are compact string representations of a simplified kind of feature structures. The first letter of an MSD encodes the part of speech, e.g., Noun or Adjective. The letters following the PoS give the values of the position determined

attributes. So, for example, the MSD `Ncfpg` expands to `PoS:Noun, Type:common, Gender:feminine, Number:plural, Case:genitive.` In case a certain attribute is not appropriate for the particular combination of features, or for the word in question, this is marked by a hyphen in the attribute's position.

To illustrate the properties of the training corpus, as well as the difference between Slovene and English we give in Table 1 the number of word tokens in the corpus, the number of different word types in the corpus (i.e., of word-forms regardless of capitalisation or their annotation), the number of different context disambiguated lemmas, and the number of different MSDs. The inflectional nature of Slovene is evident from the larger number of distinct word-forms and especially MSDs used in the corpus.

	English	Slovene
Words	104,286	90,792
Forms	9,181	16,401
Lemmas	7,059	7,903
MSDs	134	1,023

Table 1: Inflection in the MULTEXT-East corpus

4.3 Tagging Slovene

Unsurprisingly, the tagging produced by TnT trained only on the MULTEXT-East corpus had quite a low accuracy; this can be traced both to the inadequate induced lexicon, as more than a quarter of all word tokens in IJS-ELAN were unknown, as well as to n-grams applied to very different text types than what was used for training. To offset these shortcomings we employed two methods, one primarily meant to augment the n-grams, and the other the lexicon.

It is well known that "seeding" the training set with a validated sample from the texts to be annotated can significantly improve results. We selected a sample comprising 1% of the corpus segments (approx. 5,000 words) evenly distributed across the whole of the corpus. The sample was then manually validated and corrected, also with the help of Perl scripts, which pointed out certain typical mistakes, e.g., the failure of case, number and gender agreement between adjectives and nouns. The tagger n-grams were then re-learned using the concatenation of the validated ELAN sample with the Slovene MULTEXT-East corpus.

It has also been shown [14] that a good lexicon is much more important for quality tagging of inflective languages than the higher-level models, e.g., bi- and tri-grams. A word that is included in a TnT lexicon gains the information on its ambiguity class, i.e., the set of context-independent possible tags, as well as the lexical probabilities of these tags.

The Slovene part of the ELAN corpus was therefore first lexically annotated, courtesy of the company Amebis, d.o.o., which also produces the spelling checker for Slovene Word. The large lexicon used covers most of the words in the corpus; only 3% of the tokens remain unknown. This lexical annotation includes not only the MSDs but also, paired with the MSDs, the possible lemmas of the word-form.

We first tried using a lexicon derived from these annotations directly as a backup lexicon with TnT. While the results were significantly better than with the first attempt, a number of obvious errors remained and additional new errors were at times introduced. The reason turned out to be that the tagger is often forced to fall back on uni-gram probabilities, but the backup lexicon contains only the ambiguity class, with the probabilities of the competing tags being evenly distributed. So, TnT in effect often assigned a random tag from the ones available, leading to poor results. To remedy the situation, a heuristic was used to estimate the lexical frequencies of unseen words, taking as the basis the known frequencies from similar ambiguity classes taken from the training corpus.

Using this lexicon and the seeded model we then re-tagged the Slovene part of the IJS-ELAN corpus. Manually validating a small sample of the tagged corpus, consisting of around 5,000 words, showed that the current tagging accuracy is about 93%.

As had been mentioned, the lexical annotations included lemmas along with the MSDs. Once the MSD disambiguation had been performed it was therefore trivial to annotate the words with their lemmas. But while all the words, known as well as unknown, have been annotated with an MSD, we have so far not attempted to lemmatise the approximately 3% of the corpus words which are unknown.

The results of the tagging were encoded in the corpus as attribute values of the TEI `<w>` element. To illustrate, we give in Figure 4 an example sentence from the corpus: *Razlike med metropolitanskimi centri in njihovim zaledjem so ogromne. / The differences between the metropolitan centres and their hinterlands are enormous.*

```
<seg id="ecmr.sl.92">
  <w ana="Ncfpn" lemma="razlika">Razlike</w>
  <w ana="Spsi" lemma="med">med</w>
  <w ana="Aopmpi">metropolitanskimi</w>
  <w ana="Ncmpi" lemma="center">centri</w>
  <w ana="Ccs" lemma="in">in</w>
  <w ana="Ps3fpdp" lemma="njihov">njihovim</w>
  <w ana="Ncnsi" lemma="zaledje">zaledjem</w>
  <w ana="Vcip3p--n" lemma="biti">so</w>
  <w ana="Afpfpn" lemma="ogromen">ogromne</w>
  <c ctag=".">.</c>
</seg>
```

Figure 4: Linguistic annotation in the corpus

4.4 Tagging English

Tagging the English part of the corpus with the MULTEXT-East MSDs was also performed with TnT, using the English part of the MULTEXT-East corpus as the training set. However, automatic tagging with this model is bound to

contain many errors, although less than for Slovene, given the much smaller tagset.

Rather than try to improve the accuracy of our own tagging, we opted for additional annotations with other, better, models and also with a better known tagset, namely (variants of) the one used in the Brown corpus [15]. For the additional annotation of the English part we combined the output of two taggers.

First, the TnT tagger distribution already includes some English models. We chose the one produced by training on the concatenation of the Brown corpus with the Wall Street Journal corpus; this training set contained approximately 2.5 million tokens and distinguishes 38 different word-tags.

Second, we used QTag [16], which is also freely available probabilistic tri-gram tagger, although the underlying algorithm differs from that employed by TnT. The English model of QTag uses a similar, although not identical, tagset to the TnT English one. QTag is also offered via an email service, which in addition to tagging the texts, also lemmatises them; we used this lemmatisation to annotate the corpus.

To illustrate the tagging of the English part, we give an example in Figure 5.

```
<seg id="ecmr.en.92" corresp="ecmr.sl.92">
  <w ana="Pt3" ctag="EX EX"
     lemma="there">There</w>
  <w ana="Vmip-p" ctag="VBP BER"
     lemma="be">are</w>
  <w ana="Afp" ctag="JJ JJ"
     lemma="huge">huge</w>
  <w ana="Ncnp" ctag="NNS NNS"
     lemma="difference">differences</w>
  <w ana="Sp" ctag="IN IN"
     lemma="between">between</w>
  <w ana="Dd" ctag="DT DT"
     lemma="the">the</w>
  <w ana="Ncnp" ctag="?NNS NNS"
     lemma="centre">centres</w>
  <w ana="Cc-n" ctag="CC CC"
     lemma="and">and</w>
  <w ana="Ds3---p" ctag="PRP$ PP$"
     lemma="they">their</w>
  <w ana="Afp" ctag="JJ JJ"
     lemma="suburban">suburban</w>
  <w ana="Ncnp" ctag="NNS NNS"
     lemma="area">areas</w>
  <c ctag=".">.</c>
</seg>
```

Figure 5: Linguistic annotation in the English part

5 Utilising the Corpus

The IJS-ELAN corpus was thus encoded in XML/TEI, segmented, tokenised, aligned and tagged with morphosyntactic descriptions and lemmas. It was now time to turn to exploiting the corpus. Given that the texts that are included in the corpus do not have copyright restrictions (they are mostly publications of the Slovene government), it was trivial to ensure one type of "exploitation", namely to simply make the complete corpus freely available for downloading: it can be accessed at http://nl.ijs.si/elan/.

In this section we discuss two methods of utilising the corpus. The first is geared directly towards human usage, and has to do with making the corpus available for sophisticated on-line searching. The second employs a statistics-based tool that extracts a bi-lingual lexicon from the corpus.

5.1 Web concordancing

For our corpora we have developed an on-line concordancing system. This Web concordancer comprises a set of HTML pages, a simple Perl CGI script and a corpus processing back-end. The back-end is the CQP system [2], a fast and robust program, which freely available for research purposes as binaries for a number of platforms. CQP supports parallel corpora and incorporates a powerful query language that offers extended regular expressions over positional (e.g., word, lemma, MSD) and structural (e.g., <text>, <p>, <seg>) attributes.

The Web page of the concordancer contains various input fields and settings available to the user. The settings and options have associated hyperlinks, and clicking on them gives help on the particular topic. So, for example, the Display setting affects how the search results are presented: the Bilingual Display shows the hits in the target corpus, followed by their aligned segment in the translation; the KWIC Display shows the results in the familiar key-word in context format; and Word List Display gives a list of word types found in the corpus, together with their frequencies. The last option makes the most sense with fuzzy queries.

The result of the query can also be refined by specifying an additional query on the aligned corpus. This constraint can be either required or forbidden. The latter option is useful when exploring 'unexpected' translations.

The on-line concordancer has been in used at the Department of Translation and Interpreting at the University of Ljubljana for different purposes such as contrastive analysis, translation evaluation, translation-oriented lexical and terminology studies, discourse analysis, etc. The methodological aims of this work were, on the one hand, to help students gain a deeper understanding of living language and remember things they discover themselves, and, on the other, to enable them to become skilled and critical users of corpora for translation purposes.

The concordancer is also being used by translators, esp. by the volunteers of LUGOS, the Linux Users' Group of Slovenia, that are localising Linux documentation, e.g., the HOWTOs and the KDE desktop environment. As the IJS-ELAN corpus contains a whole book on Linux and the PO localisation files, it can be a welcome source of terminology translations.

5.2 Lexicon extraction

We have also performed an initial experiment in automatic bi-lingual lexicon extraction from the corpus. Extracting such lexica is one the prime uses of parallel corpora, as manual construction is an extremely time consuming process yet the resource is invaluable for lexicographers, terminologists, translators as well as machine translation systems.

A number of similar experiments had already been performed on the IJS-ELAN corpus [24, 5, 25], using a number of different tools. The software we have used here is the PWA system [21], which is a collection of tools for automatically finding translation equivalents in sentence aligned parallel corpora. The output of the system is, inter alia, a list of word token correspondences (i.e., translations in the text), a list of word type correspondences (i.e., a lexicon) and lists of monolingual collocations (i.e., a terminological glossary). The system is freely available under a research license as a binary for various platforms.

For the data, we have used one of the elements of the IJS-ELAN corpus, namely the book "Linux Installation and Getting Started" by Matt Welsh et al., and translated into Slovene by Roman Maurer. The book contains 2 × 5,773 aligned sentence segments, with the English original having 91,526 and the Slovene translation 81,955 word tokens.

For lexicon extraction we have not used the word-forms directly but rather the lemmas (where defined) of the words in question. This normalises the input and abstracts away from the rich inflections of Slovene, which would cause PWA to treat different forms of the same word as different words. Secondly, we reduced the input to the system to only adjectives, nouns and punctuation symbols. The reasoning behind this is that most useful (terminological) lexical correspondences will be noun phrases, and eliminating the other word classes reduces the chance of spurious translation correspondences. We have included punctuation signs in order to break up long stretches of nouns, which otherwise tend to get analysed as collocations.

For this input data the PWA system took 15 minutes (on a Pentium laptop) to produce the results, i.e., the list of token correspondences totalling 22,880 items, the lexicon containing 2,850 entries, and a list of collocations with 1,329 entries.

To illustrate, we show in Figure 6 data for one sentence from the text; first we give the sentence and its translation, then the equivalent input to the system, and finally the computed translation equivalents.

Most posited translations are correct, but some are less than perfect. While the system correctly identifies translation equivalents for `linux` and `system`, it misses out on the larger collocation `linux system`, and similarly for `user program` and `development tool`. The main shortcomming of the output for the example sentence is the suggested translation equivalent for `source code`, as it lacks the noun `koda`. But despite these omissions, the result is already quite useful.

English sentence: *In addition, all of the source code for the Linux system, including the kernel, device drivers, libraries, user programs, and development tools, is freely distributable.*

Slovene sentence: *Dodatno je dostopna in prosto razširljiva še vsa izvorna koda sistema Linux, vključno z jedrom, gonilniki naprav, knjižnicami, uporabniškimi programi in razvojnimi orodji.*

English input:
```
addition , source code linux
system , kernel , device driver ,
library , user program , development
tool , distributable .
```

Slovene input:
```
dostopen razširljiva izvoren
koda sistem Linux , jedro , gonilnik
naprava , knjižnica , uporabniški
program razvojen orodje .
```

Output English → Slovene translations:

source code → *izvoren*
linux → linux
system → sistem
kernel → jedro
device driver → gonilnik
library → knjižnica
user → uporabniški
program → program
development → razvojen
tool → orodje

Figure 6: Automatically extracted translation equivalents

6 Conclusions and Further Research

The paper presented the processing steps involved in building and exploiting parallel corpora and introduced the tools necessary to accomplish this task. We have tried to show how third-party publicly available software is sufficient to operationalise the complete tool chain, and how language resources can be built in a cyclic manner, with initial annotations enabling the production of language models which, in turn, enable refinement and further annotation.

The text processing model outlined in this article is especially useful in an academic environment: the software to implement it is free, and can thus be easily acquired by cash-strapped university departments. The building as well as the exploitation of the resources can be profitably used for teaching purposes, be it for computer science courses on natural language processing, or for linguistic courses on the use of language technology. As has been shown, the resources can also be used directly, by helping translators or language students make use of bi-lingual data.

As far as corpus compilation goes, our further work can be divided into two areas. The first is, of course, the acquisition of more texts, and hence the production of larger corpora. The second, and scientifically more challenging, is the addition of further markup. In the paper we have discussed only basic linguistic markup. Useful further annotations include terms (either single or multiword), named entities (proper names, acronyms, dates, etc.), chunks (esp.

noun phrases), and phrase structure (i.e., full syntactic analysis). Each of these areas has been the subject of much research but, so far, not yet attempted for Slovene.

On the exploitation side, in addition to carrying further our research on lexicon extraction, we plan to experiment with statistical machine translation, in particular to use the freely available system EGYPT [18] with the IJS-ELAN corpus as the training set.

Acknowledgements

Thanks goes to Amebis, d.o.o., for lexically annotating version 2 the Slovene part of the IJS-ELAN corpus and to Jin-Dong Kim and two anonymous reviewers for reading the draft of this paper.

References

[1] Thorsten Brants. TnT - A Statistical Part-of-Speech Tagger. In *Proceedings of the Sixth Applied Natural Language Processing Conference ANLP-2000*, pages 224–231, Seattle, WA, 2000. http://www.coli.uni-sb.de/~thorsten/tnt/.

[2] Oliver Christ. A Modular and Flexible Architecture for an Integrated Corpus Query System. In *Proceedings of COMPLEX '94: 3rd conference on Computational Lexicography and Text Research*, pages 23–32, Budapest, Hungary, 1994. http://www.ims.uni-stuttgart.de/projekte/CorpusWorkbench/.

[3] Philippe Di Cristo. MtSeg: The Multext multilingual segmenter tools. MULTEXT Deliverable MSG 1, Version 1.3.1, CNRS, Aix-en-Provence, 1996. http://www.lpl.univ-aix.fr/projects/multext/MtSeg/.

[4] Pernilla Danielsson and Daniel Ridings. Practical presentation of a "vanilla" aligner. In *TELRI Newsletter No. 5*. Institute fuer Deutsche Sprache, Mannheim, Mannheim, 1997. http://nl.ijs.si/telri/Vanilla/doc/ljubljana/.

[5] Gael Dias, Špela Vintar, Sylvie Guillore, and Jose Gabriel Pereira Lopes. Normalising the IJS-ELAN Slovene-English Parallel Corpus for the Extraction of Multilingual Terminology. In *Computational Linguistics in the Netherlands 1999, Selected Papers from the Tenth CLIN Meeting*, pages 29–40, Utrecht, 1999. UILOTS.

[6] Ludmila Dimitrova, Tomaž Erjavec, Nancy Ide, Heiki-Jan Kaalep, Vladimír Petkevič, and Dan Tufiş. Multext-East: Parallel and Comparable Corpora and Lexicons for Six Central and Eastern European Languages. In *COLING-ACL '98*, pages 315–319, Montréal, Québec, Canada, 1998. http://nl.ijs.si/ME/.

[7] Sašo Džeroski, Tomaž Erjavec, and Jakub Zavrel. Morphosyntactic Tagging of Slovene: Evaluating PoS Taggers and Tagsets. In *Second International Conference on Language Resources and Evaluation, LREC'00*, pages 1099–1104, Paris, 2000. ELRA.

[8] Tomaž Erjavec. The ELAN Slovene-English Aligned Corpus. In *Proceedings of the Machine Translation Summit VII*, pages 349–357, Singapore, 1999. http://nl.ijs.si/elan/.

[9] Tomaž Erjavec. Harmonised Morphosyntactic Tagging for Seven Languages and Orwell's 1984. In *6th Natural Language Processing Pacific Rim Symposium, NLPRS'01*, pages 487–492, Tokyo, 2001. http://nl.ijs.si/ME/V2/.

[10] Tomaž Erjavec. The IJS-ELAN Slovene-English Parallel Corpus. *International Journal of Corpus Linguistics*, 7(1):1–20, 2002. http://nl.ijs.si/elan/.

[11] William Gale and Ken W. Church. A program for aligning sentences in bilingual corpora. *Computational Linguistics*, 19(1):75–102, 1993.

[12] Greg Grefenstette and Pasi Tapanainen. What is a word, what is a sentence? problems of tokenization. In *Proceedings of The 3rd International Conference on Computational Lexicography (COMPLEX'94)*, pages 79–87. Research Institute for Linguistics, Hungarian Academy of Sciences, Budapest, 1994.

[13] Claire Grover, Colin Matheson, Andrei Mikheev, and Marc Moens. LT TTT - A Flexible Tokenisation Tool. In *Second International Conference on Language Resources and Evaluation, LREC'00*, 2000. http://www.ltg.ed.ac.uk/software/ttt/.

[14] Jan Hajič. Morphological Tagging: Data vs. Dictionaries. In *ANLP/NAACL 2000*, pages 94–101, Seatle, 2000.

[15] Henry Kučera and William Nelson Francis. *Computational Analysis of Present Day American English*. Brown University Press, Providence, Rhode Island, 1967.

[16] Oliver Mason. Qtag — a portable probabilistic tagger, 1998. http://www-clg.bham.ac.uk/QTAG/.

[17] Tony McEnery and Andrew Wilson. *Corpus Linguistics*. Edinburgh University Press, 1996.

[18] Franz Josef Och and Hermann Ney. Statistical Machine Translation. In *Proceedings of the European Association for Machine Translation Workshop, EAMT'00*, pages 39–46, Ljubljana, Slovenia, May 2001. http://nl.ijs.si/eamt00/.

[19] John Sinclair. Corpus typology. EAGLES DOCUMENT EAG–CSG/IR–T1.1, Commission of the European Communities, 1994.

[20] C. M. Sperberg-McQueen and Lou Burnard, editors. *Guidelines for Electronic Text Encoding and Interchange, The XML Version of the TEI Guidelines.* The TEI Consortium, 2002. http://www.tei-c.org/.

[21] Jorg Tiedemann. Extraction of translation equivalents from parallel corpora. In *Proceedings of the 11th Nordic Conference on Computational Linguistics*, Center for Sprogteknologi, Copenhagen, 1998. http://numerus.ling.uu.se/~corpora/plug/pwa/.

[22] Hans van Halteren, editor. *Syntactic Wordclass Tagging*. Kluwer Academic Publishers, 1999.

[23] Jean Véronis, editor. *Parallel Text Processing: Alignment and Use of Translation Corpora.* Kluwer Academic Publishers, 2000.

[24] Špela Vintar. A Parallel Corpus as a Translation Aid: Exploring EU Terminology in the ELAN Slovene-English Parallel Corpus. In *Proceedings of the 34th Colloquium of Linguistics*, Germersheim, Frankfurt, 1999. Peter Lang Verlag.

[25] Špela Vintar. Using Parallel Corpora for Translation-Oriented Term Extraction. *Babel Journal*, 47(2):121–132, 2001.

[26] W3C. Extensible markup language (XML) version 1.0. URL, 1998. http://www.w3.org/TR/1998/REC-xml-19980210.

On Formal Aspects of Zooming in Geographic Maps

Daniele Frigioni, Laura Tarantino and Tania Di Mascio
Dipartimento di Ingegneria Elettrica
Università degli Studi dell'Aquila
I-67040 Monteluco di Roio, L'Aquila - Italy
frigioni@ing.univaq.it, laura@ing.univaq.it, tania@ing.univaq.it

Keywords: Geographic Map, Poset, Human-Computer Interaction

Received: February 10, 2002

Partitions have been identified as a key concept for perception and understanding of space, and have been regarded as a primary tool for spatial analysis tasks in geographic applications. We discuss here some results regarding the exploration of geographic maps structured as nested partitions. In particular, we present a zooming model based on a Level-Of-Detail (LOD) approach, aimed at visualizing sequences of gradually simplified representations of a given area hierarchically structured in a poset. We first describe a set of basic zooming primitives defined as transitions betweens maps at different LOD, and study properties of the corresponding map space. We then introduce the notion of multiple zooming, *define two new multiple zooming primitives on top of the basic ones, extend the map space, and prove some theoretical results on the new primitives.*

1 Introduction

In this paper we report some theoretical results achieved in the framework of a project concerning the design of a visual interaction environment for Geographic Information Systems (GISs). GISs deal with storing, querying, manipulating and displaying geographic information (see, e.g., [2]). The core of a GIS is a spatial database management system. A spatial database consists of a *spatial component* encoding the geometric aspects of the physical objects under consideration (e.g., location, shape, size), and of a *thematic component* containing information about the non-geometric properties of the realm of interest (see, e.g., [15] for foundations of spatial databases). A GIS normally includes also modules devoted to application specific tasks, such as map production, spatial analysis, and data visualization. We restrict here our attention to primitives aimed at exploring the spatial component of the database.

Detail filtering in geographic maps. Technological developments in automated data capturing tools allow the collection of huge volumes of geographic data representing areas of interest with extreme accuracy. As a consequence, manipulation and rendering of complex geographic datasets may require processing resources beyond the power of state-of-the-art computers, hence introducing the necessity of detail filtering during visualization and interaction tasks. The basic idea is to dynamically adapt the resolution to the needs of specific interaction tasks, possibly taking into account user's interest in displayed data (which might imply to vary the resolution over different areas of the map).

To achieve this effect, the concept of Level-of-Detail (LOD) has been introduced, sometimes summarized as "*always use the best resolution you need - or you can afford - and never use more than that*" [6]. Of course the principle has to be characterized to the needs of the specific application, interaction task or visualization strategy. Anyhow, despite application specific differences, there is common ground among LOD models: all aims at defining sequences of gradually simplified representations of the same area that get structured either into a multi-layer model or into a hierarchical model described by a tree [16]. In the framework of a LOD approach, the interaction process can be formalized through a *map space* and *rules for traversing it* corresponding to transitions between maps at different levels of detail.

Several conceptual approaches can be used for the formalization of space and of spatial properties. Not only does the adoption of a particular model influence the type of data that can be used, but it also determines the kind of spatial analysis that can later be undertaken. The two extremes in the range of conceptual approaches are the *continuous field approach* (that views attributes of interest as varying over the space as some continuous mathematical function or field) and the *entity-based approach* (that perceives space as being occupied by entities described by their properties and mapped using a coordinate system) [2].

In the case of continuous fields, data analysis concerns the spatial properties of the fields; the geographical space is discretized into sets of single spatial units (such as square, triangular, or hexagonal cells), or into irregular triangles or polygons, tessellated to form geographic representations. A LOD representation hence consists of a collection of meshes of different sizes, each representing the object at a different resolution [16].

In the case of entities, data retrieval and analysis may concern: attributes (i.e., the nature of the entities), geographic location (i.e., spatial information about the entities), and topology (i.e., information about the relative positions of the entities). The LOD approaches must then provide different solutions to deal with attributes, geographic locations, and topology. Treatment of topological information encoded in partitions is the main concern of our work.

Objectives of our work. We restrict ourselves to databases in which the spatial component is a set of points in the 2-dimensional real space. Roughly speaking, we deal with spatially related collections of *regions* hierarchically organized as a nested partition. Partitions have been identified as a key concept for perception and understanding of space [9], and have been regarded as the primary tool for spatial analysis tasks in geographic applications and systems (see, e.g., [5, 10]). A number of papers proposed partition-based spatial analysis functions, with operations like overlay, generalization, and reclassification, all producing new partitions as a result (extended discussion on such topic can be found in [5], which also proposes a set of three powerful operations on partitions, sufficient to express all known application-specific operations). Differently from these works, we focus on primitives aimed at navigating sequences of gradually simplified representations of a given area.

In our conceptual model, the abstraction mechanism for modelling single geometric objects is the *region*. A region is a closed subset of $\mathbb{R}^2$ homeomorphic to: (i) a single point (region of dimension 0); (ii) $\mathbb{R}$ (region of dimension 1); (iii) $\mathbb{R}^2$ (region of dimension 2). We deal with a finite set $\mathcal{R} = \{R_0, R_1, \ldots, R_n\}$, where each R_i is a region.

As basic abstraction mechanisms for modelling spatially related collections of objects, in [4] we have considered the *nested partition* and the *aggregation*. The former is a recursive subdivision of the plane into pairwise disjoint regions of dimension 2 (denoted *blocks*), quite common in geographic maps (consider, e.g., countries partitioned into states, partitioned into counties, etc). The latter defines associations among regions, to model real applications where *features*, i.e., isolated points, lines, and areas, appear within regions (to represent cities, roads, rivers, lakes, etc., depending on the specific portion of the real world at hand).

According to LOD principles, we consider an exploration of the database where, starting from a map at a low LOD, more detailed information is successively disclosed on demand. In [4] we proposed a *single-focus* navigation: at each stage of the interaction session, one region of the map provides the current *focus* of interest, while the other (surrounding) regions provide the peripheral context. Roughly speaking, the overall interaction process is a sequence of user's activities composed by viewing the current map, selecting a region visible on the current map, and moving the focus to the selected region. As a result of the focus selection, a new map gets visualized, containing more detailed information on the focus region. According to the selected visualization techniques, the focus region is assigned a greater display area, while peripheral regions shrink yet preserving topology[1]. It is also possible to get maps at a lower LOD than the current one.

The single-focus navigation was supported by a set $\mathcal{IP}_{base}$ of basic primitives, defined as transitions between maps at different LOD, allowing to require/hide details regarding either the sub-blocks or the features of a focus region. Correspondingly, the map space is a graph where the set of nodes includes all possible views, and the set of arcs includes transitions between pairs of views defined by the primitives in $\mathcal{IP}_{base}$. We realized that, at a great extent, the "shape" of the graph depends on the nested partition and on the basic primitives concerning blocks (called *zooming* primitives). The complete map space, including transitions deriving from primitives concerning features, can actually be viewed as a generalization of a restricted space including only transitions deriving from zooming primitives.

Results of the paper. Since the new stage in our project concerns the definition of an extended sets of primitives on top of $\mathcal{IP}_{base}$, it appears appropriate to proceed in two steps: we now focus on new primitives related to the nested partition (and study properties of the map space in this restricted scenario), while we will extend the set of primitives to include treatment of features in a future step. This paper presents results achieved in the first step concerning the nested partition.

The idea is to consider *a set of focus regions* in place of a single focus region at a time. We hence define two new *multiple zooming* primitives called *mzoom-in* and *mzoom-out* as generalizations of the elementary zooming primitives *zoom-in* and *zoom-out*, defined in [4] to allow to require/hide details regarding the sub-blocks of a focus region. Though the definitions of the new primitives might be given independently of *zoom-in* and *zoom-out*, it is more interesting to discuss how multiple zooming can be expressed in terms of elementary zooming. These results allows us to conceive a system in which, once a kernel including algorithms corresponding to elementary primitives is designed, extensions can be easily defined by specifying appropriate sequences of *zoom-in* and *zoom-out*. In particular, we show that multiple zoom-in (resp., multiple zoom-out) corresponds to a sequence of zoom-in (resp., zoom-out) transitions, considering *in any order* the regions in the focus set.

Clearly, the main advantage of the multiple zooming is that there is no need for rendering the map after each basic *zoom-in* or *zoom-out* transition belonging to the sequence.

Structure of the paper. In Section 2 we describe our conceptual model refining the results of [4]; in Section 3 we recall the basic zooming primitives of [4] and study the properties of the corresponding map space; in Section 4 we

[1] Visualization aspects are beyond the scope of this paper. We refer to [3] for a general overview on visualization techniques, and to [4] for further discussion on the visual interaction with geographic maps.

introduce and formalize the notion of multiple zooming, and extend the map space. Finally, in Section 5 we provide some concluding remarks and outline future research directions.

2 The conceptual model of data

In our restricted model without features, the abstraction mechanism for single geometric objects is the *block*. A block is a closed subset of $\mathbb{R}^2$ homeomorphic to $\mathbb{R}^2$. In the remainder of the paper a set of blocks $\mathcal{B} = \{B_1, B_2, \ldots, B_p\}$ is called *spatial dataset*. Given a block B, a *partition* of B is a subdivision of this block into a finite set of pairwise disjoint blocks.

Definition 2.1 *Let $\mathcal{B}$ be a spatial dataset, and $\overline{B}$ be a block in $\mathcal{B}$. A partition $P = \{B_1, B_2, \ldots, B_j\}$, $j \geq 1$, of $\overline{B}$ is a* flat partition *of $\overline{B}$ in $\mathcal{B}$ if $B_i \in \mathcal{B}$, $1 \leq i \leq j$.*

The constraint on the cardinality of a partition in the above definition implies that the partition of minimum size of any block is the block itself.

Definition 2.2 *Let $\mathcal{B}$ be a spatial dataset, $\overline{B}$ be a block in $\mathcal{B}$, $P = \{B_1, B_2, \ldots, B_k\}$ be a flat partition of $\overline{B}$ in $\mathcal{B}$. A flat partition P' of $\overline{B}$ in $\mathcal{B}$ is a* refinement *of P if there exists a subset S of P, $S \neq \emptyset$, such that P' can be obtained from P by replacing each block $\hat{B}$ of P that is in S with a flat partition of $\hat{B}$ in $\mathcal{B}$.*

Notice that if P' is a refinement of P, then $|P| \leq |P'|$. In what follows we say that P' *is a refinement of P with respect to S*, where $S \subseteq P$, when we want to put emphasis on the blocks of P for which the condition of Definition 2.2 is satisfied. If $|S| = 1$ we say that P' is a *simple refinement* of P.

Example 2.1 *Let us suppose that $P = \{B_1, \ldots, B_i, \ldots, B_j, \ldots, B_k\}$ is a flat partition of $\overline{B}$ in $\mathcal{B}$, $\{B_{i,1}, \ldots, B_{i,l}\}$ is a flat partition of B_i in $\mathcal{B}$, and $\{B_{j,1}, \ldots, B_{j,m}\}$ is a flat partition of B_j in $\mathcal{B}$. Then, $P' = \{B_1, B_2, \ldots, B_{i-1}, B_{i,1}, \ldots, B_{i,l}, B_{i+1}, \ldots, B_{j-1}, B_{j,1}, \ldots, B_{j,m}, B_{j+1}, \ldots, B_k\}$ is a refinement of P with respect to $\{B_i, B_j\}$.*

Definition 2.3 *Let $\mathcal{B}$ be a spatial dataset, and B_0 be a distinguished block in $\mathcal{B}$. A* nested partition *of $\mathcal{B}$ with respect to B_0 is a finite set $\mathcal{NP}_{\mathcal{B}}(B_0) = \{P_1, P_2, \ldots, P_k\}$, $k \geq 1$, such that: 1) $\{B_0\} \in \mathcal{NP}_{\mathcal{B}}(B_0)$; 2) P_i is refinement of $\{B_0\}$, $1 \leq i \leq k$; 3) for each $P \in \mathcal{NP}_{\mathcal{B}}(B_0) \setminus \{B_0\}$ there exists $\overline{P} \in \mathcal{NP}_{\mathcal{B}}(B_0)$, $P \neq \overline{P}$, such that P is a refinement of $\overline{P}$; 4) there exists a unique $\mathcal{L} \in \mathcal{NP}_{\mathcal{B}}(B_0)$ such that $\mathcal{L}$ is a refinement of each $P \in \mathcal{NP}_{\mathcal{B}}(B_0)$; 5) $\cup_{i=1}^{k} P_i = \mathcal{B}$.*

Notice that $\{B_0\}$ is the partition of minimum size in $\mathcal{NP}_{\mathcal{B}}(B_0)$, while $\mathcal{L}$ is the partition of maximum size, and contains all blocks B of $\mathcal{B}$ whose unique flat partition in $\mathcal{B}$ is its partition of minimum size, i.e., $\{B\}$.

A relation $\preceq$ over $\mathcal{NP}_{\mathcal{B}}(B_0)$ can be defined starting from the concept of refinement as follows. Let P_i and P_j be two elements of $\mathcal{NP}_{\mathcal{B}}(B_0)$: $P_i \preceq P_j$ if and only if P_i is a refinement of P_j. The relation induces a partial order over $\mathcal{NP}_{\mathcal{B}}(B_0)$, which is hence a partially ordered set (i.e., a *poset*). In fact, the relation is:

- *reflexive*: any partition P of $\mathcal{NP}_{\mathcal{B}}(B_0)$ is refinement of itself, since we can replace any block in P with the corresponding partition of minimum size, i.e., the block itself (by Definition 2.1);
- *antisymmetric*: given P_i and P_j in $\mathcal{NP}_{\mathcal{B}}(B_0)$, if $P_i \preceq P_j$ and $P_j \preceq P_i$, then $|P_j| \leq |P_i|$ and $|P_i| \leq |P_j|$ (by Definition 2.2), and hence $|P_i| = |P_j|$. This can happen only if $P_i = P_j$;
- *transitive*: if $P_i \preceq P_j$ and $P_j \preceq P_k$ for some partitions P_i, P_j, and P_k in $\mathcal{NP}_{\mathcal{B}}(B_0)$, then $P_i \preceq P_k$. We show the case of simple refinements (i.e., $|S| = 1$), while the extension to the general case in which $|S| > 1$ is easily obtained by iteration. Given $P_k = \{B_1, \ldots, B_x, \ldots B_n\}$, two cases must be considered:

  Case 1. P_j is a refinement of P_k with respect to $\{B_x\}$, and P_i is a refinement of P_j with respect to $\{B_y\}$, where B_y is a block of P_j that does not belong to the partition of B_x used to obtain P_j from P_k. It trivially follows that P_i is a refinement of P_k with respect to $\{B_x, B_y\}$.

  Case 2. P_j is a refinement of P_k with respect to $\{B_x\}$, and P_i is a refinement of P_j with respect to $\{B_z\}$, where B_z is a block of P_j that belongs to the partition $\overline{P}$ of B_x used to obtain P_j from P_k. It is easy to see that P_i is a refinement of P_k with respect to $\{B_x\}$, since P_i is obtained from P_k by replacing B_x with a partition whose size is greater than the size of $\overline{P}$.

>From Definition 2.3 it follows that the partial order $(\mathcal{NP}_{\mathcal{B}}(B_0), \preceq)$ contains a *greatest* element (namely the partition $\{B_0\}$) and a *least* element (namely the partition $\mathcal{L}$). Furthermore, as in the case of set partitions [18], it can be shown that the pair $(\mathcal{NP}_{\mathcal{B}}(B_0), \preceq)$ forms a *lattice*, since any pair of elements in $\mathcal{NP}_{\mathcal{B}}(B_0)$ has a *least upper bound* and a *greatest lower bound* (for foundations of posets and lattices we refer to [1, 11]).

As in any partial order, we can introduce a notion of *immediate inferior*: in particular, we say that P_i is an *immediate refinement* of P_j if $P_i \preceq P_j$, $P_i \neq P_j$, and there not exists $P \in \mathcal{NP}_{\mathcal{B}}(B_0)$, $P \neq P_i$, $P \neq P_j$, such that $P_i \preceq P \preceq P_j$. It is easy to see that if P_i is immediate refinement of P_j, then P_i is a simple refinement of P_j (the converse is not true).

The notion of nested partition introduced in Definition 2.3 induces a binary relationship *sub-block* among the blocks of $\mathcal{B}$: a block B is sub-block of a block B' if B belongs to some flat partition of B'. B is *immediate sub-block*

of B' if there exists a flat partition P such that $B' \in P$ and B belongs to the immediate refinement of P with respect to B'. If B is immediate sub-block of B', then B' is the *parent-block* of B; if B is sub-block of B' then B' is an *ancestor* of B.

The relation *sub-block* over a spatial dataset $\mathcal{B}$ can be represented by a hierarchical structure, i.e., a *tree* rooted at B_0, that we denote as $\mathcal{T}_{\mathcal{B}}(B_0)$. If $B \in \mathcal{B}$, then the subtree of $\mathcal{T}_{\mathcal{B}}(B_0)$ rooted at B is denoted as $\mathcal{T}_{\mathcal{B}}(B)$. The parent-block of a block B in $\mathcal{T}_{\mathcal{B}}(B_0)$ will be denoted as $parent(B)$. We say that two blocks B_1 and B_2 of $\mathcal{T}_{\mathcal{B}}(B_0)$ are *siblings* when they are immediate sub-blocks of the same block B in $\mathcal{T}_{\mathcal{B}}(B_0)$.

Definition 2.4 *A* well-formed spatial dataset *is a pair* $\mathcal{D} = (\mathcal{B}, \mathcal{NP}_{\mathcal{B}}(B_0))$, *where: i)* $\mathcal{B}$ *is a spatial dataset; ii)* $\mathcal{NP}_{\mathcal{B}}(B_0)$ *is a nested partition of* $\mathcal{B}$ *with respect to* $B_0 \in \mathcal{B}$.

Concluding, we restrict ourselves to spatial databases in which the *spatial component* is a set of points in $\mathbb{R}^2$ that can be described by a well-formed spatial dataset.

Example 2.2 *In Figure 1 we show: (a) the tree* $\mathcal{T}_{\mathcal{B}}(B_0)$ *representing the relation sub-block associated to an example well formed spatial dataset* $\mathcal{D}$*; (b) the partition of maximum size of* $\mathcal{D}$ *composed by all the leaves of* $\mathcal{T}_{\mathcal{B}}(B_0)$*; (c) a graphical representation of the immediate refinement relationship over* $\mathcal{D}$ *provided by the Hasse diagram (i.e., the transitive reduction of the relation* $\preceq$*)* [2].

Notice that in Figure 1-(c) (and in the remaining figures as well) partitions are identified by the subscripts of the participating blocks. The example well-formed spatial dataset of Figure 1 will be referred to throughout the paper to illustrate new concepts as they are introduced.

3 Elementary zooming

In this section we study and formalize a zooming model for a well formed spatial dataset $\mathcal{D} = (\mathcal{B}, \mathcal{NP}_{\mathcal{B}}(B_0))$. The model includes a "minimal" (yet powerful) set of basic zooming primitives, denoted $\mathcal{ZP}_{base}$, which can act as a kernel for extended sets of primitives, defined on top of it. The set $\mathcal{ZP}_{base}$ includes basic primitives that, given a focus region, show/hide details related to it:

- *zoom-in* adds details regarding sub-blocks of the focus region;
- *zoom-out* subtracts details related to the focus region.

Formally, given a well formed spatial dataset $(\mathcal{B}, \mathcal{NP}_{\mathcal{B}}(B_0))$, a flat partition $P \in \mathcal{NP}_{\mathcal{B}}(B_0)$, and a block B of P:

[2] In a Hasse diagram, each poset element is represented by a dot; furthermore, given two elements A and B of the poset such that $A \preceq B$, and for no element C of the poset $A \prec C \prec B$, then the dot representing B is drawn in a position higher than the position of the dot representing A, and the two dots are connected by a line.

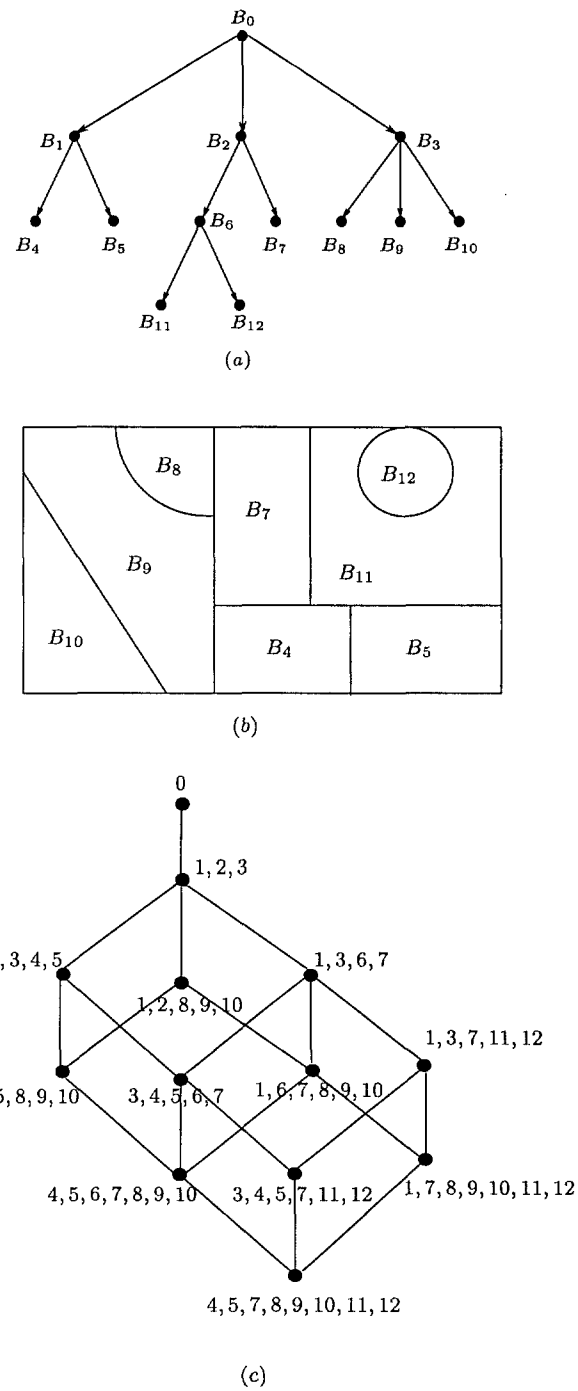


Figure 1: (a) The hierarchical structure representing the relation sub-block of an example well formed spatial dataset; (b) the associated partition of maximum size; (c) the Hasse diagram of $(\mathcal{NP}_{\mathcal{B}}(B_0), \preceq)$ in the example well formed spatial dataset.

- *zoom-in*$(P, B) = P'$, where P' is the immediate refinement of P with respect to $\{B\}$.
- *zoom-out*$(P, B) = P'$ such that P is a simple refinement of P' with respect to the parent block of B.

Notice that given P' and $B' \in P'$, there exist more than one simple refinement of P' with respect to $\{B'\}$, whereas given P and $B \in P$ there is only one P' verifying *zoom-out*$(P, B) = P'$. Intuitively, the zoom-out primitive generates the partition of maximum size among those superior of P, not containing B, and containing its parent-block.

It is worth noting that zoom-in (resp. zoom-out) cannot be applied to $\mathcal{L}$ (resp. $\{B_0\}$).

Intuitively, the zoom-in primitive generates the partition obtained from the current partition P by replacing the focus region B with its immediate sub-blocks, while the zoom-out primitive generates a partition P' as stated in the following observation.

Observation 3.1 *If zoom-out*$(P, B) = P'$, *then* P' *is obtained from* P *by subtracting those blocks of* P *that are in* $\mathcal{T}_{\mathcal{B}}(parent(B))$ *and adding* $parent(B)$.

Example 3.1 *With reference to the example of Figure 1-(c), if the current partition is* $P = \{B_1, B_7, B_8, B_9, B_{10}, B_{11}, B_{12}\}$ *and the selected block is* B_7, *then the partition resulting from zoom-out*(P, B_7) *is* $P' = \{B_1, B_2, B_8, B_9, B_{10}\}$.

Primitives in $\mathcal{ZP}_{base}$ allow us to conceive a LOD-based exploration of the well-formed spatial dataset in which, starting from the map at the lowest LOD (i.e., the partition $\{B_0\}$), maps at higher LOD are visualized by zoom-in operations, while maps at lower LOD are obtained by zoom-out operations. Correspondingly, the map space is a graph where the set of nodes coincides with $\mathcal{NP}_{\mathcal{B}}(B_0)$, and the set of arcs includes transitions between pairs of partitions defined by the primitives in $\mathcal{ZP}_{base}$. More formally, the map space is defined as follows:

Definition 3.1 *The* map space *associated to a well-formed spatial dataset* $\mathcal{D} = (\mathcal{B}, \mathcal{NP}_{\mathcal{B}}(B_0))$, *and to the set of primitives* $\mathcal{ZP}_{base}$ *is a directed graph* $\mathcal{MS}(\mathcal{D}, \mathcal{ZP}_{base}) = (N_{base}, A_{base})$ *defined as follows:*

- *for each flat partition* P *in* $\mathcal{NP}_{\mathcal{B}}(B_0)$ *there is a node* P *in* N_{base};
- *for each zoom-in/zoom-out transition there is a zoom-in/zoom-out arc in* A_{base}, *as follows: if zoom-in*$(P, B) = P'$, *then there exists a zoom-in arc from* P *to* P' *labelled by* B; *if zoom-out*$(P, B) = P'$, *then there exists a zoom-out arc from* P *to* P' *labelled by* B.

We observe that the map space is induced by the lattice: nodes are induced by elements of $\mathcal{NP}_{\mathcal{B}}(B_0)$, zoom-in arcs are induced by the immediate refinement relationship, zoom-out arcs are induced by the simple refinement relationship (but have opposite direction).

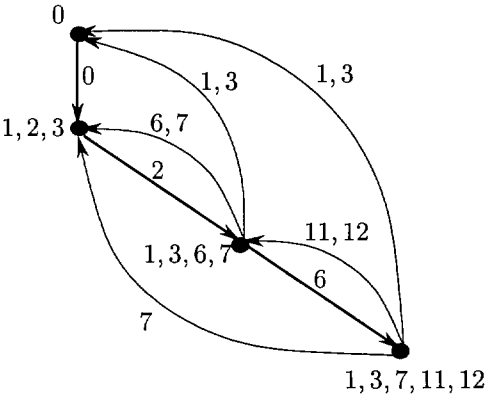


Figure 2: A portion of the map space associated to the example well-formed spatial dataset.

Example 3.2 *With reference to the well-formed spatial dataset of Example 2.2, Figure 2 shows a portion of its map space. In particular, thick arcs represent zoom-in arcs, while thin arcs represent zoom-out arcs. An arc label indicates the focus region of the transition (to avoid picture cluttering transitions differing only in the focus region have been represented by a unique arc having a list of labels). For example, if the current partition is* $P = \{B_1, B_3, B_7, B_{11}, B_{12}\}$, *then:*

- *zoom-out*$(P, B_7) = \{B_1, B_2, B_3\}$
- *zoom-out*$(P, B_{11}) = \{B_1, B_3, B_6, B_7\}$
- *zoom-out*$(P, B_{12}) = \{B_1, B_3, B_6, B_7\}$

Other transitions are easily derived from the Hasse diagram in Figure 1-(c).

An interaction process corresponds to a path in the map space. We denote as *zoom-in path* (resp., *zoom-out path*) a path in $\mathcal{MS}(\mathcal{D}, \mathcal{ZP}_{base})$ including only zoom-in arcs (resp., zoom-out arcs).

We notice that the set $\mathcal{ZP}_{base}$ is *complete*, in the sense that it allows us to reach all the partition in $\mathcal{NP}_{\mathcal{B}}(B_0)$ (in other words, there exists a path in $\mathcal{MS}(\mathcal{D}, \mathcal{ZP}_{base})$ from $\{B_0\}$ to any partition in $\mathcal{NP}_{\mathcal{B}}(B_0)$). In fact, from the definition of *zoom-in* and the properties of the lattice, it immediately follows that, if P is a partition in $\mathcal{NP}_{\mathcal{B}}(B_0)$, then there exists a zoom-in path in $\mathcal{MS}(\mathcal{D}, \mathcal{ZP}_{base})$ from P to any other partition P' in $\mathcal{NP}_{\mathcal{B}}(B_0)$, with $P' \preceq P$. All the partitions in $\mathcal{NP}_{\mathcal{B}}(B_0)$ are hence reachable from $\{B_0\}$.

Similarly, from the definition of *zoom-out* it follows that if P is a partition in $\mathcal{NP}_{\mathcal{B}}(B_0)$, then there exists a zoom-out path in $\mathcal{MS}(\mathcal{D}, \mathcal{ZP}_{base})$ from P to any other partition P' in $\mathcal{NP}_{\mathcal{B}}(B_0)$, with $P \preceq P'$. This allows us to prove the following lemma, about *traversability* of the map space.

Lemma 3.1 *Let P_1 and P_2 be any two partitions in $\mathcal{NP}_B(B_0)$. There exists a path in $\mathcal{MS}(\mathcal{D}, \mathcal{ZP}_{base})$ connecting P_1 to P_2.*

Proof. Let $\overline{P}$ be the least upper bound of P_1 and P_2. There must exist a zoom-out path from P_1 to $\overline{P}$ and a zoom-in path from $\overline{P}$ to P_2. □

According to the above result, we say that the map space $\mathcal{MS}(\mathcal{D}, \mathcal{ZP}_{base})$ is *fully traversable.*

4 Multiple zooming

The set $\mathcal{ZP}_{base}$ leads to interaction processes in which users are required to specify a great number of steps, since admitted transitions are induced by the immediate refinement and the simple refinement relationships only. In this section we define new zooming primitives that introduce additional transitions in the map space, thus providing "shortcuts" for selected paths in $\mathcal{MS}(\mathcal{D}, \mathcal{ZP}_{base})$.

Generally speaking, these new primitives are defined as generalizations of *zoom-in* and *zoom-out*, by considering *a set of focus regions* in place of a single focus region at a time.

4.1 Multiple zoom-in

The idea is to give the user the possibility of specifying a set of blocks of a given partition he/she wants to zoom-in. Formally, if $P = \{B_1, B_2, \ldots, B_k\}$ is a flat partition, and X_{in} is a subset of P, then *mzoom-in* is defined as follows:

- *mzoom-in*$(P, X_{in}) = P'$, where P' is the flat partition in $\mathcal{NP}_B(B_0)$ obtained by a sequence $\mathcal{Z}_{in}$ of zoom-in operations such that:

$$\begin{aligned} \mathcal{Z}_{in} = \; & \langle P_i \leftarrow \textit{zoom-in}(P_{i-1}, X_i) \text{ where} \\ & i = 1 \ldots n,\ n = |X_{in}| \\ & X_i \in X_{in} \\ & P_0 = P, P_n = P' \rangle \end{aligned}$$

Roughly speaking, *mzoom-in*(P, X_{in}) gives rise to a sequence $\mathcal{Z}_{in}$ of zoom-in operations having as focuses blocks of X_{in}. Each of these operations takes as parameter a partition of $\mathcal{NP}_B(B_0)$ which, at the generic step, we will call *current partition* (notice that throughout Section 4 the subscript of a flat partition denotes the step of the sequence in which the partition is generated). The sequence $\mathcal{Z}_{in}$ induces a sorting $\mathcal{S}_{in}$ of X_{in} corresponding to the order in which blocks of X_{in} are selected as focuses for the zoom-in operations in $\mathcal{Z}_{in}$.

Lemma 4.1 *If a block B is focus of a zoom-in operation $z_i \in \mathcal{Z}_{in}$, then B does not belong to any partition obtained by any zoom-in operation $z_j \in \mathcal{Z}_{in}$, with $j > i$.*

Proof. A zoom-in operation z_i with focus B generates a partition in which B has been replaced by its immediate sub-blocks. A subsequent zoom-in operation z_j, $j > i$, with focus $\overline{B}$, can make B appear again only if B is immediate sub-block of $\overline{B}$. This is not possible because, since B and $\overline{B}$ belong to X_{in}, and then to the same partition P, B cannot be immediate sub-block of $\overline{B}$. □

Notice that neither the sequence $\mathcal{Z}_{in}$ nor the corresponding sorting $\mathcal{S}_{in}$ are to be considered parameters of *mzoom*, since the user specifies the blocks he/she wants to zoom-in without providing any indication on the order in which blocks of X_{in} are to be taken into consideration (the sequence $\mathcal{Z}_{in}$ and the sorting $\mathcal{S}_{in}$ are rather related to the system's response to the users' request). We will use the notation @*mzoom-in*$(P, X_{in}, \mathcal{S}_{in})$ when we want to put emphasis on sorting details (i.e., on the execution procedure).

Example 4.1 *With reference to Figure 1-(c), if $P = \{B_1, B_2, B_8, B_9, B_{10}\}$, $X_{in} = \{B_1, B_2\}$, and $\mathcal{S}_{in} = \langle B_1, B_2 \rangle$, then @mzoom$(P, X_{in}, \mathcal{S}_{in}) = \{B_4, B_5, B_6, B_7, B_8, B_9, B_{10}\}$.*

In what follows we show that the result of *mzoom-in*(P, X_{in}) is independent of the selected sorting of X_{in} (this independence allow us to relieve the user of the burden of specifying execution details). More formally:

Theorem 4.1 *Let P be a flat partition of $\mathcal{NP}_B(B_0)$, X_{in} a subset of P, $\mathcal{S}_1$ and $\mathcal{S}_2$ two distinct sortings of X_{in}. Then, @mzoom-in$(P, X_{in}, \mathcal{S}_1) =$ @mzoom-in$(P, X_{in}, \mathcal{S}_2)$.*

Proof. The proof proceeds by contradiction, by assuming that @*mzoom-in*$(P, X_{in}, \mathcal{S}_1) \neq$ @*mzoom-in*$(P, X_{in}, \mathcal{S}_2)$. Let us denote as $\mathcal{Z}_1$ and $\mathcal{Z}_2$ the sequences inducing $\mathcal{S}_1$ and $\mathcal{S}_2$, respectively, and as P_n and Q_n the flat partitions resulting from $\mathcal{Z}_1$ and $\mathcal{Z}_2$, respectively. If $P_n \neq Q_n$, then there is at least one block B of P_n which is not in Q_n or viceversa. Without loss of generality, let us assume that $B \in P_n$ and $B \notin Q_n$. Two cases are possible, depending on whether $B \in P$ or not.

$B \in P$: If $B \notin Q_n$, then B disappeared during $\mathcal{Z}_2$ due to a zoom-in operation with focus B. Since $\mathcal{S}_1$ and $\mathcal{S}_2$ are distinct sortings of the same set X_{in}, then there must exist a zoom-in operation in $\mathcal{Z}_1$ with focus B, which hence removes B from the current partition. By Lemma 4.1, B cannot appear again during $\mathcal{Z}_1$, thus contradicting the hypothesis that $B \in P_n$.

$B \notin P$: This implies that $B \notin X_{in}$, and hence that B cannot be focus of any zoom-in operation in $\mathcal{Z}_1$ or $\mathcal{Z}_2$. Since $B \in P_n$, then B appeared during $\mathcal{Z}_1$ due to a zoom-in operation with focus $\overline{B}$, because B is an immediate sub-block of $\overline{B}$. Since $\mathcal{S}_1$ and $\mathcal{S}_2$ are distinct sortings of the same set X_{in}, there must exist a zoom-in operation in $\mathcal{Z}_2$ having $\overline{B}$ as focus. This operation adds B to the current partition since B is an immediate sub-block of $\overline{B}$. Since $B \notin X_{in}$, it cannot disappear during the remainder of $\mathcal{Z}_2$, thus contradicting the hypothesis that $B \notin Q_n$.

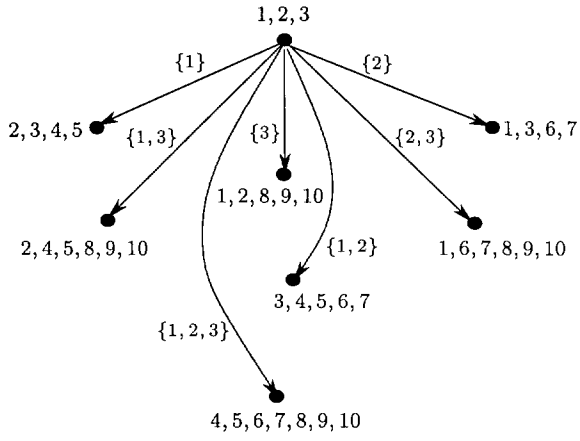


Figure 3: *mzoom-in* transitions from partition $\{1,2,3\}$ of the example well-formed spatial dataset.

Since in any case we have derived a contradiction, the theorem follows. □

As to the map space, intuitively it is obtained from $\mathcal{MS}(\mathcal{D}, \mathcal{ZP}_{base})$ by enriching the set of arcs to include mzoom-in transitions. Formally:

Definition 4.1 *The* map space *associated to a well-formed spatial dataset* $\mathcal{D} = (\mathcal{B}, \mathcal{NP}_{\mathcal{B}}(B_0))$, *and to the set of primitives* $\mathcal{ZP}_{mzin} = \{\mathcal{ZP}_{base} \cup \{mzoom\text{-}in\}\}$ *is a directed graph* $\mathcal{MS}(\mathcal{D}, \mathcal{ZP}_{mzin}) = (N, A)$ *obtained from* $\mathcal{MS}(\mathcal{D}, \mathcal{ZP}_{base}) = (N_{base}, A_{base})$ *as follows:*

- $N = N_{base}$;
- $A = A_{base} \cup A_{mzin}$ *where* A_{mzin} *is defined as follows: if* $mzoom\text{-}in(P, X_{in}) = P'$, *then there exists a multiple zoom-in arc in* A_{mzin} *from* P *to* P' *labelled by* X_{in}.

Example 4.2 *With reference to the example of Figure 1-(c), Figure 3 shows the portion of* $\mathcal{MS}(\mathcal{D}, \mathcal{ZP}_{mzin})$ *containing all arcs outgoing from the node* $P = \{B_1, B_2, B_3\}$ *(one arc for each subset of* P*). Notice that, a single step is now sufficient to go from partition* $\{B_1, B_2, B_3\}$ *to partition* $\{B_4, B_5, B_6, B_7, B_8, B_9, B_{10}\}$, *while three steps were required with elementary zooming only.*

4.2 Multiple zoom-out

Following the same approach as in the previous section, we want to give the user the possibility of specifying a set of blocks of a given partition he/she wants to zoom-out. Formally, if $P = \{B_1, B_2, \ldots, B_k\}$ is a flat partition, and X_{out} is a subset of P, then *mzoom-out* is defined as follows:

- $mzoom\text{-}out(P, X_{out}) = P'$, where P' is the flat partition in $\mathcal{NP}_{\mathcal{B}}(B_0)$ obtained by a sequence $\mathcal{Z}_{out}$ of zoom-out operations such that:

$$\mathcal{Z}_{out} = \langle P_i \leftarrow zoom\text{-}out(P_{i-1}, X_i) \text{ where } i = 1 \ldots k,\ k \leq n,\ n = |X_{out}|,\ X_i \in X_{out},\ P_0 = P, P_k = P' \rangle$$

Roughly speaking $mzoom\text{-}out(P, X_{out})$ gives rise to a sequence $\mathcal{Z}_{out}$ of zoom-out operations having as focuses blocks of X_{out}. Each of these operations takes as parameter a partition of $\mathcal{NP}_{\mathcal{B}}(B_0)$ that, at the generic step, we will call *current partition* (again, subscripts of flat partitions denote the step in which partitions are generated). The sequence $\mathcal{Z}_{out}$ induces a sorting $\mathcal{S}_{out}$ of X_{out} corresponding to the order in which blocks of X_{out} are selected as focuses for zoom-out operations in $\mathcal{Z}_{out}$.

Lemma 4.2 *Let* B *be a block of* X_{out} *and* z_i *a zoom-out operation in* $\mathcal{Z}_{out}$. *If* $B \in P_{i-1}$ *and* $B \notin P_i$, *then* $B \notin P_j$, *with* $j > i$.

Proof. By hypothesis B disappears at step i. A subsequent zoom-out operation z_j, $j > i$, with focus $\overline{B}$, can make B appear again only if B is the parent of $\overline{B}$. This is not possible because, since B and $\overline{B}$ belong to X_{out}, and then to the same partition P, B cannot be the parent of $\overline{B}$. □

The following observation is a straightforward consequence of Lemma 4.2.

Observation 4.1 *Let* B *be a block of* X_{out}. *If* $B \in P_h$, *then* $B \in P_i$, $0 \leq i \leq h$.

In fact, if $B \notin P_i$ for some $i < h$, then, by Lemma 4.2 $B \notin P_h$, thus contradicting the hypothesis.

As a consequence of Lemma 4.2 we have that the number of steps k (i.e., the number of zoom-out operations in $\mathcal{Z}_{out}$) may result less than $|X_{out}|$. In fact, if a zoom-out operation z_i in $\mathcal{Z}_{out}$ removes blocks of X_{out}, then these blocks will not appear again in partitions obtained by subsequent steps and cannot hence be considered as focuses.

We denote by $\mathcal{S}_{out}$ a sorting of X_{out} obtained by taking first the blocks used as focuses (*active* blocks) in the order in which they are considered in $\mathcal{Z}_{out}$, and then the remaining *inactive* blocks in any arbitrary order. A number of different sortings may be induced by a sequence $\mathcal{Z}_{out}$, since more than one permutation can be considered for the set of inactive blocks.

Notice that, as in the case of multiple zoom-in, neither the sequence $\mathcal{Z}_{out}$ nor a corresponding sorting $\mathcal{S}_{out}$ are to be considered parameters of *mzoom-out*, since the user specifies the blocks he/she wants to zoom-out without providing any indication on the order in which blocks of X_{out} are to be taken into considerations (the sequence $\mathcal{Z}_{out}$ and the sorting $\mathcal{S}_{out}$ are rather related to the system's response to the users' request). We will use the notation $@mzoom\text{-}out(P, X_{out}, \mathcal{S}_{out})$ when we want to put emphasis on sorting details.

Example 4.3 *With reference to the example in Figure 1-(c). If* $P = \{B_3, B_4, B_5, B_7, B_{11}, B_{12}\}$, $X_{out} = \{B_5, B_7\}$, *and* $\mathcal{S}_{out} = \langle B_7, B_5 \rangle$, *then* $@mzoom(P, X_{out}, \mathcal{S}_{out}) = \{B_1, B_2, B_3\}$.

In what follows we show that *mzoom-out*(P, X_{out}) is independent of the sorting of X_{out}. A preliminary lemma is given, directly deriving from the behavior of zoom-out, which generalizes Lemma 4.2.

Lemma 4.3 *If a block* B *of the partition* P_{i-1} *disappears as a consequence of mzoom-out*(P_{i-1}, X_i), *then* $B \notin P_h$, $h > i$.

Proof. By hypothesis B disappears at step i. By Observation 3.1 a subsequent zoom-out operation *mzoom-out*(P_{h-1}, X_h), $h > i$, can make B appear again only if B is the parent-block of X_h. On the other hand, $X_h \in X_{out}$ and $X_h \in P_{h-1}$, and then by Observation 4.1, $X_h \in P_j$, with $j < h$. Hence B and X_h belong to the same partition P_{i-1}, and B cannot be the parent-block of X_h. □

Theorem 4.2 *Let* P *be a flat partition of* $\mathcal{NP}_B(B_0)$, X_{out} *a subset of* P, $\mathcal{S}_1$ *and* $\mathcal{S}_2$ *two distinct sortings of* X_{out}. *Then,* $@mzoom\text{-}out(P, X_{out}, \mathcal{S}_1) = @mzoom\text{-}out(P, X_{out}, \mathcal{S}_2)$.

Proof. The theorem trivially follows if $\mathcal{S}_1$ and $\mathcal{S}_2$ are induced by the same $\mathcal{Z}_{out}$, and hence differ only in the order of the inactive blocks (the last blocks in the sequences). In what follows we hence assume that $\mathcal{S}_1$ and $\mathcal{S}_2$ are induced by two distinct sequences $\mathcal{Z}_1$ and $\mathcal{Z}_2$, and therefore differ in the order of both active and inactive blocks.

The remainder of the proof proceeds by contradiction. Let us assume that $@mzoom(P, X_{out}, \mathcal{S}_1) \neq @mzoom(P, X_{out}, \mathcal{S}_2)$, and denote as P_k and Q_k the flat partitions resulting from the execution of $\mathcal{Z}_1$ and $\mathcal{Z}_2$, respectively (by convention, we use the letter P for partitions generated by $\mathcal{Z}_1$, and the letter Q for partitions generated by $\mathcal{Z}_2$). If $P_k \neq Q_k$, then there exists at least one block B of P_k which is not in Q_k or viceversa. Without loss of generality, let us assume that $B \in P_k$ and $B \notin Q_k$. Two cases are possible, depending on whether $B \in P$ or not.

$B \in P$: Since $B \notin Q_k$, then B disappeared during the sequence $\mathcal{Z}_2$ as a consequence of a zoom-out operation, say *zoom-out*$(\overline{Q}, \overline{X})$, where $\overline{X}$ may or may not coincide with B. Block $\overline{X}$ can be active or inactive in $\mathcal{Z}_1$.

$\overline{X}$ is active:
Let us consider the step in which $\overline{X}$ has been taken as focus during $\mathcal{Z}_1$ for a zoom-out operation, say *zoom-out*$(\overline{P}, \overline{X})$. If $\overline{X} \equiv B$, then *zoom-out*$(\overline{P}, B)$ deletes B from $\overline{P}$, and, by Lemma 4.2, B cannot appear again during $\mathcal{Z}_1$, thus contradicting the hypothesis that $B \in P_k$. If $\overline{X} \not\equiv B$, then two cases may arise, $B \notin \overline{P}$ or $B \in \overline{P}$:

- If $B \notin \overline{P}$, then B disappeared during $\mathcal{Z}_1$ as a consequence of a previous zoom-out operation. By Lemma 4.3 B cannot appear again during $\mathcal{Z}_1$, thus contradicting the hypothesis that $B \in P_k$.
- If $B \in \overline{P}$, then it will surely disappear after *zoom-out*$(\overline{P}, \overline{X})$. In fact, let us consider the flat partition $\hat{Q}$ resulting from *zoom-out*$(\overline{Q}, \overline{X})$ during $\mathcal{Z}_2$. By Observation 3.1, $\hat{Q}$ contains *parent*$(\overline{X})$, and does not contain the blocks of $\overline{Q}$ that belong to $\mathcal{T}_B(parent(\overline{X}))$. Since $B \in \overline{Q}$ and $B \notin \hat{Q}$, then $B \in \mathcal{T}_B(parent(\overline{X}))$. This implies that during $\mathcal{Z}_1$ B will disappear from $\overline{P}$ after *zoom-out*$(\overline{P}, \overline{X})$. By Lemma 4.3 B cannot appear again during $\mathcal{Z}_1$, hence contradicting the hypothesis that $B \in P_k$.

$\overline{X}$ is inactive:
$\overline{X}$ disappeared during $\mathcal{Z}_1$ as a consequence of a zoom-out operation, say *zoom-out*$(\overline{P}', \overline{X}')$. Then, by Observation 3.1, $\overline{X} \in \mathcal{T}_B(parent(\overline{X}'))$. Since B disappeared during $\mathcal{Z}_2$ as a consequence of *zoom-out*$(\overline{Q}, \overline{X})$ then $B \in \mathcal{T}_B(parent(\overline{X}))$. It clearly follows that $B \in \mathcal{T}_B(parent(\overline{X}'))$, and then B disappears as a consequence of *zoom-out*$(\overline{P}', \overline{X}')$ in $\mathcal{Z}_1$. Since by Lemma 4.3 B cannot appear again during $\mathcal{Z}_1$, we contradict that $B \in P_k$.

$B \notin P$: Since $B \in P_k$, then B appeared during $\mathcal{Z}_1$ as a consequence of a zoom-out operation *zoom-out*$(\overline{P}, \overline{X})$, with $\overline{X} \in X_{out}$ and $B = parent(\overline{X})$, and no subsequent operation of $\mathcal{Z}_1$ removed B.

Since $B \notin Q_k$, two cases may arise: (1) B never appeared during $\mathcal{Z}_2$, or (2) B appeared during $\mathcal{Z}_2$ and then disappeared again.

1. If B never appeared during $\mathcal{Z}_2$, then $\overline{X}$, which makes B appear during $\mathcal{Z}_1$, must be inactive in $\mathcal{Z}_2$. By Observation 3.1, there must hence exist in $\mathcal{Z}_2$ a zoom-out operation *zoom-out*$(\hat{Q}, \hat{X})$ such that $\overline{X} \in \mathcal{T}_B(parent(\hat{X}))$. In other words, either (a) $\overline{X}$ is sibling of $\hat{X}$, or (b) $\overline{X}$ is sub-block of a sibling of $\hat{X}$. In case (a), since $parent(\hat{X}) = parent(\overline{X}) = B$, *zoom-out*$(\hat{Q}, \hat{X})$ adds B to the current partition, thus contradicting the hypothesis that B never appeared during $\mathcal{Z}_2$. In case (b), since $B = parent(\overline{X})$, then $B \in \mathcal{T}_B(parent(\hat{X}))$, and two subcases must be considered, depending on whether $\hat{X}$ is active or inactive in $\mathcal{Z}_1$:
   If $\hat{X}$ is active in $\mathcal{Z}_1$, there must exist in $\mathcal{Z}_1$ a zoom-out operation with focus $\hat{X}$ removing nodes in $\mathcal{T}_B(parent(\hat{X}))$ from the current partition;
   If $\hat{X}$ is inactive in $\mathcal{Z}_1$, there must exist in $\mathcal{Z}_1$ a zoom-out operation removing nodes in

$\mathcal{T}_{\mathcal{B}}(Y)$, for some ancestor Y of $\hat{X}$, from the current partition, and hence removing nodes in $\mathcal{T}_{\mathcal{B}}(parent(\hat{X}))$.

In both cases B is removed from the current partition, and by Lemma 4.3 it cannot appear again during $\mathcal{Z}_1$, thus contradicting that $B \in P_1$.

2. Regardless of the zoom-out operation of $\mathcal{Z}_2$ that makes B appear, if then B disappears, there must exist in $\mathcal{Z}_2$ a zoom-out operation *zoom-out*(Q', X') such that $B \in \mathcal{T}_{\mathcal{B}}(parent(X'))$. X' can be active or inactive in $\mathcal{Z}_1$. The two cases can be studied following the same reasoning line as for case (b) of block $\hat{X}$ in point (1), hence obtaining a contradiction.

Since in any case we have derived a contradiction, the theorem follows. □

As to the map space, intuitively it is obtained from $\mathcal{MS}(\mathcal{D}, \mathcal{ZP}_{base})$ by enriching the set of arcs to include mzoom-out transitions. Formally:

Definition 4.2 *The* map space *associated to a well-formed spatial dataset* $\mathcal{D} = (\mathcal{B}, \mathcal{NP}_{\mathcal{B}}(B_0))$, *and to the set of primitives* $\mathcal{ZP}_{mzout} = \{\mathcal{ZP}_{base} \cup \{mzoom\text{-}out\}\}$ *is a directed graph* $\mathcal{MS}(\mathcal{D}, \mathcal{ZP}_{mzout}) = (N, A)$ *obtained from* $\mathcal{MS}(\mathcal{D}, \mathcal{ZP}_{base}) = (N_{base}, A_{base})$ *as follows:*

- $N = N_{base}$;
- $A = A_{base} \cup A_{mzout}$, *where* A_{mzout} *is defined as follows: if* $mzoom\text{-}out(P, X_{out}) = P'$, *then there exists a multiple zoom-out arc in* A_{mzout} *from* P *to* P' *labelled by* X_{out}.

Concluding, we define the complete map space associated to $\mathcal{ZP}_{mz} = \{\mathcal{ZP}_{base} \cup \{mzoom\text{-}in, mzoom\text{-}out\}$ as follows.

Definition 4.3 *The* map space *associated to a well-formed spatial dataset* $\mathcal{D} = (\mathcal{B}, \mathcal{NP}_{\mathcal{B}}(B_0))$, *and to the set of primitives* $\mathcal{ZP}_{mz}$ *is a directed graph* $\mathcal{MS}(\mathcal{D}, \mathcal{ZP}_{mz}) = (N, A)$ *obtained from* $\mathcal{MS}(\mathcal{D}, \mathcal{ZP}_{base}) = (N_{base}, A_{base})$ *as follows:*

- $N = N_{base}$;
- $A = A_{base} \cup A_{mzin} \cup A_{mzout}$, *where* A_{mzin} *and* A_{mzout} *are as in Definitions 4.1 and 4.2, respectively.*

5 Conclusions and future work

In this paper we reported some theoretical results achieved in the framework of a project aimed at designing a visual interaction environment for geographical information systems. We restrict our attention to databases in which the spatial component is a set of points in the 2-dimensional real space. Our final goal is to provide the users a complete set of interaction primitives for exploring spatially related collections of *regions*, hierarchically organized as a nested partition, with *features* (i.e., isolated points, lines and areas, used to represent, e.g., towns, rivers, lakes). According to a LOD interaction approach, we focus on primitives aimed at navigating sequences of gradually simplified representations of a given area.

In a previous paper we proposed a set of basic interaction primitives allowing to require/hide details regarding sub-blocks and features of a given region [4]. Correspondingly, we defined the map space as a graph where the set of nodes includes all views (maps at different LOD) and the set of arcs includes transitions between pairs of views as defined by the interaction primitives. It has to be noticed that, conforming to a true multi-resolution approach [19], we maintain and store an implicit representation of the map space whose size is polynomial in the number of blocks in the dataset at hand (it would be unreasonable to maintain a structure containing all possible partitions generated by all possible interaction processes, since most of these partitions could be never used).

We found that the complete map space is a sort of generalization of a reduced space not including transitions related to the analysis of features. We hence restrict here our conceptual model to regions participating in the nested partition, to study new primitives and the corresponding map space in this scenario. This paper extends the results of [4] along three directions: (1) the conceptual model has been refined, (2) the properties of the map space associated to the set of basic zooming primitives ($\mathcal{ZP}_{base}$) have been analyzed, and (3) an extended set of primitives ($\mathcal{ZP}_{mz}$) has been introduced, based on the new concept of *multiple zooming*:

1. As to the conceptual model, we provide the definition of well-formed spatial datasets, and define the notions of immediate refinement and simple refinement, which constitute the basis of elementary zooming primitives. We have shown that the well formed spatial dataset constitutes a lattice. Discussions on the use of posets and lattices to represent spatial relations appeared in the literature since many years (see, e.g., [12, 13, 17]). More recently [14] proposed the lattice completion of a poset for spatial data in the topological data model, while [7, 8] discussed the expressive power of the lattice completion as a formal model for a set of spatial objects. In all these papers posets and lattices are used to model the topological relations among regions, differently from our work in which posets and lattices are used to model the nested partition, and consequently the map space.

2. As to $\mathcal{ZP}_{base}$, we have shown that the set is complete (i.e., we can reach any node from the top of the lattice), and that it guarantees full traversability of the map space (there is a path between any two nodes in the space). These properties allow us to use $\mathcal{ZP}_{base}$ as a kernel for extended set of primitives defined in terms

of elementary ones. Actually, extensions are appropriate because the full traversability of its associated map space associated to $\mathcal{ZP}_{base}$ is only a minimal requirement that does not guarantee *efficient* traversability of the graph (many steps are required to connect two nodes in the map space), and additional transitions should provide shortcuts for selected paths.

3. Multiple zooming primitives have hence been proposed here to offer shortcuts for some zoom-in paths and zoom-out paths. In particular, we showed that multiple zoom-in (resp., multiple zoom-out) corresponds to a sequence of zoom-in (resp., zoom-out) transitions, considering *in any order* the regions in the focus set.

Future work will proceed along four research lines: (1) the concept of multiple zooming will be further investigated to consider the general case in which the focus set is partitioned into two subsets, including regions to zoom-in and regions to zoom-out; (2) the map space will be further analyzed to characterize types of paths as additional primitives, thus extending $\mathcal{ZP}_{mz}$; (3) the theoretical results will be put into practice by the definition of an interaction model based on the notion of topological invariant; (4) the set of primitives will be extended to include treatment of features.

Acknowledgements

This work was partially supported by the Italian Ministry of University and Scientific and Technological Research under the project *SPADA: Representation and Processing of Spatial Data in Geographic Information Systems* and by the University of L'Aquila under the project *Representation and Interaction Techniques for Spatial Data.*

References

[1] G. Birkhoff (1967), *Lattice Theory*, volume XXV of *Colloquium Publications*, American Mathematical Society.

[2] P. A. Burrough and R. A. McDonnel (1998), *Principles of Geographical Information Systems*, Oxford University Press.

[3] S. K. Card, J. D. Mackinlay, and B. Shneiderman (1999), *Readings in information visualization: using vision to think*, Morgan Kaufmann.

[4] S. Cicerone, D. Frigioni, and L. Tarantino (2002), Exploration of geographic databases: supporting a focus+context interaction style, *Journal of Applied System Studies*, Cambridge International Science Publishing, vol. 3, n. 2.

[5] M. Erwig and M. Schneider (1997), Partition and conquer, In *Proceedings of 3rd International Conference on Spatial Information Theory (COSIT'97)*, volume 1329 of *Lecture Notes in Computer Science*, Springer, pp. 389–408.

[6] L. De Floriani, P. Magillo, and E. Puppo (1999), Data structures for simplicial multi-complexes, In *Proceedings of 6th International Symposium on Advances in Spatial Databases (SSD'99)*, volume 1651 of *Lecture Notes in Computer Science*, Springer, pp. 33–51.

[7] L. Forlizzi and E. Nardelli (1999), Characterization results for the poset based representation of topological relations – i: Introduction and models, *Informatica*, Slovene Society Informatika, vol. 23, n. 2, pp. 223–237.

[8] L. Forlizzi and E. Nardelli (2000), Characterization results for the poset based representation of topological relations – ii: Intersection and union, *Informatica*, Slovene Society Informatika, vol. 24, n. 1, pp. 83–96.

[9] A. U. Frank (1990), Spatial concepts, geometric data models and data structures, *Computer and Geosciences*, Elsevier Science, vol. 18, n. 4, pp. 409–417.

[10] A. U. Frank, G. S. Volta, and M. Mac Granaghan (1997), Formalization of families of categorical coverages, *International Journal of Geographical Information Science*, Taylor & Francis, vol. 11, n. 3, pp. 215–231.

[11] G. Gratzer (1978), *General Lattice Theory*, Academic Press, New York, NY.

[12] W. Kainz (1988), Application of lattice theory in geography, In *Proceedings of 3rd International Symposium on Spatial Data Handling*, pp. 135–142.

[13] W. Kainz (1990), Spatial relationships - topology versus order, In *Proceedings of 4th International Symposium on Spatial Data Handling*, pp. 814–819.

[14] W. Kainz, M. J. Egenhofer, and I. Greasley (1993), Modeling spatial relations and operations with partially ordered sets, *International Journal of Geographical Information Systems*, Taylor & Francis, vol. 7, n. 3, pp. 215–229.

[15] R. Laurini and D. Thompson (1992), *Fundamentals of spatial information systems*, Academic Press.

[16] E. Puppo and S. Scopigno (1997), Simplification, lod and multiresolution - principles and applications, In *Tutorial Notes PS97 TN4, in EUROGRAPHICS'97*, volume 16. Eurographics association, Balckwell Publishers, Oxford.

[17] A. Saalfeld (1985), Lattice structures in geography, In *Proceedings of 7th International Symposium on Computer Assisted Cartography (AUTOCARTO 7)*, pp. 482–489.

[18] R. S. Stanley (1986), *Enumerative Combinatorics*, Wadsworth & Brooks/Call, Monterey, CA.

[19] L. Tarantino. Detail filtering in geographic information visualization. *Encyclopedia of Life Support Systems, Theme 6.72 Advanced Geographic Information Systems*, UNESCO-EOLSS, 2001. To appear.

Parallel Aggregate-Join Query Processing

David Taniar
School of Business Systems
Monash University
PO Box 63B, Clayton, Vic 3800
Australia
David.Taniar@infotech.monash.edu.au

Y. Jiang, K.H. Liu, C.H.C. Leung
School of Communications and Informatics
Victoria University
P.O Box 14428 MCMC, Melbourne 8001
Australia

Keywords: Aggregate-Join Queries, Parallel Query Processing, Parallel Databases, and Parallel Aggregate Query Processing

Received: May 13, 2002

Queries containing aggregate functions often combine multiple tables through join operations. We call these queries "Aggregate-Join" queries. In parallel processing of such queries, it must be decided which attribute to be used as a partitioning attribute, particularly join attribute or group-by attribute. Based on the partitioning attribute, we discuss three parallel aggregate-join query processing methods, namely Join Partition Method (JPM), Aggregate Partition Method (APM), and Hybrid Partition Method (HPM). The JPM and APM models use the join attribute, and the group-by attribute, respectively, as the partitioning attribute. The HPM model combines the other two methods using a logically hybrid architecture. Our performance results show that the HPM model outperforms the others. In the performance evaluation, we also incorporate the problem of skew, which may occur at the data level, as well as at the processing level.

1 Introduction

Queries involving aggregates are very common in database processing, especially in On-Line Analytical Processing (OLAP), and Data Warehouse (Bedell, 1998, Datta and Moon, 1998). These queries are often used as a tool for strategic decision making. Queries containing aggregate functions summarize a large set of records based on the designated grouping. The input set of records may be derived from multiple tables using a join operation. In this paper, we concentrate on this kind of queries in which the queries contain aggregate functions and join operations. We call this query "Aggregate-Join" query.

As the data repository containing data for integrated decision making is growing, aggregate queries are required to be executed efficiently. Large historical tables need to be joined and aggregated each other; consequently, effective optimization of aggregate functions has the potential to result in huge performance gains. In this paper, we would like to focus on the use of parallel query processing techniques in aggregate-join queries.

The motivation for efficient parallel query processing is not only influenced by the need to performance improvement, but also the fact that parallel architecture is now available in many forms, such as systems consisting of a small number but powerful processors (i.e. SMP machines), clusters of workstations (i.e. loosely coupled shared-nothing architectures), massively parallel processors (i.e. MPP), and clusters of SMP machines (i.e. hybrid architectures) (Almasi G., and Gottlieb, 1994). We are particularly interested in formulating efficient parallel processing of aggregate-join queries, by exploiting a logically hybrid parallel database architecture, in which a set of processors is logically grouped into clusters.

We present three parallel processing methods for aggregate-join queries, Join Partition Method (JPM), Aggregate Partition Method (APM), and Hybrid Partition Method (HPM). The JPM and APM methods mainly differ in the selection of partitioning attribute for distributing workloads over the processors. The HPM method is an adaptive method based on the JPM and APM methods. In the evaluation of these three methods, we also take into account the problem of data skew since skewed load distribution may affect the query execution time significantly. The performance of the three methods has been compared under various queries and different environments with a simulation study, and the results are also presented. According to the results we have

obtained, performance of the HPM method is promising and more superior to the other two methods.

The rest of this paper is organized as follows. Section 2 explains briefly aggregate queries. Section 3 describes the three methods for parallel processing of aggregate-join queries. Section 4 presents quantitative analysis and cost models. Section 5 explains our simulation model and some simulation results. Finally, section 6 gives the conclusions and explains future work.

2 Aggregate Queries: A Brief Overview

In this section, we explain basic aggregate queries and aggregate-join queries. The details of both queries are explained as follows.

2.1 Basic Aggregate Queries

Basic aggregate queries are normally categorized into scalar aggregates and aggregate functions (Graefe, 1993). Scalar aggregate queries produce single values for a given set of records (i.e. table), whereas aggregate function queries generate a set of values for a given table. The former is like grouping the whole table and produces a single value, whereas the latter is like grouping the table into several groups, and for each group a single value is produced. To illustrate these two types of aggregate queries, we use the following tables from a Suppliers-Parts-Projects database:

```
SUPPLIER(S#,Sname,Status,City)
PARTS(P#,Pname,Colour,Weight,Price,City)
PROJECT (J#,Jname,City,Budget)
SHIPMENT (S#,P#,J#,Qty)
```

An example of a scalar aggregate query is to "retrieve the most expensive project". The input table in this case is table Project, and the single value to be produced is the largest value in attribute Budget. The SQL and a sample result of the above query are given below.

```
QUERY 1:        Select MAX(Budget)
                From PROJECT;

SAMPLE RESULT:  MAX(Budget)
                -----------
                44500
```

Notice that the query produces a single value, and this value is generated by the MAX function. Other basic functions, such as COUNT, SUM, AVG, and MIN, may also be used. Apart from these basic functions, most commercial Relational Database Management Systems (RDBMS) also include other advanced functions, such as advanced statistical functions, etc. From query processing point of view, the matter is that these functions take a set of records (i.e. a table) as their input and produce a single value as the result.

An example of an aggregate function query is to "retrieve number of suppliers for each city". The table to be used in this query is table Supplier, and the supplier records are grouped according to its city. For each group, it is then counted the number of records. These numbers will then represent number of suppliers in each city. The SQL and a sample result of this query are given below.

```
QUERY 2:        Select City, COUNT(*)
                From SUPPLIER
                Group By City

SAMPLE RESULT:  City        COUNT(*)
                ------------------
                Beijing         1
                London          9
                Melbourne       5
                Sydney          6
```

As the above query uses a *Group-By* clauses, aggregate function queries are often known as "*Group-By queries*", as this differentiates between aggregate function queries from scalar aggregate queries.

It is worth to mention that the input table may have been filtered using a Where clause (in both scalar aggregate and aggregate function queries), and additionally for aggregate function queries, the results of the grouping may be further filtered using a *Having* clause.

2.1 Aggregate-Join Queries

It is common that an aggregate function query (Group-By query) involves multiple tables. These tables are joined to produce a single table, and this table becomes an input to the group-by operation. We call this kind of aggregate query as Aggregate-Join queries, that is queries involving join and aggregate functions.

For simplicity of description and without loss of generality, we consider queries that involve only one aggregation function and a single join. The following two queries give an illustration of aggregate-join queries. Query 3 is to "retrieve project numbers, names, and total quantity of shipments for each project having the total shipments quantity of more than 1000".

```
QUERY3:
  Select PROJECT.J#, PROJECT.Jname,
  SUM(Qty) From PROJECT, SHIPMENT
  Where PROJECT.J# = SHIPMENT.J#
  Group By PROJECT.J#, PROJECT.Jname
  Having SUM(Qty)>1000
```

Another example is to "cluster the part shipment by their city locations and select the cities with average quantity of shipment between 500 and 1000". The query written in SQL is as follows.

```
QUERY 4:
  Select PARTS.City, AVG(Qty)
  From PARTS, SHIPMENT
  Where PARTS.P# = SHIPMENT.P#
  Group By PARTS.City
  Having AVG(Qty)>500 AND AVG(Qty)<1000
```

The main difference between Query 3 and Query 4 above lies in the join attributes and group-by attributes. In Query 3, the join attribute is also one of the group-by attributes. This is not the case with Query 4, where the join attribute is totally different from the group-by attribute. This difference is particularly a critical factor in

processing aggregate-join queries, especially in parallel query processing, as there are decisions to be made regarding which attribute to be used for data partitioning. When the join attribute and group-by attribute are the same as shown in Query 3 (e.g. attribute J# of both Project and Shipment), the selection of partitioning attribute becomes obvious. Therefore, instead of performing join first, the aggregate is carried out first followed by the join. Comparatively, join is a more expensive operation than aggregate functions, and it would be beneficial to reduce the join relation sizes by applying the aggregate function first. Generally, aggregate functions should always precede join whenever possible with an exception that the size reduction gained from the aggregate functions is marginal or the join selectivity factor is extremely small. In real life, early processing of the aggregate functions before join reduces the overall execution time as stated in the general query optimization rule where unary operations are always executed before binary operations if possible.

However, aggregate functions before join may not always be possible, such as Query 4 above. The semantic issues about aggregate functions and join and the conditions under which the aggregate functions would be performed before join can be found in literatures (Kim, 1982; Dayal, 1987; Bultzingsloewen, 1987; Yan and Larson, 1994). In this paper, we concentrate on more general cases where aggregate functions cannot be executed before join. Therefore, we will use Query 4 as a running example throughout this paper.

3 Parallel Aggregate-Join Query Processing Methods

Our parallel database architecture consists of a host and a set of working processors. The host accepts queries from users and distributes each query with the required base relations to all processors for execution. The processors perform the query in parallel with possibly intermediate data transmission among each other via the network, and finally send the result of the query to the host. Our previous work has proved the suitability of this architecture in for parallel database processing (Leung and Ghogomu, 1993; Leung and Taniar, 1995; Liu et al, 1996). Using this parallel architecture, parallel query processing is commonly carried out in three phases:

- Data partitioning, the operand relations of the query are partitioned and the fragments are distributed to each processor;
- Parallel processing, the query is executed in parallel by all processors and the intermediate results are produced;
- Data consolidation, the final result of the query is obtained by consolidating the intermediate results from the processors.

There is an important decision need to be made in processing aggregate-join queries, namely the selection of partitioning attribute. Selecting a proper partitioning attribute plays a crucial role in performance. Although in general any attributes of the operand relations may be chosen, two particular attributes (i.e. join attribute and group-by attribute) are usually considered.

If the join attribute is chosen, both relations are partitioned into N fragments by employing a partitioning function (e.g. a hash/range function), where N is the number of processors. The cost for parallel join operation can therefore be reduced as compared with a single processor system. However, after join and local aggregation at each processor, a global aggregation is required at the data consolidation phase, since local aggregation is performed on a subset of the group-by attribute.

If the group-by attribute is used for data partitioning, the relation with the group-by can be partitioned into N fragments, while the other relation needs to be broadcast to all processors for the join operation.

Comparing the two methods above, in the second method (partitioning based on the group-by attribute), the join cost is not reduced as much as in the first method (partitioning based on the join attribute). However, no global aggregation is required after local join and local aggregation, because records with identical values of the group-by attribute have been allocated to the same processor.

Based on these two partitioning attribute strategies, we introduce three parallel processing methods for aggregate-join queries, namely Join Partitioning Method (JPM), Aggregate Partitioning Method (APM), and Hybrid Partitioning Method (HPM). They are discussed in more details in the following sections.

3.1 Join Partition Method (*JPM*)

Given the two relations R and S to be joined, and the result is grouped-by according to the group-by attribute and possibly filtered through a having predicate, parallel processing of such query using the JPM method can be stated as follows.

Step 1: *Data Partitioning*

The relations R and S are partitioned into N fragments in terms of join attribute, i.e. the records with the same join attribute values in the two relations fall into a pair of fragments. Each pair of the fragments will be sent to one processor for execution.

Using QUERY 4 as an example, the partitioning attribute is attribute P# of both tables Parts and Shipment, which is the join attribute. Suppose we use 4 processors, and the partitioning method is a range partitioning, such as part numbers (P#) p1-p99, p100-p199, p200-p299, and p300-399 are distributed to processors 1, 2, 3, and 4, respectively. This partitioning function is applied to both tables Parts and Shipment. Consequently, processor such as processor 1 will have Parts and Shipment records where the values of its P# attribute are between p1-p99, and so on.

Step 2: *Join operation*

Upon receipt of the fragments, the processors perform in parallel, the join operation on the allocated fragments. Join in each processor is done independently to each other (DeWitt and Gray, 1992). This is possible because the two tables have been disjointly partitioned based on the join attribute.

Using the same example as above, join operation in a processor like processor 1 will produce a join result consisting of Parts-Shipment records having P# between p1-p99.

It is worth to mention that any sequential join algorithm (i.e. nested-loop join, sort-merge join, nested index join, hash join) may be used in performing a local join operation in each processor (Mishra and Eich, 1992).

Step 3: *Local Aggregation*

After the join is completed, each processor then performs a local aggregation operation. Join results in each processor is grouped-by according to the group-by attribute.

Continuing the same example as the above, each city found in the join result will be grouped. If, for example, there are three cities: Beijing, Melbourne, and Sydney, found in processor 1, the records will be grouped according to these three cities. The same aggregate operation is applied to other processors. As a result, although each processor has distinct part numbers, some of the cities, if not all, among processors may be identical (duplicated). For example, processor 2 may have three cities, such as London, Melbourne, and Sydney, where Melbourne and Sydney are also found in processor 1 as mentioned earlier, but not London.

Step 4: *Re-distribution*

A global aggregation operation is to be carried out by re-distributing the local aggregation results across all processors such that the result records with identical values of the group-by attribute are allocated to the same processors.

To illustrate this step, a range partitioning method is again used to partition the group-by attribute, such as processors 1, 2, 3, and 4 are allocated cities beginning with letter A-G, H-M, N-T, and U-Z, respectively. Using this range partitioning, processor 1 will distribute its Melbourne record to processor 2, Sydney record to processor 3, and leave Beijing record in processor 1. Processor 2 will do the same to its Melbourne and Sydney records, whereas London record will remain in processor 2.

Step 5: *Global Aggregation*

Each processor performs an N-way merging of the local aggregation results, followed by performing a restriction operation for the Having clause if required by the query.

The result of this global aggregate in each processor is a subset of the final results, meaning that each record in each processor has a different city, and furthermore, the cities in each processor will not appear in any other processors. For example, processor 1 will produce one Beijing record in the query result, and this Beijing record does not appear in any other processors. Additionally, some of the cities may then be eliminated through the Having clause.

Step 6: *Consolidation*

The host simply consolidates the partial results from the processors by a union operation, and produces the query result.

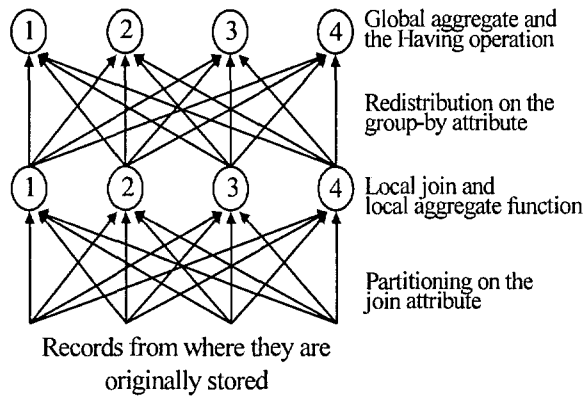


Figure 1: Join Partition Method (*JPM*).

Figure 1 gives a graphical illustration of the Join Partition Method (JPM). The circles represent processing elements, whereas the arrows denoted data flow through data partitioning or data re-distribution.

3.2 Aggregate Partition Method (*APM*)

The APM method relies on partitioning based on the group-by attribute. As the group-by attribute belongs to just one of the two tables, only the table having the group-by attribute will be partitioned. The other table has to be broadcast to all processors. This technique is often known as "Divide and Broadcast" technique, commonly used in naive parallel join operations (Leung and Ghogomu, 1993). The processing steps of the APM method are explained as follows.

Step 1: *Data Partitioning*

The table with the group-by attribute, say R, is partitioned into N fragments in terms of the group-by attribute, i.e. the records with identical attribute values will be allocated to the same processor. The other table S needs to be broadcast to all processors in order to perform the join operation.

Using QUERY 4 as an example, table Parts is partitioned according to the group-by attribute, namely City. Assume a range partitioning method is used, processors 1, 2, 3, and 3 will have Parts records having cities beginning with letter A-G, H-M, N-T, and U-Z, respectively. On the other hand, table Shipment is replicated to all four processors.

Step 2: *Join operations*

After data distribution, each processor carries out the joining of one fragment of R with the entire table S.

Using the same example, each processor joins its Parts fragment with the entire table Shipment. The results of this join operation in each processor are pairs of Parts-Shipment records having the same P# (join attribute) and the value of its City attribute must fall into the category identified by the group-by partitioning method (e.g. processor 1=A-G, processor 2=H-M, etc).

Step 3: *Aggregate operations*

The aggregate operation is performed by grouping the join results based on the group-by attribute, followed by a *Having* restriction if it exists on the query.

Continuing the above example, processor 1 will group the records based on the city and the cities are in the range of *A* to *G*. The other processors will of course have a different range. Therefore, each group in each processor is distinct to each other both within and among processors.

Step 4: *Consolidation*

Since the table *R* is partitioned on *group-by* attribute, the final aggregation result can be simply obtained by a union of the local aggregation results from the processors.

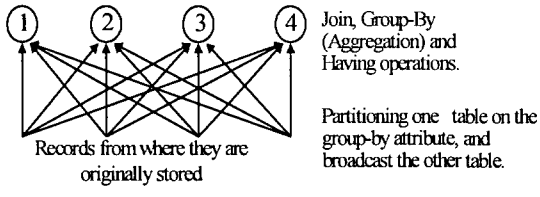


Figure 2: Aggregation Partition Method (*APM*).

Figure 2 shows a graphical illustration of the *APM* method. Notice the difference between the *JPM* and the *APM* method. The former imposes a "*two-phase*" partitioning scheme, whereas the latter is a "*one-phase*" partitioning scheme.

3.3 Hybrid Partition Method (*HPM*)

The *HPM* method is a combination of the *JPM* and *APM* methods. In the *HPM*, the total number of processors *N* is divided into *m* clusters, each of which has *N/m* processors as shown in Figure 3. Based on the proposed logical architecture, the data partitioning phase is carried out in two steps. First, the table with group-by attributes is partitioned into processor clusters in the same way of the *APM* (i.e. partitioning on the group-by attribute and the other table is broadcast to the cluster). Second, within each cluster, the fragments of the first table and the entire broadcast table are further partitioned by the join attributes as in the *JPM*. Depending on parameters such as the cardinality of the tables and the skew factors, a proper value of *m* can be chosen to minimize the query execution time.

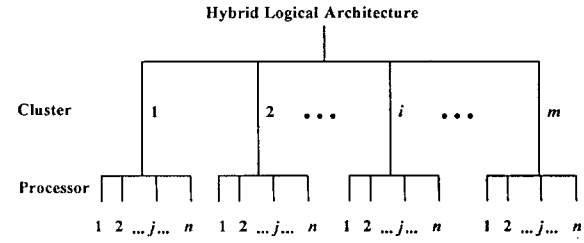


Figure 3: Logical Architecture for *HPM*.

The detailed *HPM* method is explained as follows:

Step 1:*a) Partitioning into Clusters*

Partition the table *R* on group-by attribute to *m* clusters, denoted by r_i where $1 \leq i \leq m$. Table *S* is broadcast to all clusters.

Using QUERY 4 as an example, first partition table Parts into *m* clusters based on attribute *City*. If there are three clusters, table Parts is divided into three fragments. Table Shipment is, on the other hand, replicated to all the three clusters. As a result, each cluster will have a fragment of table Parts and a full table Shipment.

b) Partitioning within Clusters

Within each cluster *i*, further partition fragment r_i and the full table S on the join attribute to n processors where $n=N/m$. Each fragment is now denoted by r_{ij} and s_j.

Suppose that there are twelve processors in total. Since we use three clusters, each cluster will contain four processors. In each cluster, a Parts fragment and the whole Shipment table are partitioned based on the join attribute *P#* into the four processors.

In practice, steps 1(a) and (b) described above are carried out at once, in order to reduce communication/distribution time. When doing the two partitioning steps as a single partitioning step, each record of the tables is applied a partitioning function that determines into which processor and cluster the record should be sent. The partitioning function takes into account whether or not the table is the group-by table. Figure 4 gives the algorithm of the partitioning step in the *HPM*. The algorithm clearly shows that the table containing the group-by attribute is purely partitioned into all processors, whereas the other table is to some degrees replicated.

```
Let processors be P
Let group-by partitioning function be G
Let join partitioning function be J
For each record r of the group-by table
   Determine cluster i for record r based on partitioning function G
   Determine processor j for record r based on partitioning function J
   Distribute record r to processor P_ij
End
For each record s of the other table
   Determine processor j for record r based on partitioning function J
   For each processor k in each cluster
       Distribute record r to processor P_kj
   End
End
```

Figure 4: Partitioning Algorithm in *HPM*.

Step 2: *Join operation*

Carry out the join operation in each processor. Join operation is executed like in the other two methods, that is, it is carried out locally and independently in each processor without involving other processors.

Step 3: *Local Aggregation*

Perform local aggregation at each processor. This local aggregation operation is carried out as like in the *JPM* method. As a result, each processor will produce a number of groups and some groups among other processors may be the same, and these groups will be later grouped in the global aggregation stage.

Step 4: *Re-distribution*

Redistribute the local aggregation results to the processors within each cluster by partitioning the results on the group-by attribute. This step is similar to that of in the *JPM* method. The main difference is that in *HPM* the re-distribution is done within each cluster, not globally involving all processors like in *JPM*.

Step 5: *Global Aggregation*

Merge the local aggregation results within each cluster. Then perform the *Having* predicate, if exists, in each cluster. Unlike the *JPM*, again, this process is done locally in each cluster. As a result of this process, each cluster contains a subset of the final query result.

Step 6: *Consolidation*

Transfer the results from the clusters to the host.

Figure 5 shows a graphical illustration of the *HPM* method. By the look at it, the flow of process is similar to that of *JPM*, in which both are a "*two-phase*" processing scheme. We, however, need to highlight two main differences. One is that the initial partitioning in *HPM* is different, and in fact, it is a combination between partitioning in *JPM* and in *APM*, where both group-by and join attributes are being utilized in the partitioning phase. This unified two-step partitioning scheme is not clearly shown in the diagram, but as shown in the algorithm the partitioning uses both group-by and join attributes. The second difference is related to the re-distribution phase where re-distribution in *HPM* is a cluster-based, not global in the sense like in *JPM*.

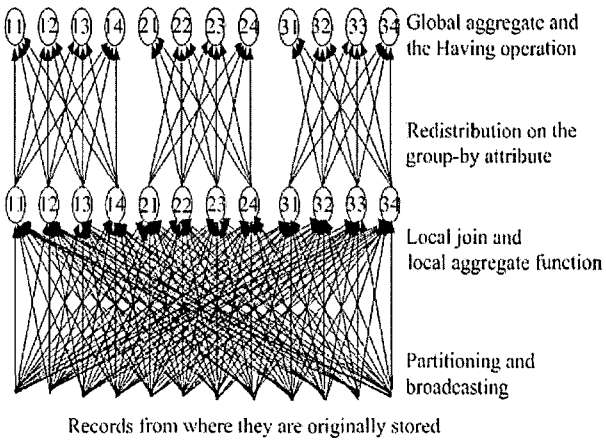


Figure 5: Hybrid Partition Method (*HPM*).

4 Cost Models

In this section, we present in this section three parallel processing methods for aggregate-join queries. The notations used in the description of the methods and in the subsequent sensitivity analysis are given in Table 1.

Parameter	Meaning
N	Total number of processors
m	Number of processor clusters
n	Number of processor in each cluster ($N = m \times n$)
r, s	Number of recor ds in base tables R and S
r_i, s_i	Number of records of fragments of tables R and S at processor i
$Sel(i)$	Join selectivity factor for fragment i
$Agg(i)$	Aggregation factor for fragment i
θ	Reduction factor after perfoming *Having* clause
α	Data partitio ning skew factor
β	Data processing skew factor
T_{comm}	Average data transmission time for each message
T_{join}	Average join time for each record
T_{agg}	Average aggregation time for each record
z	Message size in terms of the number of records

Table 1: Notations

4.1 The *JPM* Cost Model

Given the notation introduced earlier, the components of the JPM execution time expressed as follows:

Data partitioning cost for *JPM* is determined by the sum of fragments r_i and s_i which are the fragments of the two tables R and S in processor i.

$$T_{comm} \times (\max(r_i + s_i)) \quad (1)$$

Local join cost is influenced by the sequential join algorithm used in each processor. Using a nested-loop

join, the join cost can be determined by $r_i \times s_i$ which follows the *O(N2)* complexity, whereas a hash join follows a linear complexity and hence the cost is calculated by $r_i + s_i$ (Knuth, 1973). If one of the join attributes is indexed, one can use a nested index block join algorithm, which is resolved by $r_i \times log\ s_i$. Hence, the cost options for the local join operation are as follows.

$$\begin{aligned} \text{nested loop} &= T_{join} \times (\max(r_i \times s_i)) \\ \text{hash} &= T_{join} \times (\max(r_i + s_i)) \\ \text{nested block index} &= T_{join} \times (\max(r_i \times \log s_i)) \end{aligned} \quad (2)$$

Local aggregation cost takes the result of the join as its input, which is influenced by the join selectivity factor *Sel(i)*. This selectivity factor is a proportion of the cardinality of the Cartesian product between the two tables, and hence the local aggregation cost is:

$$T_{agg} \times (\max(r_i \times s_i \times Sel(i))) \quad (3)$$

Re-distribution cost is determined by number of groups formed by local aggregation. This is influenced by the aggregation factor *Agg(i)*.

$$T_{comm} \times (\max(r_i \times s_i \times Sel(i) \times Agg(i))) \quad (4)$$

Global aggregation cost is composed of the merging of the local aggregation results and the *Having* operation. The cardinality of these operations is assumed obtained from the re-distribution cardinality. Assuming that the aggregation unit cost and the *Having* operation unit cost are the same which are determined by T_{agg}, the global aggregation cost is calculated as follows.

$$T_{agg} \times (\max(r_i \times s_i \times Sel(i) \times Agg(i))) \times (1+1) \quad (5)$$

Finally, *consolidation cost* is a data transfer cost to the host. The cardinality of the consolidation cost is reduced by further from that in the global aggregation cost by a factor of q which is the reduction factor after performing the *Having* clause.

$$T_{comm} \times (\max(r_i \times s_i \times Sel(i) \times Agg(i) \times \theta)) \quad (6)$$

The *total cost for JPM* is the sum of equations (1) to (6). Assuming that a nested block index join is used, the total *JPM* cost is as follows.

$$\begin{aligned} JPM = {} & T_{comm} \times (\max(r_i + s_i)) + T_{join} \times (\max(r_i \times \log s_i)) \\ & + T_{agg} \times (\max(r_i \times s_i \times Sel(i))) \\ & + T_{comm} \times (\max(r_i \times s_i \times Sel(i) \times Agg(i))) \\ & + T_{agg} \times (\max(r_i \times s_i \times Sel(i) \times Agg(i))) \times (1+1) \\ & + T_{comm} \times (\max(r_i \times s_i \times Sel(i) \times Agg(i) \times \theta)) \end{aligned} \quad (7)$$

The maximum execution time for each of the components in the above equation varies with the degree of skewness, and could be far from average execution time. Therefore, we need to apply the skew factors to the above cost equation. Data skew, in this context, can be categorized into *data partitioning skew* and *data processing skew*.

Data partitioning skew refers to the uneven distribution of records of the operand tables over the processors and thus results in different processing loads. This is mainly caused by skewed value distribution in the partitioning attribute as well as improper use of the partitioning method.

Data processing skew, on the other hand, refers to uneven sizes of the intermediate results generated by the processors. This is caused by the different selectivity factors of the join/aggregate of the local fragments. Since the intermediate results of the join operation are processed for aggregation, data processing skew may lead to different loads of the processors even when there is no data partitioning skew.

We introduce two skew factors α and β. Data partitioning skew is denoted as α, whereas data processing skew is represented as β. Assume that a follows the *Zipf* distribution where the i^{th} common word in natural language text occurs with a frequency proportional to i (Knuth, 1973; Wolf et al, 1993a,b), i.e.,

$$p_i = \frac{1}{i \times H_N}$$

where H_N is the Harmonic number and could be approximated by $(\gamma + \ln N)$ (Leung and Liu, 1994). Notice that all elements in p_i are in order, and the first element p_1 always gives the highest probability and the last element p_N gives the lowest. Considering both operand tables R and S use the same number of processors and follow the *Zipf* distribution, the data partitioning skew factor a thus can be represented as

$$\alpha = \frac{1}{\gamma + \ln N}$$

where $\gamma = 0.57721$ known as the Euler Constant and N is the number of processors.

The other skew factor β for data processing skew is affected by the data partitioning skew factors in both operand tables since the join/aggregation results rely on the operand fragments. Therefore, the range of β falls in $[\alpha_r \times \alpha_s, 1]$. However, the actual value of β is difficult to estimate because the largest fragments from the two tables are usually not allocated to the same processor, resulting that β is much less than the product of α_r and α_s. We assume in this paper that $\alpha_r = \alpha_s = \alpha$, and $\beta = (\alpha_r \times \alpha_s + 1) / 2 = (\alpha^2 + 1) / 2$, and a detail discussion on skewness can be found in Liu et al (1995) and Leung and Liu (1994).

We also need to make the following assumptions and simplifications:

- $J_i = r_i \times s_i \times Sel(i) = J$, (i.e. the join result cardinality without skew),

- $Agg(i) = Agg$,
- $T_{join} = T_{agg} = T_{proc}$, and
- Data transmission is carried out by message passing with a size of z.

The cost equation (7) can then be re-written below

$$JPM =$$

$$T_{comm}\left\{\left[\alpha(r+s)+\frac{J\times Agg\times(1+\theta)}{\beta}\right]/z\right\}$$

$$+T_{proc}\left[\alpha r\times\log(\alpha s)+\frac{J}{\beta}+\frac{(1+1)\times J\times Agg}{\beta}\right]$$

$$=$$

$$T_{comm}\left[\left(\frac{r+s}{\gamma+\ln N}+\frac{2\times(\gamma+\ln N)^2}{1+(\gamma+\ln N)^2}\times J\times Agg\times(1+\theta)\right)/z\right]$$

$$+T_{proc}\left(\frac{r}{\gamma+\ln N}\log\left(\frac{s}{\gamma+\ln N}\right)+\frac{2\times(\gamma+\ln N)^2}{1+(\gamma+\ln N)^2}\times J\times(1+2\times Agg)\right)$$

4.2 The *APM* Cost Model

The cost components for the *APM* cost models are explained as follows:

Data partitioning cost for *APM* is the sum of the fragment of r and the full table of s. To differentiate between the *JPM* and the *APM* cost models, in the *APM* cost models, we use j as an index, not an i. Hence, the data partitioning cost is as follows.

$$T_{comm}\times(\max(r_j+s)) \tag{8}$$

Local join cost for the *APM* is similar to that in the *JPM*, with an exception that the cardinality of the second table is a full table, not a fragment. The different local join costs for the *APM* are then as follows.

$$\text{nested loop} = T_{join}\times(\max(r_j\times s))$$
$$\text{hash} = T_{join}\times(\max(r_j+s))$$
$$\text{nested block index} = T_{join}\times(\max(r_j\times\log s)) \tag{9}$$

Aggregation costs consist of the local aggregation and the *Having* operation, which are calculated as follows.

$$T_{agg}\times(\max(r_j\times s\times Sel(j))\times(1+Agg(j))) \tag{10}$$

Finally, the *consolidation cost* is a result transfer cost which is influenced by the reduction factor imposed by the *Having* operation, as like in the *JPM*.

$$T_{comm}\times(\max(r_j\times s\times Sel(j)\times Agg(j)\times\theta)) \tag{11}$$

The sum of equations (8) to (11) gives the total cost for the *APM*, which is as follows.

$$APM =$$

$$T_{comm}\times(\max(r_j+s))+T_{join}\times(\max(r_j\log s))$$
$$+T_{agg}\times(\max(r_j\times s\times Sel(j))\times(1+Agg(j)))$$
$$+T_{comm}\times(\max(r_j\times s\times Sel(j)\times Agg(j)\times\theta)) \tag{12}$$

The skew factors α and β can be added to the above equation in the same way for the *JPM* method. For the purpose of comparison of the two methods, we assume that $J_j = r_j\times s\times Sel(j) = J$ and $Agg(j) = \frac{1}{N}Agg$. Notice that J_j takes s, whereas in the *JPM*, J_i takes s_i. Therefore, it is acceptable, that $Agg(j)$ in the *APM* is equal to Agg (as in the *JPM*) divided by N. The time of *APM* method can then be expressed as

$$APM =$$

$$T_{comm}\left[\left(\alpha r+s+\frac{J\times Agg\times\theta}{\beta\times N}\right)/z\right]+$$

$$T_{proc}\left[(\alpha r)\times\log s+\frac{J}{\beta}\times\left(1+\frac{Agg}{N}\right)\right]$$

$$=$$

$$T_{comm}\left[\left(\frac{r}{\gamma+\ln N}+s+\frac{2(\gamma+\ln N)^2}{1+(\gamma+\ln N)^2}\times\frac{J\times Agg\times\theta}{N}\right)/z\right]$$

$$+T_{proc}\left(\frac{r}{\gamma+\ln N}\log s+\frac{2(\gamma+\ln N)^2}{1+(\gamma+\ln N)^2}\times J\times\left(1+\frac{Agg}{N}\right)\right)$$

4.3 The *HPM* Cost Model

The cost components for *HPM* are as follows.

Data partitioning cost is determined by the fact that the group-by table is partitioned to all processors and all clusters, whereas the other table is partitioned to all clusters but replicated among all processors within each cluster. Assuming that R is the group-by table, total data transmission time is given by:

$$T_{comm}\times(\max(r_{ij}+s_j)) \tag{13}$$

where i is in the range of $[1,m]$ and j is in the range of $[1,n]$, and there are m clusters and n processors in each cluster.

Local join cost is determined by cardinalities of the two fragments, which are indicated by r_{ij} and s_j. Depending on the join algorithm, the local join costs are as follows.

$$\text{nested loop} = T_{join} \times (\max(r_{ij} \times s_j))$$
$$\text{hash} = T_{join} \times (\max(r_{ij} + s_j))$$
$$\text{nested block index} = T_{join} \times (\max(r_{ij} \times \log s_j)) \quad (14)$$

Local aggregation cost at each processor is determined by the cardinality of the join result, which is given as follows.

$$T_{agg} \times (\max(r_{ij} \times s_j \times Sel(j))) \quad (15)$$

Re-distribution cost is determined by the cardinality of the local aggregation results. We use an aggregation reduction factor *Agg(j)* in the cost equation. The transmission time of the local aggregation results to the processors within each cluster is given as follows.

$$T_{comm} \times (\max(r_{ij} \times s_j \times Sel(j) \times Agg(j))) \quad (16)$$

Global aggregation cost is composed of the merging cost and the *Having* operation cost. Like in the *JPM* we assume that the merging unit cost has the same value as the *Having* operation cost, which is identified by T_{agg}.

$$T_{agg} \times (\max(r_{ij} \times s_j \times Sel(j) \times Agg(j))) \times (1+1) \quad (17)$$

The last term of equation (17) indicates that one is for merging and the other for the *Having* cost.

Finally, the *consolidation cost* is the cost to transfer the results from the clusters to the host. The time for data consolidation in the host is small and thus only data transmission cost is counted, i.e.

$$T_{comm} \times (\max(r_{ij} \times s_j \times Sel(j) \times Agg(j) \times \theta)) \quad (18)$$

The total execution time of the *HPM* is the sum of the time of the above equations (13) to (18) and is given as

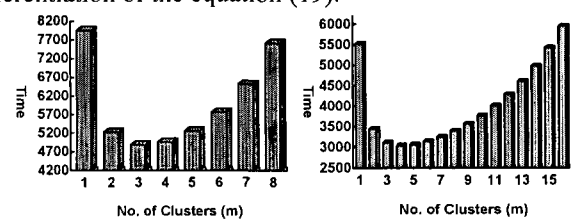
$$\begin{aligned} HPM = {} & T_{comm} \times (\max(r_{ij} + s_j)) + T_{join} \times (\max(r_{ij} \log s_j)) \\ & + T_{agg} \times (\max(r_{ij} \times s_j \times Sel(j))) \\ & + T_{comm} \times (\max(r_{ij} \times s_j \times Sel(j) \times Agg(j))) \\ & + T_{agg} \times (\max(r_{ij} \times s_j \times Sel(j) \times Agg(j))) \times 2 \\ & + T_{comm} \times (\max(r_{ij} \times s_j \times Sel(j) \times Agg(j) \times \theta)) \end{aligned} \quad (19)$$

By applying the same simplification assumptions in the previous methods and $Agg(j) = \frac{1}{m} Agg$, equation (19) is re-written as

$$\begin{aligned} HPM = {} & T_{comm} \left[\alpha_n \times \alpha_m \times r + \alpha_n \times s + \frac{(1+\theta)}{\beta_n} \times J \times \frac{Agg}{m} \right] / z \\ & + T_{proc} \left[\alpha_n \times \alpha_m \times r \times \log(\alpha_n \times s) + \frac{J}{\beta_n} \times \left(1 + 2 \times \frac{Agg}{m} \right) \right], \end{aligned}$$

where $J = r_{ij} \times s_j \times Sel(j)$, $\alpha_n = \frac{1}{\gamma + \ln n}$, $\alpha_m = \frac{1}{\gamma + \ln m}$, and $\beta_n = \frac{1 + (\gamma + \ln n)^2}{2(\gamma + \ln n)^2}$.

It can be seen from the above execution time equation that the number of clusters *m* has strong influence on the performance of *HPM*. Figures 6 show the experimentation of query execution time when the number of clusters is increased in the *HPM*. It appears that a value of $m \approx \lceil \sqrt{N} \rceil$ approximately gives the optimal cost of the query although the precise value of *m* may be worked out by finding the minimum value via the differentiation of the equation (19).



(a) 8 processors (b) 16 processors

Figure 6: Cost vs Number of Clusters in *HPM*.

5 Performance Evaluation

Performance evaluation of the three methods described in the previous sections is carried out in terms of simulation. A simulation program called *Transim*, a transputer-based simulator (Hart, 1993), was used in the experiments. The programs were written in an Occam-like language supported by *Transim*.

Using *Transim*, the number of processors and the architecture topology can be configured. Communication is done through channels, which are able to connect any pair of processors. Through this feature, the simulation model adopts a star network Master-Slave topology, where processing is done by distributing the work from the master to all slave processors. The hybrid topology can be built logically.

We conducted a sensitivity analysis by varying a number of important factors such as the degree of skewness (α and β), the aggregation factor, the table cardinality, the join selectivity factor, and the ratio of T_{comm} / T_{proc}. The default parameter values are given in Table 2.

Parameter	Value
N	16 or 32
m	$\sqrt{16}=4$ ($=\sqrt{N}$)
r	1000 records
s	1000 records
$Sel(i)$	$5(N/r)=0.08$
$Aggr(i)$	0.5
θ	0.5
α	0.2985
α_n	0.5093
α_m	0.5093
β	0.5446
β_n	0.6297
T_{comm}	0.1 standard time unit per message
T_{proc}	0.01 standard time unit per record
z	100 records per message

Table 2: Data Parameter Values.

5.1 Varying the Aggregation Factor

The aggregation factor $Agg(i)$ is defined by the ratio of result size after aggregation to the size of the base table. Figures 7 (a) and (b) show the impact of this aggregation factor on the three methods.

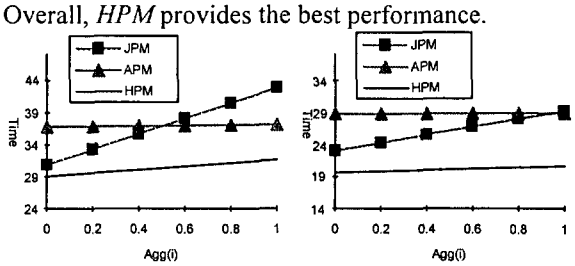
In Figure 7(a), the three methods are compared using 16 processors. The aggregation factor is varied from 0 to 1. Notice that *APM*'s performance is quite constant. The majority of the cost goes to the replication of table *S*. Once table *S* is replicated, the operations become simple. Data transmission after joining, and processing the *Having* predicate do not contribute too much to the overall cost. As a result, the performance looks quite constant. The data to select after the group-by condition (*Having*) is small, since the aggregation factor is already reduced by a factor of the number of processors.

On the other hand, *JPM*'s performance goes up as the aggregation factor increases. This is because *JPM* works in two levels, and the data to be transmitted during the processes is determined by the aggregation factor. Generally, the larger the aggregation factor the more the running time is needed.

The *HPM*'s performance provides the best performance, as it balances between the other two methods, whereby the join and aggregation are performed locally in each cluster, before the final aggregation is carried out. Using this method, the impact of the aggregation factor (as in *JPM*) and the replication (as in *APM*) is minimized.

Figure 7(b) shows a comparative performance among the three methods using 32 processors. Increasing the number of processors will reduce the running time despite data partitioning and processing skew. The trend of the three methods is similar. The main difference to note is that when the aggregation factor is small using more processors, *JPM* performance is than that of *APM*. This is due to the replication cost imposed by *APM* – the more processors, the higher the data transmission cost for data replication.

Overall, *HPM* provides the best performance.



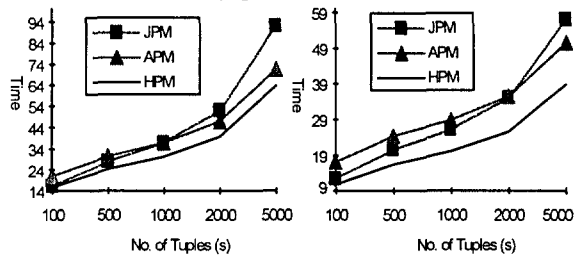
(a) 16 processors (b) 32 processors

Figure 7: Varying Aggregation Factor.

5.2 Varying the Table Cardinality

The cardinality of the operand tables is assumed the same elsewhere in the sensitivity analysis and their influences on performance are investigated here. Figure 8 shows the query execution time when we fix the cardinality of one table and other parameters and increase the cardinality of another table.

When the table size is small, *JPM* provides better performance than *APM*, while *HPM* outperforms other methods despite varying the table sizes. Comparing Figure 8(a) to Figure 8(b), increasing processors also raises the performance cross-over point of *JPM* and *APM*. With a small table, the complexity of join operation is reduced enormously, and consequently *JPM* takes advantage of the lower local processing cost and produces a comparatively good performance.



(a) 16 processors (b) 32 processors

Figure 8: Varying the Cardinality of Table S

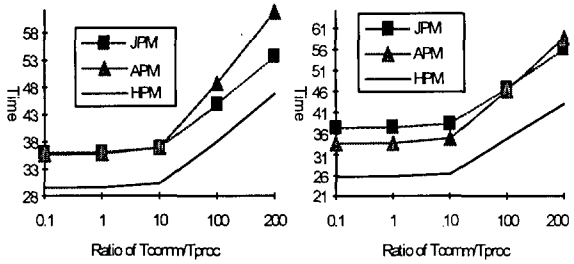
5.3 Varying the Ratio of T_{comm} / T_{proc}

The ratio of T_{comm} / T_{proc} reflects the characteristic of network used in the parallel architecture. The data communication time in parallel databases is not as critical as that in distributed systems (Almasi and Gottlieb, 1994).

As we increase the ratio, system performance decreases as shown in Figure 9 since we treat T_{proc} as a standard time unit and increase the communication cost. In the figure, higher ratio means the communication is expensive, and being a parallel database system, this ratio tends to stay small.

APM is the most sensitive to the communication cost because in our experiments both $Sel(i)$ and $Agg(i)$ are reasonably large resulting that the cost of the global aggregation in *JPM* is reasonably low. Nevertheless,

HPM always performs better than *JPM* and *APM* and the improvement comes from fragmenting tables within each cluster.

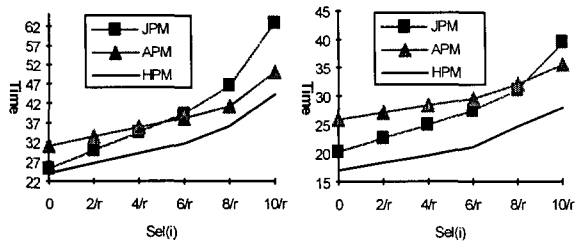


(a) 16 processors (b) 32 processors

Figure 9: Varying the Ratio of T_{comm} / T_{proc}.

5.4 Varying the Join Selectivity Factor

The join selectivity factor has significant influence on parallel aggregation processing as it determines the number of records resulting from join intermediately. After that, those records are processed for aggregation and evaluated with the predicates. Eventually, the qualified records are union-ed to form the query result. Lower selectivity factor involves less aggregation processing time and transferring time, and thus it is in favour of *JPM* as shown in Figure 10. Fewer processors will also reduce the impact of the entire second table (both communication and processing) on the overall execution time so it favours *APM*.



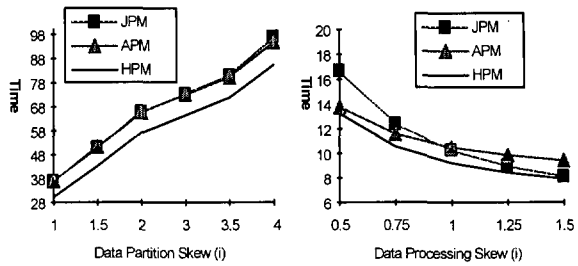
(a) 16 processors (b) 32 processors

Figure 10: Varying the Selectivity Factor.

5.5 Varying the Degree of Skewness

Figure 11(a) indicates the tendency of the performance when the data processing skew changes accordingly with the data partitioning skew whereas Figure 11(b) provides the comparison when we ignore the data partitioning skew, i.e. $\alpha = 1/N$ and alter the data processing skew. The values on the horizontal axis of both figures represent the expanding skewness factor which then is multiplied by the basic unit given by *Zipf* distribution. Unlike the factor α, the factor β is inversely proportional to data processing skew, and the larger the factor β the less the data processing skew is.

We conclude from Figure 11 that either partitioning skew or processing skew degrades the performance of parallel processing, *HPM* outperforms *APM* and *JPM* even in the presence of skewness, and *APM* is less affected by the skewness compared with *JPM*.



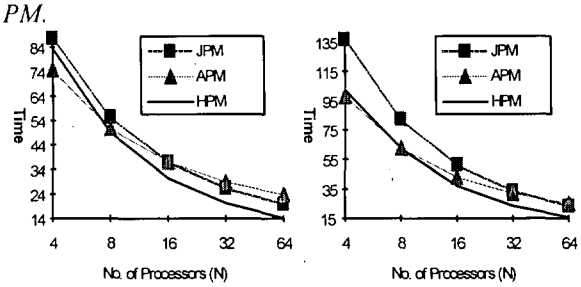
(a) Data partition skew (b) Data processing skew

Figure 11:Varying the Skewness with 16 Processors.

5.6 Varying the Number of Processors

One of the desired goals of parallel processing is to have linear scale up which can be achieved when twice as much processors perform twice as large a task in the same time. When the number of processors is increased, as we expected, the performance is upgraded in spite of the skewness shown in Figure 12.

As shown in the figure, when the number of processors is small, *APM* performs extremely well (even better than *HPM*) because the number of clusters in *HPM* may not be optimised and less processors make the communication cost insignificant which is in favour of *APM*. However, when the database system scales up, both *HPM* and *JPM* provide better performance than *APM*.



(a) (b)

(a) $Agg(i) = 0.5$, $Sel(i) = 5(N/r)$
(b) $Agg(i) = 0.6$, $Sel(i) = 10(N/r)$

Figure 12: Varying the Number of Processors.

6 Conclusions and Future Work

Traditionally, join operation is processed before aggregation operation and tables are partitioned on join attribute. In this paper, we demonstrate that group-by attribute may also be chosen as the partition attribute and present three parallel methods for aggregation queries, *JPM*, *APM*, and *HPM*. These methods differ in the way of distributing query tables, i.e. partitioning on the join attribute, on the group-by attribute, or on a combination of both; consequently, they give rise to different query execution costs. In addition, the problem of data skew has been taken into account in the proposed methods as it may adversely affect the performance advantage of parallel processing. The cost analysis shows that when the join selectivity factor is small and the degree of skewness is low, *JPM* leads to less cost; otherwise *APM* is desirable. Nevertheless, the hybrid method (*HPM*) is always superior to the other two methods since the data

partitioning is adaptively made on both join attribute and group-by attribute. In addition, it can be observed that the partitioning on group-by attribute method is insensitive to the aggregation factor and thus the method will simplify algorithm design and implementation.

Our future work is planned to investigate the behaviour of the *HPM* method in a real hybrid architecture whereby the number of clusters and the number of processors within each cluster are predetermined. We will also take into consideration the fact that it is common that processors within each cluster normally share the memory (i.e. shared-memory SMP machines), but different clusters communicate through network (i.e. shared-nothing among clusters) (Valduriez, 1993). It will be interesting to see the impact of two levels of partitioning methods like in the *HPM* model in real hybrid architectures.

Acknowledgement

I would like to thank M.Z. Ashrafi and L. Tan for formatting the camera-ready version of this paper.

References

[1] Almasi G., and Gottlieb, A., *Highly Parallel Computing*, Second edition, The Benjamin/ Cummings Publishing Company Inc., 1994.

[2] Bedell J.A. "Outstanding Challenges in OLAP", Proceedings of 14th International Conference on Data Engineering, 1998.

[3] Bultzingsloewen G., "Translating and optimizing SQL queries having aggregate", *Proceedings of the 13th International Conference on Very Large Data Bases*, 1987.

[4] Datta A. and Moon B., "A case for parallelism in data warehousing and OLAP", Proceedings. of 9th International Workshop on Database and Expert Systems Applications, 1998.

[5] Dayal U., "Of nests and trees: a unified approach to processing queries that contain nested subqueries, aggregates, and quantifiers", *Proceedings of the 13th International Conference on Very Large Data Bases*, Brighton, UK, 1987.

[6] DeWitt, D.J. and Gray, J., "Parallel Database Systems: The Future of High Performance Database Systems", *Communication of the ACM*, vol. 35, no. 6, pp. 85-98, 1992.

[7] Graefe G., "Query evaluation techniques for large databases", *ACM Computing Surveys*, Vol.25, No.2, June 1993.

[8] Hart, E., *Transim: Prototyping Parallel Algorithms*, User Guide and Reference Manual, Transim version 3.5, University of Westminster, August 1993.

[9] Kim, W., "On optimizing an SQL-like nested query", *ACM Transactions on Database Systems*, Vol 7, No 3, September 1982.

[10] Knuth D. E., "The art of computer programming", Volume 3, *Addison-Wesley Publishing Company, INC.*, 1973.

[11] Leung C. H. C. and K. H. Liu, "Skewness analysis of parallel join execution", *Technical Report*, Department of Computer and Mathematical Sciences, Victoria University of Technology, Melbourne, Victoria, Australia, 1994.

[12] Leung, C.H.C. and Ghogomu, H.T., "A High-Performance Parallel Database Architecture", *Proceedings of the Seventh ACM International Conference on Supercomputing*, Tokyo, pp. 377-386, 1993.

[13] Leung, C.H.C. and Taniar. D., "Parallel Query Processing in Object-Oriented Database Systems", *Australian Computer Science Communications*, vol 17, no 2, pp. 119-131, 1995.

[14] Liu K. H., Leung C. H. C., and Y. Jiang, "Analysis and taxonomy of skew in parallel database systems", *Proceedings of the International Symposium on High Performance Computing Systems* (HPCS'95), Montreal, Canada, pp. 304-315, July 1995.

[15] Liu K. H., Y. Jiang, and C.H.C. Leung, "Query execution in the presence of data skew in parallel databases", *Australian Computer Science Communications*, vol 18, no 2, pp. 157-166, 1996.

[16] Mishra, P. and Eich, M.H., "Join Processing in Relational Databases", *ACM Computing Surveys*, vol. 24, no. 1, pp. 63-113, March 1992.

[17] Valduriez, P., "Parallel Database Systems: The Case for Shared-Something", *Proceedings of the International Conference on Data Engineering*, pp. 460-465, 1993.

[18] Wolf J. L., Dias D. M., and P. S. Yu, "A parallel sort-merge join algorithm for managing data skew", *IEEE Transactions On Parallel And Distributed Systems*, Vol. 4, No.1, January 1993(a).

[19] Wolf J. L., Yu P. S., Turek J. and D. M. Dias, "A parallel hash join algorithm for managing data skew", *IEEE Transactions On Parallel and Distributed Systems*, vol.4, no. 12, December 1993(b).

[20] Yan W. P. and P. Larson, "Performing group-by before join", *Proceedings of the International Conference on Data Engineering*, 1994.

Developing Software Across Time Zones: An Exploratory Empirical Study

Adel Taweel and Pearl Brereton
Department of Computer Science
Keele University, Keele
Staffordshire ST5 5BG, UK
Email: a.taweel or o.p.brereton@cs.keele.ac.uk
www.keele.ac.uk/depts/cs

Keywords: global software development, collaborative working, distributed software engineering, virtual teams.

Received: May 11, 2002

Market forces and an increasingly reliable world-wide communications network have made geographically distributed software engineering a reality. Global software development enables businesses to respond more easily and more quickly to global market opportunities and to improve product and service quality. One of the many potential benefits of global development is a reduction in development time through the adoption of an 'around the clock' working practice.

This paper introduces a *sequential collaborative software engineering* process involving shift working across time zones and describes an exploratory empirical study of this working pattern involving the implementation of a small-scale software system.

The paper reports on the organisation of the study and on the results obtained through questionnaires, observations and measurements.

1 Introduction

The empirical study described in this paper was carried out as part of a research project investigating the potential for reducing the time needed to carry out software engineering tasks (and hence to deliver software products) through the use of around the clock working. The working style being addressed is one where a task is passed in a sequential manner from one software engineer to another 'across time zones'. We call this sequential collaborative software engineering (SCSE).

A general three-site scenario is illustrated in Figure-1, and shows:

- the *time differences* between sites (which may involve some overlap);
- the *reporting time* when, at the end of a working period or shift, a developer records progress made;
- the *catching up time*, when a developer catches up with the work carried out during the previous shift.

As well as undertaking an empirical study of SCSE, our research has involved the development of a set of equations which models the relationships between estimated development time using SCSE, estimated development time for single-site working, a number of contextual factors and the overheads associated with SCSE [Taweel02].

The contextual factors that are included in the model are:

- the number of sites participating in a collaboration – this is likely to be either two or three sites;
- the time differences between the collaborating sites;
- the number of participating developers;
- the level of concurrency (i.e. the task-specific potential for concurrent working).

The overheads of distribution that are accommodated are:

- extra management, since the added complexity of SCSE over single-site development is likely to require more planning and progress monitoring;
- general task-level communications, which may be needed from time to time for a range of purposes such as reviewing decisions or negotiating the overall approach to be taken;
- reporting time – i.e. the time taken by a developer to report progress at the end of a shift;
- catching up time – i.e. the time taken by a developer to catch up with work completed since completing his/her previous shift;

- distribution effort loss, which is the time lost when a developer at one site fails to complete a task on which a subsequent developer is depending.

The aims of our empirical study were to illustrate the feasibility of employing this type of work pattern and to identify the critical success factors for SCSE. We also expected to obtain some values for the associated overheads (such as reporting time and catching up time) for the particular experimental context.

The remainder of the paper is organised as follows. Section 2 and 3 describe the preparatory activities (such as the selection of subjects, the design of the study and document preparation) and the execution of the selected task. Section 4 and 5 summarise analysis and observations, critical success factors, and final remarks.

2 Preparatory activities

Any empirical study, whether it is a formal experiment, a case study involving a real project, a survey or, as in our case, an informal exploratory study, is a substantial undertaking needing a considerable amount of careful planning [Bausell86, Kitchenham97]. The task is even more challenging when the subjects need to work collaboratively and are to be located in different countries and different time zones.

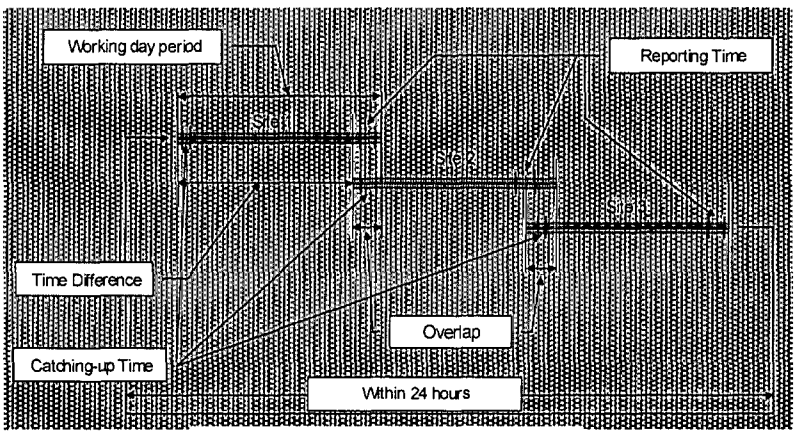


Figure 1: Three Sites

Preparatory activities include the selection of subjects, the overall design of the study and the production of a range of documents needed to support the study and transfer of knowledge between sites.

2.1 Selection of subjects

As for many empirical studies, obtaining subjects is a major problem and this was particularly difficult in our case because of the need to recruit subjects at sites located in different time zones. The sites chosen were Keele (our 'home' site in the UK) and Hebron (in Israel), which have a time difference of 2 hours. We were able to use four subjects (2 in each location) and therefore the SCSE task was executed twice (using 1 subject per site on each occasion). The two sites have similar computing facilities and the subjects have good computing and English language skills.

2.2 Design of the study

The design of the study covers a range of activities including the selection of tasks to be carried out, planning the execution in terms of the procedures to be followed, scheduling and monitoring, deciding what data should be collected and arranging the collection of the data.

2.2.1 The software task

In choosing a software task, a compromise had to be found between the available resources (especially in terms of people and their time) on the one side and the complexity of the task on the other side. Although it may be feasible to distribute other software engineering tasks (such as design and requirements analysis) we felt that the less creative nature of coding (compared especially to designing) would provide greater opportunity for successful collaboration over the necessarily limited time scale of the study. The software engineering task chosen for implementation was a calculator program based on a design that was produced at Keele. The task was selected on the basis that it should be possible to complete the implementation in the time available but that the programming effort would be non-trivial. The requirements specification and the high level design of the chosen software task are shown in Appendix 3

2.2.2 Methods and tools

One essential step was to decide what development methods and tools to use during the study. The approach taken was to choose the method and tools that were familiar to the subjects, in order to avoid the need for training, especially because carrying out training for remote subjects would be very difficult.

These constraints led us to select Java as the programming language and a hybrid of the Yourdon

method [Yourdon79, Coad91] and UML [OMG98, Rumbaugh99] as the modelling language.

2.2.3 The work pattern

Constraints on the availability of subjects led us to schedule the implementation over five shifts each of between two and three hours duration. The time difference between sites meant that the shifts spanned either two or three days. This allowed the subjects in Hebron to fit in their shifts either immediately before or immediately after their normal working hours.

The plan was to give the design document to the subjects a few days in advance and to allow them to ask for clarification from the designer if required.

An initial expected ordering and scheduling of the implementation activities was determined although the subjects were not required to adhere rigidly to either.

2.3 Data collection, knowledge transfer and preparation of documents

Because it was not possible to meet all of the subjects face-to-face, all of the necessary instructions and data collection templates needed to be in electronic form. Several documents were prepared for the different phases of the study:

- *Instructions to subjects* which included an overview of the study and the procedures that should be followed by subjects at each site for each phase;
- *Task description and design* which provided the requirements specification, design and coding convention for the system to be developed;
- *Implementation schedule* proposing an initial allocation of tasks (units of implementation) to subjects over the five shifts;
- *Development dependency chart* describing the dependencies between components of the software system being developed. This document was essential to help subjects understand the various possible orders in which the components could be implemented. This would enable subjects to devise an alternative schedule if they were unable to adhere to the proposed initial schedule;
- *Set-up completion form* to collect information about the documents received by each site and/or problems encountered during the set-up phase;
- *Knowledge transfer template* to be used by subjects during the execution phase to report the status of the task being carried out, enabling them to report problems encountered with received and sent tasks, actions taken by them and progress made. In addition to the code, this template represents the main means of knowledge transfer between sites (see Appendix 5);
- *Timing collection template* to record the catching-up time, reporting time, time spent coding and communication times for each shift;
- *Subject information form* used during the termination phase to collect information about the subjects' experience and personal information;
- *Questionnaire to collect qualitative data* about the subjects experiences and opinions of SCSE.

3 Task execution

The task execution involved three phases: the set-up phase, the execution phase and the termination phase. During the **set-up phase**, documents were distributed to the subjects and any problems addressed. The **execution phase** included the implementation of the software application, collection of quantitative data and progress monitoring. During this phase only source code files and status report of shifts (or implementation units) were transferred between subjects. In the **termination phase**, qualitative data was collected and the software application tested. During the three phases, email was used to transfer documents between subjects and between subjects and the evaluator. Details of a particular shift, which illustrates the activities performed and the information exchanged between sites, are included in Appendix 2.

4 Analysis and Observations

The quantitative data collected throughout the study enabled us to gain some insight into the overheads associated with SCSE in the particular context of the study. Data relating to time spent on the task for each shift, the catching up time, the reporting time and time spent communicating is shown in Table 1. The variations in values for catching up time and reporting time were small however the communications times for Hebron shifts were considerably longer than for the Keele shifts. This was because the Hebron subjects had to use a dial-up connection whereas Keele subjects had a permanent Internet connection.

It can be seen that on average the total knowledge transfer time (catching up time, reporting time and communications) is 31 minutes. Since the total shift time (on average) was 129 minutes, knowledge transfer made up approximately 24% of the time. Although this proportion is high, it is mainly made up of communication time. When communications facilities are good, as for the Keele site, then the proportion is less than 13%. This overhead works out at less than 9% for a working day of 8 hours under the assumptions that

the communication time remains constant and the catching up time and reporting time increase linearly.

	Overall average across 10shifts (min)	Hebron average across 6 shifts (min)	Keele average across 4 shifts (min)	SD
Development time	98	101	95	-
Catching up time	5	5	3	1
Reporting time	8	11	5	3
Communications time	18	20	6	8

Table 1: Data showing some overheads of distribution

It can be seen that on average the total knowledge transfer time (catching up time, reporting time and communications) is 31 minutes. Since the total shift time (on average) was 129 minutes, knowledge transfer made up approximately 24% of the time. Although this proportion is high, it is mainly made up of communication time. When communications facilities are good, as for the Keele site, then the proportion is less than 13%. This overhead works out at less than 9% for a working day of 8 hours under the assumptions that the communication time remains constant and the catching up time and reporting time increase linearly.

These quite low values for catching up time and reporting time are encouraging especially given the low level of automation provided during the study to support these activities. A greater level of automation is possible and likely if SCSE is used in a commercial setting.

The qualitative data collected was concerned with the subjects' opinions on a number of issues (see Table 2). Although, from their general comments, the subjects felt SCSE to be very efficient it was not a particularly popular way of working. This seems to be because, to some extent, subjects felt that their working style was constrained. Nevertheless some were keen to try the approach for larger scale projects.

Of the general comments from subjects, the most notable was an expression of the need for high quality documentation especially relating to requirements, design, design rationale and coding conventions.

In terms of product quality, the overall approach and process followed seems to have had no ill effects. In fact the final systems were fully tested and appeared to have no implementation errors (although one design fault was found).

An additional observation was that if subjects finished their allocated activities early they did not continue to work until the end of the shift (even though they had been told to do so). However, subjects also felt that they were 'under pressure' to complete the activities scheduled. Clearly a balance needs to be struck between the view that '*I have done my bit so I can stop*' and '*I can't keep up with the work rate expected of me*'.

	Yes	No
Was synchronous communications needed?	0	4
Were design documents adequate?	4	0
Were problems encountered with the communications?	1	3
Do you think that SCSE would be more suitable for bigger projects?	2	1 (+1 unsure)
Did you feel that SCSE restricted your work?	3	1
Would you prefer to use SCSE as opposed to single-site development?	0	2 (+2 unsure)

Table 2: Summary of questionnaire responses

5 Critical success factors

The aims of this work were to illustrate the feasibility of employing SCSE, to identify the critical success factors and to obtain values for the associated overheads (in the context of the study). The values for knowledge transfer time, catching-up time and reporting time are not unduly high and suggest that the overheads associated with SCSE will not prevent it from being a feasible working pattern, In addition, although SCSE was not a popular working pattern, subjects felt that it was efficient and had potential for use in larger projects. In the remainder of this section, we outline the critical success factors.

5.1 Planning

Good planning is crucial for the success of SCSE. Plans need to include both a detailed task schedule and contingencies. Further, it is apparent that timing restrictions imposed by SCSE put software engineers under pressure. This suggests that the distribution pattern with the maximum timing flexibility should be chosen. In addition, although the values of the overheads obtained cannot be generalised it is clear that such overheads are strongly dependent on organisational and management factors. It is important, therefore, to accommodate controlled deviation from planned schedules.

5.2 Documentation

Precise and complete documentation is a critical requirement throughout SCSE as well as being a consequence of the process. A documentation standard must be strictly defined to reduce ambiguities and ensure consistency. Documentation needed includes details of development schedule, requirement specifications, design method and symbols and coding conventions. SCSE also results in the actual process followed being well documented. Any difficulties arising, decisions made, rationales etc. are documented as part of the process and such information could be made available for later analysis if desired.

5.3 Software engineering tasks

Although we expect SCSE to be applicable to different types of software engineering tasks, this study only illustrates its applicability to coding and unit testing. It may be that the more creative tasks that would require a higher degree of collaboration (e.g. design) could be done using SCSE, however, it is likely that such tasks would require substantially better tool support and a great level of synchronous collaboration. The success of SCSE for a software task does not only depend on the characteristics of the software task itself but also on how these characteristics are addressed and considered during the planning and documentation.

5.4 Communications

Not surprisingly, it is essential that the communications mechanisms and tools are easy to use and reliable for SCSE to work successfully.

5.5 Development methods and tools

One of the factors that emerged during this study was the importance of having a compatible set of development methods and tools at participating sites. It was also important that subjects were familiar with their use.

6 Final remarks

In addition to the inherent advantages of SCSE in terms of reduce time scales, we can expect the code produced to be both reliable and maintainable. Three observations lead us to this conclusion:

- the process requires strict adherence to documented coding conventions;
- the code is developed by more than one software engineer so has some of the benefits, such as continuous review, of pair programming [Williams00]);
- every step and phase of development is thoroughly documented (driven by the need for extra planning and supplemented by the information accumulated through the knowledge transfer templates).

Acknowledgements

The authors acknowledge the support of the British Telecommunications plc. who funded this work and the help of the subjects who took part in the empirical study.

References

Bausell86 Bausell, R. B., *A Practical Guide to Conducting Empirical Research*, Harper & Row Publishers, 1986.

Coad91 Coad, P. & Yourdon, E., *Object Oriented Analysis*, 2nd Ed., Yourdon Press, Englewood Cliffs, N.J, 1991.

Kitchenham97 Kitchenham, B., Linkman, S. & Law, D., *DESMET: a methodlogy for evaluating software engineering methods and tools*, Computing & Control Engineering Journal, Vol. 8, No. 3, pp. 120-126, June 1997.

OMG98 OMG, *Unfiied Modeling Language Specification*, Object Management Group, Framingham, Mass, URL: www.omg.org, 1998.

Rumbaugh99 Rumbaugh, J., Jacobson, I. & Booch, G., *The Unified Modeling Language Reference Manual*, Addison Wesley Longman, Inc., 1999.

Taweel02 Taweel, A., & Brereton, O.P., *Software Development Across Time Zones*, In preparation.

Williams00 Williams, L. and Kessler R. R., Cunningham,W. and Jeffries, R., *Strengthening the Case for Pair-Programming*, IEEE Software, 2000.

Yourdon79 Yourdon, E. & Constantine L., *Structured Design: Fundamentals of a Discipline of Computer Program and Sytems Design*, Yourdon Press, Englewood Cliffs, N.J, 1979.

Appendix 1: Equations for estimating distributed development time

The **gain factors** (GF_{N_s} and GF_{N_e}) in development time (with respect to single site development) are represented by equations 1.0 and 2.0 where:

N_s is the number of utilised sites

N_e is the number of developers

O_i is the overlap between consecutive sites

GF_{N_s} is the gain factor based on the number of utilised sites

GF_{N_e} is the gain factor based on the number of developers

WD is the working day at each site (it is assumed that the working day at all sites has the same value).

$$GF_{N_s} = \sum_{i=1}^{N_s-1} \left(\frac{WD - O_i}{N_s \times WD} \right)$$

$$GF_{N_e} = \sum_{i=1}^{N_e-1} \left(\frac{WD - O_{si}}{N_e \times WD} \right)$$

Where O_{si} is the overlap between i^{th} site where the i^{th} developer is located and the i+1 site where i+1 developer (member of the team) is located. $O_{si} = 0$ for developers who are not part of a team.

The reduction in development time (Reduction Factors) is expressed in the following equations:

$$RF_{N_e} = 1 - GF_{N_e} \;,\; 0 < RF_{N_e} \leq 1$$

$$RF_{N_s} = 1 - GF_{N_s} \;,\; 0 < RF_{N_s} \leq 1$$

$$where\ RF_{N_s} \geq RF_{N_e}$$

The estimated development time for SCSE is expressed by the following equation:

$$NDT = ST \times RF_{N_s} + PT \times RF_{N_e} + Overheads$$

Where

ST is the development time estimate for sequential tasks

PT is the development time estimate for parallel tasks

EDT is the development time estimate for single site development and

PW is the level of potential concurrent working in the given software task.

NDT is the multi-site development time estimate

$$ST = EDT - EDT \times PW$$

where

$$PT = EDT \times PW$$

and

The equations derived are based on the assumption that all utilised sites work within a 24-hour period. This is expressed by the following equation:

$$Total\ working\ period = \sum_{i=1}^{N_s} WD_i - \left(\sum_{i=1}^{N_s-1} WD_i - TD_i \right)$$

Appendix 2: Details of a shift

The particular shift described is shift two for subject pair B.

At the beginning of the shift the subject received eight files as email attachments. These were:

- the knowledge transfer form (an MSWord document) which was completed by the collaborating subject at the end of shift one;
- the seven source files that had been created during the set up phase and which held all of the coding produced so far. These were made up of four Java source files - Calculator.java, MathFunction.java, Accumulator.java and Operation.java and three VisualCafe project files - Calculator.vep, Calculator.ve2 and Calculator.vpj.

The activities undertaken and the outcomes or results are shown in Table 1 below.

After approximated 70 minutes of the two-hour shift the subject had completed the scheduled activities (see row three in table 1). He had been told that if he finished early he should move on to the activities scheduled for the following shift. However, he did not do this, but instead moved to the 'end of shift' activities.

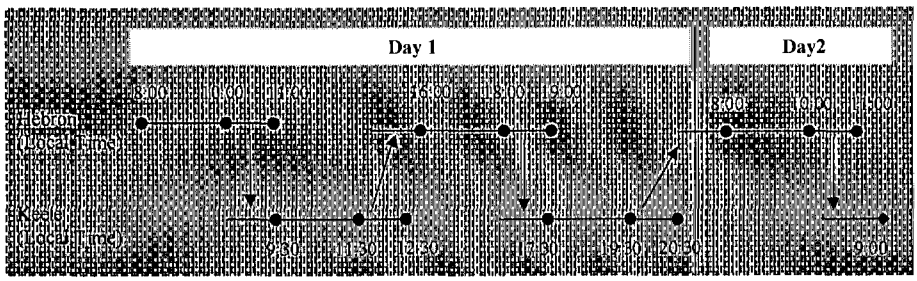


Figure 2: Shift Plan – Subject Pair A

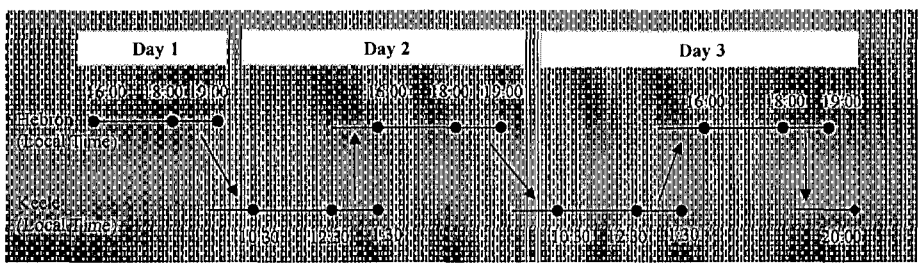


Figure 3: Shift Plan – Subject Pair B

Activity	Outcome/result
1. checked if all necessary documents were received and readable	Yes .... No problems
2. reviewed the knowledge transfer form (see Appendix 5)	One problem noted: *actionPerformed* method could not be fully tested because it calls methods that had not yet been written. To enable testing of all the class, the code written for calling these methods is kept as comments. Subjects at subsequent shifts retest respective code portions of this method when the methods being called are completed (this is part of the unit testing plan outlined in the task description and design document).
3. coded according to the implementation schedule from the point at which the collaborating subject finished	Since the scheduled activities had been completed during shift one this subject worked on methods for the *Accumulator class, Operation class and MathFunctions class*, until all scheduled activities were completed and tested. This shift also involved testing the respective part (that used the methods written in this shift) from the *actionPerformed* method according to the problem highlighted above.
4. performed 'end of shift' activities	Completed a knowledge transfer form, reporting that all files had been received successfully from shift one, all scheduled activities had been performed and no problems had been encountered. All completed tasks were marked with a Y (yes) in the implementation schedule (see Appendix 5). The knowledge transfer form and seven source files were emailed to the collaborating subject (who would work during shift three).

Table 1: Typical Shift Activities

Appendix 3: Requirements specification and class diagram

System Name: **Calculator**

Requirement Specifications:

This application is a normal common calculator. It should have the following specifications:

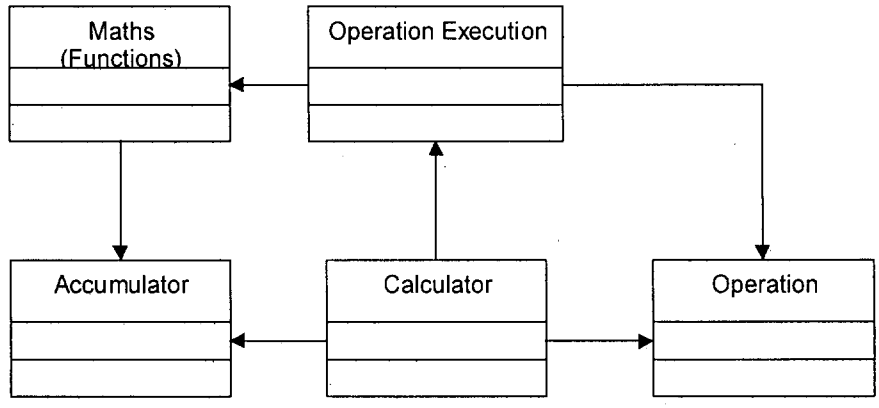
1. An applet based application that runs in a Java-based browser
2. Can be activated using the mouse.
3. Performs the following mathematical functions
   - 3.1. Addition (+)
   - 3.2. Subtraction (-)
   - 3.3. Multiplication (*)
   - 3.4. Division (/)
   - 3.5. Power (x^y)
   - 3.6. Square (x*x)
   - 3.7. Percent (%)
   - 3.8. Square Root
4. It has a standard interface

Exclusions:

1. It does not provide a help systems
2. It does not include any menu/options for upgrade.
3. It runs on a browser that supports the current version of Java (version 1.2)
4. It can only be activated using the mouse and not the keyboard

Behavioural Restrictions:

1. Display: Numbers should be left aligned with respect to the display. Calculator when started/restarted/reinitialised should display 0.0. Numbers when displayed should be in the double format (e.g. 11.0, 11.2, 0.112 etc.)
2. One Operand Functions: For these functions enter a number and then requesting a function should do the calculation on the entered number and display the result.
3. Two operand Functions: for these functions enter a number, request a function, enter another number and then requesting another function or get a result function (Equal) displays the result of the first requested function.
4. Get a result function (Equal): execute the last requested operation (for two operand function only) and displays the result of the calculation. Subsequent request of this function should not invoke any function or change displayed result

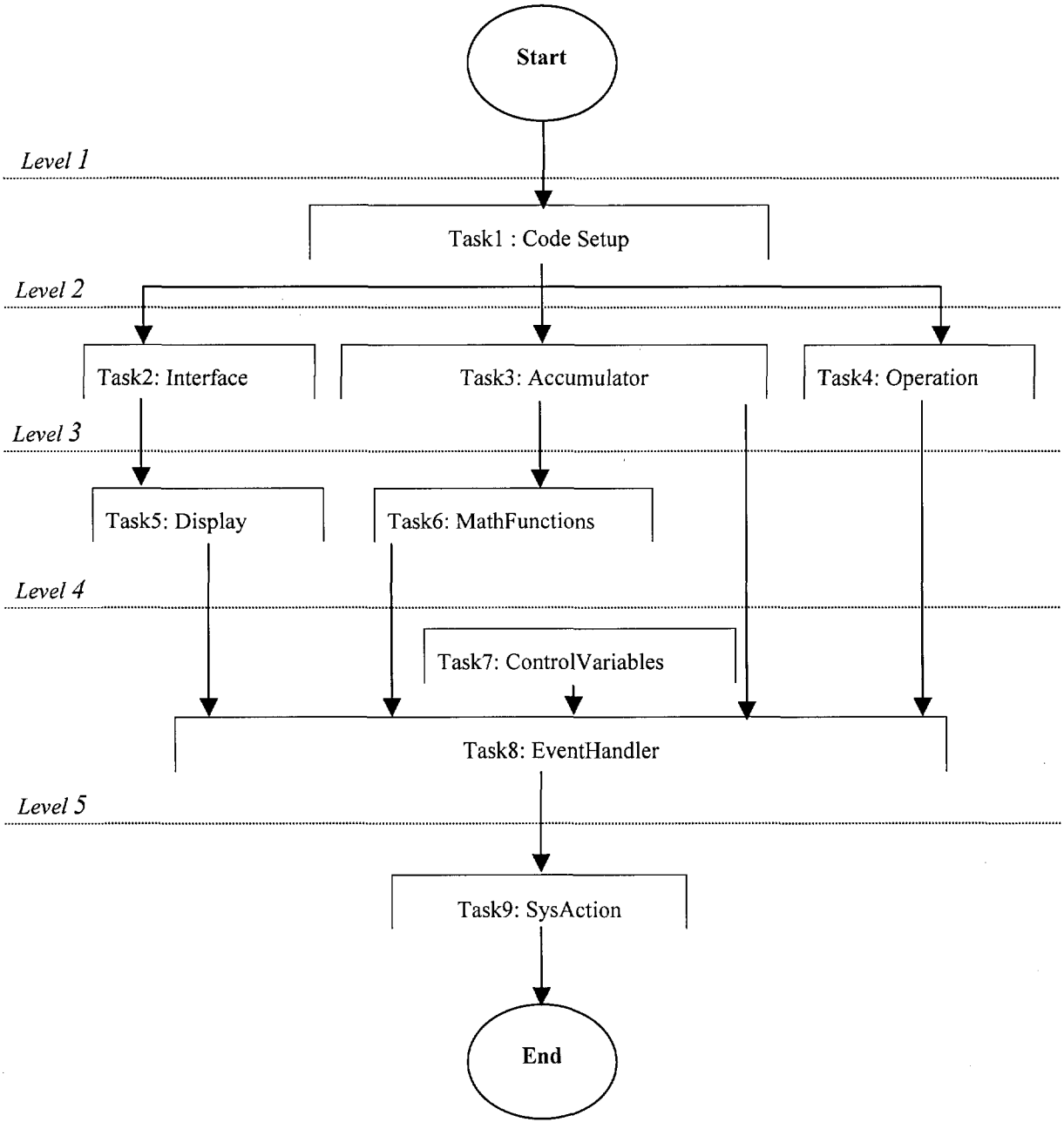


Class Diagram

Appendix 4: Implementation Schedule and Dependency Chart

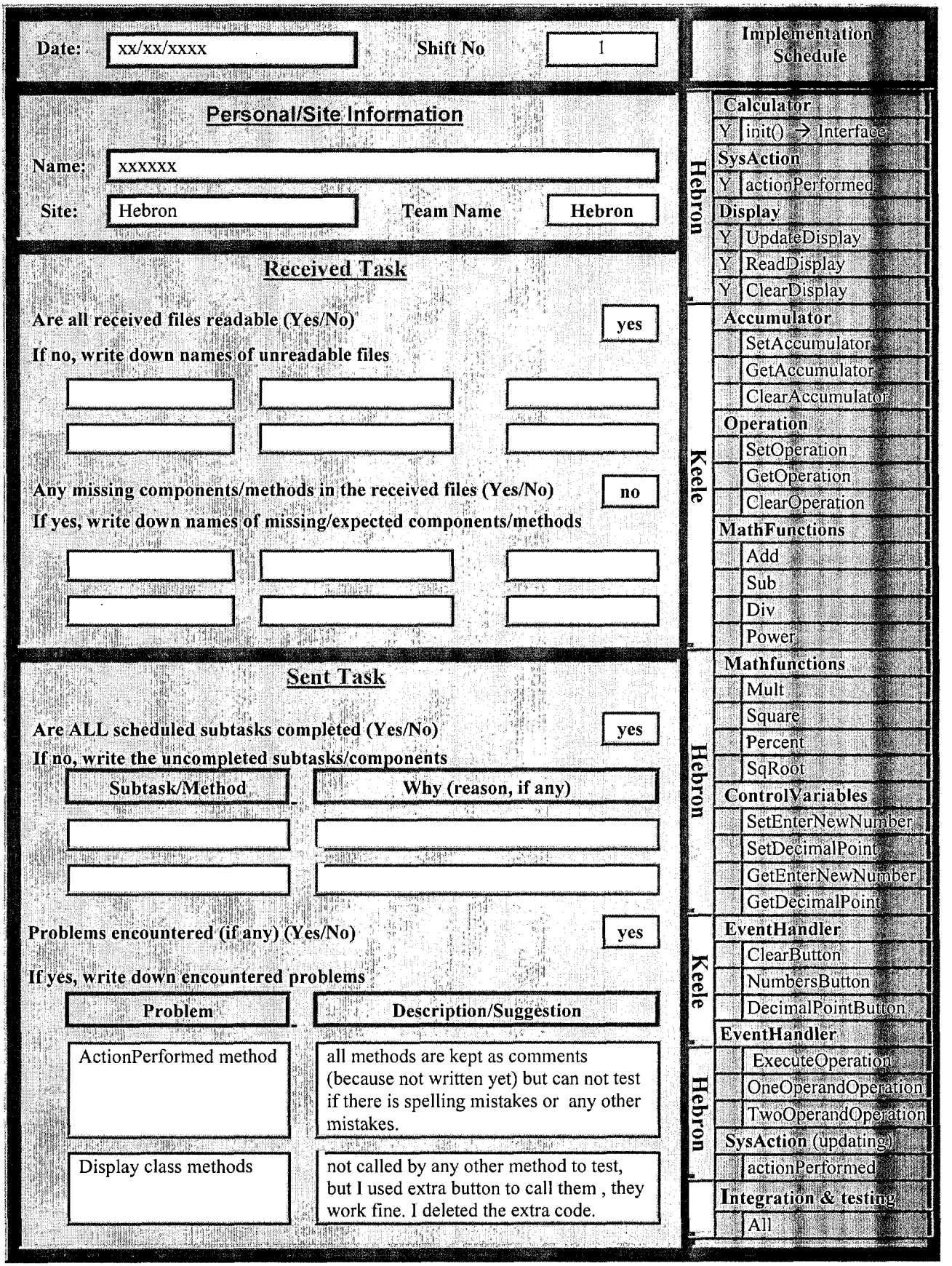
Implementation Schedule

	Hebron		Keele	
Shift 1	**Class**	**Components**		
	Calculator	init() – the interface only		
	SysAction	All		
	Display	All		
Shift 2			**Class**	**Components**
			Accumulator	All
			Operation	All
			Math Functions	Add(), Sub(), Div() Power()
Shift 3	**Class**	**Components**		
	Math Functions	Mult(), Square(), Percent() SqRoot()		
	Control Variables	All		
Shift 4			**Class**	**Components**
			EventHandler	ClearButton() NumbersButton() DecimalPointButton()
Shift 5	**Class**	**Components**		
	EventHandler	ExecuteOperation() OneOperandOperationButton() TwoOperandOperationButton EqualButton()		
	SysAction	actionPerformed() – Update		
Shift 6			**Task**	**Components**
			Integration and Testing	All



Development Dependency Chart

Appendix 5: Knowledge Transfer template



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 700 staff, has 500 researchers, about 250 of whom are postgraduates, over 200 of whom have doctorates (Ph.D.), and around 150 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.

In the last year on the site of 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.

At the present time, part of the Institute is being 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 is being developed at a particularly historical moment, characterized by the process of state reorganisation, privatisation and private initiative. The national Technology Park will take the form of a shareholding company and will host an independent venture-capital institution.

The promoters and operational entities of the project are the Republic of Slovenia, Ministry of 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 Economic Relations and Development, 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 1 219 385
Tlx.:31 296 JOSTIN SI
WWW: http://www.ijs.si
E-mail: matjaz.gams@ijs.si
Contact person for the Park: Iztok Lesjak, M.Sc.
Public relations: Natalija Polenec

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 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.

Volume 26 Number 3 November 2002 ISSN 0350-5596

Informatica

An International Journal of Computing and Informatics