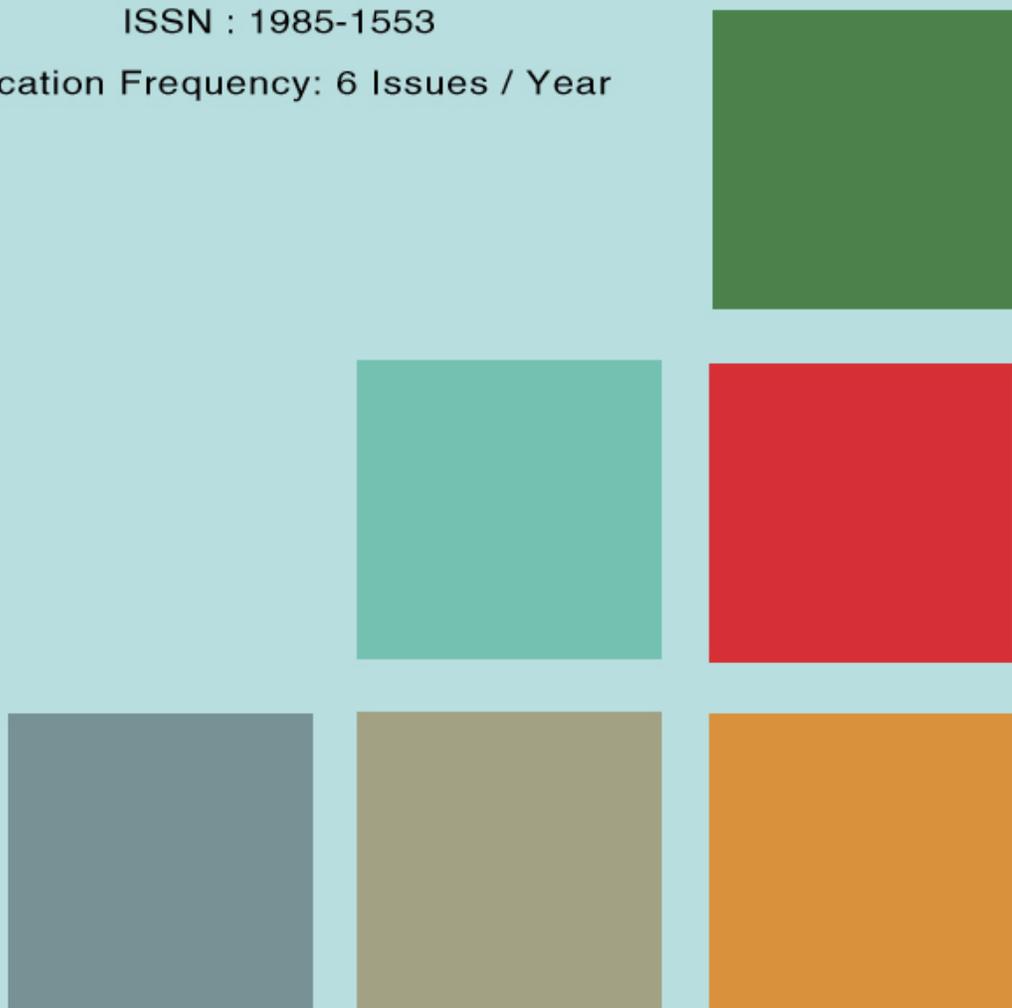


Volume 5 ▪ Issue 2 ▪ May 2011

INTERNATIONAL JOURNAL OF
COMPUTER SCIENCE AND SECURITY (IJCSS)

ISSN : 1985-1553

Publication Frequency: 6 Issues / Year



CSC PUBLISHERS
<http://www.cscjournals.org>

INTERNATIONAL JOURNAL OF COMPUTER SCIENCE AND SECURITY (IJCSS)

VOLUME 5, ISSUE 2, 2011

**EDITED BY
DR. NABEEL TAHIR**

ISSN (Online): 1985-1553

International Journal of Computer Science and Security is published both in traditional paper form and in Internet. This journal is published at the website <http://www.cscjournals.org>, maintained by Computer Science Journals (CSC Journals), Malaysia.

IJCSS Journal is a part of CSC Publishers

Computer Science Journals

<http://www.cscjournals.org>

INTERNATIONAL JOURNAL OF COMPUTER SCIENCE AND SECURITY (IJCSS)

Book: Volume 5, Issue 2, May 2011

Publishing Date: 31- 05 - 2011

ISSN (Online): 1985 -1553

This work is subjected to copyright. All rights are reserved whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, re-use of illustrations, recitation, broadcasting, reproduction on microfilms or in any other way, and storage in data banks. Duplication of this publication of parts thereof is permitted only under the provision of the copyright law 1965, in its current version, and permission of use must always be obtained from CSC Publishers.

IJCSS Journal is a part of CSC Publishers

<http://www.cscjournals.org>

© IJCSS Journal

Published in Malaysia

Typesetting: Camera-ready by author, data conversion by CSC Publishing Services – CSC Journals, Malaysia

CSC Publishers, 2011

EDITORIAL PREFACE

This is second issue of volume five of the International Journal of Computer Science and Security (IJCSS). IJCSS is an International refereed journal for publication of current research in computer science and computer security technologies. IJCSS publishes research papers dealing primarily with the technological aspects of computer science in general and computer security in particular. Publications of IJCSS are beneficial for researchers, academics, scholars, advanced students, practitioners, and those seeking an update on current experience, state of the art research theories and future prospects in relation to computer science in general but specific to computer security studies. Some important topics cover by IJCSS are databases, electronic commerce, multimedia, bioinformatics, signal processing, image processing, access control, computer security, cryptography, communications and data security, etc.

The initial efforts helped to shape the editorial policy and to sharpen the focus of the journal. Starting with volume 5, 2011, IJCSS appears in more focused issues. Besides normal publications, IJCSS intend to organized special issues on more focused topics. Each special issue will have a designated editor (editors) – either member of the editorial board or another recognized specialist in the respective field.

This journal publishes new dissertations and state of the art research to target its readership that not only includes researchers, industrialists and scientist but also advanced students and practitioners. The aim of IJCSS is to publish research which is not only technically proficient, but contains innovation or information for our international readers. In order to position IJCSS as one of the top International journal in computer science and security, a group of highly valuable and senior International scholars are serving its Editorial Board who ensures that each issue must publish qualitative research articles from International research communities relevant to Computer science and security fields.

IJCSS editors understand that how much it is important for authors and researchers to have their work published with a minimum delay after submission of their papers. They also strongly believe that the direct communication between the editors and authors are important for the welfare, quality and wellbeing of the Journal and its readers. Therefore, all activities from paper submission to paper publication are controlled through electronic systems that include electronic submission, editorial panel and review system that ensures rapid decision with least delays in the publication processes.

To build its international reputation, we are disseminating the publication information through Google Books, Google Scholar, Directory of Open Access Journals (DOAJ), Open J Gate, ScientificCommons, Docstoc and many more. Our International Editors are working on establishing ISI listing and a good impact factor for IJCSS. We would like to remind you that the success of our journal depends directly on the number of quality articles submitted for review. Accordingly, we would like to request your participation by submitting quality manuscripts for review and encouraging your colleagues to submit quality manuscripts for review. One of the great benefits we can provide to our prospective authors is the mentoring nature of our review process. IJCSS provides authors with high quality, helpful reviews that are shaped to assist authors in improving their manuscripts.

Editorial Board Members

International Journal of Computer Science and Security (IJCSS)

EDITORIAL BOARD

ASSOCIATE EDITORS (AEiCs)

Associate Professor. Azween Bin Abdullah

Universiti Teknologi Petronas,
Malaysia

Dr. Padmaraj M. V. nair

Fujitsu's Network Communication division in Richardson
Texas, USA

Dr. Blessing Foluso Adeoye

University of Lagos,
Nigeria

EDITORIAL BOARD MEMBERS (EBMs)

Professor. Abdel-Badeeh M. Salem

Ain Shams University
Egyptian

Professor. Sellappan Palaniappan

Malaysia University of Science and Technology
Malaysia

Professor Mostafa Abd-El-Barr

Kuwait University
Kuwait

Professor. Arun Sharma

Amity University
India

Dr. Alfonso Rodriguez

University of Bio-Bio
Chile

Dr. Srinivasan Alavandhar

Glasgow Caledonian University
UK

Dr. Debotosh Bhattacharjee

Jadavpur University
India

Dr. Teng li Lynn

University of Hong Kong
Hong Kong

Dr. Chiranjeev Kumar
Indian School of Mines University
India

Dr. Ghossoon M. Waleed
University Malaysia Perlis
Malaysia

Dr. Srinivasan Alavandhar
Caledonian University
Oman

Dr. Deepak Laxmi Narasimha
University of Malaya
Malaysia

Professor Mostafa Abd-El-Barr
Kuwait University
Kuwait

TABLE OF CONTENTS

Volume 5, Issue 2, May 2011

Pages

- 168 - 180 A Novel Approach for Efficient Resource Utilization and Trustworthy Web Service
Marimuthu K, Ganesh Gopal D
- 181 - 192 Three Dimensional Database: Artificial Intelligence to eCommerce Web Service Agents
R. Vadivel, K. Baskaran
- 193 - 200 A New Function-based Framework for Classification and Evaluation of Mutual Exclusion
Algorithms in Distributed Systems
Leila Omrani , Zahra Rafinezhad, Mohammadreza Kayvanpour
- 201 - 207 J48 and JRIP Rules for E-Governance Data
*Anil Rajput, Ramesh Prasad Aharwal, Meghna Dubey, S.P. Saxena, Manmohan
Raghuvanshi*
- 208 – 220 Concurrent Matrix Multiplication on Multi-core Processors
Muhammad Ali Ismail, S. H. Mirza, Talat Altaf
- 221-226 Radical Data Compression Algorithm Using Factorization
Peter Zirra, Gregory Wajiga
- 227-243 Cryptographic Algorithms For Secure Data Communication
Peter Zirra, Gregory Wajiga
- 244-276 A Study of Protocols for Grid Computing Environment
Suresh Jaganathan, A Srinivasan, A Damodaram
- 277-286 Phishing: A Field Experiment
Danuvasin Charoen

287-297

Design and Implementation of a Multi-Agent System for the Job Shop Scheduling Problem

Leila Asadzadeh, Kamran Zamanifar

A Novel Approach for Efficient Resource Utilization and Trustworthy Web Service

Marimuthu K

Assistant Professor/School of Computing
Science and Engineering
VIT University
Vellore, 632014, India

k.marimuthu@vit.ac.in

Ganesh Gopal D

Assistant Professor/School of Computing
Science and Engineering
VIT University
Vellore, 632014, India

ganeshgopal@vit.ac.in

Abstract

Many Web services are expected to run with high degree of security and dependability. To achieve this goal, it is essential to use a Web-services compatible framework that tolerates not only crash faults, but Byzantine faults as well, due to the untrusted communication environment in which the Web services operate. In this paper, we describe the design and implementation of such a framework, called RET-WS (Resource Efficient and Trustworthy Execution -Web Service). RET-WS is designed to operate on top of the standard SOAP messaging framework for maximum interoperability with resource efficient way to execute requests in Byzantine-fault-tolerant replication that is particularly well suited for services in which request processing is resource-intensive. Previous efforts took a failure masking all-active approach of using all execution replicas to execute all requests; at least $2t + 1$ execution replicas are needed to mask t Byzantine-faulty ones. We describe an asynchronous protocol that provides resource-efficient execution by combining failure masking with imperfect failure detection and checkpointing. It is implemented as a pluggable module within the Axis2 architecture, as such, it requires minimum changes to the Web applications. The core fault tolerance mechanisms used in RET-WS are based on the well-known Castro and Liskov's BFT algorithm for optimal efficiency with some modification for resource efficient way. Our performance measurements confirm that RET-WS incurs only moderate runtime overhead considering the complexity of the mechanisms.

Keywords: Distributed Systems, Fault Tolerance, Byzantine Faults, Resource Efficient.

1. INTRODUCTION

Driven by business needs and the availability of the latest Web services technology, we have seen increasing reliance on services provided over the Web. We anticipate a strong demand for robust and practical fault tolerance middleware for such Web services. Considering the untrusted communication environment in which these services operate, arbitrary faults (crash faults as well as Byzantine faults [13]) must be tolerated to ensure maximum service dependability. Middleware that provides such type of fault tolerance with resource efficient way is often termed as Resource Efficient and Trustworthy execution Protocol [12] (RET) middleware.

There exist a well-known high quality research prototype [6] that provides Byzantine fault tolerance for generic client-server applications (similar prototypes are available, but they are often tied to a specific application, such as storage [19]). In fact, Merideth *et al.* [15] have used it directly for Web services fault tolerance. However, argue against such an approach primarily for two reasons. First and foremost, the prototype uses proprietary messaging protocols (directly on top of IP multicast by default). This is incompatible with the design principles of Web services,

which call for transport independence and mandate SOAP-based communications. The use of proprietary messaging protocols compromises the interoperability of Web services. Second, this prototype lacks direct support for Web services, which requires the use of a wrapper to mediate the two components. The mediation can be achieved either through an additional socket communication, which wastes precious system resources and is inefficient, or through a Java Native Interface (the vast majority of Web services are implemented in Java, and the BFT prototype is implemented in C++), which is difficult to program and error-prone.

We believe that any type of middleware for Web services must use standard Web services technologies and must follow the design principles of Web services, and fault tolerance middleware for Web services is no exception. With this guideline in mind, we designed and implemented RET-WS, a Byzantine fault tolerance framework for Web services. To avoid reinventing the wheel and to best utilize existing Web services technology, we decide to build RET-WS by extending Sandesha2 [3], which is an implementation of the Web Service Reliable Messaging (WS-RM) standard [4] for Apache Axis2 [2] in Java. In RET-WS, all fault tolerance mechanisms operate on top of the standard SOAP messaging framework for maximum interoperability. RET-WS inherits Sandesha2's pluggability, and hence, it requires minimum changes to the Web applications (both the client and the service sides). The core fault tolerance mechanisms in RET-WS are based on the well-known Castro and Liskov's BFT algorithm [6] for optimal runtime efficiency. The performance evaluation of a working prototype of RET-WS shows that it indeed introduces only moderate runtime overhead verses the original Sandesha2 framework considering the complexity of the Byzantine fault tolerance mechanisms.

2. LITERATURE SURVEY

2.1. Byzantine Fault Tolerance

A Byzantine fault is an arbitrary fault that occurs during the execution of an algorithm by a distributed system. It encompasses both omission failures (e.g., crash failures, failing to receive a request, or failing to send a response) and commission failures (e.g., processing a request incorrectly, corrupting local state, and/or sending an incorrect or inconsistent response to a request.) The early solutions were described by Lamport, Shostak, and Pease in 1982. One solution considers scenarios in which messages may be forged, but which will be *Byzantine-fault-tolerant* as long as the number of traitorous generals does not equal or exceed one third.

A second solution requires unforgeable signatures (in modern computer systems, this may be achieved in practice using public key cryptography but maintains Byzantine fault tolerance in the presence of an arbitrary number of traitorous generals.

The Query/Update (Q/U) protocol is a new tool that enables construction of fault-scalable Byzantine fault-tolerant services. The optimistic quorum-based nature of the Q/U protocol allows it to provide better throughput and fault-scalability than replicated state machines using agreement-based protocols. Moreover, the performance of the Q/U protocol decreases by only 36% as the number of Byzantine faults tolerated increases from one to five, whereas the performance of the replicated state machine decreases by 83%[20].

The Q/U protocol [21], describing the weakness of approaches and show how to adapt Byzantine quorum protocols, which had previously been mostly limited to a restricted read/write interface [22], to implement Byzantine-fault-tolerant state machine replication. This is achieved through a client-directed process that requires one round of communication between the client and the replicas when there is no contention and no failures.

A Byzantine faulty process may behave arbitrarily, in particular, it may disseminate different information to other processes, which constitutes a serious threat to the integrity of a system. Since a Byzantine faulty process may also choose not to respond to requests, it can exhibit crash fault behavior as well (*i.e.*, crash faults can be considered as a special case of Byzantine faults). Byzantine fault tolerance (BFT) refers to the capability of a system to tolerate Byzantine faults.

For a client-server system, RET can be achieved by replicating the server and by ensuring all server replicas to execute the same request in the same order. The latter means that the server replicas must reach an agreement on the set of requests and their relative ordering despite Byzantine faulty replicas and clients. Such an agreement is often referred to as Byzantine agreement [13].

Byzantine agreement algorithms had been too expensive to be practical until Castro and Liskov invented the BFT algorithm mentioned earlier [6]. The BFT algorithm is designed to support client-server applications running in an asynchronous distributed environment with a Byzantine fault model. The implementation of the algorithm contains two parts. At the client side, a lightweight library is responsible to send the client's request to the primary replica, to retransmit the request to all server replicas on the expiration of a retransmission timer (to deal with the primary faults), and to collect and vote on the replies. The main RET algorithm is executed at the server side by a set of $3f+1$ replicas to tolerate f Byzantine faulty replicas. One of the replicas is designated as the primary while the rest are backups.

As shown in FIGURE 1, the normal operation of the (server side) RET algorithm involves three phases. During the first phase (called pre-prepare phase), the primary multicasts a pre-prepare message containing the client's request, the current view and a sequence number assigned to the request to all backups.

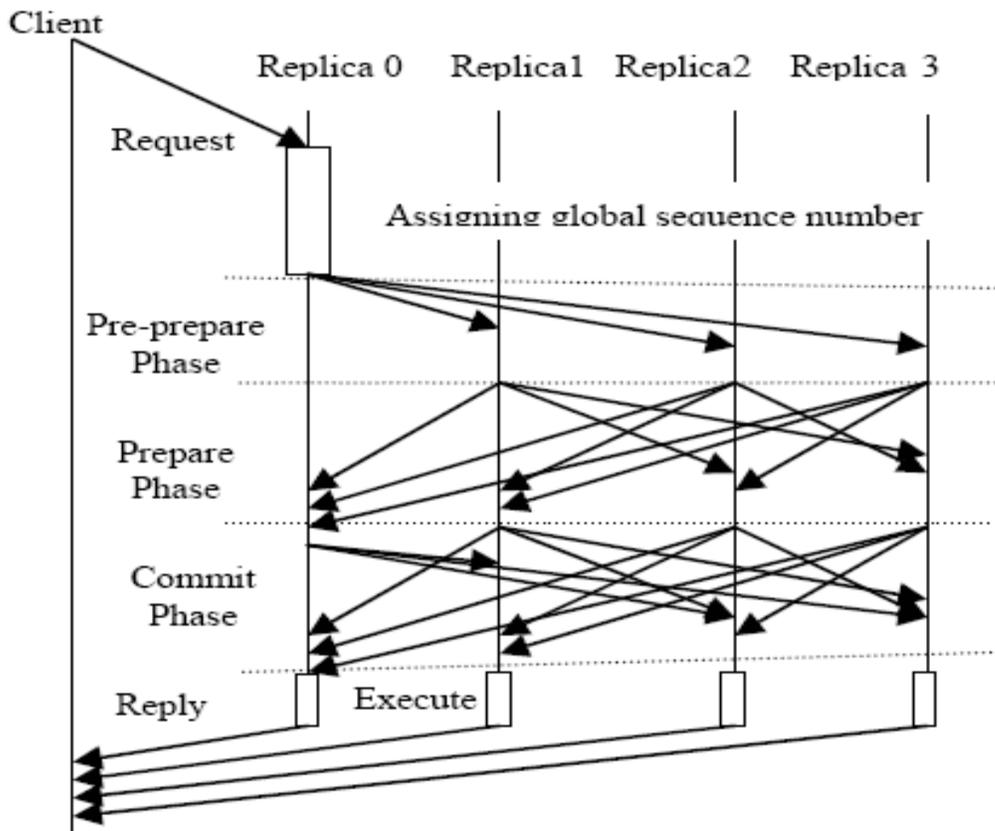


FIGURE 1: Normal operation of the RET algorithm

A backup verifies the request message the ordering information. If the backup accepts the message, it multicasts to all other replicas a prepare message containing the ordering information and the digest of the request being ordered. This starts the second phase, *i.e.*, the prepare phase. A replica waits until it has collected $2f$ prepare messages from different replicas (including

the message it has sent if it is a backup) that match the prepare message before it multicasts a commit message to other replicas, which starts the commit phase. The commit phase ends when a replica has received $2f$ matching commit messages from other replicas. At this point, the request message has been totally ordered and it is ready to be delivered to the server application if all previous requests have already been delivered. If the primary or the client is faulty, a Byzantine agreement on the ordering of a request might not be reached, in which case, a new view is initiated, triggered by a timeout on the current view. A different primary is designated in a round-robin fashion for each new view installed.

2.2. Web Services Reliable Messaging

The Web Services Reliable Messaging (WS-RM) standard describes a reliable messaging (RM) protocol between two endpoints, termed as RM source (RMS) and RM destination (RMD). The core concept introduced in WSRM is *sequence*. A sequence is a unidirectional reliable channel between the RMS and the RMD. At the beginning of a reliable conversation between the two endpoints, a unique sequence (identified by a unique sequence ID) must first be created (through the create-sequence request and response). The sequence is terminated when the conversation is over (through the terminate-sequence request and response). For each message sent over the sequence, a unique message number must be assigned to it. The message number starts at 1 and is incremented by 1 for each subsequent message. The reliability of the messaging is achieved by the retransmission and positive acknowledgement mechanisms. At the RMS, a message sent is buffered and retransmitted until the corresponding acknowledgement from the RMD is received, or until a pre defined retransmission limit has been exceeded. For efficiency reason, the RMD might not send acknowledgement immediately upon receiving an application message, and the acknowledgements for multiple messages can be piggybacked with another application message in the response sequence, or be aggregated in a single explicit acknowledgement message.

Because it is quite common for two endpoints to engage in two-way communications, the RMS can include an Offer element in its create-sequence request to avoid an explicit new sequence establishment sRET for the traffic in the reverse direction. Most interestingly, WS-RM defines a set of delivery assurances, including AtMostOnce, AtLeastOnce, Exactly-Once, and InOrder. The meanings of these assurances are self-explanatory. The InOrder assurance can be used together with any of the first three assurances. The strongest assurance is ExactlyOnce combined with InOrder delivery.

The WS-RM standard has been widely supported and there exist many implementations, most of which are commercial. We choose to use Sandesha2 [3] for this research, due to its open-source nature and its support for Axis2, the second generation open-source SOAP engine that supports pluggable modules.

3. SYSTEM ARCHITECTURE

The overview of the RET-WS architecture is shown in FIGURE 2. RET-WS is implemented as an Axis2 module. During the out-flow of a SOAP message, Axis2 invokes the RET-WS Out Handler during the user phase, and invokes the Rampart (an Axis2 module that provides WS-Security [17] features) handler for message signing during the security phase. Then, the message is passed to the HTTP transport sender to send to the target endpoint. During the in-flow of a SOAP message, Axis2 first invokes the default handler for preliminary processing (to find the target object for the message based on the URI and SOAP action specified in the message) during the transport phase, it then invokes the Rampart handler for signature verification during the security phase. This is followed by the invocation of the RET-WS Global In Handler during the dispatch phase. This handler performs tasks that should be done prior to dispatching, such as duplicate suppression at the server side. If the message is targeted toward a RETWS-enabled service, the RET-WS In Handler is invoked for further processing during the user-defined phase, otherwise, the message is directly dispatched to the Axis2 message receiver. For clarity, FIGURE 2 shows only a one-way flow of a request from the client to the replicated Web service. The response flow

is similar. Also not shown in FIGURE 2 are the multicast process and the internal components of the RET-WS module.

Note that for the Rampart module to work (required by the RET algorithm to authenticate the sender, so that a faulty replica cannot impersonate another correct replica), each replica has a pair of public and private RSA keys. Similarly, each client must also possess a public and private key pair. We assume that the public keys are known to all replicas and the clients, and the private keys of the correct replicas and clients are kept secret. We further assume the adversaries have limited computing power so that they cannot break the digital signatures of the messages sent by correct replicas or clients.

The main components of the RET-WS module are illustrated in FIGURE 3. The client side bears a lot of similarity to the Sandesha2 client side module, with the exception of the addition of RET-WS Voter, the replacement of Sandesha Sender by a Multicast Sender, and the replacement of the Sandesha Out Handler by the RET-WS Out Handler. The server side contains more additions and modifications to the Sandesha2 components. Furthermore, a set of actions are added to the module configuration to allow total-ordering of messages, view change management and replica state synchronization. Besides the Multicast Sender, the server side introduced a Total Order Manager, and replaced the original Global In Handler, In Handler, and In-Order handler, by RET-WS Global In Handler, RET-WS In Handler and Total Order Invoker, respectively. The storage framework in Sandesha2 is not changed. The functions of these components (both Sandesha2 original and the modified or new components) are elaborated in the following subsections, starting with the components dealing with the out-flow, and then the components for the in-flow.

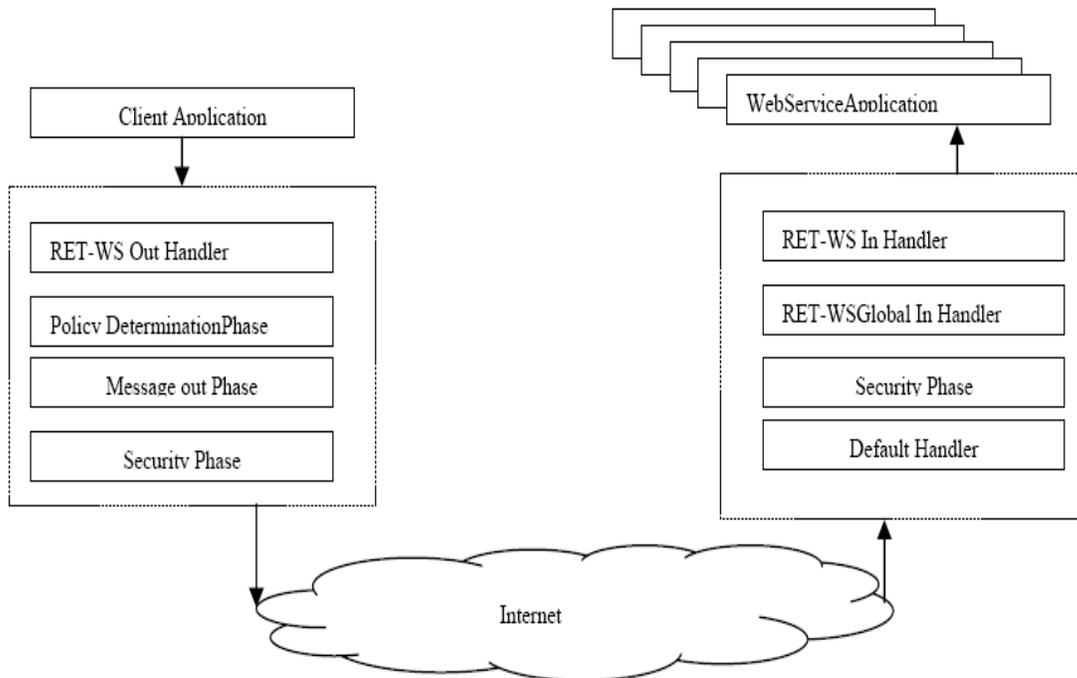


FIGURE 2: The Overview of the RET-WS architecture

Note that even though what described in this section are specific to Axis2, we believe that our Byzantine fault the tolerance mechanisms are generic enough to be ported to other Web services infrastructure without great barrier.

3.1. RET-WS Out Handler

This handler performs out-flow processing for reliable messaging. In particular, it generates a create-sequence request when the application sends the first message of a new sequence, and sends a terminate-sequence request after the last message of a sequence is transmitted. The difference between the RET-WS Out Handler and the original Sandesha Out Handler lies in the creation and handling of the create-sequence message. In the original implementation, the create-sequence message does not contain any element that can be used for the server side to perform duplicate detection. If the create-sequence request contains an Offer element, it may be used as a way to check for duplicate. However, not all create-sequence requests contain such an element, because its existence is specified by the client application. To address this problem, we propose to include a UUID string in the create-sequence request. The UUID is embedded in the Create Sequence/any element, an optional element specified by the WS-RM standard to enable extensibility.

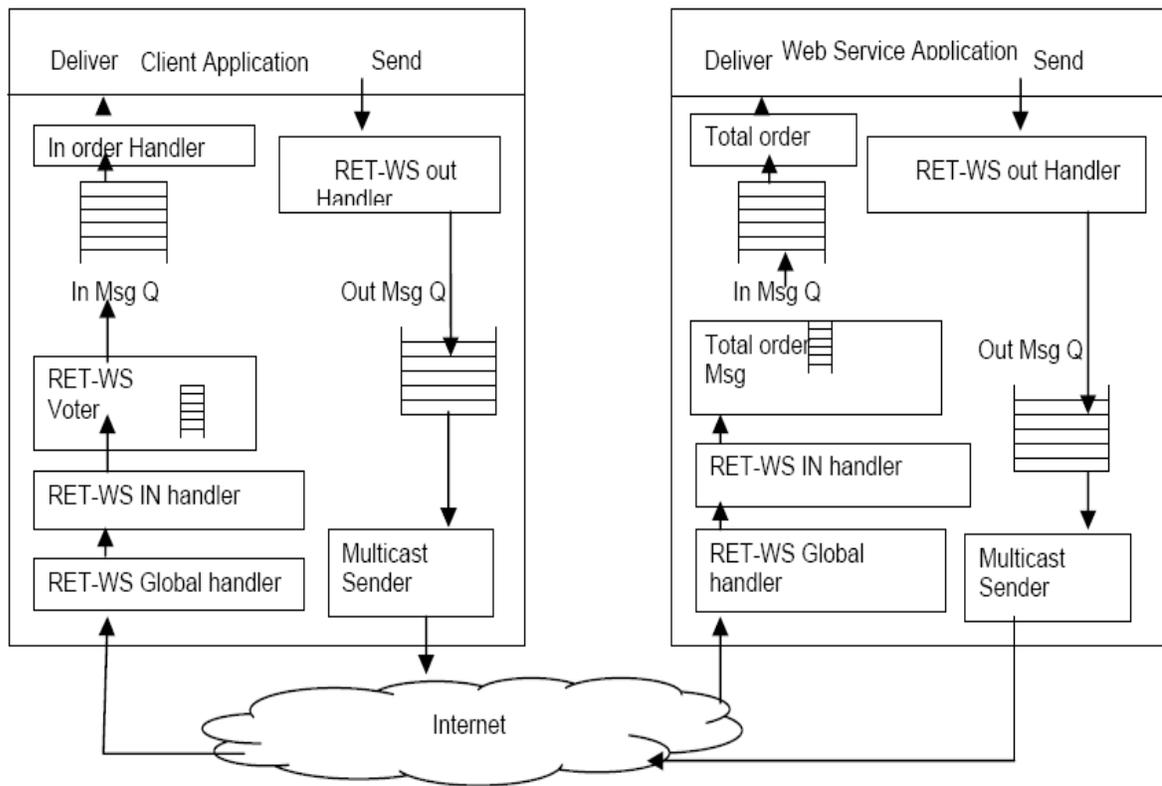


FIGURE 3: The main components of the RET-WS modules

The addition of this UUID element also helps alleviate a tricky problem that would cause replica inconsistency. The WS-RM standard does not specify how the sequence ID for the newly created sequence should be determined. In Sandesha2, a UUID string is generated and used as the sequence ID at the server side. If we allow each replica to generate the sequence ID unilaterally in this fashion, the client would adopt the sequence ID present in the first create-sequence response it receives. This would prevent the client from communicating with other replicas, and would prevent the replicas from referring to the same sequence consistently when ordering the application messages sent over this sequence. Therefore, we modified the create-sequence request handling code to generate the sequence ID deterministically based on the client supplied UUID and the Web service group endpoint information.

3.2. Multicast Sender

In RET-WS, the sequence between the client and the service provider endpoints is mapped transparently to a virtual sequence between the client and the group of replicas. The same sequence ID is used for the virtual sequence so that other components can keep referring to this sequence regardless if it is a one-to-one or a one-to-many (or many-to-one) sequence. The mapping is carried out by the multicast sender.

To make the mapping possible, we assume that each service to be replicated bears a unique group endpoint, in addition to the specific endpoint for each replica. Higher level components, including the application, must use the group endpoint when referring to the replicated Web service. When a message to the group endpoint is detected, including application messages and RET-WS control messages, the multicast sender translates the group endpoint to a list of individual endpoints and multicasts the message to these endpoints. We assume the mapping information is provided by a configuration file. The Multicast Sender runs as a separate thread and periodically poll the Out Message Queue for messages to send.

One additional change is the garbage collection mechanism. For point-to-point reliable communication, it is sufficient to discard a buffered message as soon as an acknowledgement for the message is received. However, this mechanism does not work for reliable multicast for apparent reasons. Consequently, a message to be multicast is kept in the buffer until the acknowledgements from all destinations have been collected, or a predefined retransmission limit has been exceeded.

Note that in RET-WS, the client multicasts its requests to all replicas via the Multicast Sender component. Even though it may be less efficient in some scenarios, such as when the client is geographically farther away from the Web service and the Web service replicas are close to each other, this design is more robust against adversary attacks since the clients do not need to know which replica is currently serving as the primary. Without such information, the adversary can only randomly pick up a replica to attack, instead of focusing on the primary directly. From the availability perspective, the compromise of the primary can result in much severe performance degradation than that of a backup. It is important to encapsulate internal state information as much as possible to improve system robustness. Information encapsulation also reduces the dependency between the clients and the Web services.

3.3. RET-WS Global In Handler

The Sandesha Global In Handler performs duplicate filtering on application messages. This is fine for the server side, however, it would prevent the client from performing voting on the responses. Therefore, the related code is modified so that no duplicate detection is done on the client side. The other functionalities of this handler, *e.g.*, generating acknowledgement for the dropped messages, is not changed.

3.4. RET-WS In Handler

Axis2 dispatches all application messages targeted to the RET-WS-enabled services and the RET-WS control messages to this handler. The RET-WS In Handler operates differently for the client and the server sides.

At the client side, all application messages are passed immediately to the RET-WS Voter component for processing. The rest of control messages are processed by the set of internal message processors as usual.

At the server side, all application messages are handled by an internal application message processor. Such messages are stored in the In Message Queue for ordering and delivery. All RET-related control messages, such as pre-prepare, prepare, commit, and view change messages, are passed to the Total Order Manager for further processing. The WS-RM-related control messages such as create-sequence and terminate-sequence requests, are handled by

the internal message processors available from the original Sandesha2 module, with the exception of the handling of sequence ID creation.

3.5. RET-WS Voter

This component only exists at the client side. The Voter verifies the authenticity of the application messages received and temporarily stores the verified messages in its data structure. For each request issued, the Voter waits until it has collected $f + 1$ identical response messages from different replicas before it invokes the application message handler to process the response message. When the processing is finished, the message is passed to the In Message Queue for delivery.

3.6. Storage Manager

This component consists of the In Message Queue, the Out Message Queue, and a number of other subcomponents for sequence management, acknowledgement and retransmission management, and in-order delivery. This component comes with the Sandesha2 module. It is instrumented only for the purpose of performance profiling.

3.7. Total Order Invoker

This component replaces the Sandesha InOrder Invoker. This invoker runs as a separate thread to poll periodically the received application messages (stored in the In Message Queue) for ordering and delivery. To be eligible for ordering, the message must be in-order within its sequence, *i.e.*, all previous messages in the sequence has been received and ordered (or being ordered). If the message is eligible for ordering, the Total Order Manager is notified to order the message. Note that only the primary initiates the ordering of application messages.

The Total Order Invoker asks the Total Order Manager for the next message to be delivered. If there is a message ready for delivery, the Invoker retrieves the message from the In Message Queue and delivers it to the Web service application logic via the Axis2 message receiver.

3.8. Total Order Manager

This component is responsible for imposing a total order on all application requests according to the RET algorithm. To facilitate reliable communication among the replicas themselves, each replica establishes a sequence with the rest of the replicas. The reliability of the control messages sent over these sequences are guaranteed by the WS-RM mechanisms and the Multicast Sender. For clarity, we first describe the RET algorithm assuming that a unique global sequence number is assigned for each application message, then we elaborate on the batching mechanism which is needed to ensure optimal runtime performance. Due to space limitation, the view change and state transfer algorithms are omitted.

For each application request to be ordered, the Total Order Manager at the primary assigns the next available global sequence number to the message and constructs a preprepare message. The pre-prepare message contains the following information: The global sequence number n , the current view number v , and the digest d of the application message m to be ordered. The pre-prepare message is then passed to the Out Message Queue for sending.

The Total Order Manager uses a TotalOrderBean object to keep track of the ordering status for each application message. When a pre-prepare message is created at the primary, the Manager stores the message in the corresponding TotalOrderBean (a new TotalOrderBean is created on the creation of the first pre-prepare message for each application message).

At the backup, the Total Order Manager accepts a preprepare message if the message is signed properly, and it has not accepted a pre-prepare message for the same global sequence number n in view v . If the backup accepts the preprepare message, it creates a TotalOrderBean for the message and stores the pre-prepare message in the TotalOrderBean. The backup also constructs a prepare message containing the following information: The global sequence number n , the current view number v , the digest d of the application message m . The prepare message is

dropped to the Out Message Queue for sending and the TotalOrderBean is updated correspondingly.

When a replica receives a prepare message, it verifies n and v , and compares the digest d with that of the application message. It accepts the prepare message if the check is passed and updates the TotalOrderBean. When a replica has collected $2f$ prepare messages from other replicas, including the pre-prepare message received from the primary (if the replica is a backup), it constructs a commit message with the same information as that of the prepare message, and passes the commit message to the Out Message Queue for sending. Again, the TotalOrderBean is updated for the sending of the commit message.

A replica verifies a commit message in the similar fashion to that for a prepare message. When a replica has collected $2f$ correct commit messages for n in view v , the message m is committed to the sequence number n in view v , *i.e.*, a total order for m has now been established. A totally ordered message can be delivered if all previously ordered messages have been delivered. We now describe the batching mechanism. The primary does not immediately order an application request when the message is in-order within its sequence if there are already k batches of messages being ordered, where k is a tunable parameter and it is often set to 1. When the primary is ready to order a new batch of messages, it assigns the next global sequence number for a group of application requests, at most one per sequence, and the requests ordered must be in-order within their own sequences.

4. IMPLEMENTATION

Our performance evaluation is carried out on a test bed consisting of Windows servers connected by a 100Mbps Ethernet. Each server is equipped with a single Pentium V 2.8GHz processors and 1GB memory.

We focus on reporting the runtime overhead of our RET-WS framework during normal operation. A backup failure virtually does not affect the operation of the RET-algorithm, and hence, we see no noticeable degradation of runtime performance. However, when the primary fails, the client would see a significant delay if it has a request pending to be ordered or delivered, due to the timeout value set for view changes. The timeout is usually set to 2 seconds in our experiment which is in a LAN environment. In the Internet environment, the timeout would be set to a higher number. If there are consecutive primary failures, the delay would be even longer.

An echo test application is used to characterize the runtime overhead. The client sends a request to the replicated Web service and waits for the corresponding reply within a loop without any “think” time between two consecutive calls. The request message contains an XML document with varying number of elements, encoded using AXIOM (Axis Object Model) [1]. At the replicated Web service, the request is parsed and a nearly identical reply XML document is returned to the client.

In each run, 1000 samples are obtained. The end-to-end latency for the echo operation is measured at the client. The throughput is measured at the replicated Web service. In our experiment, we keep the number of replicas to 4 (to tolerate a single Byzantine faulty replica), and vary the request sizes in terms of the number of elements in each request, and the number of concurrent clients.

In FIGURE 4, the end-to-end latency of the echo operation is reported for RET- replication with 4 replicas and a single client. For comparison, the latencies for two other configurations are also included.

The first configuration involves no replication and no digital signing of messages. The second configuration involves no replication, but with all messages digitally signed. The measurements for the two configurations reveal the cost of digital signing and verification. As can be seen, such

cost ranges from 90ms for short messages to 130ms for longer messages. The latency overhead of running RET- replication is significant.

For comparison, the latency for the no-replication configuration with and without digital signing of messages are included as well. However, the overhead is very reasonable considering the complexity of the RET- algorithm. Comparing with the latency for the no replication-with-signing configuration, the overhead ranges from 150ms for short messages to over 310ms for longer messages.

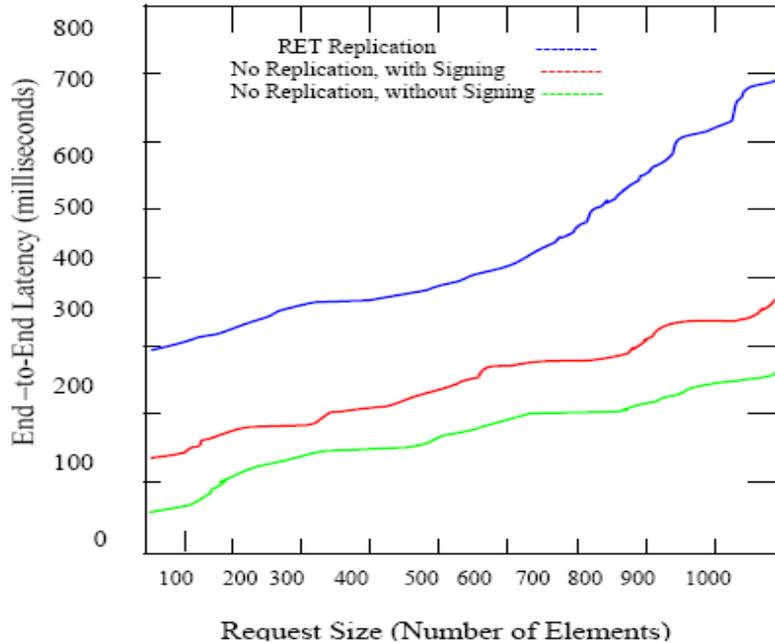


FIGURE 4: The end-to-end latency

The increased overhead for larger messages is likely due to the CPU contention for processing of the application requests (by the Web service) and the RET- replication mechanisms (by our framework). Future work is needed to fully characterize the sources of the additional cost for longer messages.

The latency cost for each step of processing in a request-reply round trip for a particular run with a single client and 4 replicas is summarized in TABLE 1. Both the request and the reply contain 1000 elements. As can be seen, the major costs come from message ordering, request multicasting, and message signing and verification.

5. SUMMARY OF RESULTS

A large number of high availability solutions for Web services have been proposed in the last several years [5, 7- 11, 14-16, 18]. Most of them are designed to cope with crash faults only. Furthermore, none of them has taken our approach, which integrates the replication mechanisms into the SOAP engine for maximum interoperability. Thema [15] is the only complete BFT-framework for Web services which we know. In [18], an alternative solution is proposed for BFT-Web services, but no implementation details or performance evaluations are reported.

5.1 Comparative Evaluation

In the paper [15] they used automatic outcome with BAwCC protocol. All the messages are protected with the timestamp. The end to end latency is measured at initiator only and the throughput is measured at the coordinator. Moreover the test case uses only one replica at the

initiator and four at the coordinator. The latency overhead is less than 20%. Throughput reduction is less than 20% when replication is enabled.

In our paper we used the SOAP for better operability. For the comparison we have taken the messages with digital signature and without also for the better results. We use four replicas for the better overhead and able to achieve that. The results show that both the latency overhead and throughput has been improved comparatively.

Similar to our work, Thema [15] also relies on the BFT- algorithm to ensure total ordering of application messages. However, a wrapper is used to interface with an existing implementation of the RET- algorithm [6], which is based on IP multicast, rather than the standard SOAP/HTTP transport, as such, it suffers from the interoperability problem we mentioned in the beginning of this paper. This approach limits its practicality. That said, it does provide richer functionality than our current RET-WS framework in that it supports multi-tiered applications and the interaction between a replicated Web service as client and another non-replicated Web service as server with resource efficient way. We plan to add similar functionality to RET-WS in the next stage of our project.

In [18], attempts to address some problems in Thema when a replicated client interacts with a replicated Web service which may have been compromised. It proposes to use the RET- algorithm for the client replicas to reach consensus on the reply messages to avoid the situation which different client replicas accept different reply messages for the same request made, when the server is compromised. However, it is not clear to us the value of such an approach. If the Web service has been compromised, the integrity of the service is no longer guaranteed, the service could easily send the *same* wrong reply to all client replicas, which cannot be addressed by the mechanism proposed in [18], and yet, the end-to-end latency is doubled as a result.

Processing SRET	Latency (ms)
Request out-processing	9.9
Request multicast	127.7
Request in-processing	10.8
Request ordering and delivery	307.0
Request processing at application	43.8
Reply out-processing	5.6
Reply send	78.7
Reply voting	11.9
Reply in-processing	8.9
Reply delivery	16.5
Message signing & verification (derived)	130.2
Total	751

TABLE 1: Detailed latency measurement for a particular run with a single client and 4 replicas.

6. CONCLUSION

In this paper, we presented the design and implementation of RET-WS, a Byzantine fault tolerance middleware framework for Web services with resource efficient way. It uses standard Web services technology to build the Byzantine fault tolerance service, and hence, it is more suitable to achieve interoperability. We also documented in detail the architecture and the major components of our framework. We anticipate that such descriptions are useful to practitioners as well as researchers working in the field of highly dependable Web services. Finally, our framework has been carefully tuned to exhibit optimal performance, as shown in our performance evaluation results. Future work will focus on the expansion of the feature set of RET-WS, such as the support of multi-tiered Web services and transactional Web services.

7. REFERENCES

- [1] Apache Axiom. <http://ws.apache.org/commons/axiom/>.
- [2] Apache Axis2. <http://ws.apache.org/axis2/index.html>.
- [3] Apache Sandesha2. <http://ws.apache.org/sandesha/sandesha2/index.html>.
- [4] R. Bilorusets et al. “*Web Services Reliable Messaging Specification*”, February 2005.
- [5] K. Birman. “*Adding high availability and autonomic behavior to web services*”. In Proceedings of the International Conference on Software Engineering, Scotland, UK, May 2004.
- [6] M. Castro and B. Liskov. “*Practical Byzantine fault tolerance*”. In Proceedings of the Third Symposium on Operating Systems Design and Implementation, New Orleans, USA, February 1999.
- [7] P. Chan, M. Lyu, and M. Malek. “Making services fault tolerant”. *Lecture Notes in Computer Science*, 4328:43–61, 2006.
- [8] V. Dialani, S. Miles, L. Moreau, D. D. Roure, and M. Luck. “*Transparent fault tolerance for web services based architecture*”. *Lecture Notes in Computer Science*, 2400:889–898, 2002.
- [9] G. Dobson. “*Using WS-BPEL to implement software fault tolerance for Web services*”. In Proceedings of the 32nd EUROMICRO Conference on Software Engineering and Advanced Applications, July 2006.
- [10] A. Erradi and P. Maheshwari. “*A broker-based approach for improving Web services reliability*”. In Proceedings of the IEEE International Conference on Web Services, Orlando, FL, July 2005.
- [11] C. Fang, D. Liang, F. Lin, and C. Lin. “*Fault tolerant web services*”. *Journal of Systems Architecture*, 53:21–38, 2007.
- [12] H.V. Ramasamy, C. Cachin, A. Agbaria, and W.H. Sanders, “*A Parsimonious Approach for Obtaining Resource efficiency and trustworthy Execution*”, *IEEE Transactions on Dependable and Secure Computing*, vol. 4, no. 1, pp.1-17 january-march 2007.
- [13] L. Lamport, R. Shostak, and M. Pease. “*The Byzantine generals problem*”. *ACM transactions on Programming Languages and Systems*, 4(3):382–401, July 1982.
- [14] N. Looker, M. Munro, and J. Xu. “*Increasing web service dependability through consensus voting*”. In Proceedings of the 29th Annual International Computer Software and Applications

Conference, pages 66–69, 2005.

- [15] M. Merideth, A. Iyengar, T. Mikalsen, S. Tai, I. Rouvellou, and P. Narasimhan. “*Thema: Byzantine-fault-tolerant middleware for web services applications*”. In Proceedings of the IEEE Symposium on Reliable Distributed Systems, pages 131–142, 2005.
- [16] L. Moser, M. Melliar-Smith, and W. Zhao. “*Making web services dependable*”. In Proceedings of the 1st International Conference on Availability, Reliability and Security, pages 440–448, Vienna University of Technology, Austria, April 2006.
- [17] A. Nadalin, C. Kaler, P. Hallam-Baker, and R. Monzillo. “Web Services Security: SOAP Message Security 1.0”. OASIS, oasis standard 200401 edition, March 2004.
- [18] S. L. Pallemulle, I. Wehrman, and K. J. Goldman. “*Byzantine fault tolerant execution of long-running distributed applications*”. In Proceedings of the IASTED Parallel and Distributed Computing and Systems Conference, pages 528–523, November 2006.
- [19] S. Rhea, P. Eaton, D. Geels, H. Weatherspoon, B. Zhao, and J. Kubiatowicz. “*Pond: the OceanStore prototype*”. In Proceedings of the 2nd USENIX Conference on File and Storage Technologies, March 03.
- [20] Ken Birman, Robbert van Renesse, Werner Vogels¹, Dept. of Computer Science, Cornell University. “Adding High Availability and Autonomic Behavior to Web Services” IEEE Proceedings(ICSE04).
- [21] ABD-EL-MALEK, M., GANGER, G. R., GOODSON, G. R., REITER, M. K., AND WYLIE, J. J. Fault-scalable byzantine fault-tolerant services. In SOSP '05: Proceedings of the twentieth ACM symposium on Operating systems principles (New York, NY, USA, 2005), ACM Press, pp. 59–74
- [22] MALKHI, D., AND REITER, M. Byzantine Quorum Systems. Journal of Distributed Computing 11, 4 (1998), 203–213.
- [23] Wenbing Zhao, Honglei Zhang, Wenbing Zhao and Honglei Zhang “Byzantine Fault Tolerant Coordination for Web Services Business Activities” 2008 IEEE International Conference on Services Computing

Three Dimensional Database: Artificial Intelligence to eCommerce Web service Agents

R.Vadivel

*Department of Computer Science
Karpagam University
Coimbatore, 641024, INDIA*

vadivel.rangasamy@gmail.com

Dr K. Baskaran

*Associate Professor
Department of Computer Science
Government College of Technology
Coimbatore, 641 015, INDIA*

baski_101@yahoo.com

Abstract

A main objective of this paper is using artificial intelligence technique to web service agents and increase the efficiency of the agent communications. In recent years, web services have played a major role in computer applications. Web services are essential, as the design model of applications are dedicated to electronic businesses. This model aims to become one of the major formalisms for the design of distributed and cooperative applications in an open environment (the Internet). Current commercial and research-based efforts are reviewed and positioned within these two fields. A web service as a software system designed to support interoperable machine-to-machine interaction over a network. It has an interface described in a machine-process able format (specifically Web Services Description Language WSDL). Other systems interact with the web service in a manner prescribed by its description using SOAP messages, typically conveyed using HTTP with an XML serialization in conjunction with other Web-related standards. Particular attention is given to the application of AI techniques to the important issue of WS composition. Within the range of AI technologies considered, we focus on the work of the Semantic Web and Agent-based communities to provide web services with semantic descriptions and intelligent behavior and reasoning capabilities. Re-composition of web services is also considered and a number of adaptive agent approaches are introduced and implemented in publication domain with three dimensional databases and one of the areas of work is eCommerce.

Keywords: Web Services, Semantic Web, eCommerce, Artificial Intelligence, Publication Domain, Dynamic Web, Three Dimensional Database.

1. INTRODUCTION

Currently, Web services give place to active research and this is due both to industrial and theoretical factors. On one hand, Web services are essential as the design model of applications dedicated to the electronic business. On the other hand, this model aims to become one of the major formalisms for the design of distributed and cooperative applications in an open environment (the Internet). Research in the field of semantic web / web service (WS) and artificial intelligence (AI) communities are coming together to develop solutions that will take us to the next and more mature generation of the web application. The composition of web services to create a value-chain greater than the sum of the parts is a key part of what can be expected. The fulfillment of the vision of the web as an information-providing and world-altering provider of services is not far away. More futuristic is the notion of serendipitous. In both visions the services and outcomes may be the same.

However, the difference between the two visions is that the first can be achieved through static and manual solutions and the second requires dynamic and automated solutions. While helpful

for the first, the addition of semantic content on the web is essential to enable automatic discovery and composition of multiple services. It is natural that earlier work in the field of AI will assist in realization of the (artificially) intelligent web. The work on the Web Services Modeling Framework (WSMF) is an example of AI being applied to this field. WSMF offers the combined use of ontology, goal (problem-type) repositories, web service descriptions and mediators to handle interoperability issues. The agent community, which is primarily AI-based, has also been actively conducting WS related research.

Our own distributed agent-based work and the Agent Factory, originates from our earlier AI research into complex knowledge based systems and generic task based configuration. On the one hand, our work on planning and automated configuration offers a way of composing eCommerce web services. On the other hand, WSs potentially provide us with components needed to achieve an implementation of our design. Through the addition of techniques from the Semantic Web community, the benefits of combining our agent technology with WSs has been mutual.

This paper offers a review of research that overlaps the fields of WS and AI. In the following section we describe web services and the need for semantics to be added. In section B we look at how the Semantic Web communities, within the field of AI, are offering semantics. In section C we present AI-based research to address the discovery of WSs. In section D we consider both commercial and AI based techniques for WS composition. In section E, the notion of re-composition of WS is considered and how adaptive agent technology, including our own, can address this problem. We conclude with future directions for the role of AI in the web services field.

2. RELATED WORKS

Implement artificial intelligence concepts to agent's communication and improve the efficiency of the communication between the agents. We are going implement this concept to publication domain and that domain application has used three dimensional databases architecture has been played major rule on the creation dynamic fields. Database architecture for creating dynamic web controls is a three dimensional structure, where we use three terms static, meta and dynamic. Here Static data is generally creating the tables and fields to the database. Meta data is a bridge between static and dynamic data. Dynamic data is the dynamic resultant tables or views that the user needs. An output of database is XML format and it contains data definition and data values. XSL is a presentation part which transforms XML data to output HTML.

In three dimensional databases has used two types of SQL statements Static and Dynamic. Static SQL is SQL statements in an application that do not change at runtime and, therefore, can be hard-coded into the application. Dynamic SQL is SQL statements that are constructed at runtime; for example, the application may allow users to enter their own queries. Thus, the SQL statements cannot be hard-coded into the application.

2.1 Web Services

Web services are typically application programming interfaces (API) or web APIs that can be accessed over a network, such as the Internet, and executed on a remote system hosting the requested services.

Web services are a new way of connecting business. Web services are platform-neutral and vendor-independent protocols that enable any form of distributed processing to be performed using XML and Web-based technologies.

2.1.1 Just-in-time Integration

The Web Services architecture describes the principles behind the next generation of e-business architectures, presenting a logical evolution from object-oriented systems to systems of services. Web Services systems promote significant decoupling and dynamic binding of components: All components in a system are services, in that they encapsulate behavior and publish a messaging

API to other collaborating components on the network. Services are marshaled by applications using service discovery for dynamic binding of collaborations. Web Services reflect a new service-oriented architectural approach, based on the notion of building applications by discovering and orchestrating network-available services, or just-in-time integration of applications.

2.2 Semantic Description of Web Services

WSDL, SOAP and UDDI are seen as steps in the right direction but ones that will fail to achieve the goals of improved automation and interoperability, because they rely on a priori standardizations and require humans in the loop. To support automated reasoning, knowledge representations (such as markup languages) will be needed that express both data and rules for reasoning. The ability to dynamically locate and compose web services based on their semantic description will rely on the richness of the description and the robustness of the matching techniques used. Ontology will be used to enable definition and comprehension of meaningful content. These are the concerns of the Semantic Web community. Additionally, agents will be needed to interpret the content and transform user requests into optimized delivered solutions. The Intelligent Brokering Service for Knowledge-Component Reuse on the WWW (IBROW)⁴ can be seen as a forerunner of the Semantic Web. In IBROW problem solving methods (PSMs) and ontologies were the components being configured, the current focus is on WS configuration. PSMs and ontologies when used together are also capable of delivering services. The most significant work that has been done to describe web services has been conducted by the DAML-S coalition. The matching of service providers and service requesters via semantic descriptions of the services are key goals of this work. DAML-S uses the DAML+OIL specification language (which extends the weak semantics of RDF(S)) to define a number of ontologies that can be specifically used to describe web services. DAML-S is built on the AI-based action metaphor where each service is either an atomic/primitive or composite/complex action. Knowledge preconditions and knowledge effects are handled via the inputs and outputs of the web service. The DAML-S coalitions are providing solutions to work with current WS standards. For example, a DAML-S service grounding definition can be mapped to a WSDL definition of the service. A number of approaches to service discovery and composition that we discuss in the following sections use or extend the DAML-S web service ontology.

2.3 Discovering Web Services

Discovery involves locating and/or matchmaking against some selection criteria. An earlier AI system, Lark, which involved annotation of agent capabilities to enable them to be located and brokered, clearly solved a problem similar to the discovery of WS by a middle agent. This work has developed into the DAML-S Matchmaker⁵. To support matchmaking a number of filters may be configured by the user to achieve the desired trade off between performance and matching quality. These filters include: word frequency comparison, ontology similarity matching, ontology subsumption matching, and constraint matching.

Offer an alternative to sequential searching when matchmaking an agent with a service request. They point out that finding possible partners via matching of service advertisements with requests is not enough. To support runtime interactions we need smarter behavior to handle components that are not quite what was requested and combining several partial components to meet the original request. The solution to overcome sequential searching is the conversion of the concepts into number intervals and the use inheritance hierarchies to determine subclass and equality relations. A generalized search tree is used to handle partial matches.

The feasibility of matchmaking largely depends on the annotation of web services. AI can also be applied to this problem. A number of markup tools have been developed for document markup and these could be applied to the semantic description of WSs. The SHOE Knowledge Annotator [19] uses ontologies to guide knowledge annotation. To produce RDF-based markup, COHSE or AeroDAML can be used. These approaches start with descriptions in DAML+OIL and DAML, respectively. These approaches support automatic conversion of markup languages but do not support information extraction or automatic mark-up. OntoMat does support some form of

automated extraction of semantics. OntoMat combines the resource with its DAML-S markup. The MnM approach additionally stores the annotations in a knowledge base. Automated markup in MnM is achieved using techniques from knowledge engineering; machine learning and natural language processing have developed a query language that is used to find services. The solution to finding services is to first describe the service using the process ontology with the assistance of the MIT Process Handbook. The Handbook is large and allows reuse to assist in ontology definition. Next, the ontology is indexed by breaking it down into its components such as attributes, ports, dependencies, subtasks and exceptions. The requester can form a query in the query language that will use the index to find matches.

Clearly AI is already contributing solutions for locating, matchmaking, querying and annotation of WS to facilitate their discovery. Discovery of web services is an important issue as it is a prerequisite to their use. However, the real value of web services lies in their composition.

2.4 Composing Web Services

Web service composition can be simply defined as: “the problem of composing autonomous services to achieve new functionality”. WS composition is not just an alternative to application development but a means of reducing the application backlog problem because: many services are moving online; integration is easier since WSs conform to the HTTP protocol and many independent providers have related services that need to be combined to satisfy user requirements. The rigidity and lack of intelligence of current solutions has spawned a number of research projects from a number of other research fields.

The work by has arisen from experience in the distributed systems and networking fields. They have developed the Infrastructure for Composability at Runtime of Internet Services (ICARIS). They have extended WSDL to develop the Web Services Offerings Language (WSOL). They offer flexibility and adaptability but their approach is very alternative. Instead of trying to solve the problem of how to find services dynamically and combine them, they focus on the situation where providers and requestors are already matched but will at times either make changes to their services or requests. A service is seen to have numerous offerings. The functionality will be the same but the constraints will differ such as authorization rights and QoS. They suggest that a limited number of classes of services be offered and described. Then using WSOL they are able to specialize the classes into offerings. Their solution offers dynamic switching between offerings. From a commercial point of view the notion of offerings makes sense as customers probably prefer to do business with companies they already know and businesses want to maintain their existing client base.

The work at Hewlett Packard laboratories on eFlow is similar in that dynamic composition involves automatic adaptation of the configuration at runtime according to the requests of the individual customer. The approach is driven by the view that composition adds value but to stay competitive, composition needs to be dynamic as services offered need to adapt to stay competitive. Their goal is to allow dynamic change in service processes with no or minimal human intervention. While they take a business process perspective they point out that web services are less static, predictable or repetitive compared to “traditional” business processes. Similar to most current commercial solutions, dynamic composition is made possible due to the use of a central repository that has clients and providers already attached to it.

The notion of generic solutions that are customized according to user constraints is a recurring theme in much of the literature. Also look at composition as the selection of possible services based on user specified criteria. They offer a centralized, pipes and filters architecture with two main components: a composer (user interface) and an inference engine (IE) component (which includes a knowledge base of known services). The inference engine is an OWL reasoned and includes axioms to find all relevant entailments, such as the inheritance relation between two classes which may not have been made explicit. The user identifies some criteria that the service must satisfy. The matchmaker (IE) selects services that might be suitable based on those criteria and the composer shows them to the user. Suitable services for composition are ones whose

output can be an input to a selected service. While execution of WS may be performed automatically, the actual task of composition is performed by a human using the services suggested by the system.

Model-based reasoning is a common technique employed in AI approaches. In SWORD entity relationship modeling of services is performed by “base service modelers” to produce a “world model”. After building a world model for each service, a composition model is developed that models each service as an action. An expert system is used to automatically determine if the composite service can be created with existing services and if so a plan of execution is generated.

In summary, a number of solutions are offered to provide web service composition. The approaches described in this section show that composition can be assisted through the use of class definitions, inheritance hierarchies and model and rule-based reasoning. In many cases, decision making is left to humans. The only automated composition offered is in limited situations where a central repository is used and the requestor and provider are part of the same system. However, the web is distributed in nature. Intelligent reasoning and collaboration between services is needed to handle this complexity. Agents are capable of both.

2.5 Agents and Web Services

The autonomous and reasoning capabilities of agents make them well suited for handling cross-organizational decision making. For example, agents can be used to (re)negotiate contracts which would then require: determination of which processes are needed to fulfil the contract; creation of new business processes; and adaptation of existing business processes. Two main agent-oriented approaches exist: use wrappers to make WS behave like agents and; using agents to orchestrate WS.

2.5.1 Adding Behavior to WS Via Agents Wrappers

WS are componential, independent, software applications similar to agents. However, agents are also reactive, social and capable of reasoning. If we wish web services to work together, we need to give them social and reasoning capabilities. This can be achieved by wrapping a service in an agent. In the work of, a composition language is used to create an agent wrapper which allows services to collaborate. The created agent has first-order reasoning abilities that have been derived from the DAML-S description of the service. This then allows one agent wrapped service to know what other agent-wrapped services are capable of doing and whether they can assist in the service/agent meeting its goals. Also offer an agent-based wrapper approach to web services. They have developed a tool for creating wrappers so that web sources can be queried in a similar manner to databases. They then use an interactive, hierarchical constraint propagation system to perform integration. As in, the end user interacts via a GUI to manage the orchestration. The Racing project⁶ offers a mediator architecture also using agent wrappers that are structured into a hierarchy. A number of different agent wrappers are supported: user, query translation, query planning, resource wrapper, ontology, matchmaking, and cloning and coordination agents. The use of agent wrappers is a way of allowing multi-agent system technology to be applied to web services.

2.5.2 Composing Web Services Using Agents

The work of combines ideas from the Semantic Web, Knowledge Representation and Agent communities to allow WSs to be composed. Their goal is to “construct reusable, high-level generic procedures, and to archive them in shareable (DAML-S) generic-procedures ontologies so that multiple users can access them”. In the approach, WSs and user constraints are marked up in DAML-S. A generic task procedure is selected by the user and given to the DAML(-S) enabled agent, who customizes the procedure according to the user specific constraints. The generic procedures are written in an extended version of ConGolog, a situation calculus agent programming language, and executed using a Prolog inference engine. Others provide agent-oriented languages for web service description. Propose an Agent Service Description Language (ASDL) and Agent Service Composition Language (ASCL). ASDL is an extension to WSDL and

captures external behavior via a finite state machine. Their work is based on the argument that composition requires more than description of the data, but also requires a strong representation of actions and processes. A number of approaches are focused on the design of agent systems with web services as the components have developed WARP (Workflow Automation through Agent-based Reflective Processes) that uses the XML and WSDL standards. The goal is automatic configuration and management of low-level services (components). The software engineering development process that has been developed is semi-automatic involving multiple software agents and a human workflow designer. They support visualization of the process based on activity diagrams in UML.

2.5.3 (Re-)composition and Adaptable Agents

The ability of agents to adapt according to changes in system requirements and the environment is important to enable dynamic and reactive behavior.

Agents may be adapted in a number of different ways. The knowledge and facts that an agent uses may be adapted for example the agent may use a client profile that changes according to the clients activities (e.g. this type of adaptation typically involves machine learning, e.g. An agent may also adapt its interface according to the platform on which it is being used (e.g.[brand]). A third type of adaptation, and the type of adaptation we are concerned with, is adaptation of the agent's functionality. There is limited work in this area. Semi-automatic agent creation tools such as AGENTBUILDER, D'AGENTS/ AGENT/TCL, ZEUS and PARADE could possibly be extended to support agent adaptation.

Following the use of compositionality in the major software engineering paradigms (e.g. functional programming, object-oriented programming, component-based programming and the Factory design pattern, we have developed an Agent Factory. The approach is based on the use of components, the general agent model (GAM) and the DESIRE formal knowledge level modeling and specification framework for multi-agent systems. Our agent (re-)structuring approach allows an agent to automatically adapt by reusing existing components. Our approach is a combination of process-oriented and object-oriented approaches by treating processes as the 'active' parts of our agent, which are our agent components and classes as the 'passive' part of our agent, which are the data types used in the agent components. We are currently exploring whether DAML-S descriptions of web services are adequate for automated configuration of web services by the Agent Factory.

the Agent Factory and based on the notion of design patterns, assists human designers in functional design, and the configuration of software components to fulfill the conceptual design specified by the designers, depending on the agent platform that is to be used. Our approach does more: it automates the creation and redesign of both the conceptual and operational design based on the requirements on function, behavior and state of an agent. Our use of web services as components is a further distinguishing feature.

While not currently working in the WS area, the Adapt agent approach, bring together adaptive workflow and agent research. They consider how agents can be used to collaborate to perform a workflow and make workflow more intelligent and how workflow can be used to organize a set of agents and coordinate interaction between people and agents.

The reuse of knowledge has also been a widely researched topic and the creation of libraries of problem solving methods and generic task models offer a similar idea to the functional components in our agent factory. The IBROW project, mentioned earlier, has even more in common with our approach by semi automatically configuring intelligent problem solvers using problem solving methods as building blocks. They use mappings to act as glue between the components which are modeled as CORBA objects. Unlike our approach, their architecture is restricted to specific languages and architectures, they only support semi-automation and they do not distinguish between conceptual and implementation level designs.

3. EXPERIMENTAL RESULTS

In this section we evaluate the performance of the proposed artificial intelligence to eCommerce web service agent. We have created common web service application to integrate the all web and windows application who want to integrate eCommerce application to their application. Refer Fig 1 – 3 work flow of AI based eCommerce web service. Solis architecture has used three dimensional database that is Static, Meta and Dynamic. Here Static data is generally creating the tables and fields to the database. Meta data is a bridge between static and dynamic data. Dynamic data is the dynamic resultant tables or views that the user needs. It's provided three main areas of functionality – self-updating interface on the web, robust database administration, searchable front-end for end users. That system is designed so that dynamic data at the core of the integrated system is available in any output or view. The data administrator has control over the data content, various templates and user permissions, thereby giving an unrivalled level of flexibility and control in content collection, management and presentation.

3.1 Application Overview

The Order Management System (eCommerce) has been written to provide a common means to create simple orders and process credit card transactions. The first version works only with PayPal's PayFlowProw service but can be updated to work with other online merchant services (e.g. Authorize). When the need to use an alternative provide comes up we'll code the core library accordingly. Any changes here will not affect the way you use the service.

3.3.1 Not a User Management System

The system doesn't offer any user management capabilities like sign in. It assumes the calling application knows who the user is let's it take care of any user authentication required. When you're coding your shopping carts you need to handle all of this. The Order Management System simply provides the relevant methods to create Processing Sessions, Shopping Carts etc you just need to implement them.

The Order Management System does keep track of "users" (customers) through the use of an email address. This is the primary means of identification and is required before you can process any payments.

3.3.2 Typical Lifecycle / Process Flow

It's important to understand how the Order Management System (OMS) works so you can make use of the methods in the most efficient way.

The first thing you'll need to do is to create a Processing Session in the OMS. This is done by calling the `GetSessionForKnownUser()` method and passing in the email address of the current user. The OMS will create a session and a shopping cart for the user. It's recommended that you store the ID of the session in your application cookie or in your database so it can be reused. It's not efficient to create a session every time!

To retrieve the shopping cart you simply call the `GetShoppingCart()` method. To add an item to the shopping cart simply call the `AddItemToShoppingCart()` method passing in a properly constructed `ShoppingCartItem` object. To remove an item from the shopping cart simply call the `RemoveItemFromShoppingCart()` method passing in the item to remove. You can also empty the shopping cart by calling the `EmptyShoppingCart()` method.

3.3.3 Ready to Checkout

Once you've populated the shopping cart, authenticated your user you're ready to process the transaction and turn the Shopping Cart into an Order. To do this you must create a `PaymentProcessingKey`. Think of this is a temporary key allowing you to make a credit card transaction. To create one you call the `GeneratePaymentProcessingKet()` method passing in the session. It will configure it with the session and the associated shopping cart ready for processing.

3.3.4 Payment Information Page

This page is the one responsible for taking the credit card information and processing the payment through the online payment gateway (e.g. PayFlowPro). The Order summary is displayed at the top of the page so the user can make sure they're purchasing the correct item(s). The next section prompts for the credit card information including the CVV2 security code location on the credit card.

The last section prompts the user for the billing address that's associated with the credit card they're using. If the user is purchasing physical goods they should also populate a shipping address. If the order consists of only electronic items the shipping address can be left blank.

3.3.5 Processing the Payment

Once the user is happy that all the information has been entered correctly they should click the "Purchase Now" button to initiate the payment transaction. Processing payments is actually using a two step process:

1. The first step is to authorize the payment. The reason we do this is to basically test to see if the payment information is correct and that the payment card will accept the new payment being attempted without actually taking the funds. The reason we do this is to make sure the transaction will succeed. If this step fails we send the user back to the payment information page and display them the error. It basically means that we'll never process an order unless the payment succeeds.

2. The second step is to then retrieve the actual funds allocated during the first authorization step. At this point we're 99.9% confident that the transaction will succeed because the authorization was successful.

After a successful transaction the system performs some cleanup routines and processes the order:

1. Updates the status of the order to Complete
2. Constructs an invoice and sends this to the user
3. Constructs a notification email and sends this to the person setup in the installation configuration
4. Calls the Call-back page defined in the installation configuration. This page is located on calling application and is generally responsible for firing any triggers based on the products that were just purchased. For example it might need to perform an upgrade of a profile or add a new feature.

Once all this is complete the user is sent to the Order Confirmation Page where a summary of the order is presented.

Premium Listings	Select	Price				
1. Platinum Listing Includes: Priority placement, expanded listing, company logo and document file attachment. You will receive email instructions for submitting your logo and document. View Example	<input checked="" type="radio"/>	Monthly Recurring - \$ 49.95/month				
2. Gold Listing Includes: Priority placement, expanded listing and company logo. You will receive email instructions for submitting your logo. View Example	<input type="radio"/>	Monthly Recurring - \$ 39.95/month				
3. Silver Listing Includes: Priority placement and expanded listing. View Example	<input type="radio"/>	Monthly Recurring - \$ 29.95/month				
Order Total:		\$49.95				
About Recurring Billing Your listing will automatically renew each month and the monthly fee will be charged to credit card. You may cancel at any time. I have read and agree to the FEES AND PAYMENT POLICY						
<input type="button" value="I Decline"/> <input type="button" value="I Agree.. Check Out>>"/>						
Order Detail(s)						
Order ID	Account Number	Create Date	Name	Firm Name	Order Status	
2	mkt60	7/23/2009 2:19:15 PM	Vadivel	ABACUS INS BROKERS INC	Order Completed	View Full Detail
44	mkt60	8/24/2009 4:07:23 AM	Vadivel	ABACUS INS BROKERS INC	Order Completed	View Full Detail
231	mkt60	6/10/2010 12:28:03 AM	Vadivel	ABACUS INS BROKERS INC	Order Completed	View Full Detail

FIGURE 1: Product select and Checkout page

[Cancel and go back](#)

Order Summary

Item Name	Unit Price	Quantity	Total Cost
Platinum	\$49.95	1	\$49.95
Subtotal:			\$49.95
Tax:			\$0.00
Order Total:			\$49.95

Payment Information

Credit card Type: 

Credit card number:

Credit card CW2:

Expiry month / year:

Name on card:

Billing Address

This address must be the address on file for the credit card above.

First name:

Last name:

Address 1:

Address 2:

City:

State:

Country:

Zip / Postcode:

Telephone:

Shipping Address

Leave this address blank if your order is to be sent to the billing address or if not applicable.

First name:

Last name:

Address 1:

Address 2:

City:

State:

Country:

Zip / Postcode:

Telephone:

Submit Order

Your credit card will be charged \$49.95. Please check all entries and click "Place Order Now" to continue. Once the order has been submitted do not refresh your browser or you may be charged more than once.



FIGURE 2: Credit card and Billing address page

Order Confirmation

 Print page

Your payment was successful, Thank you.

Billed To:
vadivelr@365media.in
 Vadivel Rangasamy
 #64 South Street No:2 Avarampalayam
 Coimbatore, Tamil nadu 641006
 919787778365

Order Number: BMHQOEMO-535
Receipt Date: 6/14/2010 3:45:00 AM
Total items purchased: 1
Order Total: \$49.95

Item Name	Unit Price	Quantity	Total Cost
Platinum	\$49.95	1	\$49.95
Subtotal:			\$49.95
Tax:			\$0.00
Order Total:			\$49.95

[Click here to continue](#)

FIGURE 3: Order confirmation page

4. CONSLUSION & FUTURE WORK

The work of the Semantic Web community to provide semantic description of web services will play a key role in enabling agents to automatically compose web services. In this eCommerce application has implemented in embedded windows and web applications with cross-platforms and it's successfully interoperability of applications. A standard communication between the agents is clearly defined and very less amount of data loss.

Existing agent platforms may need to be adapted to handle the specific requirements of web services. But in this system with no trouble to adaptable all kind of computer applications and tested in real world applications. The RETSINA functional architecture includes four basic types of agents: interface, task, information and middle agents who communicate via a special agent communication language. Each of these agents includes four reusable modules: communication and coordination, planning, scheduling and monitoring. The middle agent plays a critical role in matching providers with requesters and is offered as a solution to the heterogeneous nature of agents over the web.

In future work will continue on artificial intelligence to natural language technology research will assist discovery of web services and agents will play an important role in using web services to satisfy user requests.

5. REFERENCES

- [1] Boutrous Saab, C.; Coulibaly, D.; Haddad, S.; Melliti, T.; Moreaux, P.; Rampacek, S. "An Integrated Framework for Web Services Orchestration", Idea Group Publishing, 2009
- [2] Raymond Y. K. Lau, "Towards a web services and intelligent agents-based negotiation system for B2B eCommerce", Elsevier Science Publishers B. V., October 2007
- [3] Sabou, M., Richards, D. and van splunter, S. An experience report on using DAML-S , Workshop on E-Services and the Semantic Web, Budapest, Hungary, May, 2003
- [4] B.Y. Wu and K.M. Chao. Spanning Trees and Optimization Problems. CRC Press, New York, USA, 2009.
- [5] Xia Yang Zhang Qiang Xu Zhao Zhang Ling, "Research on Distributed E-Commerce System Architecture", IEEE , August 2007
- [6] Lixiao Geng Zhenxiang Zeng Yajing Jiang, "Research on E-Commerce personalized service based on intelligent agent technology", IEEE, November 2008
- [7] T. Finin, J. Mayeld, C. Fink, A. Joshi, and R. S. Cost. Information retrieval and the semantic web, January 2004.
- [8] Scott Short, "Building XML Web Services for the Microsoft .NET Platform", Microsoft Press, 2002
- [9] James Murty, "Programming Amazon Web Services", O'Reilly Media, March 2008
- [10] <http://www.daml.org/ontologies/>, daml ontology library, by daml.
- [11] <http://www.schemaweb.info/>, schema web.
- [12] <http://www.semwebcentral.org/>, semwebcentral, by infoether and bbn.

[13] <http://www.w3.org/2004/ontaria/>, ontaria, by w3c.

A New Function-based Framework for Classification and Evaluation of Mutual Exclusion Algorithms in Distributed Systems

Leila Omrani

*Department of Computer Engineering
Islamic Azad University of Qazvin
Qazvin , 34185-1416,Iran*

L.Omrani@Qiau.ac.ir

Zahra Rafinezhad

*Department of Computer Engineering
Islamic Azad University of Qazvin
Qazvin , 34185-1416,Iran*

Z.Rafinezhad@Qiau.ac.ir

Mohammadreza Keyvanpour

*Department of Computer Engineering
Islamic Azad University of Qazvin
Qazvin , 34185-1416,Iran*

Keyvanpour@Qiau.ac.ir

Abstract

This paper presents a new function-based framework for mutual exclusion algorithms in distributed systems. In the traditional classification mutual exclusion algorithms were divided in to two groups: Token-based and Permission-based. Recently, some new algorithms are proposed in order to increase fault tolerance, minimize message complexity and decrease synchronization delay. Although the studies in this field up to now can compare and evaluate the algorithms, this paper takes a step further and proposes a new function-based framework as a brief introduction to the algorithms in the four groups as follows: Token-based, Permission-based, Hybrid and K-mutual exclusion. In addition, because of being dispersal and obscure performance criteria, introduces four parameters which can be used to compare various distributed mutual exclusion algorithms such as message complexity, synchronization delay, decision theory and nodes configuration. Hope the proposed framework provides a suitable context for technical and clear evaluation of existing and future methods.

Keywords: Mutual Exclusion, Critical Section, Token

1. INTRODUCTION

The mutual exclusion problem involves the allocation of a single, non shareable resource among n processes [1], by means that just one process can execute in its critical section at a given time. Mutual exclusion problem was first introduced in centralized systems. In these systems mutual exclusion ensure with preserving semaphores and monitors, and one of the nodes function as a central coordinator which is fully responsible for having all the information of the system and processes ask only the coordinator for permission to enter their critical sections. But in distributed systems the decision making is distributed across the entire system and the solution to the mutual exclusion problem is far more complicated because of the lake of common shared memory and physical clock. So obtain a complete knowledge of the total system is difficult.

Lots of algorithms are proposed in distributed systems. They classified into two groups traditionally. One of them is token-based, in this group there is a unique token in the system which ensure mutual exclusion. So the requesting node must have it to enter the critical section. Another one is permission-based group, that requesting node has to ask all other nodes for their permissions to enter the critical section [5]. According to approach of new algorithms, in this paper we proposed a new function-based framework which classified these algorithms in four groups as Token-based, Permission-based, Hybrid and K-mutual exclusion. Also in new approach spite of synchronization delay and message complexity in light and

heavy loads, we introduced two new measures including decision theory and nodes configuration.

This paper is organized as follows: in section 2, presents general model of distributed system and formally describes the mutual exclusion problem. In section 3, introduces the proposed framework and measures with classification of algorithms. Finally in last section concludes our work.

2. MODEL AND PROBLEM DEFINITION

2.1. System Model

In general, most of mutual exclusion algorithms use a common model. In this model, a distributed system is a set of independent and autonomous computers. These computers are called as node or site and connected via a communication network. Each node has abstract view of whole system that communicates with message passing [4]. The most important purposes of distributed system are assigned as providing an appropriate and efficient environment for sharing resource, having an acceptable speed and high reliability and availability.

2.2. The Mutual Exclusion Problem

Mutual exclusion problem in distributed systems has received great consideration in recent 3 decades. This problem ensures that concurrent processes access common resource and data sequentially. In addition each process that executes in its critical section for a finite time, must do without interfering with other processes. Also, when no process is in a critical section, any process that request entry to its critical section must be permitted to enter without delay [3]. Eventually mutual exclusion must be without deadlock and starvation.

3. PROPOSED FRAMEWORK AND CRITERIA FOR CLASSIFICATION OF MUTUAL EXCLUSION ALGORITHMS

By reason of the mutual exclusion importance in distributed system for keeping system consistency and increasing concurrently, various algorithms are proposed. In order to evaluate performance of these algorithms various criteria are defined as synchronization delay and message complexity. Synchronization delay is the average time delay in granting critical section. Message complexity is the number of messages exchanged by a process per critical section entry. Also evaluates message complexity in different heavy and light load of system state.

In addition, two new measures are proposed as decision theory and nodes configuration. In the first one, if each node need not keep information about the concurrent state of the system, the algorithm will be called static. On the other hand the algorithm is called dynamic. Also, in nodes configuration if nodes are assumed to be arranged in logical configuration, the algorithm is called structural, otherwise is called nonstructural. In next sections, the proposed framework is presented for classifying mutual exclusion algorithms with their evaluation as follows: 1.Token-based, 2.Permission-based, 3.Hybrid, 4.K-mutual exclusion.

3.1.Description of Token-based Approach

In this approach the right to enter a critical section is produced by a unique message, named token. The concurrent owner of the token chooses the next token owner and sends it the token. So granting the privilege to enter the critical section performs by the owner of the token. In 1985, Suzuki and Kasami [5], presented an algorithm that token by means of privilege message transmitted to requesting process based on sequence number.

Some of the algorithms use a logical structure like in Raymond [6]. In this algorithm, nodes (each process performs in a node) arranged in rootless tree structure and every node is related only to its neighbors and is aware of their information. In Naimi and Trehel algorithm [7], each node sends its request only to another one that knows as a current root and waits for its permission. In addition this algorithm uses two data structures, one of them is a queue for keeping requests and the other is a logical rooted dynamic tree for assigning token. But this algorithm is so sensitive to node failure and recovery. In consequence in [8] presented a dynamic algorithm which is able to failure detection, regenerates lost token and robust against failures.

In 2006 a new algorithm presented as queue migration [9], that nodes arranged in a distributed and fully connected network. In this network, nodes in logical group can communicate directly together for the purpose of entering the critical section. One node from each group is selected to form part of the global group this node is called a link node. Link node collects logical and global requests and sends them to the owner of token. So token transmits among local and global groups for ensuring mutual exclusion.

3-1-1. Evaluation of Token-based Approach

The most advantage of Token-based approach is simplicity. In this approach, for example if the logical structure of algorithm is a ring then the token transmits from a node to another continuously. Table1 represent the comparison and evaluation of well-known algorithms which mentioned in previous section according to proposed measures. Under light load, this method is so expensive because token is broadcasted without using by any nodes. But under heavy load it's so efficient. According to results that show in table1 Suzuki and Kasami's algorithm [5] has least message complexity under light load and queue migration algorithm [9] has least message complexity under heavy load. As you see in table1, the algorithm which has a structural configuration and dynamic decision theory can has less synchronization delay.

Token-based approach is so capable to lose token and deadlock occurs if it can't regenerate token. But if algorithm assumes the existing token is lost and regenerate another one, it will violate mutual exclusion [10]. Another problem of this approach is low scalability because all nodes arranged in a logical structure. So, when the number of nodes increment, the average of waiting time increase. Most of these algorithms have a structural configuration and decision theory make dynamically.

3.2. Description of Permission-based Approach

In Permission-based approach, requesting node asks to obtain permissions from a set of nodes in the systems. A priority or an order of events have to be established between competing requesting nodes, so only one of them receives permission from all other nodes in the set [5]. After receiving permission from a sufficient number of nodes, it is allowed to enter the critical section. When a node is completed its execution in critical section must sends release message to the other nodes. The main problem is finding a minimal number of nodes from which a node has to obtain permission to enter its critical section. Many algorithms have been developed to find this minimal, such as Lamport algorithm [11]. This algorithm uses a mechanism based on logical clocks for the total ordering of requests in the system.

Algorithm name	Evaluation measures					Description
	Message complexity		synchronization delay	configuration	decision theory	
	Heavy load	Light load				
Suzuki-Kasami	N	0	-	Non structural	dynamic	Token as a privilege message
Raymond with tree structure	O(log N)	O(log N)	((log N)/2)T	structural	static	Nodes in rootless tree
Naimi-Trehel	O(log N)	O(log N)	T	structural	dynamic	Use two structures :queue& logical tree
Dynamic tree	O(log N)	O(log N)	T	structural	dynamic	With failure detection & recovery
Queue migration	2	$O(\sqrt{N})$	-	structural	dynamic	With global & local groups

TABLE 1: comparison and evaluation of Token-based approach

In Ricart and Agrawala algorithm [12], when a node receives a request compares its sequence number with previous request and allows the request with smallest sequence

number to enter the critical section. After that in [13], both of above algorithms exchanged and introduced a new algorithm which is the best known algorithm that guarantees fairness in the same sense. It means, when there is a high priority request to do, the low priority request is delayed.

3.2.1. The Group Mutual Exclusion Problem

In this problem, every request for a critical section is associated with a type or group. It means at any time none of two processes which have requested critical sections belonging two different groups in their critical sections are simultaneously. In addition it's free of starvation it means a process wishing to enter critical section succeeds eventually.

Also, concurrent entry property is the most important issue in group mutual exclusion, it means if all requests are for critical sections belonging to the same group, then no requesting process should wait for entry in to its critical section until some other processes have left it [14]. The concept of quorum is used to solve this problem. In fact quorum is a subset of processes. Each process enters its critical section only after it has successfully locked all nodes in its quorum. There are two properties in the concept of quorum which can ensure mutual exclusion requirements: the first one is intersection and the second one is minimally [14].

At first Maekawa in 1985[15], used the concept of quorum in his algorithm. In this algorithm requests serviced based on their sequence numbers. After that another algorithm [16] represented by using the concept of dynamic quorum. The purpose of this algorithm was high scalability and low message complexity. This algorithm acts independent on its quorum system. In Maekawa algorithm a node never changes its quorum or requests from the other quorums but here a node can link to member of other quorums and respond all of its requests by making some changes during execution. Hence the quorums change dynamically. Then, the delay optimal algorithm was presented in [17]. This algorithm has least delay with fix message complexity among of above algorithms.

3.2.2. Evaluation of Permission-based Approach

Fairness is a very important measure for solutions to the most contention problems. In the concept of mutual exclusion is that requests for access to the critical section are satisfied in the order of their timestamps. This concept is obvious in the algorithms which mentioned above, clearly [13].

Table2 presents the comparison and evaluation of well-known algorithms which mentioned in previous section according to proposed measures. Beside the number of messages are exchanged for accessing the resources capture the overhead imposed on the system. In the various algorithms are tried to get an optimal value. This value in group mutual exclusion algorithms is limited to the number of quorum members [16].

As mentioned in section 3.1.1, most of the token-based algorithms configuration was structural, which decreased scalability. Also detection of lost token and regenerate were the other problems of this approach. But in permission-based approach, are tried to solve these problems. Here configuration of nodes in most of the primary algorithms was nonstructural. However after introducing the group mutual exclusion, is imposed the logical structure to the system and the configuration is became structural. But as show in table2 synchronization delay increased when the configuration of algorithms became structural. So delay optimal algorithm solved this problem.

3.3. Description of Hybrid Approach

Providing deadlock-free distributed mutual exclusion algorithms is often difficult and it involves passing many messages, so the hybrid algorithms are introduced. Since call such algorithm hybrid that uses both Token-based and Permission-based approach for mutual exclusion assurance, simultaneously. One of the algorithms of this group is proposed by Paydar et al[10]. In which sets n nodes in 2-dimension array. This array is composed of \sqrt{n} rows and \sqrt{n}

Algorithm name	Evaluation measures					Description
	Message complexity		synchronization delay	configuration	decision theory	
	Heavy load	Light load				
Lamport	$3(N-1)$	$3(N-1)$	T	Non structural	static	Give priority with timestamp
Ricart-Agrawala	$2(N-1)$	$2(N-1)$	T	Non structural	dynamic	Get n-1 permissions
fair	$2(N-1)$	$(N-1)$	T	Non structural	dynamic	Give priority with FIFO
Maekawa	$5\sqrt{N}$	$3\sqrt{N}$	$2T$	structural	static	Use quorum
quorum Dynamic	$O(q)$	$O(q)$	$3T$	structural	dynamic	Generate dynamic quorum
Delay optimal	$6(\sqrt{N} - 1)$	$6(\sqrt{N} - 1)$	T	structural	dynamic	Least synchronization delay

TABLE 2: comparison and evaluation of Permission-based approach

columns. Every node i ($i=1,2,3,\dots,n$) with the other $\sqrt{n} - 1$ nodes in the same row and the other $\sqrt{n} - 1$ nodes in the same column form the quorum Q_i . Every node can enter to critical section when it obtains the permissions of all nodes in the same quorum and gets the token too.

Kakugawa et al [18] presented an algorithm that used the coterie concept. Coterie is a set of quorums, each of which is a subset of the process set and any two quorums share at least one process. In coterie concept both intersection and minimally properties are satisfied. This algorithm uses two classes of token, main-token and sub-token. If each process of a coterie requests to enter the critical section, the owner of the main-token by generating necessary sub-tokens can respond their requests.

Then an algorithm based on Suzuki and Kasami's algorithm [27] is proposed in [19,20]. This algorithm uses non-uniform groups, in which, the same groups set in one session. Also, uses two kind of tokens to enhance concurrency of Suzuki and Kasami algorithm: primary and secondary. The owner of the primary token can grant the secondary token to others.

3.3.1. Evaluation of Hybrid Approach

In algorithms of this approach, every node begin its function in a permission-based manner but continues in token-based. This approach can overcome token-based problems, Because of using both token-based and permission-based techniques.

Table3 represents the comparison and evaluation of well-known algorithms which mentioned in previous section according to proposed measures. Most of the algorithms in this group improve message complexity and increase the degree of concurrency. This issue avoids unnecessary blocking [19,20], also they can increase fault tolerance by using failure recovery and failure detection mechanisms.

According to table3 these algorithms because of using quorums have structural configuration and their decision theory is dynamically, so these features help them to overcome the only token-based or permission-based algorithm's problems.

3.4. Description of K-mutual Exclusion Approach

The K-mutual exclusion problem is a fundamental distributed problem that completes the mutual exclusion issue. It guarantees the integrity of the k units of a shared resource by restricting the number of processes that can access them simultaneously [21]. Likewise these algorithms divided in to token-based and permission-based groups. In token-based group, k tokens are generated to let requested processes to enter their critical sections. But in permission-based group a node gets in to the critical section only after sending requests to

Algorithm name	Evaluation measures					description
	Message complexity		synchronization delay	configuration	decision theory	
	Heavy load	Light load				
Paydar algorithm	$4\sqrt{N}$	$4\sqrt{N}$	-	structural	Dynamic	Quorum contains of same column and row nodes
Kakugawa	$5 Q +1$	0	3T	structural	Dynamic	With coterie and main token & sub token
Non-uniform groups	2N-1	2N-1	T	structural	Dynamic	With non-uniform groups

TABLE 3: Comparison and evaluation of Hybrid approach

the $n - 1$ other nodes and receiving permission from $n - k$ nodes [22].

One of the algorithms of this group that uses queue migration is extended on [9]. In this algorithm for ensuring mutual exclusion, the link node is the owner of parent token which has the capacity of generating $k - 1$ tokens. In [23] the Raymond's algorithm [6] extended. Although in this algorithm each node has a sequence number, it has to obtain $n - k$ permissions to enter the critical section. In [21] the Raymond's algorithm is extended again and it increases fault tolerance to $n - 1$ nodes, in which in Raymond's algorithm it was $k - 1$ nodes. So it ensures that even occurs failure, k processes can execute simultaneously.

3-4-1.Evaluation of K-mutual Exclusion Approach

Most of the k-mutual exclusion algorithms focus on fault tolerance. They provide using more resource at the same time. For example, some of these algorithms can prevent the system security until occurs $k - 1$ failures. It means when a node fails, It is impossible to any others can collect $n - k$ permissions and enter to the critical section. But after $k - 1$ failures, the most number of processes that can execute simultaneously reach to one. As a result the algorithm's performance decreases. Also, if the number of the received permissions decreased by occurring failures dynamically, the value of fault tolerance will reach to $n - 1$ nodes. So it ensures that even occur failures, k processes can execute in their critical sections simultaneously [23,24].

Table 4 represents the comparison and evaluation of well-known algorithms which mentioned in previous section according to proposed measures. As you see decision theory in these algorithms are dynamically.

Algorithm name	Evaluation measures					description
	Message complexity		synchronization delay	configuration	decision theory	
	Heavy load	Light load				
k-queue migration	$O(\sqrt{N})$	0	4T	structural	dynamic	Use parent token & generate k-1 tokens
Raymond with multi entries	2N-1	2N-k-1	-	Non structural	dynamic	K resources & n-k replies
Obtain n-k replies from n_i correct nodes	2N-1	2N-k-1	-	Non structural	dynamic	Extended of Raymond algorithm

TABLE 4:comparison and evaluation of K-mutual approach

4. CONCLUSION

According to importance of mutual exclusion in achieving goals of distributed systems, various approaches are proposed. In this research try to make a brief analyzing on most

common distributed mutual exclusion algorithms. Also, present a new classification based on their functions in four groups: Token-based, Permission-based, Hybrid and K-mutual exclusion. This framework helps novice researchers to set each new algorithm in specific category. To achieving this purpose focus on four measures such as message complexity, synchronization delay, decision theory and nodes configuration for comparison and evaluation of them. Hope the proposed framework makes a convenient way for future researches.

5. REFERENCES

- [1] N.A.Lynch . "*Distributed Algorithms*" , Morgan Kaufmann Publishers, pp.255-327,(1996)
- [2] P.C.Saxena, and J.Rai. "*A Survey Of Permission-based Distributed Mutual Exclusion Algorithms*". Computer Standards & Interfaces, 25: 159-181, 2003
- [3] M.G.Velaquez. "*A Survey Of Distributed Mutual Exclusion Algorithms*". Technical Report CS. Colarido state university, September 1993
- [4] W.Stallings. "*Operating Systems Internals and Design Principls*", Prentice Hall, pp.205-261 (2009)
- [5] I.Suzuki,and T.Kasami."*A Distributed Mutual Exclusion Algorithm*", ACM Transactions On Computer Systems, Vol.3(No.4): 344-349, November 1985
- [6] K.Paymond. "*A TreeBased Algorithm For Distributed MutualExclusion*",ACM Transactions On Computer System, Vol.7(No.1): 61-77, February 1989
- [7] M.Naimi, M.Trehel, and A.Arnold. "*A Log(n) Distributed Mutual Exclusion Algorithm Based On The Path Reversal*", Journal Of Parallel And Distributed Computing,34(1): 1-13 April 1996
- [8] J.Sopena , L.Arantes, M.Bertier, and Pierre Sens. "*A Fault-tolerant Token-based Mutual Exclusion Algorithm Using A Dynamic Tree*". Euro Par.LNCS 3648, 2005
- [9] P.chaudhuri, and Tomas Edward. "*An $O(\sqrt{n})$ Distributed Mutual Exclusion Algorithm Using Queue Migration*". *Journal Of Universal Computer Science*, Vol.21(No.2):140-159, 2006
- [10] S.Paydar, M.Naghizadeh , and A.Yavari. "*A Hybrid Distributed Mutual Exclusion Algorithm*", IEEE International Conference On Emerging Technologies In Pakistan, November 2006
- [11] L.Lamport. "*Times,Clocks,And The Ordering Of Events in a Distributed System*". Communications OF The ACM, Vol.21(No.7): 558-565, July 1978
- [12]G.Ricart, and Ashok.K.Agrawala. "*An Optimal Algorithm For Mutual Exclusion in Computer Networks*". *Communication of The ACM*, Vol.24(No.1):9-17, January 1981
- [13] S.Lodha, and A.Kshemkalyani. "*A Fair Distributed Mutual Exclusion algorithm*".IEEE Transactions On Parallel And Distributed Systems. Vol.11(No.6), June 2000
- [14] R.Atreyya , and N.Mittal. "*A Dynamic Group Mutual Exclusion Using Surrogate-Quorums*". Proc,IEEE Int'l Conf. Distributed Computing System, June 2005
- [15] M.Maekawa. "*A \sqrt{n} Algorithm For Mutual Exclusion In Decentralized Systems*". ACM Transactions On Computer Systems , Vol.3 (No.2):145-159, May 1985
- [16] R.Atreyya , and N.Mittal. "*a Quorum-based Group Mutual Exclusion Algorithm For A Distrinbuted System With Dynamic Group Set*". IEEE Transactions On Parallel And Distributed Systems, Vol.18(No.10), October 2007

- [17] G.Cao , and M.Singhal. "*A Delay-optimal Quorum-based Mutual Exclusion Algorithm For Distributed Systems*". IEEE Transactions On Parallel And Distributed Systems, Vol.12(No.12), December 2001
- [18] H.Kakugawa , S.Kamei, and T.Masuzawa. "*A Token-based Distributed Group Mutual Exclusion Algorithm With Quorums*", IEEE Transactions On Parallel And Distributed Systems, Vol.19(No.9), 2008
- [19] N.Mittal,and P.Mohan."*An Efficient Distributed Group Mutual Exclusion Algorithm For Non-uniform group access*", proceedings Of The IASTED International Conference On Parallel And Distributed Computing And Systems , Phoenix,Arizona,USA, 2005
- [20] N.Mittal,and P.Mohan. "*A Priority-based Distributed Group Mutual Exclusion Algorithm When Group Access is Non-uniform*". Journal Of Parallel And Distributed Computing, No.67: 797-815, March 2007
- [21] M.Bouillaguet,L.Arantes,and P.Sens. "*Fault Tolerant K-mutual Exclusion Algorithm Using Failure Detector*" . International Symposium On Parallel And Distributed Computing, 2007
- [22] P.Chaudhuri, and T.Edward. "*An Algorithm for K-mutual Exclusion In Decentralized Systems*". Computer Communications 31: 3233-3235, 2008
- [22] K.Raymond. "*A Distributed Algorithm For Multiple Entries To A Critical Section*", Information Processing Letters, North-Holland, No.30, February 1989

J48 and JRIP Rules for E-Governance Data

Anil Rajput

*Principal, Bhabha Engineering Research
Institute-MCA, Bhopal-26, India*

drar1234@yahoo.com

Ramesh Prasad Aharwal

*Asstt. Prof., Department of Mathematics and
Computer Science, Govt. P.G. College
Bareilly (M.P.), 464668, India*

ramesh_ahirwal_neetu@yahoo.com

Meghna Dubey

*Asstt. Prof., Department of Computer science,
SCOP College, Bhopal (M.P.) - India.*

S.P. Saxena

*HOD, T.I.T. Engineering college-MCA,
Bhopal, India*

Manmohan Raghuvanshi

*Asstt. Prof. BIST Bhopal (M.P.)
India*

Abstract

Data are any facts, numbers, or text that can be processed by a computer. Data Mining is an analytic process which designed to explore data usually large amounts of data. Data Mining is often considered to be "a blend of statistics. In this paper we have used two data mining techniques for discovering classification rules and generating a decision tree. These techniques are J48 and JRIP. Data mining tools WEKA is used in this paper.

Keywords: Data Mining, Jrip, J48, WEKA, Classification.

1. INTRODUCTION

Data mining is an interdisciplinary research area such as machine learning, intelligent information systems, database systems, statistics, and expert systems. Data mining has evolved into an important because of theoretical challenges and practical applications associated with the problem of extracting interesting and previously unknown knowledge from huge real-world databases. Indeed, data mining has become a new paradigm for decision making, with applications ranging from E-commerce to fraud detection, credit scoring, even auditing data before storing it in a database. The fundamental reason for data mining is that there is a lot of money hidden in the data. Without data mining all we have are opinions, we need to understand the data and translate it into useful information for decision making. According to Han and Kamber (2001), the term 'Data Mining' is a misnomer.

2. CLASSIFICATION

Classification problems aim to identify the characteristics that indicate the group to which each case belongs. This pattern can be used both to understand the existing data and to predict how New instances will behave. Data mining creates classification models by examining already Classified data (cases) and inductively finding a predictive pattern. These existing cases may come from historical database. They may come from an experiment in which a sample of the entire database is tested in the real world and the results used to create a classifier. Sometimes an expert classifies a sample of the database, and this classification is then used to create the model which will be applied to the entire database (TCC, 1999). A number of data mining algorithms have been introduced to the community that perform summarization of the data, classification of data with respect to a target attribute, deviation detection, and other forms of

data characterization and interpretation. One popular summarization and pattern extraction algorithm is the association rule algorithm, which identifies correlations between items in transactional databases.

In data mining tasks, classification and prediction is among the popular task for knowledge discovery and future plan. The classification process is known as supervised learning, where the class level or classification target is already known. There are many techniques used for classification in data mining such as Decision Tree, Bayesian, Fuzzy Logic and Support Vector Machine (SVM). In fact, there are many techniques from the decision tree family such as C4.5, NBTtree, SimpleCart, REPTree, BFTree and others. The C4.5 classification algorithm is easy to understand as the derived rules have a very straightforward interpretation. Due to these reasons, this study is aimed to use this classification algorithm to handle issue on E-governance data.

2.1 Decision Tree

Decision tree can produce a model with rules that are human-readable and interpretable. According to Hamidah Jantan et al 2010 (H. Jantan et a), the classification task using decision tree technique can be performed without complicated computations and the technique can be used for both continuous and categorical variables. This technique is suitable for predicting categorical outcomes (H. Jantan et a). Decision tree classifiers are quite popular techniques because the construction of tree does not require any domain expert knowledge or parameter setting, and is proper for exploratory knowledge discovery. In present, there are many research that in use decision tree techniques such as in electricity energy consumption (G. K. F. Tso and K. K. W. Yau), prediction of breast cancer (D. Delen), accident frequency (L. Y. Chang). It is stated that, the decision tree is among the powerful classification algorithms some of decision tree classifiers are C4.5, C5.0, J4.8, NBTtree, SimpleCart, REPTree and others (H. Jantan et a).

2.2 Decision Tree Classifier

The C4.5 technique is one of the decision tree families that can produce both decision tree and rule-sets; and construct a tree. Besides that, C4.5 models are easy to understand as the rules that are derived from the technique have a very straightforward interpretation. J48 classifier is among the most popular and powerful decision tree classifiers. C5.0 and J48 are the improved versions of C4.5 algorithms. WEKA toolkit package has its own version known as J48. J48 is an optimized implementation of C4.5.

3. DATA SOURCES AND DESCRIPTION

Data is taken from questioners which is fillip from individuals. A questionnaire contains ten questions and demographic information. These Questionnaires have fill up from Bhopal which is a capital of Madhya Pradesh. After the initial data collection, new database was created in Ms Excel format. Ms Excel was used for preparing the dataset into a form acceptable by the selected data mining software, and Knowledge studio Weka. The database table is for E-governance data with 15 columns and 397 rows.

Question No.	Descriptions	Values
Q1	Whether are you have T. V.	True/false
Q2	Purpose of T. V.	True/false
Q3	How many mobiles are you have in your home?	True/false
Q4	Whether are you having Computer in your home?	True/false
Q5	Purpose of Computer at home	True/false
Q6	Whether you have a internet at your home	True/false
Q7	Whether you use a internet	True/false
Q8	Whether are you know about E-Governance	True/false
Q9	Whether are you know about Common Service Centre	True/false
Q10	Whether are you gain information from E-Governance	True/false

TABLE 1: Descriptions of each questions

4. EXPERIMENT

This study has three phases; the first phase is the data collection process which involved the data cleaning and data preprocessing. The second phase is to generate the classification rules using j48 classifier for the training dataset. In this case, we use all the selected attributes defined in Table 1. The J48 classifier produced the analysis of the training dataset and the classification rules. In the experiment, the third phase of experiment is the evaluation and interpretation of the classification rules using the unseen data. In the experiment we have use 66% instances of the database as a training datasets and remaining instances for tested dataset. The analyses were performed using WEKA environment. Inside the Weka system, there exist many classification algorithms which can be classified into two types; rule induction and decision-tree algorithms (N. Ulutađdemir and Ö. Dađlı). Rule induction algorithms generate a model as a set of rules. The rules are in the standard form of IFTHEN rules. Meanwhile, decision-tree algorithms generate a model by constructing a decision tree where each internal node is a feature or attribute.

4.1 Data Mining Tool Selection

Data mining tool selection is normally initiated after the definition of problem to be solved and the related data mining goals. However, more appropriate tools and techniques can also be selected at the model selection and building phase. Selection of appropriate data mining tools and techniques depends on the main task of the data mining process. In this paper we have used WEKA software for extracting rules and built a decision tree. The selected software should be able to provide the required data mining functions and methodologies. The data mining software selected for this research is WEKA (to find interesting patterns in the selected dataset). The suitable data format for Weka data mining software are MS Excel and arff formats respectively. Scalability-Maximum number of columns and rows the software can efficiently handle. However, in the selected data set, the number of columns and the number of records were reduced. Weka is developed at the University of Waikato in New Zealand. "Weka" stands for the Waikato Environment of Knowledge Analysis. The system is written in Java, an object-oriented programming language that is widely available for all major computer platforms, and Weka has been tested under Linux, Windows, and Macintosh operating systems. Java allows us to provide a uniform interface to many different learning algorithms, along with methods for pre and post processing and for evaluating the result of learning schemes on any given dataset. Weka expects the data to be fed into to be in ARFF format (WEKA website).

4.2 Screen Shots Which is Generated During Experiment

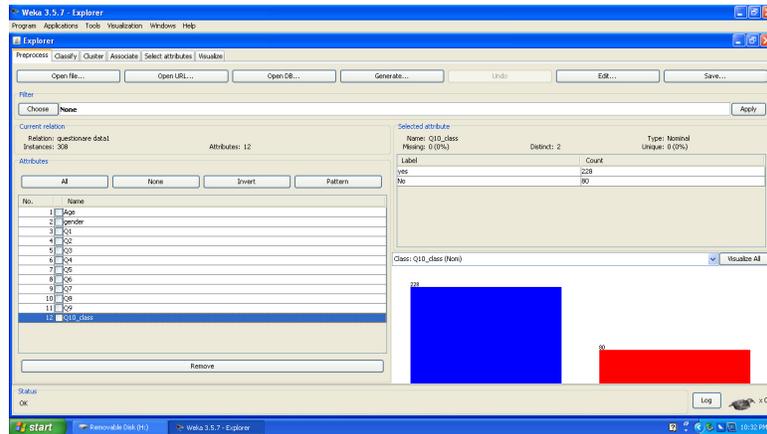


FIGURE: 1 WEKA explorer

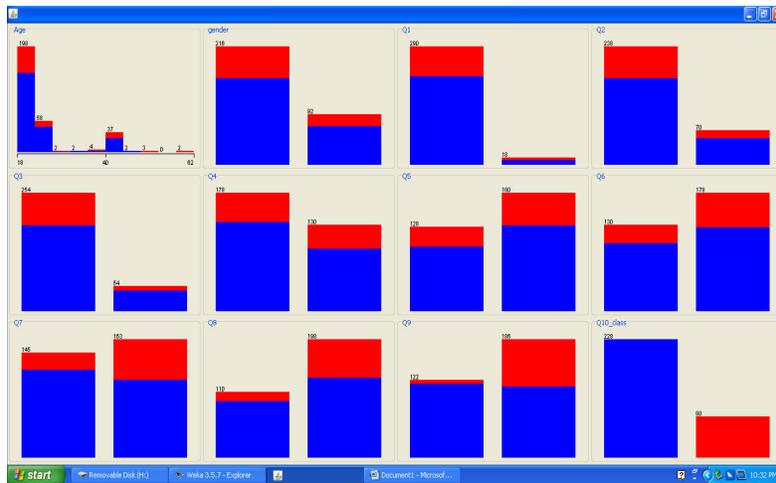


FIGURE: 2: Visualization of each attributes of experimental data

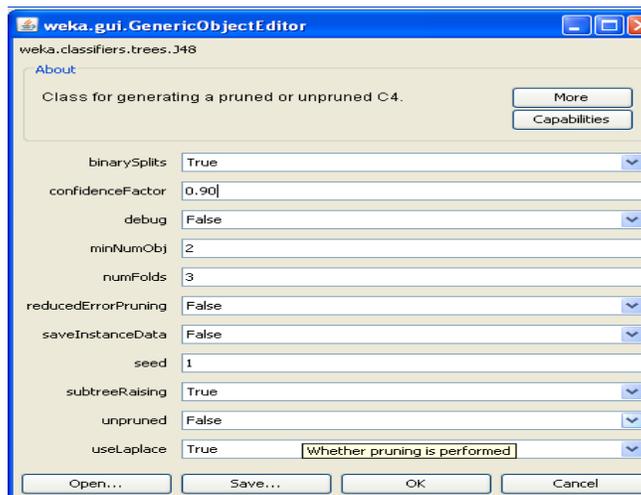


FIGURE 3: parameter setting

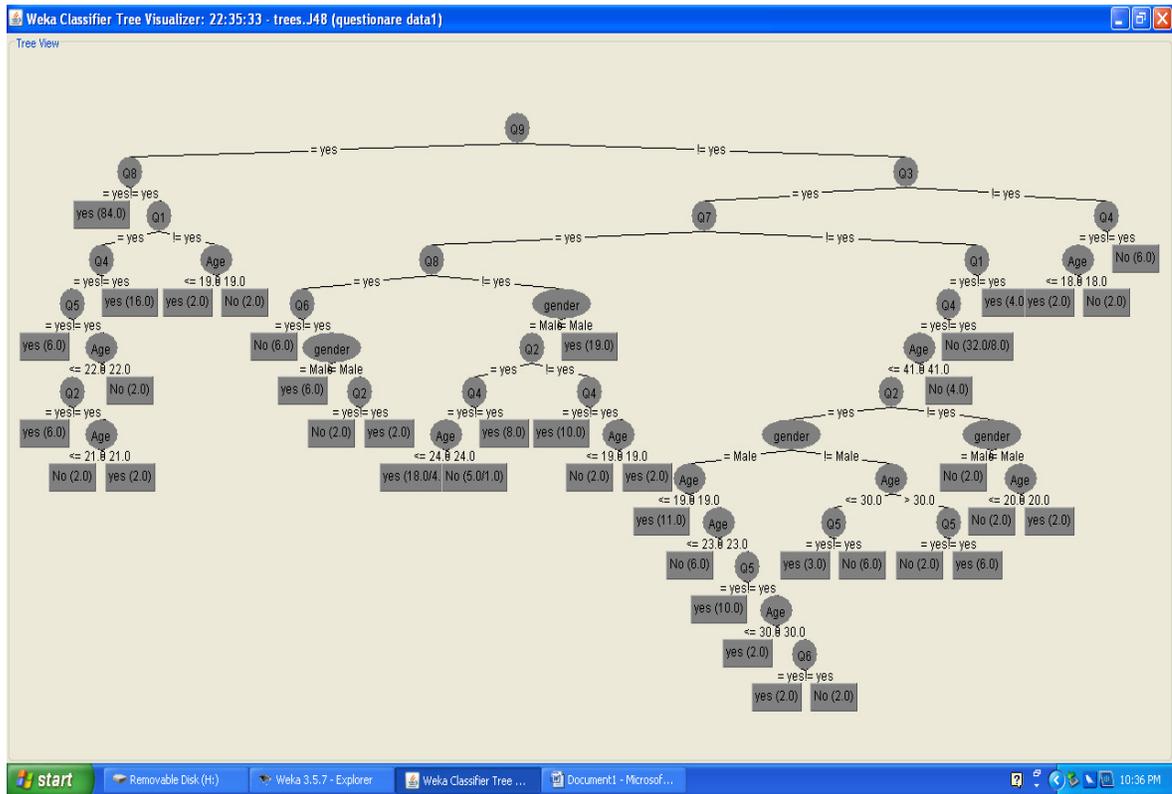


FIGURE 4: Decision tree generated from WEKA

Experimental Result

=== Run information ===

Scheme: weka.classifiers.trees.J48 -C 0.9 -B -M 2 -A

Relation: questionare data1

Instances: 308

Attributes: 12

Age, gender, Q1, Q2, Q3, Q4, Q5, Q6, Q7, Q8, Q9, Q10_class

Test mode: split 66% train, remainder test

=== Evaluation on test split ===

=== Summary ===

Correctly Classified Instances	90	85.7143 %
Incorrectly Classified Instances	15	14.2857 %

=== Confusion Matrix ===

a b <-- classified as

68 8 | a = yes

7 22 | b = No

==== Predictions on test split ====

5. JRIP RULES CLASSIFIERS

JRip (RIPPER) is one of the basic and most popular algorithms. Classes are examined in increasing size and an initial set of rules for the class is generated using incremental reduced error JRip (RIPPER) proceeds by treating all the examples of a particular judgment in the training data as a class, and finding a set of rules that cover all the members of that class. Thereafter it proceeds to the next class and does the same, repeating this until all classes have been covered.

==== Run information ====

Scheme: weka.classifiers.rules.JRip -F 5 -N 2.0 -O 2 -S 1 -D

Relation: questionnaire data1

Instances: 308

Attributes: 12

Age,gender, Q1, Q2, Q3, Q4, Q5, Q6,Q7,Q8, Q9, Q10_class

Test mode: split 66% train, remainder test

JRIP rules:

=====

(Q9 = No) and (Q7 = No) and (Q4 = No) => Q10_class=No (42.0/12.0)

(Q9 = No) and (Q8 = yes) and (gender = Female) => Q10_class=No (10.0/2.0)

(Q9 = No) and (Q7 = No) and (Age >= 20) and (Age <= 22) => Q10_class=No (10.0/0.0)

(Age >= 30) and (Age >= 49) => Q10_class=No (5.0/0.0)

(Q8 = No) and (Q3 = No) and (Age >= 19) => Q10_class=No (4.0/0.0)

=> Q10_class=yes (237.0/23.0)

5.1 Interpretations of Rules

In this section we try to interpret Jrip rule.

Rule first interpreted as If a person do not know about common service center and do not use a computer and also he does not have computer at his home then he do not gain information from E-governance.

Rule second interpreted as If a person does not know about common service center and he know about E-governance and he is a female then he does not gain information from E-governance.

Third rule is interpreted as If a person do not know about common service center and he does not use a computer and age is above 19 and below 23 then he do not gain information from E-governance.

Similarly last rule is interpreted as, if a person do not know about E-governance and he have not mobile at home and age is greater and equal 19 years then he gain information from E-governance

Same way other rules can be interpreted as above

6. CONCLUSION

This paper focuses on the use of decision tree and JRIP classifiers for E-governance data. Decision tree classifier generates the decision tree. From generated decision tree useful and

meaningful rules can be extracted. JRIP classifier generates some useful rules which are interpreted in above section.

7. REFERENCES

- [1] L. Y. Chang and W. C. Chen, "Data mining of tree-based models to analyze freeway accident frequency," *Journal of Safety Research*,36(1): 365-375, 2005.
- [2] D. Delen, G. Walker, and A. Kadam, "Predicting breast cancer survivability: A comparison of three data mining methods," *Artificial Intelligent in Medicine*, 34:113- 127, 2005.
- [3] H. Jantan et al. "Classification for Prediction", *International Journal on Computer Science and Engineering*, 2(8): 2526-2534, 2010.
- [4] J. Han and M. Kamber, "Data Mining: Concept and Techniques". Morgan Kaufmann (2006).
- [5] G. K. F. Tso and K. K. W. Yau, "Predicting electricity energy consumption: A comparison of regression analysis, decision tree and neural networks," *Energy*, 32 : 1761-1768, 2007.
- [6] N. Ulutagdemir and Ö. Dagi, "Evaluation of risk of death in hepatitis by rule induction algorithms", *Scientific Research and Essays*, 5(20): 3059-3062, 2010, ISSN 1992-2248.
- [7] Weka website: Data Mining Software in Java, <http://www.cs.waikato.ac.nz/ml/weka/>

Concurrent Matrix Multiplication on Multi-Core Processors

Muhammad Ali Ismail

*Assistant Professor, Faculty of Electrical & Computer Engineering
Department of Computer & Information Systems Engineering
NED University of Engineering & Technology
Karachi, 75270, Pakistan*

maismail@neduet.edu.pk

Dr. S. H. Mirza

*Professor
Usman Institute of Technology
Karachi, 75300, Pakistan*

shmirza@uit.edu

Dr. Talat Altaf

*Professor, Faculty of Electrical & Computer Engineering
Department of Electrical Engineering
NED University of Engineering & Technology
Karachi, 75270, Pakistan*

deanece@neduet.edu.pk

Abstract

With the advent of multi-cores every processor has built-in parallel computational power and that can only be fully utilized only if the program in execution is written accordingly. This study is a part of an on-going research for designing of a new parallel programming model for multi-core architectures. In this paper we have presented a simple, highly efficient and scalable implementation of a common matrix multiplication algorithm using a newly developed parallel programming model SPC³ PM for general purpose multi-core processors. From our study it is found that matrix multiplication done concurrently on multi-cores using SPC³ PM requires much less execution time than that required using the present standard parallel programming environments like OpenMP. Our approach also shows scalability, better and uniform speedup and better utilization of available cores than that the algorithm written using standard OpenMP or similar parallel programming tools. We have tested our approach for up to 24 cores with different matrices size varying from 100 x 100 to 10000 x 10000 elements. And for all these tests our proposed approach has shown much improved performance and scalability.

Keywords: Multi-Core, Concurrent Programming, Parallel Programming, Matrix Multiplication.

1. INTRODUCTION

Multi-core processors are becoming common and they have built-in parallel computational power and which can only be fully utilized only if the program in execution is written accordingly. Writing an efficient and scalable parallel program is much complex. Scalability embodies the concept that a programmer should be able to get benefits in performance as the number of processor cores increases. Most software today is grossly inefficient, because it is not written with sufficient parallelism in mind. Breaking up an application into a few tasks is not a long-term solution. In order to make most of multi-core processors, either, lots and lots of parallelism are actually needed for efficient execution of a program on larger number of cores, or secondly, concurrent execution of multiple programs on multiple cores [1, 2].

Matrix Multiplication is used as building block in many of applications covering nearly all subject areas. Like physics makes use of matrices in various domains, for example in geometrical optics and matrix mechanics; the latter led to studying in more detail matrices with an infinite number of rows and columns. Graph theory uses matrices to keep track of distances between pairs of vertices in a graph. Computer graphics uses matrices to project 3-dimensional space onto a 2-dimensional screen. Matrix calculus generalizes classical analytical concept such as derivatives of functions or exponentials to matrices etc [4, 11, 13]. Serial and parallel matrix multiplication is always be a challenging task for the programmers because of its extensive computation and memory requirement, standard test set and broad

use in all types of scientific and desktop applications. With the advent of multi-core processors, it has become more challenging. Now all the processors have built-in parallel computational capacity in form of cores and existing serial and parallel matrix multiplication techniques have to be revisited to fully utilize the available cores and to get the maximum efficiency and the minimum executing time [2, 3, 8, 9].

In this paper we have presented a concurrent matrix multiplication algorithm and its design using a new parallel programming model SPC³ PM, (**S**erial, **P**arallel, and **C**oncurrent **C**ore to **C**ore **P**rogramming **M**odel) developed for multi-core processors. It is a serial-like task-oriented multi-threaded parallel programming model for multi-core processors that enables developers to easily write a new parallel code or convert an existing code written for a single processor. The programmer can scale it for use with specified number of cores. And ensure efficient task load balancing among the cores.

The rest of the paper is organized as follows. In section 2, the related studies on parallel and concurrent matrix multiplication are briefly reviewed. The characteristics of SPC³ PM are described in section 3. Section 4 deals with the programming in SPC³ PM. The concurrent matrix multiplication algorithm based on SPC³PM is presented in section 5. In section 6 and 7, the experimental setup and results are discussed respectively. Finally, conclusion and future work are given in section 8.

2. RELATED WORK

Many of parallel matrix multiplication algorithms and implementations for SMPs and distributed systems have been proposed. Like Systolic algorithm [5], Cannon's algorithm [], Fox's algorithm with square decomposition, Fox's algorithm with scattered decomposition [6], SUMMA [7], DIMMA [10], 3-D matrix multiplication [12] etc. Majority of the parallel implementations of matrix multiplication for SMPs are based on functional parallelism. The existing algorithms for SMPs are not so efficient for multi-core and have to be re-written using some multi-core supported language [1, 2]. These algorithms are also difficult for common programmer to understand as they require detailed related subject knowledge. On the other hand distributed algorithms which are usually base on data parallelism also cannot be applied on the shared memory multi-core processors because of the architectural change.

Some attempts have also been made to solve matrix multiplication using data parallel or concurrent approaches on cell or GPUs [14, 15, 16, 17, 18, 19]. But the associated problem with these approaches is architectural dependence and cannot be used for general purpose multi-core processors.

3. SPC³ PM

SPC³ PM, (**S**erial, **P**arallel, **C**oncurrent **C**ore to **C**ore **P**rogramming **M**odel), is a serial-like task-oriented multi-threaded parallel programming model for multi-core processors, that enables developers to easily write a new parallel code or convert an existing code written for a single processor. The programmer can scale it for use with specified number of cores. And ensure efficient task load balancing among the cores.

SPC³ PM is motivated with an understanding that existing general-purpose languages do not provide adequate support for parallel programming. Existing parallel languages are largely targeted to scientific applications. They do not provide adequate support for general purpose multi-core programming whereas SPC³ PM is developed to equip a common programmer with multi-core programming tool for scientific and general purpose computing. It provides a set of rules for algorithm decomposition and a library of primitives that exploit parallelism and concurrency on multi-core processors. SPC³ PM helps to create applications that reap the benefits of processors having multiple cores as they become available.

SPC³ PM provides thread parallelism without the programmers requiring having a detailed knowledge of platform details and threading mechanisms for performance and scalability. It helps programmer to control multi-core processor performance without being a threading expert. To use the library a programmer specifies tasks instead of threads and lets the library map those tasks onto threads and threads onto cores in an efficient manner. As a result, the programmer is able to specify parallelism and concurrency far more conveniently and with

better results than using raw threads.. The ability to use SPC³ PM on virtually any processor or any operating system with any C++ compiler also makes it very flexible.

SPC³ PM has many unique features that distinguish it with all other existing parallel programming models. It supports both data and functional parallel programming. Additionally, it supports nested parallelism, so one can easily build larger parallel components from smaller parallel components. A program written with SPC³ PM may be executed in serial, parallel and concurrent fashion. Besides, it also provides processor core interaction to the programmer. Using this feature a programmer may assign any task or a number of tasks to any of the cores or set of cores.

3.1 Key Features

The key features of SPC³ are summarized below.

- SPC³ is a new shared programming model developed for multi-core processors.
- SPC³ PM works in two steps: defines the tasks in an application algorithm and then arranges these tasks on cores for execution in a specified fashion.
- It provides Task based Thread-level parallel processing.
- It helps to exploit all the three programming execution approaches, namely, Serial, Parallel and Concurrent.
- It provides a direct access to a core or cores for maximum utilization of processor.
- It supports major decomposition techniques like Data, Functional and Recursive.
- It is easy to program as it follows C/C++ structure.
- It can be used with other shared memory programming model like OpenMP, TBB etc.
- It is scalable and portable.
- Object oriented approach

4. PROGRAMMING WITH SPC³ PM

SPC³ PM provides a higher-level, shared memory, task-based thread parallelism without knowing the platform details and threading mechanisms. This library can be used in simple C / C++ program having tasks defined as per SPC³ PM Task Decomposition rules. To use the library, you specify tasks, not threads, and let the library map tasks onto threads in an efficient manner. The result is that SPC³ PM enables you to specify parallelism and concurrency far more conveniently, and with better results, than using raw threads.

Programming with SPC³ is based on two steps. First describing the tasks as it specified rules and then programming it using SPC³ library. The figure 1 shows the step by step development of an application using SPC³PM.

4.1 Rules for Task Decomposition

- Identify the parts of the code which can be exploited using Functional, Data or Recursive decomposition
- Defined all those piece of code specified in step 1 as Tasks.
- Identify the loops for the loop parallelism and also defined them as Tasks
- Identify portions of the application algorithm which are independent and can be executed concurrently
- A Task may be coded using either C/C++/VC++/C# as an independent unit.
- Tasks should be named as Task1, Task2,..... TaskN.
- There are no limits for Tasks.
- Arrange the tasks using SPC³ library in the main program file according to the program flow.
- A Task may be treated as a function.
- A Task may only intake pointer structure as a parameter. Initialize all the parameters in a structure specific to a Task.
- A structured may be shared or private.
- A Task may or may not return the value. The Task named with suffix 'V' do not return any value. The Task with suffix 'R' do return value.

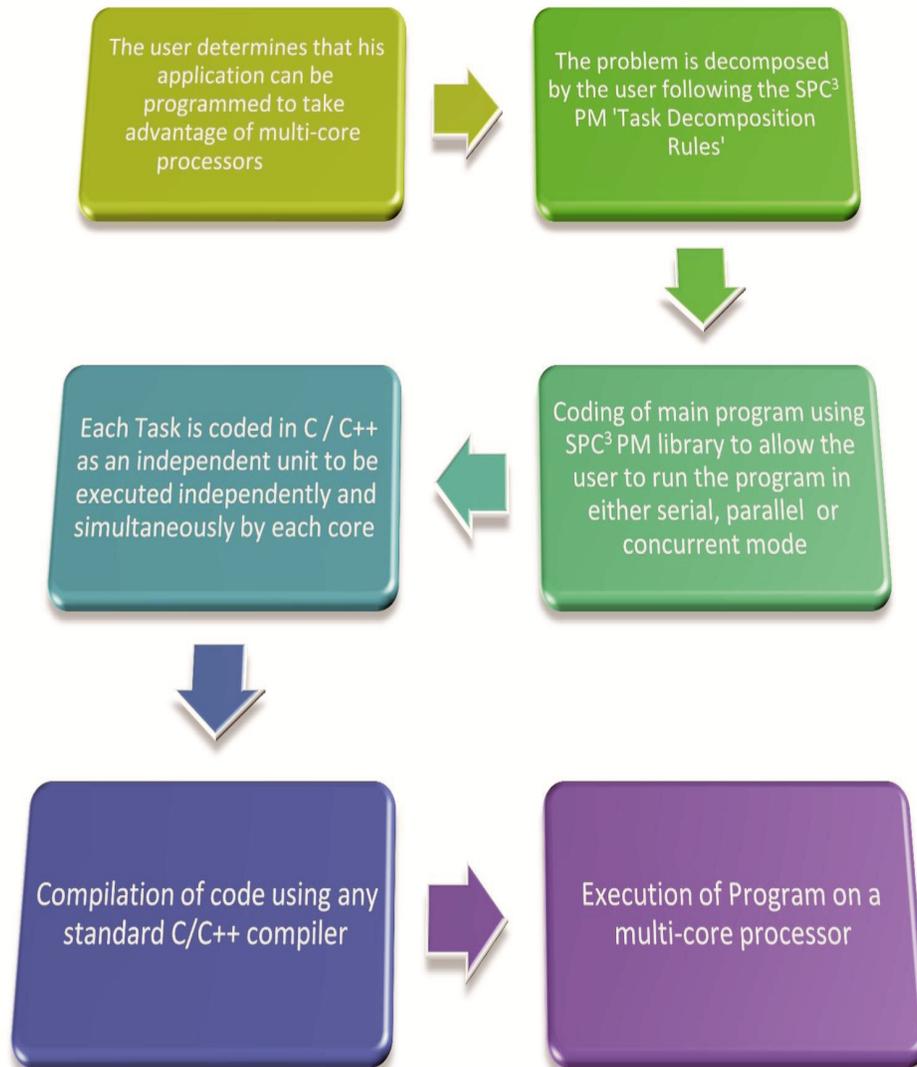


FIGURE 1: Steps involved in programming with SPC³ PM

4.2 Program Structure

```

Define Task1
Define Task2
Define Task3
Define Task4
...
Define TaskN
    
```

Structure	Structure	Structure	Structure
<i>STRUCT_NAME</i>	<i>STRUCT_NAME</i>	<i>STRUCT_NAME</i>	<i>STRUCT_NAME</i>
{	{	{	{
//The structure	//The structure	//The structure	//The structure
//having private	//having private	//having private	//having private
//or global	//or global	//or global	//or global
//parameters	//parameters	//parameters	//parameters
//associated with a	//associated with a	//associated with a //associated with a
//specified task	//specified task	//specified task	//specified task
}	}	}	}
<i>STRUCT_NAME</i>	<i>STRUCT_NAME</i>	<i>STRUCT_NAME</i>	<i>STRUCT_NAME</i>
*P_TASK1	*P_TASK2	*P_TASK3	*P_TASKN

```

Task1(LPVOID)   Task2(LPVOID)   Task3(LPVOID)   .....   TaskN(LPVOID)
{
//performing      //performing      //performing      .....      //performing
//some            //some            //some            .....      //some
//computation     //computation     //computation     .....      //computation
}
    
```

```

void main( void)
{
    // any declaration;
    // any piece of code ;

    Serial (Task1, P_TASK1);           //execution of task 1 in serial

    // Any other code ;

    Parallel (Task2, P_TASK2);         //execution of task 2 in parallel

    // any other code ;

    Concurrent (Task3 , P_TASK3, Task4, P_TASK4); //execution of task 3 and 4 concurrently
}
    
```

4.3 SPC³ PM Library

SPC³ PM provides a set of specified rules to decompose the program into tasks and a library to introduce parallelism in the program written using C/ C++. The library provides three basic functions.

- Serial
- Parallel
- Concurrent

Serial: This function is used to specify a Task that should be executed serially. When a Task is executed with in this function, a thread is created to execute the associated task in sequence. The thread is scheduled on the available cores either by operating system or as specified by the programmer. This function has three variants. *Serial (Task i)* {Basic}, *Serial (Task i, core)* {for core specification} and **p Serial (Task i, core, *p)* {for managing the arguments with core specification}

Parallel: This function is used to specify a Task that should be executed in parallel. When a Task is executed with in this function, a team of threads is created to execute the associated task in parallel and has an option to distribute the work of the Task among the threads in a team. These threads are scheduled on the available cores either by operating system or as specified by the programmer. At the end of a parallel function, there is an implied barrier that forces all threads to wait until the work inside the region has been completed. Only the initial thread continues execution after the end of the parallel function. The thread that starts the parallel construct becomes the master of the new team. Each thread in the team is assigned a unique thread id to identify it. They range from zero (for the master thread) up to one less than the number of threads within the team. This function has also four variants. *Parallel (Task i)* {Basic}, *Parallel (Task i, num-threads)* {for defining max parallel threads}, *Parallel (Task i, core list)* {for core specification} and **p parallel (Task i, core, *p)* {for managing the arguments with core specification}

Concurrent: This function is used to specify the number of independent tasks that should be executed in concurrent fashion on available cores. These may be same tasks with different data set or different tasks. When the Tasks are executed defined in this function, a set of threads equal or greater to the number of tasks defined in concurrent function is created such that each task is associated with a thread or threads. These threads are scheduled on the available cores either by operating system or specified by the programmer. In other words, this function is an extension and fusion of serial and parallel functions. All the independent tasks defined in concurrent functions are executed in parallel where as each thread is being executed either serially or in parallel. This function has also three variants. *Concurrent (Task i, Task j,Task N)* {Basic}, *Concurrent (Task i, core, Task j, core,)* {for core specification} and *Concurrent (Task i, core, *p, Task j, core, *p)* {for managing the arguments with core specification}.

5. CONCURRENT MATRIX ALGORITHM

We have selected a standard and basic matrix multiplication algorithm in which the product of a $(m \times p)$ matrix A with a $(p \times n)$ matrix B is a $(m \times n)$ matrix denoted C such that

$$C_{i,j} = \sum_{k=1}^p A_{i,k} B_{k,j}$$

Where $1 \leq i \leq m$ is the row index and $1 \leq j \leq n$ is the column index. This algorithm is implemented using two different approaches. The first is the standard parallel approach using OpenMP. The other is in C++ using the concurrent function of SPC³ PM. Pseudo code for both of the algorithms are shown in table 1.

In OpenMP implementation the basic computations of addition and multiplication are placed within the three nested 'for' loops. The outer most is parallelized using OpenMP keyword 'pragma omp parallel for'. The row level distribution of matrices is followed. The matrix is divided into set of rows equal number of parallel threads defined by the variable 'core' such that each row set is computed on a single core.

For SPC³ PM using concurrent function, a Task is defined having the basic algorithm implementation. The idea is to execute this task concurrently on different cores with different data set. Every Task has its own private data variables defined in a structure 'My_Data'. All the private structures are associated with their tasks and initialized accordingly. Using the Concurrent function of SPC³ PM, the required number of concurrent tasks are initialized and executed.

Matrix Multiplication Algorithm OpenMP (Parallel)	Matrix Multiplication Algorithm SPC ³ PM, Concurrent
<pre> Void main (void) { // inintilillig the matrices int A[][],B[][],C[][] int core ; // number of parallel threads omp_set_num_threads(core); // initializing the parallel loop #pragma omp parallel for private(i,j,k) for (i=0; i<n; i++) { for (j=0; j<n; j++) { c[i][j]=0; for (k=0;k<n;k++) { c[i][j]=c[i][j]+ a[i][k]*b[k][j]; } } } } </pre>	<pre> Task(LPVOID) { P_MY_DATA data; data=(P_MY_DATA)lp; for(i=data->val3; i<data->val1; i++) for(j=0; j< data->val2; j++) { for(k=0;k< data->val2 ;k++) c[i][j]=c[i][j]+ a[i][k]*b[k][j]; } } void main (void) { typedef struct My_Data { int val1, val2, val3; int A[][],B[][],C[][] } MY_DATA, *P_MY_DATA[n]; //initialize P_MY_DATA_1; //initialize P_MY_DATA_2; //initialize P_MYDATA_N; concurrent(Task,P_MYDATA_1,Task,P_MY_DATA_2 Task,P_MY_DATA_N); } </pre>

TABLE 1: Parallel Matrix Algorithm for OpenMP and SPC³ Concurrent

6. EXPERIMENTAL SETUP

For the execution of the algorithms we used quad Intel Xeon processor 5500 series based SR1670HV, server systems having 48 cores and dual Intel Xeon processors 5500 series based SR1600UR server systems with 24 cores. Operating systems used are windows server 2003 and 2008. We tested our approach for up to 24 cores with different matrices size varying from 100 x 100 to 10000 x 10000 elements.

7. PERFORMANCE EVOLUTION

The following tables 2 to 5 show the execution time in seconds for each of two approaches, OpenMP and SPC³ PM Concurrent with different sizes of matrices for 4, 8, 12 and 24 parallel / concurrent threads respectively.

Matrix Size	Number of Parallel Threads	Execution Time (Sec)	
		OpenMP (Parallel)	SPC ³ PM, Concurrent
100 X 100	4	1	1
1000 X 1000		3	3
2000 X 2000		36	23
3000 X 3000		162	85
4000 X 4000		404	202
5000 X 5000		738	396
6000 X 6000		1244	682
7000 X 7000		2078	1086
8000 X 8000		3093	1619
9000 X 9000		4558	2303
10000 X 10000		5425	3161

TABLE 2: Execution Time (Sec) for parallel matrix multiplication using OpenMP and SPC³ PM Concurrent for 4 parallel threads

Matrix Size	Number of Parallel Threads	Execution Time (Sec)	
		OpenMP (Parallel)	SPC ³ PM, Concurrent
100 X 100	8	1	1
1000 X 1000		2	1
2000 X 2000		25	12
3000 X 3000		83	44
4000 X 4000		212	104
5000 X 5000		433	204
6000 X 6000		703	351
7000 X 7000		1099	559
8000 X 8000		1742	833
9000 X 9000		2503	1186
10000 X 10000		3276	1626

TABLE 3: Execution Time (Sec) for parallel matrix multiplication using OpenMP and SPC³ PM Concurrent for 8 parallel threads

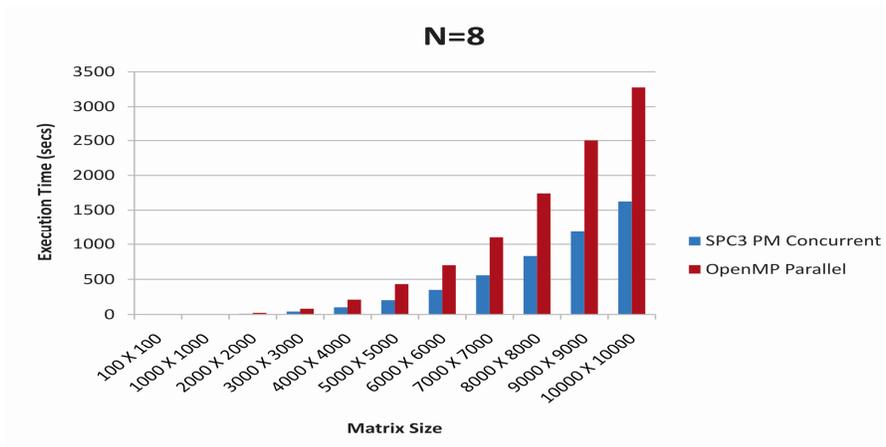
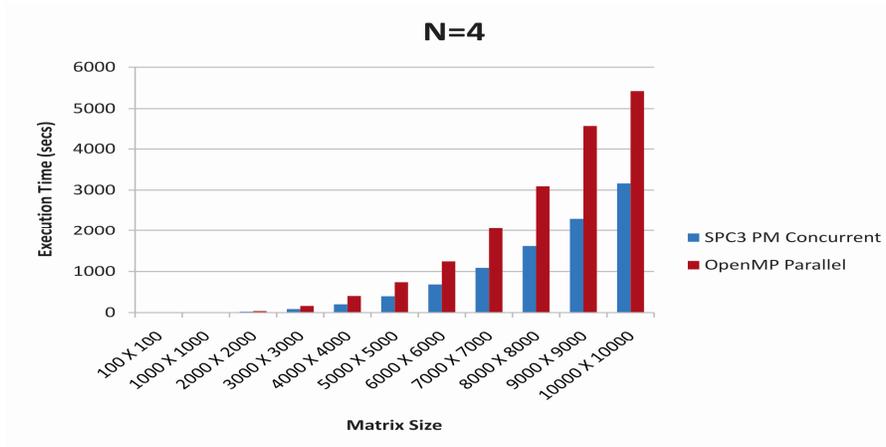
Matrix Size	Number of Parallel Threads	Execution Time (Sec)	
		OpenMP (Parallel)	SPC ³ PM, Concurrent
100 X 100	12	1	1
1000 X 1000		1	1
2000 X 2000		18	8
3000 X 3000		65	30
4000 X 4000		164	72
5000 X 5000		330	141
6000 X 6000		573	242
7000 X 7000		842	384
8000 X 8000		1291	575
9000 X 9000		1799	816
10000 X 10000		2664	1126

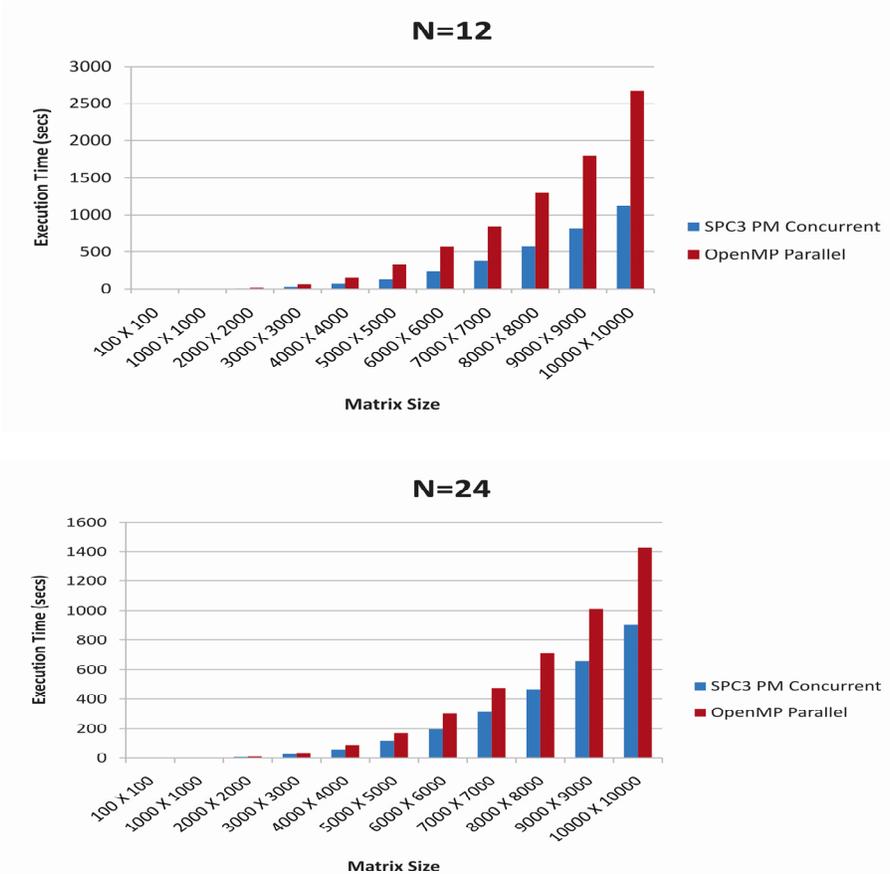
TABLE 4: Execution Time (Sec) for parallel matrix multiplication using OpenMP and SPC³ PM Concurrent for 12 parallel threads

Matrix Size	Number of Parallel Threads	Execution Time (Sec)	
		OpenMP (Parallel)	SPC ³ PM, Concurrent
100 X 100	24	1	1
1000 X 1000		1	1
2000 X 2000		10	7
3000 X 3000		36	26
4000 X 4000		86	58
5000 X 5000		171	113
6000 X 6000		303	197
7000 X 7000		476	314
8000 X 8000		710	467
9000 X 9000		1011	661
10000 X 10000		1431	905

TABLE 5: Execution Time (Sec) for parallel matrix multiplication using OpenMP and SPC³ PM Concurrent for 24 parallel threads

The figures 2-5 compare the execution time based on table 2-5 for each of the two approaches, OpenMP and SPC³ PM Concurrent with different sizes of matrices for 4, 8, 12 and 24 parallel / concurrent threads respectively.





FIGURES 2-5: comparisons of the execution time for each of the two approaches, OpenMP and SPC³ PM Concurrent

Based on table 1, the following table 6 shows the speedup obtained for the SPC³ PM concurrent function for different matrices size and number of concurrent threads. Figure 6 shows the comparison of speedup based on table 2 for SPC³ PM concurrent function with 4,8,12 and 24 concurrent threads.

Matrix Size	Time (Sec) Serial C++	Speedup			
		SPC ³ Concurrent N=4	SPC ³ Concurrent N=8	SPC ³ Concurrent N=12	SPC ³ Concurrent N=24
100 X 100	1	1.00	1.00	1.00	1.00
1000 X 1000	13	4.33	13.00	13.00	13.00
2000 X 2000	154	6.70	12.83	19.25	22.00
3000 X 3000	551	6.48	12.52	18.37	21.19
4000 X 4000	1374	6.80	13.21	19.08	23.69
5000 X 5000	2707	6.84	13.27	19.20	23.96
6000 X 6000	4642	6.81	13.23	19.18	23.56
7000 X 7000	7285	6.71	13.03	18.97	23.20
8000 X 8000	10925	6.75	13.12	19.00	23.39
9000 X 9000	15606	6.78	13.16	19.13	23.61
10000 X 10000	21497	6.80	13.22	19.09	23.75

TABLE 6: Speedup obtained for the SPC³ PM concurrent function with different number of concurrent threads and matrices size

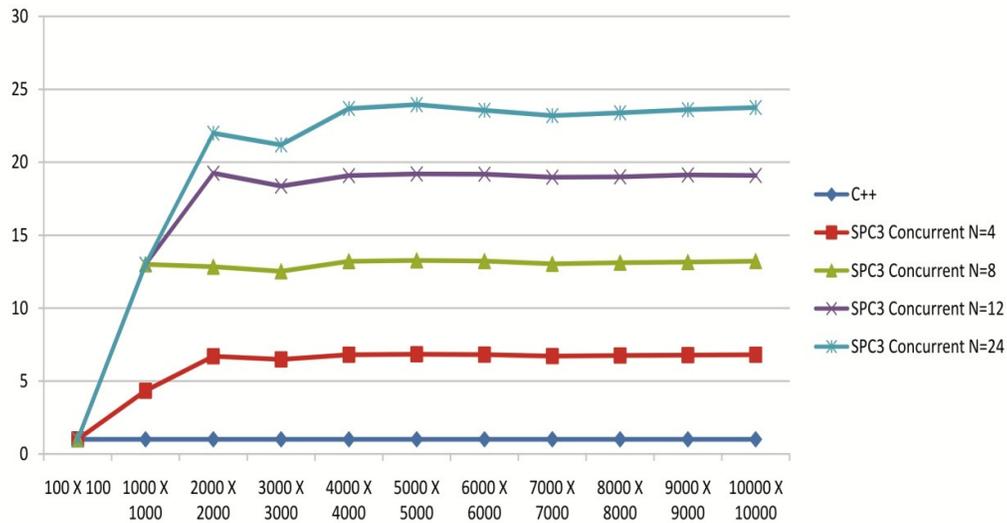


FIGURE 6: Comparison of speedup based on table 6 for SPC³ PM concurrent function with 4, 8, 12 and 24 concurrent threads.

From the tables 2 to 5 and figures 2 to 5 it can clearly seen that concurrent function of SPC³ PM takes much lesser execution time than that of the OpenMP. Another observation can be made that this performance gain increases as the either number of cores increases or the problem size increases. From Table 6 and figure 6 it can be clearly observed that the speedup obtained using SPC³ PM are uniform and it is not affected by the problem size. These results proof the enhanced and scalable implementation of a concurrent matrix multiplication using SPC³ PM.

The proposed concurrent matrix multiplication using SPC³ PM provides an efficient, simple and scalable parallel matrix multiplication implementation for multi-core processors. The other parallel implementations of matrix multiplication proposed for SMPs are usually based on functional parallelism. The functional parallelism exploited alone with in a program cannot make most of multi-core processors. For maximum utilization of multi-core processors, program or programs in execution should be able to execute concurrently on available cores. The Concurrent function of SPC³ PM provides the facility of concurrent execution of program or program on the available cores to maximize the utilization of multi-core processors. Besides, the existing parallel matrix multiplication algorithms for SMPs are very specific and difficult for common programmer to understand as they require detailed related subject knowledge whereas the proposed concurrent implementation of parallel matrix multiplication using SPC³ PM is based on a simple and fundamental algorithm of matrix multiplication and does not requires any detailed related subject knowledge. In comparison to the attempts which have been made to solve matrix multiplication using data parallel or concurrent approaches on cell or GPUs, the proposed approach is more generic, architectural independent and found suitable for general purpose multi-core processors.

8. CONCLUSION AND FUTURE WORK

The results from this study show that the SPC³ PM provides a simpler, effective and scalable way to perform matrix multiplication on multi-core processors. With the concurrent function of SPC³ PM, the programmer can execute a simple and standard matrix multiplication algorithm concurrently on multi-core processors in much less time than that of standard parallel OpenMP. The proposed approach also shows more scalability, better and uniform speedup and better utilization of available cores than that of OpenMP. This SPC³ PM will be further worked out for the introduction of some more parallel and concurrent functions and synchronizing tools.

9. REFERENCES

- [1] N. Vachharajani, Y. Zhang and T. Jablin, "Revisiting the sequential programming model for the multicore era", *IEEE MICRO*, Jan - Feb 2008.
- [2] M. D. McCool, "Scalable Programming Models for Massively Multicore Processors", *Proceedings of the IEEE*, vol. 96(5), 2008.
- [3] G. Goumas, "Performance evaluation of the sparse matrix-vector multiplication on modern architectures", *Journal of Supercomputing*, pp. 1-42, Nov. 2008.
- [4] R. Vuduc, H. Moon, "Fast sparse matrix-vector multiplication by exploiting variable block structure", *Lecture notes in computer science*, vol. 3726, pp. 807-816, 2005.
- [5] H. T. Kung, C. E. Leiserson, "Algorithms for VLSI processor arrays"; in "Introduction to VLSI Systems", Addison-Wesley, 1979.
- [6] G. C. Fox, S. W. Otto and A. J. G. Hey, "Matrix algorithms on a hypercube I: Matrix multiplication", *Parallel Computing*, vol. 4(1), pp.17-31, 1987.
- [7] R. A. van de Geijn, J. Watts, "SUMMA_ Scalable Universal Matrix Multiplication Algorithm", *TECHREPORT*, 1997.
- [8] A. Ziad, M. Alqadi and M. M. El Emary, "Performance Analysis and Evaluation of Parallel Matrix Multiplication Algorithms", *World Applied Sciences Journal*, vol. 5 (2), pp. 211-214, 2008.
- [9] Z. Alqadi and A. Abu-Jazzar, "Analysis of program methods used for optimizing matrix Multiplication", *Journal of Engineering*, vol. 15(1), pp. 73-78, 2005.
- [10] J. Choi, "Fast Scalable Universal Matrix Multiplication Algorithm on Distributed-Memory Concurrent Computers" in Proceeding of *11th International Symposium on Parallel Processing IPPS '97 IEEE*, 1997.
- [11] P. Alonso, R. Reddy, A. Lastovetsky, "Experimental Study of Six Different Implementations of Parallel Matrix Multiplication on Heterogeneous Computational Clusters of Multi-core Processors" in Proceedings of *Parallel, Distributed and Network-Based Processing (PDP)*, Pisa, Feb. 2010.
- [12] R. C. Agarwal, S. M. Balle, F. G. Gustavson, M. Joshi, P. Palkar, "A three-dimensional approach to parallel matrix multiplication", *IBM Journal of Research and Development*, vol. 39(5). pp. 575, 1995.
- [13] A. Buluc, J. R. Gilbert, "Challenges and Advances in Parallel Sparse Matrix-Matrix Multiplication", in proceedings of *37th International Conference on Parallel Processing, ICPP '08*, Portland, Sep 2008.
- [14] L. Buatois, Caumon, G. Lévy, "Concurrent number cruncher: An efficient sparse linear solver on the GPU", in Proceedings of the *High-Performance Computation Conference (HPCC)*, Springer LNCS, 2007.
- [15] S. Sengupta, M. Harris, Y. Zhang, J.D. Owens, "Scan primitives for GPU computing". In *Proceedings of Graphics Hardware*, Aug. 2007.
- [16] J. A. Stratton, S. S. Stone, Hwu, "M-CUDA: An efficient implementation of CUDA kernels on multicores", *IMPACT Technical Report 08-01*, University of Illinois at Urbana-Champaign, 2008.
- [17] K. Fatahalian, J. Sugerman, P. Hanrahan, "Understanding the efficiency of GPU algorithms for matrix-matrix multiplication" in Proceeding of the *conference on Graphics hardware ACM SIGGRAPH/EUROGRAPHICS HWWS '04*, 2004.

- [18] J. Bolz, I. Farmer, E. Grinspun, P. Schröder, "Sparse matrix solvers on the GPU: conjugate gradients and multigrid", *ACM Transactions on Graphics (TOG)*, vol. 22(3), 2003.
- [19] S. Ohshima, K. Kise, T. Katagiri and T. Yuba, "Parallel Processing of Matrix Multiplication in a CPU and GPU Heterogeneous Environment", *High Performance Computing for Computational Science – VECPAR, 2006*.

Radical Data Compression Algorithm Using Factorization

Zirra Peter Buba

*Department of Mathematical Sciences
Adamawa State University
Mubi, 650221, Nigeria.*

zirrapeter@yahoo.com

Gregory Maksha Wajiga

*Department of Mathematics and Computer Science
Federal University of Technology
Yola, 640284, Nigeria.*

gwajiga@gmail.com

Abstract

This work deals with encoding algorithm that conveys a message that generates a “compressed” form with fewer characters only understood through decoding the encoded data which reconstructs the original message. The proposed factorization techniques in conjunction with lossless method were adopted in compression and decompression of data for exploiting the size of memory, thereby decreasing the cost of the communications. The proposed algorithms shade the data from the eyes of the cryptanalysts during the data storage or transmission.

Keywords: Data Compression, Cryptography, Lossless, Algorithm

1. INTRODUCTION

In this paper the generic term message is used for the object to be compressed. Compression consists of two components, an encoding algorithm that takes a message and generates a “compressed” representation (with fewer characters), and a decoding algorithm that reconstructs the original message from the compressed representation.

There are "lossless" and "lossy" forms of data compression [1]. Lossless (reversible) data compression is used when the data has to be uncompressed exactly as it was before compression. Text files are stored using lossless techniques, since losing a single character can in the worst case make the text dangerously misleading. Archival storage of master sources for images, video, and audio data generally needs to be lossless as well. However, there are strict limits to the amount of compression that can be obtained with lossless compression.

Lossy (irreversible) compression, in contrast, works on the assumption that the data doesn't have to be stored perfectly. Much information can be simply thrown away from images, video, and audio data, and when uncompressed, such data will still be of acceptable quality. Compression ratios can be an order of magnitude greater than those available from lossless methods [2].

The proposed factorization techniques of data compression uses lossless method of compressing and decompressing of data to exploit the size of memory, thereby decreasing the cost of the communications and shading the data from the eyes of the cryptanalysts during the data storage or transmission.

2. DATA COMPRESSION

Data compression is defined as the reduction of the volume of a data file without loss of information [3]. As pointed out by [4] and [5], data compression aims to condense the data in order to reduce the size of a data file.

Data compression has important application in the areas of data transmission and data storage [6]. Many data processing applications require storage of large volumes of data, and the number of such applications is constantly increasing as the use of computers extends to new disciplines.

At the same time, the proliferation of computer communication networks is resulting in massive transfer of data over communication links. Compressing data to be stored or transmitted reduces storage and communication costs. When the amount of data to be transmitted is reduced, the effect is that of increasing the capacity of the communication channel [7]. Similarly, compressing a file to half of its original size is equivalent to doubling the capacity of the storage medium. It may then become feasible to store the data at a higher, thus faster, level of the storage hierarchy and reduce the load on the input/output channels of the computer system.

3. THEORETICAL FRAME WORK

The main feature of the radical data compression algorithm takes as an input a plaintext, compress the string of the characters and produce an output in ciphertext. To decompress, the ciphertext requires a decompression key [8]. The decompression key is transmitted to the receiver through different media such as email or Short Message Service (SMS) to strengthen its security. The whole process is depicted in Figure 1.

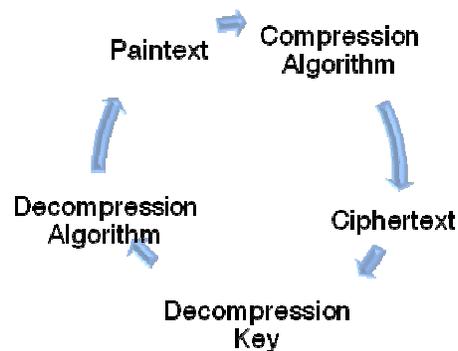


FIGURE 1: Compression and Decompression Process

4. THE PROPOSED RADICAL COMPRESSION ALGORITHMS

4.1 Compression Algorithm

The compression of the data is achieved through the following:

- i. Each character or symbol in the message such as number, white space and punctuation are assigned a numerical value.
- ii. The encoding scheme in Table 1 is employed.

4.1.1 Compression Key

- a. Serialization of string.
- b. Write down character of the input string without repetition.
- c. The encoding scheme in Table 1.

Symbol	Count	Locations	Product	Sum	Maximum Location
S_j	C_j	ℓ_{jk}	$\alpha_j = \prod_{k=1}^{C_j} \ell_{jk}$	$\beta_j = \sum_{k=1}^{C_j} \ell_{jk}$	$h_j = \max(\ell_{jk})$

TABLE 1: Encoding Scheme

Illustrative example 1: If we are going to compress the random message “*WHO IS PROMISSING WHO*”, we would have an encoded message that is given in *Table 3* using *Table 2*.

S/No.	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Symbol	W	H	O		I	S		P	R	O	M	I	S	I	N	G		W	H	O

TABLE 2: Serialization of plaintext

Symbol	S_j	White space	W	H	O	I	S	P	R	M	N	G
Location	ℓ_{jk}	4,7,17	1,18	2,19	3,10,20	5,12,14	6,13	8	9	11	15	16
Character Count	C_j	3	2	2	3	3	2	1	1	1	1	1
Sum	$\beta_j = \sum_{k=1}^{C_j} \ell_{jk}$	28	19	21	33	31	19	8	9	11	15	16
Product	$\alpha_j = \prod_{k=1}^{C_j} \ell_{jk}$	476	18	38	600	840	78	8	9	11	15	16
Maximum Location	$h_j = \max(\ell_{jk})$	17	18	19	20	14	13	8	9	11	15	16

TABLE 3: Encoded Symbols

Hence the compressed data is *WHOISPRMNG* and a white space.

We can easily find where characters ‘white space’, ‘G’, ‘H’, ‘I’, ‘M’, ‘N’, and so on occurred in the message by exploring the compression algorithm in section 4.2 *visa vice* Table 3.

4.2 Decompression Algorithm

The algorithm, which describes each step in compression process, is listed in the Algorithm_Decompression (input: symbol) below

Algorithm_Decompression (input: symbol)

- Step 1: Read the product (α_j), sum (β_j) and the maximum occurrence of the character (h_j) as produced in Table 3.

Step 2: Begin
 Step 3: If $C_j = 1$, then output h_j
 Step 4: Else
 Step 5: Compute C_j factor(s) of $\alpha_j \leq h_j$ such that the factor(s) satisfy the value α_j and β_j
 Step 6: Endif
 Step 7: If (3) or (5) is true, then output the character S_j at location(s) (E_{jk})
 Step 8: Else goto step 5
 Step 9: Repeat steps (1) to (8) for C_j of S_j until $S_j = 0$
 Step 10: EndBegin
 EndAlgorith_Decompression

4.2.1 Decompression key

E_j factors of $\alpha_j \leq h_j$ such that the factor(s) satisfy the values of α_j and β_j in Table 3 or if $E_j = 1$, write h_j .

This key is transmitted through a different channel (email) to the recipient in order to be able to decompress the characters.

Illustrative example 2: To decode the character symbol 'white space':

From the Algorithm_Decompression (input: symbol), the computed value of $C_j = 3$, $\alpha_j = 476$, $\beta_j = 28$, and $h_j = 17$, as depicted in Table 3.

- a) We compute the factors of $\alpha_j \leq h_j$, which are 1, 2, 4, 7, 14 and 17
- b) From (a) we take C_j factors of $\alpha_j \leq h_j$ that satisfied the value of α_j and h_j . These are 4, 7 and 17.

Since the value of α_j and h_j are satisfied, a blank space is created at locations 4, 7 and 17.

Taking the second character symbol G: the value $C_j = 1$, $\alpha_j = 16$, $\beta_j = 16$, and $h_j = 1$

- c) We compute the factors of $\alpha_j \leq h_j$, which are 1 and 16.
- d) From (c) we take C_j factors of $\alpha_j \leq h_j$ that satisfied the value of α_j and h_j and that is $h_j = 16$

Since the value of α_j and h_j are satisfied, a character G is written at locations 16 only.

Similarly, to decode the character symbol "H", we take the value of $C_j = 2$, $\alpha_j = 38$, $\beta_j = 21$, and $h_j = 19$, from the Algorithm_Decompression (input: symbol).

- e) We compute the factors of $\alpha_j \leq h_j$, which are 1, 2 and 19
- f) From (e) we take C_j factors of $\alpha_j \leq h_j$ that satisfied the value of α_j and h_j . These are 2 and 19

Since the value of α_j and h_j are satisfied, the character "H" is printed at locations 2.

This is continued until all the characters of the plaintext have been recovered to its original plaintext "WHO IS PROMISING WHO" as depicted in Table 2.

5. RESULTS AND DISCUSSION

From Table 3, and the results of illustrative example 2, showed how efficient and practicable [9] the proposed data compression scheme in Table 1 and decompression algorithms in section 4.2 in handling any kinds of plaintext in an unsecure channel and how it increases storage capacity for faster data processing [3].

The meaningless results in Table 3 is in line with the proposed data compression and decompression algorithms which helped to conceal the content of sensitive information from the terrorist [9,10]. These meant that the proposed algorithms has proved to prevent the hacker from gaining insight of what the content of information was all about while in either transit or store.

The result of the study has also successfully reduced the large volume of the plaintext that contained the same information as the original string in Table 2 but whose length was small as possible [3,4,7]. These proved that it may then become feasible to store the data at a higher, faster, level of the storage hierarchy [5] and reduce the load on the input/output channels of the computer systems which is the main goal of data compression techniques [11,12].

The results of the study also shown that one key is used for encoding the data and another different key were used for decoding of the data. The results revealed that the receiver obtained these key from the sender secretly through either SMS or Fax machine. The strength of cryptography systems in the computer era rests on the secrecy of the key, not on the algorithm [8]. It is clear that our information is not easily visible to an unauthorized person without a decoding key. The concealment of the key rather than the encryption technique is one of the most important features of a strong cryptography system [10].

6. CONCLUSION

The new data compression transforms a string of characters into a new string (encoded) using compression key and reverses the encoded string using decompression key. The encoded string contains the same information as the original string but whose length is small as possible. These shade the information contents from the eyes of the intruders, but thereby increasing the capacity of the communication channel. Therefore, the algorithms will be found useful by organizations that deal with sensitive documents.

7. REFERENCES

- [1] W.S. Steven. *The Scientist and Engineer's Guide To Digital Signal Processing*. California: Technical Publishing, 2007.
- [2] E. B. Guy. "Introduction to Data Compression". Internet: www.eecs.harvard.edu/~michaelm/CS222/compression.pdf, 2010 [Jan. 24, 2011].
- [3] A.G. Fraser. "Data Compression and Automatic Programming". Internet: www.comjnl.oxfordjournals.org, 2010 [Oct. 16, 2010].
- [4] N. Ziviani, E. Moura, G. Navarro, and R. Baeza-Yates. "Compression: A key for next generation text retrieval systems". *IEEE Computer Society*, 2000, 33(11), pp. 37- 44. 2000
- [5] A. Kattan. "Universal lossless compression technique with built in encryption". M. Sc. Thesis, University of Essex, UK., 2006.
- [6] S. Khalid. *Introduction to Data Compression (2nd ed.)* New York : Morgan Kaufmann Publishers Inc., 2000, pp 151-218.
- [7] E.J.D. Garba and S.E Adewumi. "A Cryptosystems Algorithm Using Systems of Nonlinear Equations". *Iranian Journal of Information Science And Technology*, 2003, 1(1), pp. 43-55.
- [8] M. Milenkovic. *Operating System: Concepts and Design*, New York: McGrew-Hill, Inc., 1992.

- [9] V. Singla, R. Singla, and S. Gupta. "Data Compression Modelling: Huffman and Arithmetic" *International Journal of The Computer, the Internet and Management*, 2008, 16(3), pp 64- 68
- [10] *Cryptography FAQ (03/10: Basic Cryptology)*, "3.5. What are some properties satisfied by every strong cryptosystem?" [cited 23 August 2006]; Available: <http://www.faqs.org/faqs/cryptographyfaq/part03/index.html>.
- [11] A. Hauter, M.V.C., R. Ramanathan. *Compression and Encryption. CSI 801 Project Fall 1995*. December 7, 1995 [cited 10 March2006]; Available: <http://www.science.gmu.edu/~mchacko/csi801/proj-ckv.html>.
- [12] *How does cryptography work*. 12 March 2006 [cited 12 March 2006]; Available: <http://www.pgpi.org/doc/pgpintro>.

Cryptographic Algorithms for Secure Data Communication

Zirra Peter Buba

*Department of Mathematical Sciences
Adamawa State University
Mubi, 650221, Nigeria.*

zirrapeter@yahoo.com

Gregory Maksha Wajiga

*Department of Mathematics and Computer Science
Federal University of Technology
Yola, 640284, Nigeria.*

gwajiga@gmail.com

Abstract

Personal privacy is of utmost importance in the global networked world. One of the best tools to help people safeguard their personal information is the use of cryptography. In this paper we present new cryptographic algorithms that employ the use of asymmetric keys. The proposed algorithms encipher message into nonlinear equations using public key and decipher by the intended party using private key. If a third party intercepted the message, it will be difficult to decipher it due to the multilevel ciphers of the proposed application.

Keywords: Cryptographic Algorithm, Asymmetric key, Communication, Nonlinear System

1. INTRODUCTION

Some vital information that are disseminated within an office, across offices, between branches, of an organization and other external bodies and establishments at times get into the hands of the unauthorized persons who may tamper with the contents of the information. And if no security measures are taken, there is no doubt that such data and other sensitive information will be exposed to threats such as impersonation, insecrecy, corruption, repudiation, break-in or denial of services [1,2] that may cause serious danger on the individual or organization.

A secure system should maintain the integrity, availability, and privacy of data [3]. Data integrity usually means protection from unauthorized modification, resistance to penetration and protection from undetected modification.

Therefore, algorithms which help prevent interception, modification, penetration, disclosure and enhance data/information security are now of primary importance. This paper suggests new methods for secured means of communication over unsecure channel. This is to ensuring that the intruders do not have access to the plaintext without a secret key.

2. PRELIMINARIES

2.1 Cryptography

One way to strengthen security [4,5] in computer systems is to encrypt sensitive records and messages in transit and in storage. The basic model of a cryptographic system is illustrated in Figure 1. The original unenciphered text is called the plaintext. The act of converting a plain text message to its ciphertext form is called enciphering [6]. In its cipher form, a message cannot be read by anyone but the intended receiver. Reversing that act (i.e., ciphertext form to plain text message) is deciphering. Enciphering and deciphering are more commonly referred to as encryption and decryption, respectively.

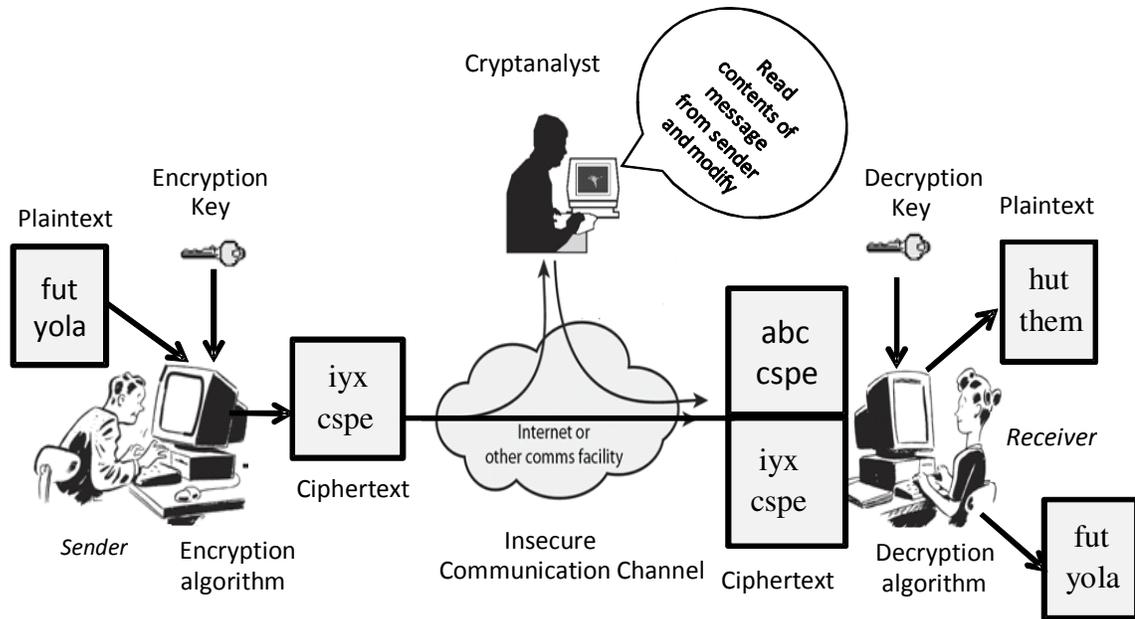


FIGURE 1: Encryption and Decryption Process

2.2 Modern Key-Based Cryptographic Techniques

There are several modern key-based cryptographic techniques. The two common key based encryption techniques are symmetric and asymmetric key cryptography [7].

In symmetric key cryptography, the same key is used for both encryption and decryption. In asymmetric schemes, one key is used for encryption and another is used for decryption [8]. The increased confidence in the integrity of systems that use encryption is based on the notion that ciphertext should be very difficult to decipher without knowledge of the key [3].

2.3 Types of Attacks

There are several types of code-breaking attacks. The first, known as the ciphertext attack, occurs when an adversary comes into possession of only the ciphertext [3]. The known plaintext problem occurs when the intruder has some matched portions of the ciphertext and the plaintext [9]. The most dangerous is the chosen plaintext problem, in which the attacker has the ability to encrypt pieces of plaintext at will. Brute-force is the ultimate attack on a cipher, by which all possible keys are successively tested until the correct one is encountered [9]. Codebook attacks are attacks that take advantage of the property by which a given block of plaintext is always encrypted to the same block of ciphertext as long as the same key is used. A "man-in-the-middle" attack is an attack that is placed by an active attacker who can listen to the communication between two entities and can also change the contents of this communication. While performing this attack, the attacker pretends to be one of the parties in front of the other party.

2.4. Types of Encryption Algorithms

Currently, there are several kinds of key based encryption software in the market categorized by their function and target groups. The most common key based encryption techniques are given [7,10,11] as follows:

- i. The Caesar Cipher- one of the earliest cryptographic algorithms linked and attributed to Julius Caesar in the Gallic war for its usage. Julius Caesar used cipher to protect the messages to his troops by replacing each letter in a message by the third letter further along in the

alphabet. '*abc*' becomes '*def*'. Obviously, this is extremely weak cryptographic algorithm in today's.

- ii. Data Encryption Standard (DES)- was the first encryption standard to be recommended by National Institute of Standards and Technology (NIST). DES is (64 bits key size with 64 bits block size). Since that time, many attacks and methods recorded the weaknesses of DES, which made it an insecure block cipher.
- iii. International Data Encryption Algorithm (IDEA) is a cryptosystem developed by X. Lai and J. Massey in 1991 to replace the DES standard. It is a symmetric block cipher, operating on 8 bytes at a time, just like DES, but with a key of 128 bits.
- iv. Rivest Cipher 4 (RC4) - a cipher invented by Ron Rivest, a proprietary system by RSA Security, is used in a number of commercial systems like Lotus Notes and secure Netscape.
- v. *Blowfish* is block cipher 64 bits. It takes a variable-length key, ranging from 32 to 448 bits; default 128 bits. Blowfish is unpatented, license-free, and is available free for all uses. Blowfish has variants of 14 rounds or less.
- vi. Unix Crypt - Many Unix systems come supplied with an encryption system called crypt. This routine should never be used for encrypting anything because there exist programs on the net for producing the decrypted text and the key.
- vii. Ron Rivest, Adi Shamir, and Leonard Adleman Algorithm (RSA) - a cipher algorithm based on the concept of a trapdoor function, which is easily calculated, but whose inverse is extremely difficult to calculate. The RSA algorithm is named after Ron Rivest, Adi Shamir and Len Adleman, who invented it in 1977. The RSA algorithm can be used for both public key encryption and digital signatures. Its security is based on the difficulty of factoring large integers.
- viii. Pretty Good Privacy (PGP) - a public key system for encrypting electronic mail using the RSA public key cipher. It encrypts the message using the IDEA cipher with a randomly generated key. It then encrypts the key using the recipient's public key. When the recipient receives the message, PGP uses his private RSA key to decrypt the IDEA key and then uses that IDEA key to decrypt the message.
- ix. Diffie-Hellman (DH)- is the first published public key cryptographic algorithm which allows two users to exchange a secret key over an insecure medium without any prior secrets. The original protocol had two system parameters, p and g . They are both public and may be used by all the users in a system. The Diffie-Hellman key exchange was vulnerable to a man-in-the-middle attack, as Diffie-Hellman key exchange does not authenticate the participants. Parameter p is a prime number and parameter g is an integer less than p , with the following property: for every number n between 1 and $p-1$ inclusive, there is a power k of g such that $n = g^k \pmod p$, where k is kept secret.

3. PROPOSED CRYPTOGRAPHIC ALGORITHM

The proposed encryption algorithm consists of a three level cipher attempt to keep your personal data secure. The first level is achieved through the words compression flowchart in Figure 2, the second level is realized by transforming the compressed words from Figure 2 into systems of nonlinear equations and the third level is achieved by the applying the δ_p encoding principles.

3.1 Encryption and Decryption Keys

Asymmetric encryption key is used which means two keys are shared: a public key to encrypt the message and a private key to decrypt it.

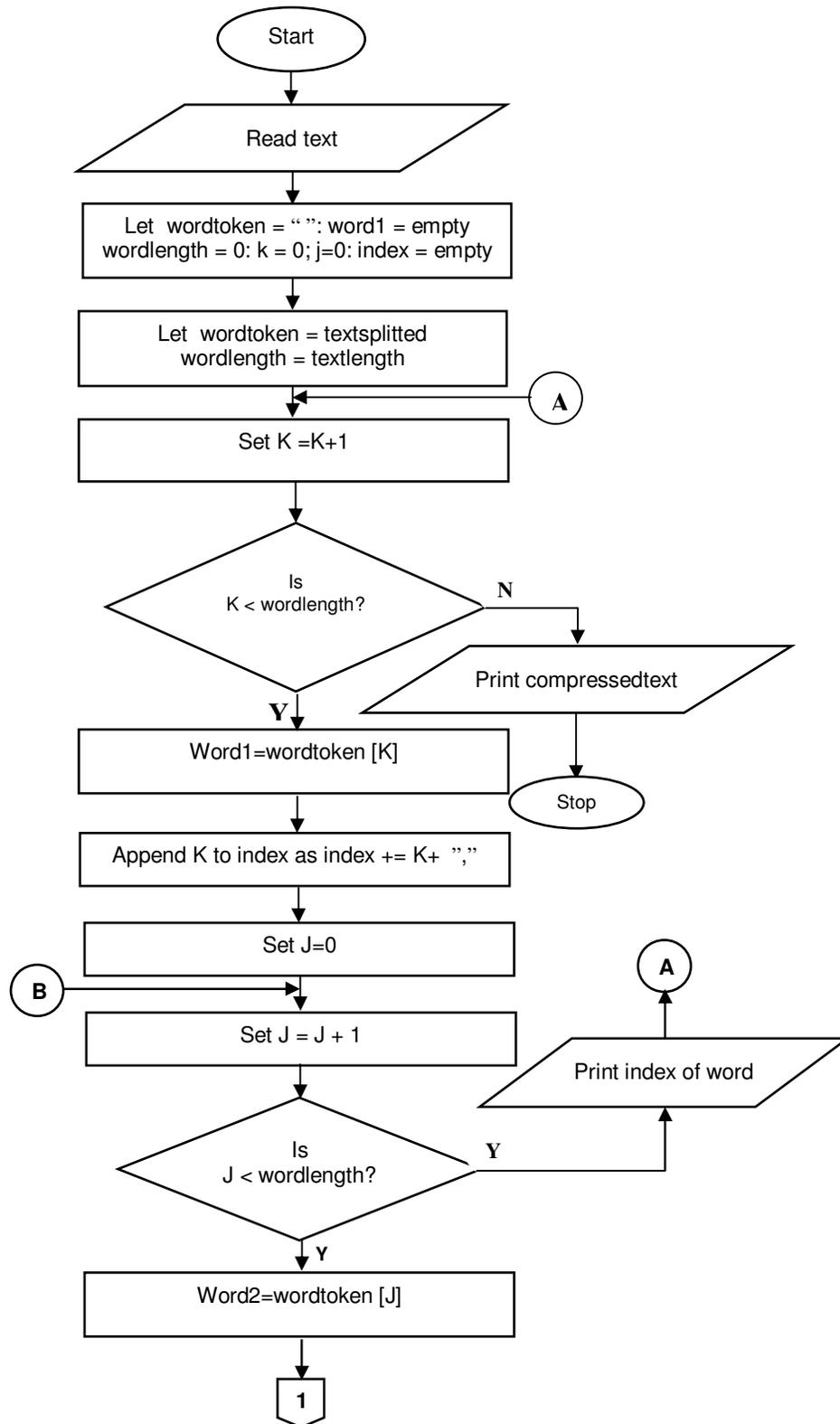
3.2 Encryption and Decryption Rules

- a) The encryption process requires that the sender must have the message and number of words that are left from Figure 2.
- b) To decipher the message in Equation (1b), the intended receiver must be given the following key which contains the associated terms and corresponding code.
 - i. Solutions obtained from the system of nonlinear equations.
 - ii. The δ_p values associated with the variable index in Table 2.
 - iii. The formula $s_k = \sum_{p=1}^k \delta_p$.
 - iv. A lookup character position in Table 1.

	2	3	4	5	6	7
0	space	0	@	P	`	P
1	!	1	A	Q	a	q
2	“	2	B	R	b	r
3	#	3	C	S	c	s
4	\$	4	D	T	d	t
5	%	5	E	U	e	u
6	&	6	F	V	f	v
7	‘	7	G	W	g	w
8	(8	H	X	h	x
9)	9	I	Y	i	y
a	*	:	J	Z	j	z
b	+	;	K	[k	{
c	,	<	L	\	l	}
d	-	=	M]	m	
e	.	>	N	^	n	□
f	/	?	O	_	o	N

TABLE 1: A Lookup Character Position

- c) The key is typically shared by trusted entities and be kept secret from the unauthorized users.
- d) The variable indexes that represent the compressed characters in Equation (1b) are further hid in a file using delta encoding principle before transmission to the intended receiver to further create a state of confusion to the intruders.
- e) A copy of the generated decryption key is saved in a file and sends to recipient email or via any secure communication medium such as telephones Short Message Service (SMS) on or before the message reaches the intended recipient.



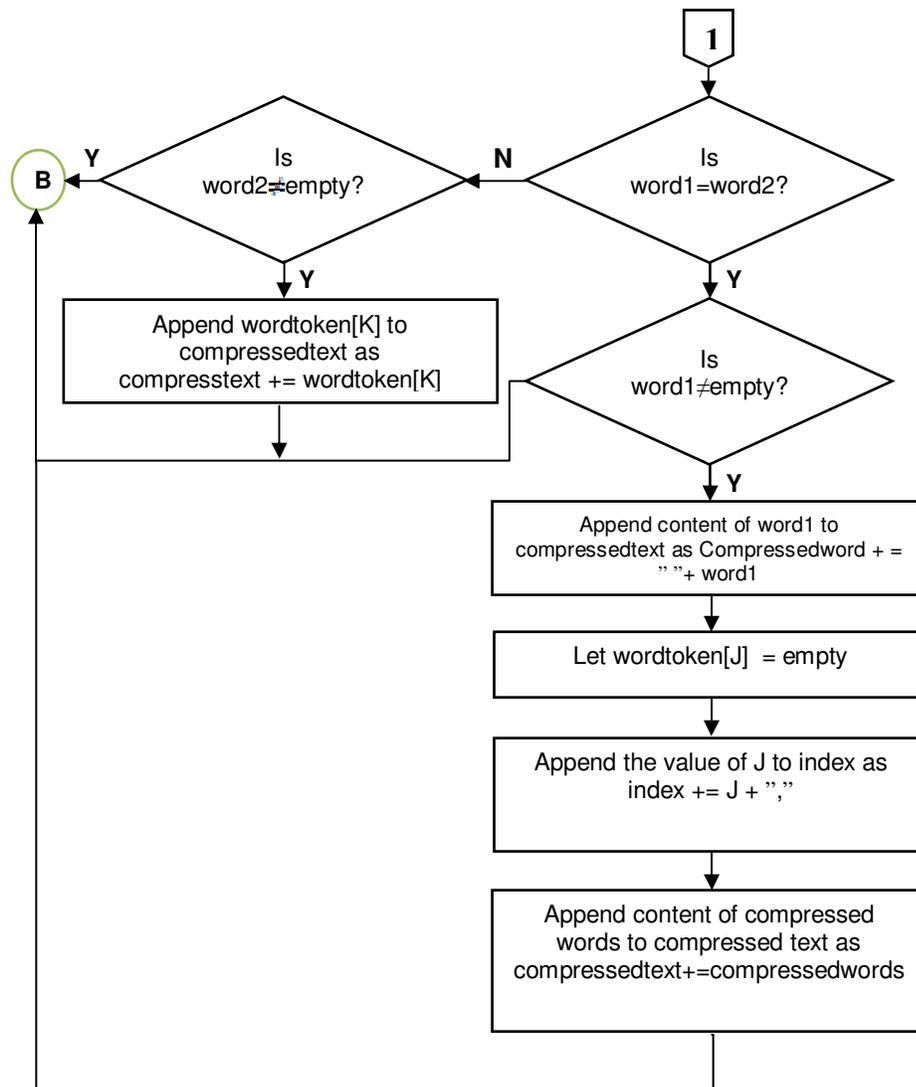
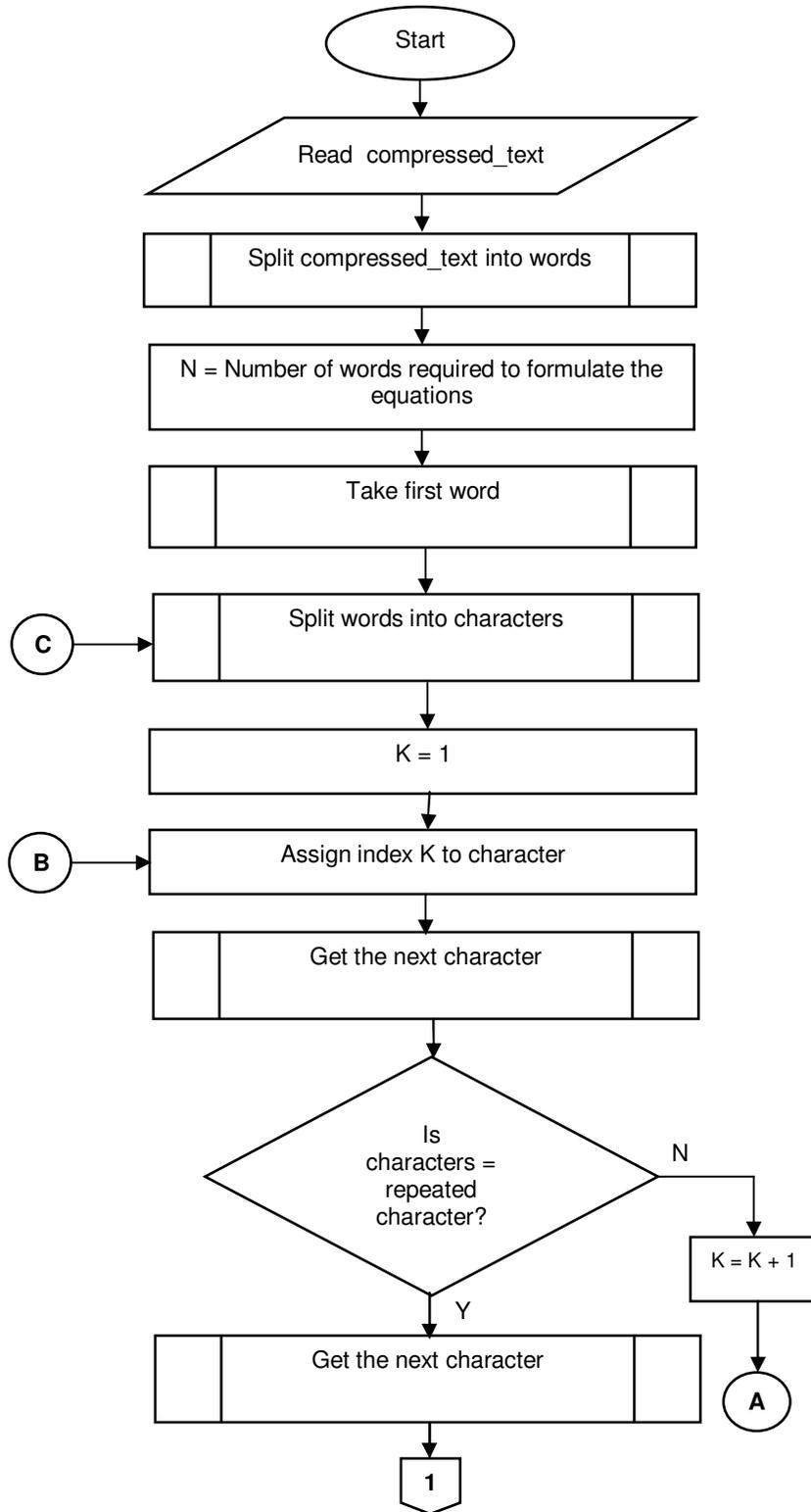


FIGURE 2: Word Compression Flowchart



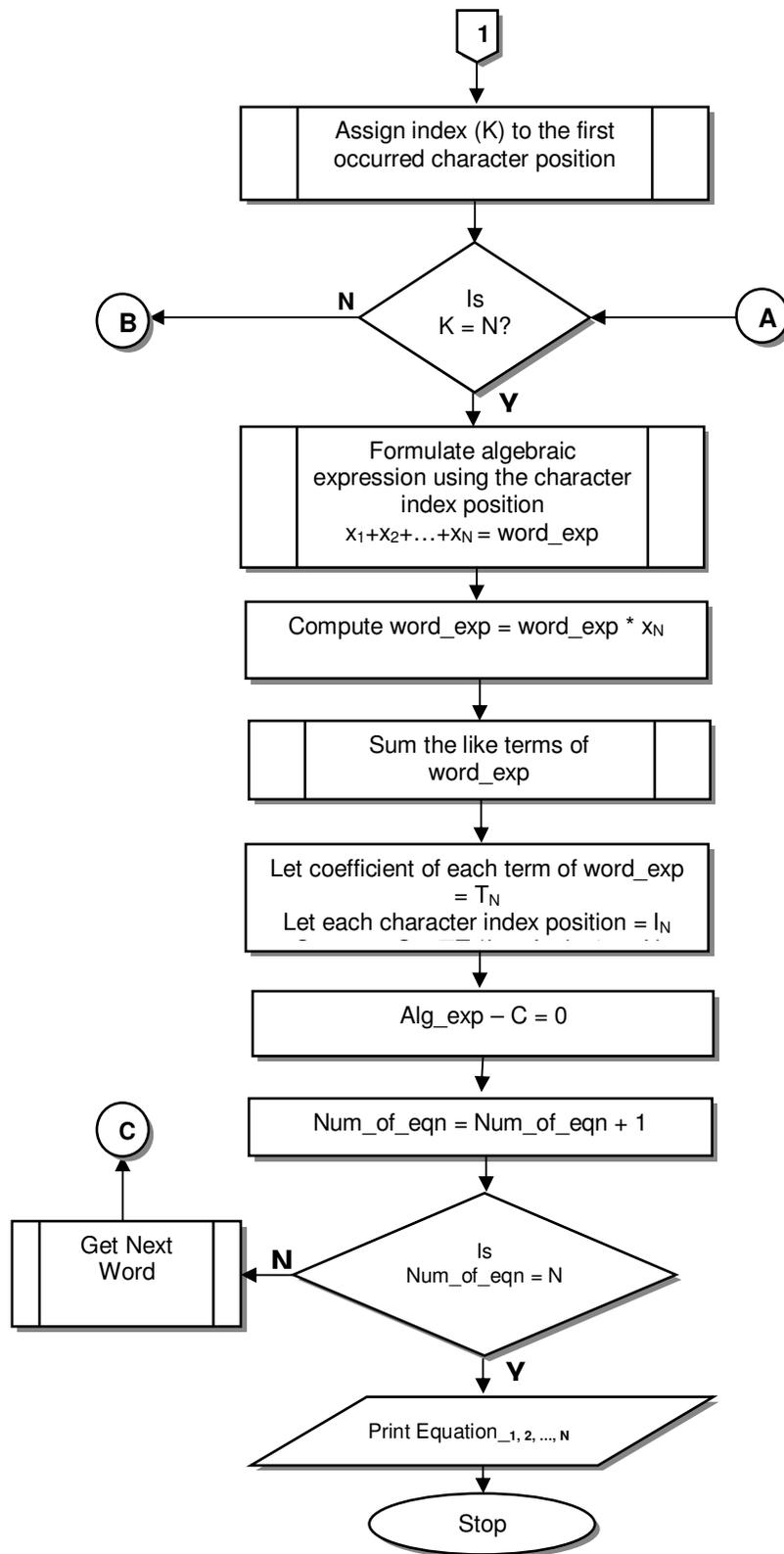


FIGURE 3: Flowchart to Formulate System of Nonlinear Equations

Illustrative example 1: To encrypt the message “who is promising who” we apply the procedure in Figure 2 and 3. The result yields the ciphertext in Equation (1b).

$$\begin{array}{l}
 \text{who:} \quad x_1^2 + x_1x_2 + x_1x_3 = 6 \\
 \text{is:} \quad x_1^2 + x_1x_2 = 3 \\
 \text{promising:} \quad 4x_1^2 + 2x_1x_2 + 3x_1x_3 = 17
 \end{array} \tag{1a}$$

This above representation can be written as:

$$f(x_1, x_2, x_3) = \left. \begin{array}{l} 2x_1^2 + x_1x_2 + x_1x_3 = 6 \\ x_1^2 + x_1x_2 = 3 \\ 4x_1^2 + 2x_1x_2 + 3x_1x_3 = 17 \end{array} \right\} \tag{1b}$$

Equation (1b) becomes systems of nonlinear equations to be transmitted to the recipient in place of the plaintext “who is promising who”

3.3 Decryption Process

The decryption is performed simply by solving the systems of nonlinear equations in Equation (1b) using the below Algorithm 1.

The Newton’s *Algorithm 1*

- 1: guess an approximation solution x_0
- 2: calculate $J(x_p)$ and $f(x_p)$, where $J(x_p)_{ij} = \partial f_i(x) / \partial x_j$, for $1 \leq i, j \leq p$
- 3: solve the linear system $J(x_p)\delta_p = -f(x_p)$
- 4: set $x_{p+1} = x_p + \delta_p$

Algorithm 1: Newton’s Algorithm

For the purpose of this paper,

- a. Explicit computation of the inversion of Jacobi (i.e. $J(x_p)^{-1}$) is avoided as this will involve additional iteration for determining $J(x_p)^{-1}$
- b. Instead we employ the linear system $J(x_p)\delta_p = -f(x_p)$ at the next iterate, thus $x_{p+1} = x_p + \delta_p$.

Illustrative example 2: To decipher the message in Equation (1b):

- i. The receiver must solve and obtain the solutions of the system of nonlinear equations in equation (1) as follows:
- ii.

$$\begin{array}{l}
 J(x_n)\delta_x = -F(x_n) \Rightarrow \\
 \left[\begin{array}{ccc|ccc}
 2x_1 + x_2 + x_3 & x_1 & x_1 & & & \\
 2x_1 + x_2 & x_1 & & & & \\
 8x_1 + 2x_2 + 3x_3 & 2x_1 & 3x_1 & & &
 \end{array} \right] \delta x = - \left[\begin{array}{l} x_1^2 + x_1x_2 + x_1x_3 - 6 \\ x_1^2 + x_1x_2 - 3 \\ 4x_1^2 + 2x_1x_2 + 3x_1x_3 - 17 \end{array} \right]
 \end{array} \tag{2}$$

We take an initial guess of $x_0 = (1, 1, 1)^T$ and substitute in equation (2), to obtain $J(x_0)$ and $f(x_0)$ as in equation (3) in matrix notation.

Hence, from equation (2) we obtain

$$J(x_0)(x_1 - x_0) = -F(x_0) \Rightarrow \begin{bmatrix} 4 & 1 & 1 \\ 3 & 1 & 0 \\ 13 & 2 & 3 \end{bmatrix} \begin{bmatrix} x_1 - 1 \\ x_2 - 1 \\ x_3 - 1 \end{bmatrix} = \begin{bmatrix} 3 \\ 1 \\ 8 \end{bmatrix} \tag{3}$$

Solving equations (3) by Gussaian elimination or any other methods convenient yield the approximate solutions $x_1 = 1, x_2 = 2$ and $x_3 = 3$.

Equation (2) yields equation (4) on input of $x_1 = 1, x_2 = 2$ and $x_3 = 3$. This is a test of convergence of the approximate solutions.

$$J(x_0)(x_2 - x_1) = -F(x_1) \Rightarrow \begin{bmatrix} 7 & 1 & 1 \\ 4 & 1 & 0 \\ 20 & 2 & 3 \end{bmatrix} \begin{bmatrix} x_1 - 1 \\ x_2 - 2 \\ x_3 - 3 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \quad (4)$$

Therefore, the approximate solutions are $x_1 = 1, x_2 = 2$ and $x_3 = 3$.

- iii. Get the lookup character position in Table 1 above.
- iv. Use the values of the variables (solutions) in conjunction with the delta encoded in Table 2, and the formula $s_k = v_{2j} + \sum_{i=1}^k \delta_x$ to finally to obtain the original plaintext in Table 5.

Virtual position of characters (x_2)	76	66	6c		68	71		6d	71	6c	6b	68	71	68	6b	6b
Solution of Equations (v_{2j})	1	2	3		1	2		3	1	3	2	1	2	1	3	1
δ_x	76	-10	6		68	9		6d	04	-05	-01	-03	09	-09	03	-05

TABLE 2: Delta Encoding for “who is promising”

Illustrative example 3: To encrypt the message “Kill all Hippopotamus in the river Mississippi” we apply the procedure in Figure 2 and 3. The result yields the ciphertext in Equation (5b).

Kill	$x_1^2 + x_1x_2 + 2x_1x_3 = 9$	}
all	$2x_1x_3 + x_1x_4 = 10$	
Hippopotamus	$x_1^2 + 2x_1x_2 + x_1x_3 + 2x_1x_4 + x_1x_5 + 3x_1x_6 + 2x_1x_7 = 53$	
in	$x_1x_2 + x_1x_3 = 7$	
(5a)		
the	$x_1^2 + x_1x_6 + x_1x_7 = 14$	
river	$2x_1^2 + 2x_1x_2 + x_1x_7 = 13$	
Mississippi	$4x_1x_2 + x_1x_3 + 4x_1x_4 + 2x_1x_6 = 39$	

Equation (5a) can be written as shown in equation (5b)

$x_1^2 + x_1x_2 + 2x_1x_3 - 9 = 0$	}
$2x_1x_3 + x_1x_4 - 10 = 0$	
$x_1^2 + 2x_1x_2 + x_1x_3 + 2x_1x_4 + x_1x_5 + 3x_1x_6 + 2x_1x_7 - 53 = 0$	
$x_1x_2 + x_1x_3 - 7 = 0$	
(5b)	
$x_1^2 + x_1x_6 + x_1x_7 - 14 = 0$	
$2x_1^2 + 2x_1x_2 + x_1x_7 - 13 = 0$	
$4x_1x_2 + x_1x_3 + 4x_1x_4 + 2x_1x_6 - 39 = 0$	

Equation (5a) becomes the ciphertext to be transmitted to the recipient in place of the plaintext “Kill all Hippopotamus in the river Mississippi”. The variable solutions of Equation (5b) are further concealed in a delta encoding file and then send to the intended receiver as shown in Table 3 to further create confusion to the intruder.

formulated whose variables must not exceed the number of the words) and one indexed word that previously occurred.

Using Figure 3 gives the systems of nonlinear equations of the text “Credit A/c No: 6711645138110 with my VISA debit card 123456789101112” as shown in equation (8)

$$\left. \begin{array}{l}
 \text{Credit} \quad x_1^2 + x_1x_2 + x_1x_3 + x_1x_4 + x_1x_5 + x_1x_6 = 21 \\
 \text{A/c} \quad x_1x_7 + x_1x_8 + x_1x_9 = 24 \\
 \text{No:} \quad x_1^2 + x_1x_2 + x_1x_{10} = 13 \\
 \text{6711645138110} \quad 2x_1x_3 + x_1x_4 + 5x_1x_5 + x_1x_6 + x_1x_7 + x_1x_8 + x_1x_9 + x_1x_{10} = 75 \\
 \text{with} \quad x_1^2 + x_1x_2 + x_1x_3 + x_1x_6 = 14 \\
 \text{my} \quad x_1x_3 + x_1x_4 = 7 \\
 \text{VISA} \quad x_1x_5 + x_1x_6 + 2x_1x_7 = 25 \\
 \text{debit} \quad x_1x_3 + x_1x_4 + x_1x_5 + x_1x_6 + x_1x_8 = 26 \\
 \text{card} \quad x_1x_2 + x_1x_4 + 2x_1x_9 = 24 \\
 \text{123456789101112} \quad x_1^2 + x_1x_3 + x_1x_4 + 5x_1x_5 + x_1x_6 + x_1x_7 + x_1x_8 + x_1x_9 + 3x_1x_{10} = 93
 \end{array} \right\} \quad (8)$$

Equation (8) becomes the enciphertext to be transmitted to the recipient without the words appearing against the Equation (8) in place of the plaintext “Credit A/c No: 6711645138110 with my VISA debit card No: 123456789101112”. As usual, the variable solutions of Equation (8) are further conceal in a file called delta encoding file before transmitting it to the intended receiver as shown in Table 4 to enforce protection of the message.

x_n	42	70	62	60	64	6e	3a	27	5a	44	6e	38	33	33	2c	2c	33	2e	2e	2c
	2b	2f	2c	2c	26	76	64	6e	66	6a	75	51	43	4c	3a	60	62	5a	64	6e
	5a	58	70	60	2c	28	2b	2e	2e	33	33	2f	38	2c	26	2c	2c	2c	28	
v_s	1	2	3	4	5	6	7	8	9	a	1	2	3	4	5	5	3	6	7	5
	8	9	5	5	a	1	5	6	2	3	4	5	6	7	7	4	3	8	5	6
	9	9	2	4	5	a	8	6	7	3	4	9	1	5	a	5	5	5	a	
δ_x	42	2e	-	-	04	0a	3a	-	33	44	2a	-	33	00	-	00	07	-	00	-
	-	04	-	00	-	76	-	0a	-	6a	0b	51	-	09	-	60	02	-	0a	0a
	01	-	03	-	06	-	12	-	08	-	-	0e	-	12	-	06	00	-	08	-
	5a	-	18	-	2c	-	03	03	00	05	00	-	09	-	06	00	00	-	-	-
		02	-	10	-	04	03	03	00	05	00	04	09	0c	06	00	00	04		

TABLE 4: Delta Encoding: Credit A/c No: 6711645138110 with my VISA debitcard 123456789101112

As usual, we compute $J(x_0)$ and $F(x_0)$ of equation (8). Then take initial guess of $x_0 = (1, 1, 1, 1, 1, 1, 1, 1, 1, 1)^t$ and substitute in the equation, this will give the equation (9) in matrix form.

$$\begin{bmatrix} 7 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 3 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \\ 4 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 13 & 0 & 2 & 1 & 5 & 1 & 1 & 1 & 1 & 1 \\ 5 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 2 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 4 & 0 & 0 & 0 & 1 & 1 & 2 & 0 & 0 & 0 \\ 5 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 4 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \\ 16 & 0 & 1 & 1 & 5 & 1 & 1 & 1 & 1 & 3 \end{bmatrix} \begin{bmatrix} x_1 - 1 \\ x_2 - 1 \\ x_3 - 1 \\ x_4 - 1 \\ x_5 - 1 \\ x_6 - 1 \\ x_7 - 1 \\ x_8 - 1 \\ x_9 - 1 \\ x_{10} - 1 \end{bmatrix} = \begin{bmatrix} 15 \\ 21 \\ 10 \\ 62 \\ 10 \\ 5 \\ 21 \\ 21 \\ 20 \\ 78 \end{bmatrix} \quad (9)$$

Similarly, equations (9), can be solve by applying the procedure in section 3.3 to obtain the following approximate solutions:

$$x_1 = 1, x_2 = 2, x_3 = 3, x_4 = 4, x_5 = 5, x_6 = 6, x_7 = 7, x_8 = 8, x_9 = 9, x_{10} = 10$$

To decrypt the message in Equation (8), the receiver would further need other secret keys as stated in section 3.2. Using these keys, the intended receiver can now recover the encrypted text as shown in the result Table 7.

4. RESULTS

Table 5 shows the result of the deciphered text from the enciphered text (Equation (1b)) obtained from the proposed algorithms.

Position of Word s_k	$s_k = v_s + \sum_{i=1}^k \delta_i$, where v_s = variable solution	From Table I (R.C)
1	$x_1 = 1 + 76 = 77$	$77 = w$
	$x_2 = 2 + 76 - 10 = 68$	$68 = h$
	$x_3 = 3 + 76 - 10 + 6 = 6f$	$6f = o$
2	$x_1 = 1 + 68 = 69$	$69 = i$
	$x_2 = 2 + 68 + 9 = 73$	$73 = s$
3	$x_3 = 3 + 6d = 70$	$70 = p$
	$x_1 = 1 + 6d + 4 = 72$	$72 = r$
	$x_3 = 3 + 6d + 4 - 5 = 6f$	$6f = o$
	$x_2 = 2 + 6d + 4 - 5 - 1 = 6d$	$6d = m$
	$x_1 = 1 + 6d + 4 - 5 - 1 - 3 = 69$	$69 = i$
	$x_2 = 2 + 6d + 4 - 5 - 1 - 3 + 9 = 73$	$73 = s$
	$x_1 = 1 + 6d + 4 - 5 - 1 - 3 + 9 - 9 = 69$	$69 = i$
	$x_3 = 3 + 6d + 4 - 5 - 1 - 3 + 9 - 9 + 3 = 6e$	$6e = n$
	$x_1 = 1 + 6d + 4 - 5 - 1 - 3 + 9 - 9 + 3 - 5 = 67$	$67 = g$
4	$x_1 = 1 + 76 = 77$	$77 = w$
	$x_2 = 2 + 76 - 10 = 68$	$68 = h$
	$x_3 = 3 + 76 - 10 + 6 = 6f$	$6f = o$

TABLE 5: Plaintext Recovery

Table 6 shows the result of the deciphered text from the enciphered text (Equation (5b))

Position of Word s_k	$s_k = v_s + \sum_{n=1}^k \delta_n$, where v_s = variable solution	From Table 1 (R.C)
1	$x_1 = 1 + 4a$	$4b = K$
	$x_2 = 2 + 4a + 1d$	$69 = i$
	$x_3 = 3 + 4a + 1d + 0e$	$6c = l$
	$x_3 = 3 + 4a + 1d + 0e + 00$	$6c = l$
2	$x_4 = 4 + 5d$	$61 = a$
	$x_5 = 3 + 5d + 0c$	$6c = l$
	$x_5 = 3 + 5d + 0c + 00$	$6c = l$
	:	:
7	$x_7 = 3 + 4a$	$4d = M$
	$x_7 = 2 + 4a + 1d$	$69 = i$
	$x_8 = 4 + 4a + 1d + 0e$	$73 = s$
	$x_8 = 4 + 4a + 1d + 0e + 00$	$73 = s$
	$x_7 = 2 + 4a + 1d + 0e + 00 - 0e$	$69 = i$
	$x_8 = 4 + 4a + 1d + 0e + 00 - 0e + 0e$	$73 = s$
	$x_8 = 4 + 4a + 1d + 0e + 00 - 0e + 0e + 00$	$73 = s$
	$x_7 = 2 + 4a + 1d + 0e + 00 - 0e + 0e + 00 - 0e$	$69 = i$
	$x_8 = 6 + 4a + 1d + 0e + 00 - 0e + 0e + 00 - 0e + 0e$	$70 = p$
	$x_8 = 6 + 4a + 1d + 0e + 00 - 0e + 0e + 00 - 0e + 0e + 00$	$70 = p$
	$x_7 = 2 + 4a + 1d + 0e + 00 - 0e + 0e + 00 - 0e + 0e + 00 - 0e$	$69 = i$

TABLE 6: Plaintext Recovery for “Kill all Hippopotamus in the river Mississippi”

Table 7 shows the result of the deciphered text from the enciphered text (Equation (8))

Position of Word s_k	$s_k = v_s + \sum_{n=1}^k \delta_n$, where v_s = variable solution	From Table 1 (R.C)
1	$x_1 = 1 + 4e$	$43 = C$
	$x_2 = 2 + 4e + 2e$	$72 = r$
	$x_3 = 3 + 4e + 2e - 0e$	$65 = e$
	$x_4 = 4 + 4e + 2e - 0e - 0e$	$64 = d$
	$x_5 = 5 + 4e + 2e - 0e - 0e + 0e$	$69 = i$
	$x_6 = 6 + 4e + 2e - 0e - 0e + 0e + 0e$	$74 = t$
2	$x_7 = 7 + 3a$	$41 = A$
	$x_8 = 8 + 3a - 11$	$2f = /$
	$x_9 = 9 + 3a - 11 + 33$	$63 = c$
3	$x_{10} = a + 44$	$4e = N$
	$x_1 = 1 + 44 + 2a$	$6f = o$
	$x_2 = 2 + 44 + 2a + 33$	$3a = :$
	.	.
10	$x_5 = 5 + 2c$	$31 = 1$
	$x_{10} = a + 2c - 0e$	$32 = 2$
	$x_8 = 8 + 2c - 0e + 0e$	$33 = 3$
	$x_6 = 6 + 2c - 0e + 0e + 0e$	$34 = 4$
	$x_7 = 7 + 2c - 0e + 0e + 0e + 00$	$35 = 5$

$x_3 = 3 + 2c - 04 + 03 + 03 + 00 + 05$	$36 = 6$
$x_4 = 4 + 2c - 04 + 03 + 03 + 00 + 05 + 00$	$37 = 7$
$x_9 = 9 + 2c - 04 + 03 + 03 + 00 + 05 + 00 - 04$	$38 = 8$
$x_1 = 1 + 2c - 04 + 03 + 03 + 00 + 05 + 00 - 04 + 09$	$39 = 9$
$x_5 = 5 + 2c - 04 + 03 + 03 + 00 + 05 + 00 - 04 + 09 - 0c$	$31 = 1$
$x_{10} = 10 + 2c - 04 + 03 + 03 + 00 + 05 + 00 - 04 + 09 - 0c - 06$	$30 = 0$
$x_5 = 5 + 2c - 04 + 03 + 03 + 00 + 05 + 00 - 04 + 09 - 0c - 06 + 06$	$31 = 1$
$x_5 = 5 + 2c - 04 + 03 + 03 + 00 + 05 + 00 - 04 + 09 - 0c - 06 + 06 + 00$	$31 = 1$
$x_5 = 5 + 2c - 04 + 03 + 03 + 00 + 05 + 00 - 04 + 09 - 0c - 06 + 06 + 00 + 00$	$31 = 1$
$x_{10} = 10 + 2c - 04 + 03 + 03 + 00 + 05 + 00 - 04 + 09 - 0c - 06 + 06 + 00 + 00 - 04$	$32 = 2$

TABLE 7: Plaintext Recovery: Credit A/c No: 6711645138110 with my VISA debit card No: 123456789101112

In decompression, the characters are written as they appeared coupled with spaces between them and where there is an index value in place of new character, the index is interpreted and written in the place they appeared. Each time a word is written, a space is allowed between them.

4. DISCUSSION

Examination of Tables 5, 6 and 7, showed that simple attack identified by [9] to find the decryption key by the cryptanalyst requires solving the systems of nonlinear equations in equations (1b), (5b) and (8), obtaining the δ_x values associated with the variable index in Tables 2, 3 and 4, the formula $S_x = V_x + \sum_{i=1}^n \delta_x$, and the lookup character position in table 1. This is notoriously difficult to obtain due to their high mathematical formulation. A good encryption algorithm should be designed so that, when used with sufficiently long keys, it becomes computationally infeasible to break as reported [12,13]. This is in accordance with another related literature that revealed that the strength of an encryption algorithm relied on the mathematical soundness of the algorithm [3]. It is also in agreement with an earlier study by [8] who revealed that resources required for revealing a secret message should be strong and complex enough through a hiding key. This study is designed on similar encryption techniques that use sufficiently long keys.

The study also indicated that one key is used to encipher plaintext into ciphertext and another different key to decipher that ciphertext into plaintext as depicted in Tables 5, 6, and 7 [14,15,16]. The proposed scheme avoids the problem of sharing keys associated with the symmetric cryptography [16,17] that there is less risk associated with a public key than the symmetric key and the security based on that key is not compromised [18]. This study is designed on similar encryption techniques that use asymmetric key.

The study showed that, the decipher keys were transmitted to the intended receiver secretly through a different medium such as email, short message service or fax machine to the receiver before the receiver can have access to the plaintext. On the basis of this results, it is evident that unauthorized user will find it difficult to decrypt the message without the knowledge of the secret keys [3], since they were not transmitted together with the ciphered message. The strength of an encryption scheme is relies on the secrecy of the key [12]. This placed another level of security on the data in store or transit.

From the results of the study it is clear that there is confidentiality, non- repudiation and integrity of our sensitive and classified information over the Internet from the hands of Internet terrorist as highlighted by [2] and [16]. This is due to the robustness design of the proposed algorithms.

5. CONCLUSION

This paper has practically demonstrated how people can secure their vital and sensitive information stored or transmitted via insecure communication channels from cryptanalysts by using strong encryption and decryption keys. The proposed algorithm has proven to withstand any type of the attack.

7. REFERENCES

- [1] B. Figg. (2004). *Cryptography and Network Security*. Internet: <http://www.homepages.dsu.edu/figgw/Cryptography%20&%20Network%20Security.ppt> [March 16, 2010].
- [2] A. Kahate, *Cryptography and Network Security (2nd ed.)*. New Delhi: Tata McGraw Hill, 2008.
- [3] M. Milenkovic. *Operating System: Concepts and Design*, New York: McGraw-Hill, Inc., 1992.
- [4] P.R. Zimmermann. *An Introduction to Cryptography*. Germany: MIT press. Available: <http://www.pgpi.org/doc/pgpintro>, 1995, [March 16, 2009].
- [5] W. Stallings. *Cryptography and Network Security (4th ed.)*. Englewood (NJ):Prentice Hall, 1995.
- [6] V. Potdar and E. Chang. "Disguising Text Cryptography Using Image Cryptography," International Network Conference, United Kingdom: Plymouth, 2004.
- [7] S.A.M. Diao, M.A.K. Hatem, and M.H. Mohiy (2010). "Evaluating The Performance of Symmetric Encryption Algorithms" *International Journal of Network Security*, 2010, 10(3), pp.213-219
- [8] T. Ritter. "Crypto Glossary and Dictionary of Technical Cryptography". Internet: www.ciphersbyritter.com/GLOSSARY.HTM , 2007, [August 17, 2009]
- [9] K.M. Alallayah, W.F.M. Abd El-Wahed, and A.H. Alhamani. "Attack Of Against Simplified Data Encryption Standard Cipher System Using Neural Networks". *Journal of Computer Science*, 2010, 6(1), pp. 29-35.
- [10] D. Rudolf. "Development and Analysis of Block Cipher and DES System". Internet: <http://www.cs.usask.ca/~dtr467/400/>, 2000, [April 24, 2009]
- [11] H. Wang. (2002). *Security Architecture for The Teamdee System*. An unpublished MSc Thesis submitted to Polytechnic Institution and State University, Virginia, USA.
- [12] G.W. Moore. (2001). *Cryptography Mini-Tutorial*. Lecture notes University of Maryland School of Medicine. Internet: <http://www.medparse.com/whaticryp.htm> [March 16, 2009].
- [13] T. Jakobsen and L.R. Knudsen. (2001). Attack on Block of Ciphers of Low Algebraic Degree. *Journal of Cryptography*, New York, 14(3), pp.197-210.
- [14] N. Su, R.N. Zobel, and F.O. Iwu. "Simulation in Cryptographic Protocol Design and Analysis." Proceedings 15th European Simulation Symposium, University of Manchester, UK., 2003.

- [15] C.K. Laudan, and C.G. Traver. *E-Commerce .Business .Technology .Society (2nd ed.)*. New York: Pearson Education, Inc., 2004.
- [16] G.C. Kessler. *Handbook on Local Area Networks: An Overview of Cryptography*. United Kingdom: Auerbach. Available <http://www.garykessler.net/library/crypto.html>. 2010, [January 3, 2010].
- [17] M.A. Yusuf. Data Security: Layered Approach Algorithm. An unpublished MSc Thesis submitted to Abubakar Tafawa Balewa University, Bauchi, Nigeria, 2007.
- [18] J. Talbot and D. Welsh. *Complexity and Cryptography: An Introduction*. New York: Cambridge University Press, 2006

A Study of Protocols for Grid Computing Environment

Suresh Jaganathan

Associate Professor

*Department of Computer Science and Engineering
Sri Sivasubramania Nadar College of Engineering
Chennai, Tamilnadu, India*

whosuresh@gmail.com

Srinivasan A

Professor

*Department of Computer Science and Engineering
Misrimal Navajee Munoth Jain Engineering College
Chennai, Tamilnadu, India*

asrini30@gmail.com

Damodaram A

Professor

*Department of Computer Science and Engineering
Jawaharlal Nehru Technological University
Hyderabad, Andra Pradesh, India*

damodarama@jntuh.ac.in

Abstract

Using grid systems efficiently has to face many challenges. One of them is the efficient exchange of data between distant components exacerbated by the diversity of existing protocols for communicating participants. Grid computing will reach its vast potential if and only if, the underlying networking infrastructure is able to transfer data across quite long distances in a very effective manner. Experiences show that advanced distributed applications executed in existing large scale computational grids are often able to use only a small fraction of available bandwidth. The reason for such a poor performance is the TCP, which works only in low bandwidth and low delay networks. Several new transport protocols have been introduced, but a very few are widely used in grid computing applications, these protocols can be categorized in three broad categories viz. TCP based, UDP based and Application layer protocols. We study these protocols and present its performance and research activities that can be done in these protocols.

Keywords: Communication Protocol, Grid Computing, Bandwidth Delay Networks and Performance

1. INTRODUCTION

GRID computing [1, 2] is a technology for coordinating large scale resource sharing and problem solving among various autonomous group. Grid technologies are currently distinct from other major technical trends such as internet, enterprise distributed networks and peer to peer computing. Also it has some embracing issues in QoS, data management, scheduling, resource allocation, accounting and performance.

Grids are built by various user communities to offer a good infrastructure which helps the members to solve their specific problems which are called a grand challenge problem. A grid consists of different types of resources owned by different and typically independent organizations which results in heterogeneity of resources and policies. Because of this, grid based services and applications experience a different resource behavior than expected. Similarly, a distributed infrastructure with ambitious service put more impact on the capabilities of the interconnecting networks than other environments.

Grid High Performance Network Group [3] works on network research, grid infrastructure and development. In their document the authors listed six main functional requirements, which are

considered as mandatory requirements for grid applications. They are: i) high performance transport protocol for bulk data transfer, ii) performance controllability, iii) dynamic network resource allocation and reservation, iv) security, v) high availability and vi) multicast to efficiently distribute data to group of resources.

New trends of data communication protocols are needed for grid, because of these reasons, i) networks emerging with long fat network [LFN], high bandwidth and long delay such as LambdaGrid [4], OptiPuter [5], CANARIE [6] and sensor networks for grid [7], ii) communication patterns which are shifting from one-to-one communication to many-to-one and many-to-many communication and iii) there is a unique type of communication needs in transport characteristics for each grid application, such as rate, loss ratio and delay. For all these data communications, standard transport protocols such as TCP [8] and UDP [9] are not always sufficient, for example, traditional TCP reacts adversely when there is an increase in bandwidth and delay leading to poor performance in high BDP networks [10]. It is still a challenge for grid applications to use the bandwidth available, due to the limitations of current and today's network transport protocols.

For instance, a remote digital image processing application may use unreliable fixed rate data transfer, whereas a fast message parsing application can demand reliable data transfer with minimum delay, and a wireless grid application can demand for minimized data traffic to prolong battery life. As TCP acts as a reliable transport protocol and UDP with no guarantees, these two protocols cannot provide the optimal mix of communication attributes for different grid applications. More number of transport protocols have been developed and proposed over the last 3 decades. In this paper, we study the various variants of such transport protocols based on TCP and UDP, and compare the protocols in various points for grid computing. Each protocol is reviewed based on the i) operation, ii) operation mode, iii) implementation, iv) congestion control, v) fairness, vi) throughput, vii) TCP friendly, viii) security, ix) quality of service and x) usage scenario.

Section II highlights the issues in designing high performance protocols and briefly how TCP and UDP play an important role in implementing grid computing protocols. Section III surveys the various protocols for bulk data transfer, high speed, and high bandwidth delay and with high performance with TCP as base. Section IV surveys UDP based protocols. Section V deals with application layer protocols and Section VI concludes the paper with a summary.

2. ISSUES IN COMMUNICATION PROTOCOLS FOR GRID COMPUTING

The emerging high-performance grid can have a wide range of network infrastructures and different communication patterns for different types of applications, these combinations and factors made researchers develop new data communication protocols especially for grid environments. For these applications, available standard protocols (TCP and UDP) are not sufficient, because of their properties and lack of flexibility. Performance of TCP not only depends on transfer rate, but also on the product of round-trip delay and transfer rate. This *bandwidth*delay* product measures the amount of data that would fill the pipe, and that amount of data is the buffer space required at sender and receiver side to obtain maximum throughput on the TCP connection over the path. TCP performance problems arise when the *bandwidth*delay* product becomes large. Three fundamental performance problems with the TCP over high bandwidth delay network paths are; i) window size limit, ii) recovery from losses, and iii) round-trip measurement.

In grid high performance research document [3], it summarizes the networking issues available in grid applications and gives some consideration for designing a protocol for grid computing, they are: i) slow start, ii) congestion control, iii) ACK clocking, iv) connection setup and teardown and in [11] author lists some parameters which relate to TCP performance in high speed networks, they are: i) cwnd increase function, ii) responsiveness requirements, iii) scalability and iv) network dynamics. In [12] some general set of transport attributes are listed for grid computing, i) connection oriented vs. connectionless, ii) reliability, iii) latency and jitter, iv) sequenced vs. unsequenced, v) rate shaping vs. best effort, vi) fairness and vii) congestion control. The rapid

growth of network infrastructures and communication methods made the grid community to demand enhanced and diverse transport functionality. Because of these growth in network environment, QoS guarantees and its properties depends on these parameters, i) type of link (dedicated or shared), ii) communication methods (point-to-point or multipoint-to-point) and iii) application requirements.

2.1. Need of High Performance Communication Protocol

2.2.1. Increasing Capabilities

Increasing in computing power has a doubling time of 18 months, storage device capacity doubles every 12 months and communication speed doubles every 9 months. The difference in rate of increase creates a situation in all areas, although significant power available, because of traditional implementation tradeoff method changes. This increasing capability has resulted in work to define new ways to interconnect and manage computing resources.

2.2.2. New Application Requirements

New applications are being developed in physics, biology, astronomy, visualization, digital image processing, meteorology etc. Many of the anticipated applications have communication requirements between a small numbers of sites. This presents a very different requirement than presented by “typical” Internet use, where the communication is among many sites. Applications can be defined in three classes: 1) lightweight “classical” Internet applications (mail, browsing), 2) medium applications (business, streaming, VPN) and 3) heavyweight applications (e-science, computing, data grids, and virtual presence). The total bandwidth estimate for all users of each class of network application is 20 Gb/sec for the lightweight Internet, 40 Gb/sec for all users of the intermediate class of applications and 100 Gb/sec for the heavyweight applications. Note that the heavyweight applications use significantly more bandwidth than the total bandwidth of all applications on the classical Internet. Different application types value different capabilities. Lightweight applications give importance to interconnectivity, middleweight applications to throughput and QoS, while the heavyweight applications to throughput and performance.

An example of heavyweight application, the Large Hadron Collider (LHC) in CERN, has requirements well beyond what is available now. The LHC produce and distribute gigantic amounts of data. The data rates from the LHC in CERN are estimated to be in the order 100 to 1500 Megabytes per second. Dedicated fiber will carry data from the Atlas Collector in CERN to the initial storage and computing devices around the world, bypassing the Internet. Another example of heavyweight application are the CAVE Research Network, a work being done at the Electronic Visualization Lab (EVL) at the University of Illinois at Chicago which uses very large communication rates to implement total immersion projects. Other examples include the Electron Microscopy work being done at the University of California San Diego (UCSD) and Osaka.

TCP works well on the commodity internet, but has been found to be inefficient and unfair to concurrent flows as bandwidth and delay increase [13, 14, 15, 16]. Its congestion control algorithm needs very long time to probe the bandwidth and recover from loss in high BDP links. Moreover, the existence of random loss on the physical link, the lack of a buffer on routers, and the existence of concurrent bursting flows prevent TCP from utilizing high bandwidth with a single flow. Furthermore, it exhibits a fairness problem for concurrent flows with different round trip times (RTTs) called RTT bias. The success of TCP is mainly due to its stability and the wide presence of short lived, web-like flows on the Internet. However, the usage of network resources in high performance data intensive applications is quite different from that of traditional internet applications. First, the data transfer often lasts a very long time at very high speeds. Second, the computation, memory replication, and disk I/O at the end hosts can cause bursting packet loss or time-outs in data transfer. Third, distributed applications need cooperation among multiple data connections. Finally, in high performance networks, the abundant optical bandwidth is usually shared by a small number of bulk sources. These constraints made the researchers to design a new transport protocol for high performance domains.

2.2. Congestion Control

Moving bulk data quickly over high-speed data network is a requirement for many applications. These applications require high bandwidth link between network nodes. To maintain the stability of internet, all applications should be subjected to congestion control. TCP is well-developed, extensively used and widely available Internet transport protocol. TCP is fast, efficient and responsive to network congestion conditions, but one objection to using TCP congestion control is that TCP's AIMD congestion back-off algorithm [17], which is too abrupt in decreasing the window size, thus it hurts the data rate.

The performance of the congestion control system, TCP algorithm and the link congestion signal algorithm has many facets and these variables can impact the Quality of Service (QoS). A variety of merits described they are:

Fairness: The Jain index [18] is a popular fairness metric that measures how equally the sources sharing a single bottleneck link. A value of 1 indicates perfectly equal sharing and smaller values indicate worse fairness.

Throughput: Throughput is simply the data rate, typically in Mbps, delivered to the application. For a single source this should be close to the capacity of the link. When the BDP is high, that is, when the link capacity or RTT or both are high, some protocols are unable to achieve good throughput.

Stability: The stability metric measures the variations of the source rate and/or the queue length in the router around the mean values when everything else in the network is held fixed. Stability is typically measured as the standard deviation of the rate around the mean rate, so that a lower value indicates better performance. If a protocol is unstable, the rate can oscillate between exceeding the link capacity, and thus resulting in poor delay jitter and throughput performance.

Responsiveness: measures how fast a protocol reacts to a change in network operating conditions. If the source rates take long time to converge to a new level, say after the capacity of the link changes, either the link may become underutilized or the buffer may overflow. The responsiveness metric measures the time or the number of round trips to obtain the right rate.

Queuing delay: Once congestion window is greater than the BDP, the link is well utilized and however, if the congestion window is increased more, queuing delay builds up. Different TCP and AQM protocol combinations operate on how to minimize the Queuing Delay.

Loss recovery: packet loss can be a result because of overflowing buffers, which indicates network congestion, and also of transmission error, such as bit errors over a wireless channel. It is desirable that, when packet loss occurs due to transmission error, the source continues to transmit uninterrupted. However, when the loss is due to congestion, the source should slow down. Loss recovery is typically measured as the throughput that can be sustained under the condition of a certain random packet loss caused by transmission error. Loss based protocols typically cannot distinguish between congestion and transmission error losses.

In the past few years, more number of TCP variant were developed, that address the under-utilization problem most notably due to the slow growth of TCP congestion window, which makes TCP unfavorable for high BDP networks. Table 1 list some TCP variants addressing the conservative approach of TCP to update its congestion window under congestion condition.

Congestion Control Approaches	TCP Variants
Loss-based TCP congestion control	High Speed TCP, BIC TCP, Scalable TCP, CUBIC TCP and Hamilton TCP
Delay-based congestion control	TCP-Vegas, Fast-TCP
Mixed loss-delay based TCP congestion control	Compound TCP ,TCP Africa
Explicit congestion Notification	XCP

TABLE 1: TCP Variants based on various congestion control approaches

Loss based high speed algorithms are aggressive to satisfy bandwidth requirement but this aggressiveness causes TCP unfairness and RTT unfairness. Delay based approaches provide

RTT fairness but it is difficult to meet TCP fairness, where in third approach, a synergy of delay based and loss based approach address the problem in the two approaches.

2.3. Previous Work

Delivering communication performance in high bandwidth-delay product network is a major research challenge. Compared to shared, packet-switched IP networks, the key distinguishing characteristics of grid computing communications are: i) no internal network congestion and very small end system congestion, ii) small numbers of endpoints and iii) very high speed links i.e. more than 10GB.. Traditional TCP and its variants were developed for shared networks, where the bandwidth on internal links is a critical and has limited resource. As such, available congestion control techniques manage internal network contention, providing a reasonable balance of non-aggressive competition and end-to-end performance. This results in slow start, causing TCP to take a long time to reach full bandwidth when RTT is large and takes a long time to recover from packet loss because of its AIMD control law. In [19, 20], a simulation-based performance analysis of HighSpeed TCP is presented and the fairness to regular TCP is analyzed. In [21], the author deals with the performance of Scalable TCP and analyzes the aggregate throughput of the standard TCP and Scalable TCP based on an experimental comparison. In [22], the performance of different TCP versions, such as HighSpeed TCP, Scalable TCP, and FAST TCP, are compared in an experimental test-bed environment. In all cases, the performance among connections of the same protocol sharing bottleneck link is analyzed and different metrics such as throughput, fairness, responsiveness, stability are presented.

To provide good transport performance for network with high bandwidth-delay product optical network links, researchers have proposed many delay based TCP variations and recently, rate-based reliable transport protocols are proposed based on UDP. They explicitly measure packet loss, and adjust transmission rates in response. A number of delay based TCP variants are proposed for shared, packet-switched networks and few user level rate-based protocols, targeting different network scenarios and communication methods. Evaluation of these protocols are done using point to point single flow, parallel point to point flows, and multipoint convergent flows. From the evaluation results, rate-based protocols deliver high performance in high bandwidth-delay product networks, but because of their aggressiveness, there occurs high rate of packet loss when multiple flows exist.

3. TCP BASED PROTOCOLS FOR GRID NETWORKS

New challenges for TCP have been addressed by several research groups in the last 3 decades and, as a result, a number of new TCP variants have been developed. An overview and study of protocols that are designed for high speed bulk data transfer in high bandwidth delay networks are given here, and the TCP variants analyzed in this paper are summarized in Figure 1.

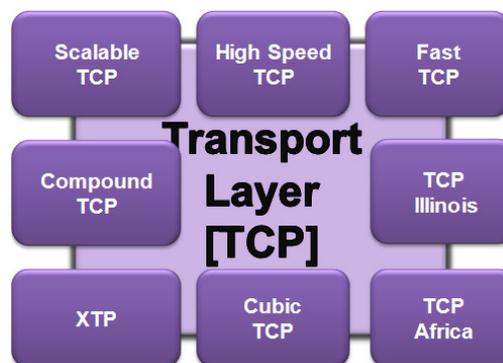


FIGURE 1: TCP Variants

Emerging networks bring new challenges. First, the new architecture, heterogeneous networks, mobile and wireless environments exhibit different network characteristics requiring more attention on the dynamical aspects of the operation. As a consequence of this, the dynamic behavior of TCP flows has to be taken into consideration. On the one hand, it is obvious that the dynamic effects have significant impact on the performance and throughput of the TCP flows [23]. On the other hand, the fairness also needs to be reconsidered from the aspects of dynamic behavior. The performance analyses of various TCP variants are included in many papers [24, 25, 26].

3.1. Scalable TCP (S-TCP)

Performance of today’s network is to be increased due to applications like distributed simulation, remote laboratories, and multi gigabyte data transfers. TCP fails to capture the available bandwidth in high performance network, because of two reasons: i) size of socket buffer at the end-hosts which limits the transfer rate and the maximum throughput and ii) packet loss causing multiplicative cwnd value decrease and additive increase of cwnd when in absence of loss, reduces the average throughput. To address these issues researchers focused in three approaches: i) modifying available TCP, ii) parallel TCP and iii) automatic buffer sizing.

Modifying TCP congestion control scheme and also in routers, can lead to significant benefits for both applications and networks. Parallel TCP connection increases the aggregate throughput, but fails in maintaining fairness and also in increasing the window rate. Several researchers have proposed TCP modifications, mainly on congestion control schemes making TCP more effective in high performance paths. Kelly proposed Scalable TCP [21]. The main feature in the Scalable TCP is it adopts constant window increase and decrease factors in case of congestion and multiplicative increase in absence of congestion.

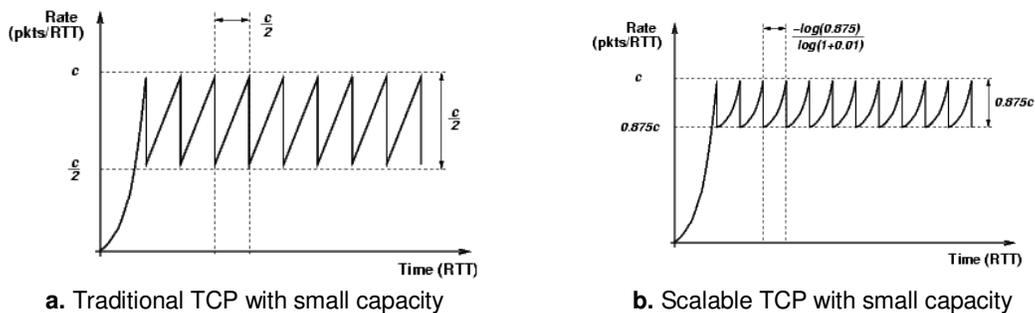
Scalable TCP is designed in such a way that it automatically switches to traditional TCP stacks in low bandwidth and to incremental method at the time of high bandwidth available. STCP is a simple sender side modification to TCP congestion control, and it employs Multiplicative Increase Multiplicative Decrease (MIMD) technique. Using Scalable TCP, better utilization of a network link with the high bandwidth-delay product can be achieved. If STCP is mixed with regular TCP then STCP dominates the bandwidth for sufficiently large bandwidth-delay product region, which results in unfriendliness towards standard TCP. Congestion Window [cwnd] value is changed according to the equation shown below:

$$cwnd = cwnd + 0.01 \quad \text{for each ack received and no loss}$$

$$cwnd = 0.875 * cwnd \quad \text{on each loss event}$$

Scalable TCP response function is calculated as $W_{scalable} = \frac{0.0745}{p}$ where ‘p’ is packet loss rate.

Figure 2 shows the probing results of traditional and Scalable TCP. The main difference between the two depends on round trip time and rate, traditional TCP are proportional to sending rate and round trip time, where are in Scalable TCP it is proportional only to round trip time and not depends on rate.



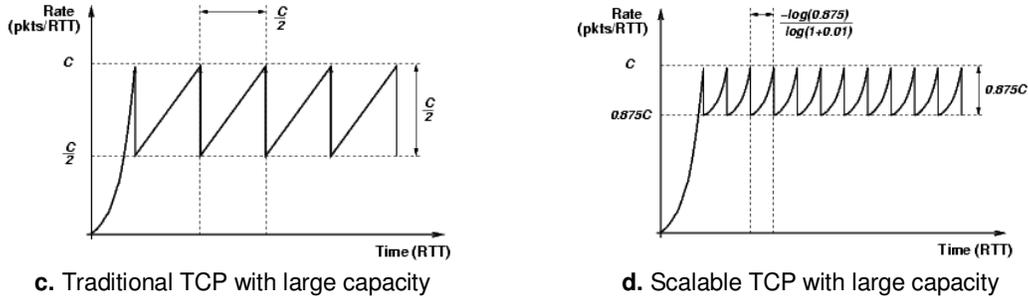


FIGURE 2: Scaling properties Traditional TCP vs. Scalable TCP

Table 2 shows the recovery time for standard TCP congestion control and Scalable TCP congestion control for various sending rates and packet loss in less than one RTT.

Rate	Recovery Time	
	Standard TCP	Scalable TCP
1Mbps	1.7s	2.7s
10Mbps	17s	3.0s
100Mbps	2mins	4.7s
1Gbps	28mins	2.7mins
10Gbps	4hrs 43mins	1 hrs 24mins

TABLE 2: Recovery Times

The recovery time after packet loss is 13.42RTT, i.e. proportional to the RTT and independent of congestion window size. An STCP connection can recover even a large congestion window in a short time and so that it makes efficient use of the bandwidth in high-speed networks. For e.g. for a TCP connection with 1500 byte MTU and RTT value of 200ms, then for 10Gbps network, congestion window recovery time after packet loss for STCP is 1hrs 24mins whereas that for Standard TCP is approximately 4hrs 43mins. STCP can be used in high performance environment because of i) its efficient congestion control, ii) improved performance in high BDP and iii) increased throughput, but it does has fairness issue.

3.2. High Speed TCP (HS-TCP)

Networking applications such as multimedia web streaming seek more bandwidth and bulk-data transfer. These applications often operate over a network with high BDP, so the performance over these networks is a critical issue [27]. Recent experiences indicate that TCP has difficulty in utilizing high-speed connections, because of this, network applications are not able to take utilize the available bandwidth in high speed network [28]. The amount of packets that are needed to fill a gigabit pipe using present TCP is beyond the limit of currently achievable in fiber optics and the congestion control is not so dynamic, also most users are unlikely to achieve even 5 Mbps on a single stream wide-area TCP transfer, even the underlying network infrastructure can support rates of 100 Mbps or more.

HSTCP (HighSpeed TCP) is a variant to the TCP, which is specifically designed for use in high speed, high bandwidth network. Congestion management allows the protocol to react and to recover from congestion and operate in a state of high throughput yet sharing the link fairly with other traffic.

The performance of a TCP connection is dependent on the network bandwidth, round trip time, and packet loss rate. At present, TCP implementations can only reach the large congestion window, necessary to fill a pipe with a high bandwidth delay product, when there is an exceedingly low packet loss rate. Otherwise, random losses lead to significant throughput deterioration when the product of the loss probability and the square of the bandwidth delay are

larger than one. The HighSpeed TCP for large congestion window [29] was introduced as a modified version of TCP congestion control mechanism. It is designed to have different responses in very low congestion event rate, and also to have the standard TCP responses with low packet loss rates. Further, TCP behavior is unchanged when there is mild to heavy congestion and doesn't increase the congestion collapse.

HighSpeed TCP's modified response function comes in effect with higher congestion window, and TCP behavior is unchanged when there is heavy congestion and doesn't introduce any congestion collapse. HSTCP uses three parameters, W_L , W_H , and P_H . To ensure TCP compatibility, when the congestion window is W_L at most, HSTCP uses the standard TCP response function, and uses the HSTCP response function when congestion window is greater than W_L . When the average congestion window is greater than W_L , then the response function is calculated as follows:

$$W = \left(\frac{p^s}{P_L^s}\right)W_L, \text{ where } s = \frac{(\log W_H - \log W_L)}{(\log P_H - \log P_L)}$$

HSTCP keeps average congestion window W_H and W_L , when packet loss rates are P_H and P_L , respectively. Recommended parameters are: $W_L = 38$, $W_H = 83000$ and $P_H = 10^{-7}$. Even though there is a loss rate in high-speed environments, it still allows acceptable fairness for the HighSpeed response function when compared with Standard TCP environments with packet drop rates of 10^{-4} or 10^{-5} . The HSTCP response function is computed as shown below:

$$W_{\text{highspeed}} = \frac{0.12}{p^{0.001}}$$

HSTCP is more aggressive than standard TCP and a HighSpeed TCP connection would receive ten times the bandwidth of a standard TCP in an environment with packet drop rate of 10^{-6} , which is unfair. The HighSpeed TCP response function is represented by new additive increase and multiplicative decrease parameters and these parameters modify the function parameters according to congestion window value.

HighSpeed TCP performs well in high-speed long-distance links. It falls back to standard TCP behavior if the loss ratio is high. In case of burst traffic, its link utilization is improved but at the same time there are some issues regarding fairness.

3.3. TCP Africa

The limitations of standard TCP are apparent in networks with large bandwidth-delay-products, which are becoming increasingly common in the modern Internet. For example, supercomputer grids, high energy physics, and large biological simulations require efficient use of very high bandwidth Internet links, often with the need to transfer data between different continents. The desired congestion window used by standard TCP is roughly equal to the BDP of the connection. For high bandwidth-delay-product links, this desired congestion window is quite high, as high as 80,000 packets for a 10 Gbps link with 100 ms RTT.

TCP's deficiencies with a purely loss based protocol leads to super-aggressive protocols, which raises fairness and safety concerns. By contrast, delay based protocols, which make use of the wealth of information about the network state that is provided by packet delays, can indeed achieve excellent steady state performance, as well as minimal self-induced packet losses. These delay-responsive protocols do not cause enough packet drops to the non-delay-responsive protocols to force them to maintain only their fair share. TCP-Africa, an Adaptive and Fair Rapid Increase Congestion Avoidance mechanism for TCP is proposed [30], to solve this problem which is a new delay sensitive congestion avoidance mode, which has scalability, aggressive behavior.

TCP-Africa, a new delay sensitive two-mode congestion avoidance rule for TCP, promises excellent utilization, efficiency, and acquisition of available bandwidth, with significantly improved safety, fairness, and RTT bias properties. This new protocol uses an aggressive, scalable window

increase rule to allow quick utilization of available bandwidth, but uses packet round trip time measurements to predict eminent congestion events. Once the link is sensed to be highly utilized, the protocol reacts by resorting to the conservative congestion avoidance technique of standard TCP.

TCP-Africa is a hybrid protocol that uses a delay metric to determine whether the bottleneck link is congested or not. In the absence of congestion, the *fast mode*, where it uses aggressive, scalable congestion avoidance rule. In the presence of congestion, the *slow mode*, it switches to the more conservative standard TCP congestion avoidance rule.

TCP-Africa, in its scalable “fast” mode, quickly grabs available bandwidth. Once the delay increases, the protocol senses that the amount of available bandwidth is getting small. So, it switches to a conservative “slow” mode.

TCP-Africa detects congestion using the following metric:

$$\frac{W(aRTT - minRTT)}{aRTT} \geq \alpha$$

The quantity $(aRTT - minRTT)$ gives us an estimate of the queuing delay of the network. Since the overall round trip time is $minRTT + (aRTT - minRTT)$, the quantity $(aRTT - minRTT) / aRTT$ is the proportion of the round trip time that is due to queuing delay rather than propagation delay. TCP maintains an average sending rate of $\frac{W}{aRTT}$ packets per second and in case of TCP algorithm it is $\frac{W(aRTT - minRTT)}{aRTT}$ an estimate for the number of packets that the protocol currently has in the queue. The α parameter is a constant, usually set as a real number larger than one. The choice of α determines how sensitive the protocol is to delay. By tracking this metric, TCP Africa can detect when its packet are beginning to en-queue at the bottleneck link, and thus, determine an opportune time to switch into slow growth mode.

The congestion avoidance steps followed by the protocol are:

$$\begin{aligned} &\text{if } (W(aRTT - minRTT) < \alpha \times aRTT) \\ &\quad \{W = W + fast_increase(W) / W\} \\ &\text{else} \\ &\quad \{W = W + 1/W\} \end{aligned}$$

The function *fast_increase(W)* is specified by a set of modified increase rules for TCP. The Table 3 shows the experimental results of TCP Africa.

RTT Ratio	Throughput Ratio	Packet Losses
1	1.0132	440
2	2.3552	581
3	3.6016	529
6	8.4616	336

TABLE 3: TCP Africa Experimental Results

Analysis of this protocol shows, switching to a one packet per RTT rate can greatly decrease the frequency of self inflicted packet loss. Thus, TCP-Africa can be quick to utilize available bandwidth, but slow when it comes to inducing the next congestion event.

3.4. Fast TCP

Advances in computing, communication, and storage technologies in global grid systems started to provide the required capacities and an effective environment for computing and science [31]. The key challenge to overcome the problem in TCP is, using Fast TCP congestion control algorithm which does not scale to this advancement. The currently deployed TCP implementation is a loss-based approach. It uses additive increase multiplicative decrease (AIMD) and it works

well at low speed, but additive increase (AI) is too slow and multiplicative decrease (MD) too drastic leading to low utilization of network resources. Moreover, it perpetually pushes the queue to overflow and also discriminates against flows with large RTTs. To address these problems, Fast TCP was designed, which adopts delay based approach.

Fast TCP has three key differences: i) it is an equation based algorithm [32], ii) for measuring congestion, it uses queuing delay as a primary value, iii) has stable flow dynamics and achieves weighted fairness in equilibrium that it does not penalize long flows. The advantages of using queuing delay as the congestion measure are, i) queuing delay is estimated more accurately than loss probability, and also loss samples provide coarser information than queuing delay samples. This makes easier stabilize a network into a steady state with a high throughput and high utilization and ii) the dynamics of queuing delay have the right scaling with respect to network capacity.

In Fast TCP, congestion is avoided by using maximum link utilization, which can be attained by adjusting the source's sending rate so that resource is shared by all TCP connections. Fast TCP uses two control mechanisms for achieving its objective, they are: i) dynamically adjusting the send rate and ii) using aggregate flow rate to calculate congestion measure. Under normal network conditions, Fast TCP periodically updates the congestion window 'w' based on the average RTT according to below equation

$$w = \min \left\{ 2w, (1 - \gamma)w + \gamma \left(\frac{\text{baseRTT}}{\text{RTT}} w + \alpha \right) \right\}$$

where $\gamma \in (0,1)$, RTT is the current average round-trip time, baseRTT is the minimum RTT, and α is a protocol parameter that controls fairness and the number of packets each flow buffered in the network [33]. It is proved that, in the absence of delay, this algorithm is globally stable and converges exponentially to the unique equilibrium point where every bottleneck link is fully utilized and the rate allocation is proportionally fair. Table 4 shows the experimental results of FAST TCP for implementation in Linux v2.4.18.

Flow	Transfer [Gb]	Utilization	Delay [ms]	Throughput [Mbps]
1	380	95%	180	925
2	750	92%	180	1797
7	15300	90%	85	6123
9	3750	90%	85	7940
10	21650	88%	85	8609

TABLE 4: FAST TCP experimental results

Fast TCP performs well in these areas: i) throughput, ii) fairness, iii) robust and iv) stability, but stability analysis was limited to a single link with heterogeneous sources and feedback delay was ignored. Further, many experimental scenarios were designed to judge the properties of Fast TCP but these scenarios are not very realistic.

3.5. TCP-Illinois

Several new protocols have been introduced to replace standard TCP in high speed networks. For all the protocols, the increase in window size initially slow, when the network is in absence of congestion, window size is small and in case congestion, the window size becomes large. As a result, the window size curve between two consecutive loss events is convex. This convex nature is not desirable. First, the slow increment in window size, at the time of absence of congestion is inefficient. Second, the fast increment in window size, at the time of congestion causes heavy loss. Heavy congestion causes more frequent timeouts, more synchronized window bakeoffs, and is more unfair to large RTT users. So the main problem with AIMD algorithm is the convexity of the window size curve. An ideal window curve should be concave, which is more efficient and avoids heavy congestion. To overcome and rectify this problem, a general AIMD algorithm is modified in such a way that it results in a concave window curve.

TCP-Illinois a variant of TCP congestion control protocol [34], developed at the University of Illinois at Urbana-Champaign, targeted at high-speed, long distance and long fat networks. It achieves high average throughput than the standard TCP, by modifying congestion control in sender side. It allocates the network resource fairly, a TCP friendly protocol and provides incentive for TCP users to switch.

TCP-Illinois is a loss-delay based algorithm, which uses packet loss as the primary congestion signal, and it uses queuing delay as the secondary congestion signal to adjust the pace of window size [35]. Similar to the standard TCP, TCP-Illinois increases the window size W by α / W for each acknowledgment, and decreases W by βW for each loss event. Unlike the standard TCP, α and β are not constants. Instead, they are the functions of average queuing delay d_q , $\alpha = f_1(d_q)$, $\beta = f_2(d_q)$. In detail

$$\alpha = f_1(d_q) = \begin{cases} \frac{\alpha_{max}}{k_1} & \text{if } d_q \leq d_1 \\ \frac{\alpha_{max}}{k_2 + d_q} & \text{otherwise} \end{cases}$$

$$\beta = f_2(d_q) = \begin{cases} \beta_{min} & \text{if } d_q \leq d_2 \\ k_3 + k_4 d_q & \text{if } d_2 < d_q < d_3 \\ \beta_{max} & \text{otherwise} \end{cases}$$

where $\alpha_{max} = \frac{k_1}{k_2 + d_1}$, $\beta_{min} = k_3 + k_4 d_2$, $\beta_{max} = k_3 + k_4 d_3$.

Suppose d_m is the maximum average queuing delay and denoted as $\alpha_{min} = f_1(d_m) = \frac{k_1}{k_2 + d_m}$, where $k_1 = \frac{(d_m - d_1) \alpha_{min} \alpha_{max}}{\alpha_{max} - \alpha_{min}}$, $k_2 = \frac{(d_m - d_1) \alpha_{min}}{\alpha_{max} - \alpha_{min}} - d_1$, $k_3 = \frac{(\beta_{min} d_3 - \beta_{max} d_1)}{d_3 - d_2}$, $k_4 = \frac{\beta_{max} - \beta_{min}}{d_3 - d_2}$.

TCP-Illinois increases the throughput much more quickly than TCP when congestion is far and increases the throughput very slowly when congestion is imminent. As a result, the average throughput achieved is much larger than the standard TCP. It also has many other desirable features, like fairness, compatibility with the standard TCP, providing incentive for TCP users to switch, robust against inaccurate delay measurement.

3.6. Compound TCP

The physicists at CERN LHC conduct physics experiments that generate gigabytes of data per second, which are required to be shared among other scientists around the world. These bulk data has to move quickly over high speed data network, currently most of the applications use the TCP for this purpose. TCP is a reliable data transmission with congestion control algorithm, which takes care of congestion collapses in the network by adjusting the sending rate according to the available bandwidth of the network but in the to-day's Internet environment, TCP substantially underutilizes network bandwidth over high-speed and long distance networks.

Compound TCP (CTCP) is a synergy of delay-based and loss-based approach [36]. The sending rate of Compound TCP is controlled by both sender and receiver components. This delay-based component rapidly increases sending rate at the time of underutilized, but retreats slowly in a busy network when bottleneck queue is built.

Compound TCP integrates a scalable delay-based component into the standard TCP congestion avoidance algorithm [37, 38]. This scalable delay-based component has a fast window increase function when the network is underutilized and reduces the sending rate when a congestion event is sensed. Compound TCP maintains the following state variables, *cwnd* (congestion window), *dwnd* (delay window) and *awnd* (receiver advertised window). TCP sending window is calculated as $W_{tm} = \min(cwnd + dwnd, awnd)$ where *cwnd* is updated in the same way as controlled by standard TCP, whereas in CTCP, on arrival of an ACK, *cwnd* is modified as:

$$cwnd = cwnd + \left(\frac{1}{cwnd + dwnd} \right)$$

CTCP efficiently uses the network resource and achieves high bandwidth utilization. In theory, CTCP can be very fast to obtain free network bandwidth, by adopting a rapid increase rule in the delay-based component, e.g. multiplicative increase.

CTCP has similar or even improved RTT fairness compared to regular TCP. This is due to the delay-based component employed in the CTCP congestion avoidance algorithm. When compared to other TCP variants, fairness value is good for CTCP. Because of the delay-based component, CTCP can slowly reduce the sending rate when the link is fully utilized. Hence, there is no self-induced packet loss when compared to standard TCP, maintained good fairness to other competing TCP flows and opted for high performance computing protocol.

3.7. CUBIC TCP

Yet another variant of TCP, called CUBIC that enhances the fairness property, scalability and stability [39]. The main feature of CUBIC is that its window growth function is defined in real time so that its growth will be independent of RTT.

CUBIC TCP is an enhanced version of Binary Increase Congestion Control shortly BIC [40]. It simplifies the BIC window control function and is TCP-friendliness and RTT fairness. As the name of the protocol represents, the window growth function of CUBIC is a cubic function in terms of the elapsed time accounting from the last loss event. The protocol keeps the window growth rate independent of RTT, which keeps the protocol TCP friendly under short and long RTTs. The congestion period of CUBIC is determined by the packet loss rate alone. As TCP's throughput is defined by the packet loss rate as well as RTT, the throughput of CUBIC is defined only by the packet loss rate. Thus, when the loss rate is high and/or RTT is short, CUBIC can operate in a TCP mode. More specifically, the congestion window of CUBIC is determined by the following function:

$$W_{cubic} = C(t - K)^3 + W_{max}$$

where C is a scaling factor, t is the elapsed time from the last window reduction, W_{max} is the window size just before the last window reduction, and $K = \sqrt[3]{W_{max}\beta/C}$ where β is a constant multiplication decrease factor applied for window reduction at the time of loss event (i.e., the window reduces to βW_{max} at the time of the last reduction).

An important feature of CUBIC is that it keeps the epoch fairly long without losing scalability and network utilization. Generally, in AIMD, a longer congestion epoch means slower increase (or a smaller additive factor). However, this would reduce the scalability of the protocol, and also the network would be underutilized for a long time until the window becomes fully open.

Unlike AIMD, CUBIC increases the window to W_{max} very quickly and then holds the window there for a long time. This keeps the scalability of the protocol high, while keeping the epoch long and utilization high. This feature is unique in CUBIC.

3.8. Xpress Transport Protocol (XTP)

In near future, data transfer over network increases, thus requiring high speed protocols, standard TCP fails to utilize the available bandwidth, due to its congestion control algorithm. Some variants of congestion control algorithm are proposed, which improves the throughput, but has poor fairness. Xpress Transport Protocol (XTP) is a transport layer protocol for high-speed networks developed by XTP Forum [41] to replace TCP. XTP provides protocol options for error control, flow control, and rate control. Instead of separate protocols for each type of communication, XTP controls packet exchange prototype to produce different models, e.g. reliable datagram's, transactions, unreliable streams, and reliable multicast connections.

XTP provides for the reliable transmission of data in an inter-networked environment, with real time processing of the XTP protocol i.e., the processing time for incoming or outgoing packets is

no greater than transmission time. XTP contains error, flow and rate control mechanisms similar to those found in other more modern transport layer protocols in addition to multicast capability. Timer management is minimized in XTP as there is only one timer at the receiver, used in closing the context. Address translation, context creation, flow control, error control, rate control and host system interfacing can all execute in parallel. The XTP protocol is considered a lightweight protocol for several reasons. First, it is a fairly simple yet flexible algorithm. Second, packet headers are of fixed size and contain sufficient information to screen and steer the packet through the network. The core of the protocol is essentially contained in four fixed-sized fields in the header — KEY, ROUTE, SEQ and the command word. Additional mode bits and flags are kept to a minimum to simplify packet processing.

Xpress Transport Protocol (XTP) is a next generation protocol, designed to be used in high-speed networks and for multimedia applications. It meets the data transfer requirements of many real time systems and has flexible transfer capabilities, in conjunction with various underlying high performance network technology. It provides good support for event-type distributed systems, in both LAN and other internetwork topologies. XTP has been attracting, as a most suitable protocol for high speed network applications. As it provides a wide range of options to the applications to select the different services they need, which makes XTP very attractive particularly for multimedia and other applications, when constrained by the limitations of current protocols.

XTP is able to increase the performance due to its efficient control and error handling algorithms. When the network is more congested or in high speed networks, XTP shows significant higher throughput, maintaining the bandwidth under control and minimizing the CPU utilization. And also this protocol is designed so as to provide more flexibility to distributed applications, supports priority, transmission control rate, selective retransmission, works under UDP with reliable datagram, has rate and burst control, error and flow control, selective acknowledgement, which makes another suitable protocol for high performance computing.

3.9. Summary

Loss-based congestion control algorithms, like HS-TCP and Scalable TCP use packet loss as primary congestion signal, increase window size for each ACK and decrease window size for packet loss. The advantage of delay-based algorithms is that they achieve better average throughput, since they can keep the system around full utilization. As a comparison, the loss based algorithms purposely generate packet losses and oscillate between full utilization and under utilization. However, existing delay-based algorithms suffer from some inherent weaknesses. First, they are not compatible with standard TCP. FAST TCP yields non-unique equilibrium point if competing with other TCP variants. TCP Illinois gives the option to switch over standard TCP, but it increases the throughput slowly. CTCP has a very good embedded congestion control and avoidance algorithm in standard TCP, takes care of fairness value, has increased throughput, but fails in high speed high bandwidth networks and particularly for multimedia applications. XTP comes in picture, where it supports and delivers good transfer rate for multimedia applications in high speed networks. Table 5 presents the summary of TCP based protocols for high performance grid computing environment.

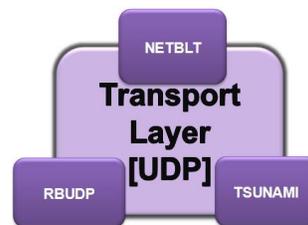
Protocols	Contributors	Year	Changes Done	Remarks
<i>S-TCP</i>	Tom Kelly	2003	Sender Side congestion Control	Exploration is needed
<i>HS-TCP</i>	E. Souza, D. Agarwal	2004	Congestion Control	Loss ratio is high
<i>TCP Africa</i>	Ryan King, Richard Baraniuk, Rudolf Riedi	2005	Delay based CC	Simulation Results using ns2.27 + More overhead than TCP Reno
<i>Fast TCP</i>	Cheng Jin et.al.	2006	Delay based CC	No realistic experiments

<i>TCP Illinois</i>	S. Liu, T. Basar, and R. Srikant	2007	C-AIMD Algorithm	Only Simulated Results using NS2 & Heterogenous users
<i>CTCP</i>	Tan, K. Song, J. Zhang, Q. Sridharan, M.	2007	synergy of delay-based and loss-based Congestion Control	Implemented in windows OS+ effectively utilizes the link capacity
<i>CUBIC TCP</i>	Sangtae Ha, Injong Rhee, Lisong Xu	2008	Modifies the window value using a cubic function	Linux Kernel 2.6.23.9
<i>XTP</i>	Diogo R. Viegas and Rodrigo Mario A. R. Dantas and Michael A. Bauer	2009	Implements multicast options in standard protocol	Multicast Communication Protocol

TABLE 5: Summary of TCP Based Protocols for HPC

4. UDP BASED PROTOCOLS FOR GRID NETWORKS

UDP-based protocols provide much better portability and are easy to install. Although implementation of user level protocols needs less time to test and debug than in kernel implementations, it is difficult to make them as efficient, because user level implementations cannot modify the kernel code, there may be additional context switches and memory copies. At high transfer speeds, these operations are very sensitive to CPU utilization and protocol performance. In fact, one of the purposes of the standard UDP protocol is to allow new transport protocols to be built on top of it. For example, the RTP protocol is built on top of UDP and supports streaming multimedia. In this section we study some UDP based transport protocol for data intensive grid applications. Figure 3 gives some of the UDP variants that are analyzed in this paper.

**FIGURE 3:** UDP Variants

4.1 NETBLT

Bulk data transmission is needed for more applications in various fields and it is must for grid applications. The major performance concern of a bulk data transfer protocols is high throughput. In reality, achievable end-to-end throughput over high bandwidth channels is often an order of magnitude lower than the provided bandwidth. This is because it is often limited by the transport protocols mechanism, so it is especially difficult to achieve high throughput and reliable data transmission across long delay, unreliable network paths.

Throughput is often limited by the transport protocol and its flow control mechanism and it is difficult to achieve high throughput, reliable data transmissions across long delay network paths. To design a protocol, with high throughput, robust in long delay networks, a new flow control mechanism has to be implemented. NETBLT checks the validity of the rate based flow control mechanism, and gains implementation and testing experience with rate control.

Generally speaking, errors and variable delays are two barriers to high performance for all transport protocols. A new transport protocol, NETBLT [42, 43], was designed for high throughput, bulk data transmission application and to conquer the two barriers. NETBLT was first proposed in 1985 (RFC 969). The primary goal of the NETBLT is to get high throughput and robust in a long delay and high loss of network. Seeing the problems with window flow control

and timers, NETBLT builders decided that the goal can be achievable only by employing a new flow control mechanism by implementing and testing experience with rate control.

NETBLT works by opening a connection between two clients (the sender and the receiver) transferring data in a series of large numbered blocks (buffers), and then closing the connection. NETBLT transfer works as follows: the sending client provides a buffer of data for the NETBLT layer to transfer. NETBLT breaks the buffer up into packets and sends the packets using the internet datagram's. The receiving NETBLT layer loads these packets into a matching buffer provided by the receiving client. When the last packet in that buffer has arrived, the receiving NETBLT part will check to see if all packets in buffer have been correctly received or if some packets are missing. If there are any missing packets, the receiver requests to resend the packets. When the buffer has been completely transmitted, the receiving client is notified by its NETBLT layer. The receiving client disposes the buffer and provides a new buffer to receive more data. The receiving NETBLT notifies the sender that the new buffer is created for receiving and this continues until all the data has been sent. The experimental results show that NETBLT can provide the performance predictability than in parallel TCP.

NETBLT protocol is tested in three different networks (RFC1030). The first network is a token ring with 10Mbit/second, served as a reference environment for possible performance. The second network is Ethernet with better performance and third in high bandwidth network. NETBLT's performance in Ethernet allowed the researchers to account it for high bandwidth network. The test used several parameters such as: i) burst interval, ii) burst size, iii) buffer size, iv) data packet size and v) number of outstanding buffers. Table 6 gives the performance details of NETBLT protocols, in three different networks.

Networks →	Token Ring	Ethernet	Wideband
Transfer Size	2 – 5 Million Bytes	2 – 5 Million Bytes	500 – 750 Kilobytes
Data Packet Size	1990 Bytes	1438 Bytes	1432 Bytes
Buffer Size	19900 Bytes	14380 Bytes	14320 Bytes
Burst Size	5 Packets	5 Packets	5 Packets
Burst Interval	40 ms	30 ms	55ms

TABLE 6: Performance test results for NETBLT

In NETBLT, one important area needs to be explored, i.e. dynamic selection and control of NETBLT's transmission parameters (burst size, burst interval) has to be studied. Some research work on dynamic rate control is going on, but more work needs to be done before integrating NETBLT in high performance computing.

4.2 Reliable Blast UDP (RBUDP)

RBUDP [44] is designed for extremely high bandwidth, dedicated or quality of service enabled networks, which require high speed bulk data transfer which is an important part. RBUDP has two goals: i) keeping the network buffer full during data transfer and ii) avoiding TCP's per packet interaction and sending acknowledgements at the end of a transmission.

Figure 4 shows the data transfer scheme of RBUDP. In the data transfer phase (1 to 2 in figure 4 on sender side) RBUDP sends the entire payload to the receiver, using the receiver specified rate. Since the full payload is sent using UDP which is an unreliable protocol, some datagram's may be lost, therefore the receiver has to keep a tally of datagram's received in order to determine which has to be retransmitted. At the end, the sender sends an end signal by sending 'DONE' via TCP (3 in figure 4 on receiver side) to the receiver and receiver acknowledges by sending a total number of received datagram's sequence numbers to the sender (4 in figure 4 on sender side). The sender checks the acknowledgement and resends the missing datagram's to the receiver.

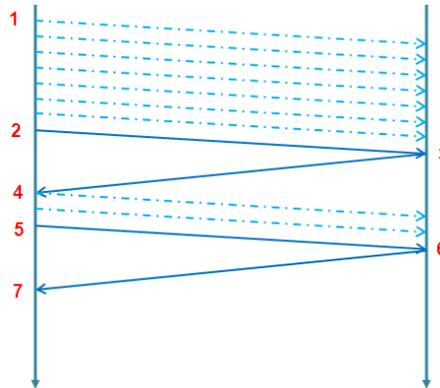


FIGURE 4: Time Sequence Diagram of RBUDP

In RBUDP, the most important input parameter is the sending rate of UDP blasts. To minimize loss, the sending rate should not be larger than the bandwidth of the bottleneck link. Tools such as Iperf [45] and Netperf [46] are typically used to measure the bottleneck bandwidth. There are 3 version of RBUDP available:

- i) Version 1: without scatter/gather optimization - this is naive implementation of RBUDP where each incoming packet is examined and then moved.
- ii) Version 2: with scatter/gather optimization this implementation takes advantage of the fact that most incoming packets are likely to arrive in order, and if transmission rates are below the maximum throughput of the network, packets are unlikely to be lost.
- iii) Version 3: Fake RBUDP this implementation is the same as the scheme without the scatter/gather optimization except that the incoming data is never moved to application memory.

The implementation result of RBUDP shows that it performs very efficiently over high speed, high bandwidth, and Quality of Service enabled networks such as optically switched network. Also through mathematical modeling and experiments, RBUDP has proved that it effectively utilizes available bandwidth for reliable data transfer.

4.3 TSUNAMI

A reliable transfer protocol, Tsunami [47], is designed for transferring large files fast over high speed networks. Tsunami is a protocol which uses inter-packet delay for adjusting the rate control instead of sliding window mechanism. UDP is used for sending data and TCP for sending control data. The goal of Tsunami is to increase the speed of file transfer in high speed networks that use standard TCP. The size of the datagram is avoided in the initial setup and length of file is exchanged in the initial step [connection setup]. A thread is created at each end [sender and receiver], which handles both network and disk activity. The receiver sends a retransmission signal which has higher priority, at the time of datagram loss and at last it sends an end signal. In Tsunami, the main advantage is, the user can configure the parameters such as size of datagram, threshold error rate, size of transmission queue and acknowledgement interval time. The initial implementation of the Tsunami protocol consists of two user-space applications, a client and a server. The structure of these applications is illustrated in Figure 5.

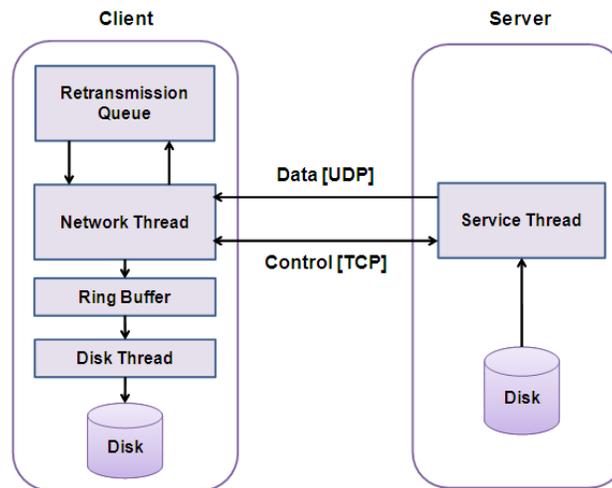


FIGURE 5: Tsunami Architecture

During a file transfer, the client has two running threads. The network thread handles all network communication, maintains the retransmission queue, and places blocks that are ready for disk storage into a ring buffer. The disk thread simply moves blocks from the ring buffer to the destination file on disk. The server creates a single thread in response to each client connection that handles all disk and network activity. The client initiates a Tsunami session by connecting it to the TCP port of the server. Upon connection, the server sends a random data to the client. The client checks the random data by using XOR with a shared secret key and calculates a MD5 check sum, then transmits it to the server. The server does the same operation and checks the check sum and if both are same, the connection is up. After performing the authentication and connection steps, the client sends the name of file to the server. In the server side, it checks whether the file is available and if it is available, it sends a message to client. After receiving a positive message from server, client sends its block size, transfer rate, error threshold value. The server responds with the receiver parameters and sends a time-stamp. After receiving the timestamp, client creates a port for receiving file from the server and server sends the file to the receiver.

Tsunami is an improvement of Reliable Blast UDP in two points. First, Tsunami receiver makes a retransmission request periodically (every 50 packets) and it doesn't wait until finishing of all data transfer, then it calculates current error rate and sends it to the sender. Second, Tsunami uses rate based congestion control. Tsunami has best performance in networks with limited distance, when it comes to long distance network, bandwidth utilization goes down, absence of flow control affects its performance and issues like fairness and TCP friendliness has to be studied.

4.4 Summary

Three UDP-based transfer protocols have much better performance than TCP. Among them, Tsunami has the best performance and RBUDP also performs well when file size is small in size. However, these protocols lack in some ways. First the bandwidth utilization is comparatively low (<70%) which makes not suitable for high speed long distance networks. Second, both RBUDP and Tsunami doesn't estimate bandwidth and uses some third party tools such as Iperf and Netperf. RBUDP has no congestion control and flow control. Tsunami has no flow control, which makes more packet loss in receiver side because of not knowing the receiver's receiving capacity. These existing UDP-based protocols are still to be enhanced in many ways, first, protocol with high throughput for bulk data is to be designed. Then, the issues like fairness and TCP-friendliness are to be considered. Table 7 presents the summary of UDP based protocols for high performance grid computing environment.

Protocols	Contributors	Year	Changes Done	Remarks
NETBLT	David D Clark, Mark L Lambert, Lixia Zhang	1987	New flow control mechanism	High Throughput
RBUDP	Eric He, Jason Leigh, Oliver Yu, Thomas A. DeFanti	2002	SACK + pipe full	Depends on lperf & Netperf
TSUNAMI	Mark R. Meiss	2002	Threads	User Level ease, no flow control

TABLE 7: Summary of UDP Based Protocols for HPC

5. APPLICATION LAYER BASED PROTOCOLS FOR GRID NETWORKS

The need for high performance data transfer services is becoming more and more critical in today's distributed data intensive computing applications, such as remote data analysis and distributed data mining. Although efficiency is one of the common design objectives in most network transport protocols, efficiency often decreases as the bandwidth delay product (BDP) increases. Other considerations, like fairness and stability, make it more difficult to realize the goal of optimum efficiency. Another factor is that many of today's popular protocols were designed when bandwidth was only counted in bytes per second, so performance was not thoroughly examined in high BDP environments. Implementation also becomes critical to the performance as the network BDP increases. A regular HTTP session sends several messages per second, and it does not matter when message processing is delayed for a short time. Whereas, in data intensive applications, the packet arrival speed can be as high as 100 packets per seconds and any such delay matters. The protocol needs to process each event in a limited amount of time and inefficient implementations can lead to packet loss or time-outs. People in the high performance computing field have been looking for application level solutions. One of the common solutions is to use parallel TCP connections [48] and tune the TCP parameters, such as window size and number of flows. However, parallel TCP is inflexible because it needs to be tuned on each particular network scenario. Moreover, parallel TCP does not address fairness issues. In this section we review some application level protocols for high performance computing purposes especially for grid applications. Figure 6 gives some of the applications layer protocols analyzed in this paper.

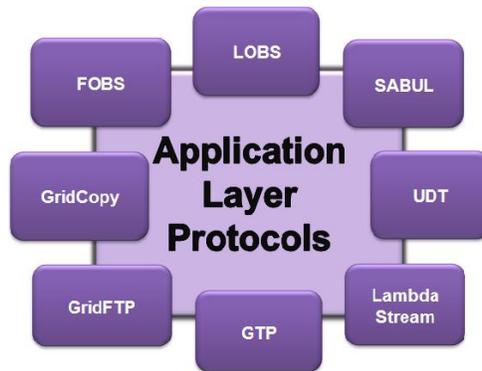


FIGURE 6: Application Layer Variants

5.1. Fast Object Based File Transfer System (FOBS)

A FOBS is an efficient, application level data transfer system for computer grids [49]. TCP is able to detect and respond to network congestion and because of its aggressive congestion control mechanism results in poor bandwidth utilization even when the network is lightly loaded.

FOBS is a simple, user level communication mechanism designed for large scale data transfers in the high bandwidth, high delay network environment typical of computational Grids. It uses

UDP as the data transport protocol, and provides reliability through an application level acknowledgement and retransmission mechanism.

FOBS employs a simple acknowledgement and retransmission mechanism where the file to be transferred is divided into data units called chunks. Data is read from the disk, transferred to the receiver and writes in the disk in units of chunks. Each chunk is subdivided into segments, and the segments are further subdivided into packets. Packets are 1470 bytes (within the MTU of most transmission medium), and a segment consists of 1000 packets. The receiver maintains a bitmap for each segment in the current chunk depicting received / not received status of each packet in the segment. These bitmaps are sent from the data receiver to the data sender at intervals dictated by the protocols and triggers (at a time determined by the congestion / control flow algorithm) a retransmission of the lost packets. The bitmaps are sent over a TCP sockets.

FOBS is a lightweight application level protocol for grids. It is UDP based and acts as a reliable transfer protocol. FOBS uses rate controlled congestion control mechanism, which reduces packet loss and outperforms TCP. For the further usage in high performance computing, congestion control mechanism can be still enhanced to give optimal performance.

5.2. Light Object Based File Transfer System (LOBS)

LOBS is a mechanism for transferring file in high performance computing networks which are optimized for high bandwidth delay network, especially for computational grids. Heart of this environment is the ability of transferring vast data in an efficient manner. LOBS rectify the problem found in the TCP for the data transfer mechanism for grid based computations. In LOBS the increase in performance optimization is done and the order of data delivered is not considered. A LOBS is built directly on top of FOBS [50]. The TCP window size plays a vital role for achieving best performance in high bandwidth delay networks; this leads to tune the size at runtime. In LOBS, the size of TCP window is tuned using different approach, i.e. using UDP stream. UDP is used because of these reasons: i) user level not kernel level, ii) to avoid multiplexing TCP streams in kernel level and iii) to provide user level enhancements.

The basic working concept of LOBS is, it creates threads in sender part for controlling its data buffer, to read file from the disk and fills the data buffer. Once the buffer is full, it is transferred to the client over the network. When the data is in transfer mode, the other threads start reading the data from the file and fill the data buffer and the steps are repeated again until the full file is transferred. In LOBS, the goal is to make use of network I/O operation and the disk I/O operation to the largest extent as possible.

Two protocols closely related to LOBS are RBUDP [44] and SABUL [51, 52]. Primary differences between these two protocols are how loss of packet is interpreted and how to minimize the packet loss impacts affecting the behavior of the protocol. SABUL assumes that packet losses indicate congestion, and it reduces the rate based on the perceived congestion, where as in LOBS it is assumed that some packet loss is inevitable and does not make any changes in the sender rate. Primary difference between LOBS and RBUDP is based on the type of network for which the protocols is designed.

LOBS is used to transfer large files between two computational resources in a grid and this mechanism is lightweight and has lesser functionalities than GridFTP [59]. LOBS supports the primary functionality required for computation grids (i.e. fast and robust file transfer).

5.3. Simple Available Bandwidth Utilization Library (SABUL)

SABUL [51, 52] is an application level library, designed for data intensive grid application over high performance networks and to transport data reliably. SABUL uses UDP for the data channel and it detects, retransmits dropped packets. Using TCP as a control channel reduces the complexity of reliability mechanism. To use available bandwidth efficiently, SABUL estimates the bandwidth available and recovers from congestion events as soon as possible. To improve performance, SABUL does not acknowledge every packet, but instead acknowledges packets at

constant time interval which is called as selective acknowledgement [53]. SABUL is designed to be fair, so that grid applications can employ parallelism and also designed in a way so that all flows ultimately reach the same rate, independent of their initial sending rates and of the network delay. SABUL is designed as an application layer library so that it can be easily deployed without any changes in operating systems network stacks or to the network infrastructure.

SABUL is a reliable transfer protocol with loss detection and retransmission mechanism. It is light in weight with small packet size and less computation overhead, hence can be deployed easily in public networks and is also TCP friendly. In SABUL, both the sender and the receiver maintain a list of the lost sequence numbers sorted in ascending order. The sender always checks the loss list first when it is time to send a packet. If it is not empty, the first packet in the list is resent and removed; otherwise the sender checks the number of unacknowledged packets flow size, and if not, it packs a new packet and sends it out. The sender then waits for the next sending time decided by the rate control. The flow window serves the job of limiting the number of packet loss upon congestion when TCP control reports about the occurrence of delay and the maximum window size is set as $Bandwidth * RTT$ (use SYN instead of RTT if $SYN > RTT$).

After each constant synchronization (SYN) time, the sender triggers a rate control event that updates the inter packet time. The receiver receives and reorders data packets. The sequence numbers of lost packets are recorded in the loss list and removed when the resent packets are received. The receiver sends back ACK periodically if there is any newly received packet. The ACK interval is the same as SYN time. The higher the throughput the less ACK packets generated. NAK is sent once loss is detected. The loss will be reported again if the retransmission has not been received after $k*RTT$, where k is set to 2 and is incremented by 1 each time the loss is reported. Loss information carried in NAK is compressed, considering that loss is often continuous. In the worst case, there is 1 ACK for every received DATA packet if the packet arrival interval is not less than the SYN time, there are $M/2$ NAKs when every other DATA packet gets the loss for every M sent DATA packets.

5.4. UDP based Data Transfer Protocol (UDT)

UDT [54, 55] is a high performance data transfer and is an alternative data transfer protocol for the TCP when its performance goes down. Goal of UDT is to overcome TCP's inefficiency in BDP networks and in connection oriented unicast and duplex networks. The congestion control module is an open source, so that different control algorithms can be deployed apart from native or default control algorithm based on AIMD.

UDT is more complex than Reliable Blast UDP and Tsunami, but similar to TCP. UDT which is based on UDP adds reliability, congestion control and flow control mechanism. For reliability, UDT uses selective acknowledgement (SACK) at a fixed interval and sends negative acknowledgement (NACK) once a loss is detected. In case of congestion control, UDT adopts DAIMD (AIMD with decreasing increase) algorithm to adjust sending rate. To estimate the link capacity it uses receiver based packet pairs and flow control to limit the number of unacknowledged packets.

Rate control tunes the inter-packet time at every constant interval, which is called SYN. The value of SYN is 0.01 seconds, an empirical value reflecting a trade off among efficiency, fairness and stability. For every SYN time, the packet loss rate is compared with the last SYN time, when it is less than a threshold then the maximum possible link Bit Error Rate (BER) and the number of packets that will be sent in the next SYN time is increased by the following equation,

$$tmc = \max \left(10^{\log_{10}(B-C)-MTC-S} * \frac{\beta}{MTC} * \frac{1}{MTU} \right)$$

where B is the estimated bandwidth and C is the current sending rate, both in number of packets per second, β is a constant value of 0.0000015. MTU is the maximum transmission unit in bytes, which is the same as the UDT packet size. The inter-packet time is then recalculated using the total estimated number of sent packets during the next SYN time. The estimated bandwidth B is

probed by sampling UDT data packet pairs and is designed for scenarios where numbers of sources share more bandwidth than expected. In other scenarios, e.g., messaging or low BDP networks, UDT can still be used but there may be no improvement in performance.

UDT adopts complex congestion control algorithm and flow control algorithm which makes not suitable for bulk data transfer in dedicated networks and fairness issue, TCP friendliness also has to be studied.

5.5. Lambda Stream

Network researchers have reached a consensus that the current TCP implementations are not suitable for long distance high performance data transfer. Either TCP needs to be modified radically or new transport protocols should be introduced. Long Fat Networks (LFNs), where the round-trip latencies are extremely high, this latency results in gross bandwidth under-utilization when TCP is used for data delivery. Several solutions have been proposed, one such solution is to provide revised versions of TCP with better utilization of the link capacity. Another solution is to develop UDP-based protocols to improve bandwidth usage. The Simple Available Bandwidth Utilization Library (SABUL [51], Tsunami [47], Reliable Blast UDP (RBUDP) [44] and the Group Transport Protocol (GTP) [58] are few recent examples.

LambdaStream is an application-layer transport protocol [56]. Correspondingly, key characteristics of LambdaStream include a combination loss recovery and a special rate control, which avoids packet loss inherent in other congestion control schemes. To efficiently utilize bandwidth and quickly converge to a new state, the protocol sets the initial sending rate as the quotient of the link capacity over the maximum number of flows, which is easily obtained in a dedicated network.

It adapts the sending rate to dynamic network conditions while maintaining a constant sending rate whenever possible. One advantage of this scheme is that the protocol avoids deliberately provoking packet loss when probing for available bandwidth, a common strategy used by other congestion control schemes. Another advantage is that it significantly decreases fluctuations in the sending rate. As a result, streaming applications experience small jitter and react smoothly to congestion. Another important feature is that the protocol extends congestion control to encompass an end-to-end scope. It differentiates packet loss and updates the sending rate accordingly, thus increasing throughput.

LambdaStream builds on experiences from a high performance networking protocol called Reliable Blast User Datagram Protocol (RBUDP) which transports data using UDP and TCP for control packets. In LambdaStream, the congestion control scheme is developed in such manner it decreases jitter and improve RBUDP's adaptation to network conditions. LambdaStream is an application-layer library, for two reasons. Firstly, application-layer tool makes development easier and simplifies deployment for testing purposes. Secondly, an application-layer protocol can measure end-to-end conditions as applications actually experience them, allowing the protocol to distinguish packet loss and avoid unnecessarily throttling throughput. The key characteristics of the congestion control in LambdaStream are: it is rate based [57], it uses receiving interval as the primary metric to control the sending rate, it calculates rate decrease/increase at the receiver side during a probing phase, and it maintains a constant sending rate after probing for available bandwidth. LambdaStream uses the receiving interval as a metric because 1) the receiving interval is closely related with the link congestion and the receiver's processing capability; 2) the receiving interval can be used to detect incipient congestion and loss differentiation.

The congestion control is composed of two parts. One part is to distinguish a packet loss and adjusts sending rate accordingly, thus avoiding unnecessarily throttling of the sending rate. Another part is to update the sending rate based on the ratio between the average receiving interval and the sending interval. Incipient congestion leads to a higher ratio, which triggers the protocol to decrease the sending rate. The protocol increases its sending rate if the ratio is close to one and the available bandwidth is greater than zero.

LambdaStream extends the congestion control to encompass an end-to-end scope. It distinguishes packet loss and adjusts the sending rate accordingly. The protocol also applies a ratio sampling approach to detect incipient congestion and combines it with a bandwidth estimation method for *proactively* probing for an appropriate sending rate.

LambdaStream protocol's throughput converges very well for a single flow, even for the initial sending rate in between 1720Mbps or 172Mbps. The protocol manages to maintain the throughput at an almost fixed sending rate, about 950Mbps. The experimental results show that LambdaStream achieves 950Mbps throughput in a 1Gbps channel. It exhibits small application level jitter and reacts smoothly to congestion, which is very suitable for streaming applications and also works well for continuous data streams of varying payloads.

5.6. Group Transport Protocol

Group Transport Protocol (GTP), is a receiver-driven transport protocol that exploits information across multiple flows to manage receiver contention and fairness. The key novel features of GTP include 1) request-response based reliable data transfer model, flow capacity estimation schemes, 2) receiver-oriented flow co-scheduling and max-min fairness rate allocation, and 3) explicit flow transition management.

Group Transport Protocol (GTP) is designed to provide efficient multipoint-to-point data transfer while achieving low loss and max-min fairness among network flows [58]. In a multipoint-to-point transfer pattern, multiple endpoints terminate at a receiver and aggregate a much higher capacity than the receiver can handle. In a sender-oriented scheme (e.g. TCP), this problem is more severe because the high bandwidth-delay product of the network makes it difficult for senders to react to congestion in a timely and accurate manner. To address this problem, GTP employs *receiver-based* flow management, which locates most of the transmission control at the receiver side, close to where packet loss is detected. Moreover, a receiver-controlled rate-based scheme in GTP, where each receiver explicitly tell senders the rate at which they should follow, allow flows to be adjusted as quickly as possible in response to detected packet loss.

In order to support multi-flow management, enable efficient and fair utilization of the receiver capacity, GTP uses a receiver-driven centralized rate allocation scheme. In this approach, receivers actively measure progress (and loss) of each flow, estimate the actual capacity for each flow, and then allocate the available receiver capacity fairly across the flows. Because GTP is receiver-centric rate-based approach, it manages all senders of a receiver, and enables rapid adaptation to flow dynamics by adjusting, when flows join or terminate.

GTP is a *receiver-driven response-request* protocol. As with a range of other experimental data transfer protocols, GTP utilizes light-weight UDP (with additional loss retransmission mechanism) for bulk data transfer and a TCP connection for exchanging control information reliably. The sender side design is simple: send the requested data to receiver at the receiver-specified rate (if that rate can be achieved by sender). Most of the management is at the receiver side, which includes a *Single Flow Controller* and *Single Flow Monitor* for each individual flow, and *Capacity Estimator* and *Max-min Fairness Scheduler* for centralized control across flows.

GTP implements two levels of flow control. For each individual flow, the receiver explicitly controls the sender's transmission rate (by sending rate requests to senders). This allows the flow's rate to be adjusted quickly in response to packet loss (detected at the receiver side). Ideally any efficient rate-based point-to-point flow control scheme could be 'plugged in' and it acts as a centralized scheduler. The scheduler at the receiver manages across multiple flows, dealing with any congestion or contention and performing max-min rate amongst them. The receiver actively measures per-flow throughput, loss rate, and uses it to estimate bandwidth capacity. It then allocates the available receiver capacity (can be limited by resource or the final link) across flows. This allocation is done once for each control interval in Max-min fair manner. Correspondingly, the senders adjust to transmit at the revised rates.

Results got after implementation of GTP shows that for point-to-point single flow case, GTP performs well, like other UDP-based aggressive transport protocols (e.g. RBUDP, SABUL), achieving dramatically higher performance than TCP with low loss rates. In case of multipoint-to-point, GTP still achieves high throughput with 20 to 100 times lower loss rates than other aggressive rate-based protocols. In addition, simulation results show, unlike TCP, which is unfair to flows with different RTT, GTP responds to flow dynamics and converge to max-min fair rate allocation quickly.

GTP outperforms other rate based protocols, for multipoint-to-point data transmission, GTP reduces the packet loss, which are caused by aggressiveness of rate based protocols. GTP focus on Receiver based flow contention management, however detailed rate allocation and fairness among flow are not yet considered. Some works has to be done regarding GTP's performance such as achieving max-min fairness, estimating TCP flow's capability and tuning TCP parameters to achieve target rate [57]. Table 8 presents the results for GTP, compared with other rate based protocols such as RBUDP and SABUL. Table 9 displays the data flow statistics for GTP and compares with other protocols.

Protocol	RBUDP	SABUL	GTP
Initial Rate	User defined	Fixed rate	negotiated by sender and receiver
Multipoint to point	No	No	Yes
Rate Adaptation	Optional	Rate based with delay compensation	Rate estimation and delay compensation
Fairness among flows	Not considered	Some extent	max-min fairness among flows at receiver side

TABLE 8: Comparison of GTP with other Rate Based Protocols

Protocol	RBUDP		SABUL		GTP	
	Parallel	Multi-flows	Parallel	Multi-flows	Parallel	Multi-flows
Aggregate Rate (Mbps)	931	443	912	811	904	865
Average Loss	2.1%	53.3%	0.1%	8.7%	0.03%	0.06%
System Stability	Yes	No	Yes	No	Yes	Yes
Fairness	Fair	No	Fair	No	Fair	Yes

TABLE 9: Parallel vs. Convergent Flow Statistics for GTP

5.7. GridFTP

A common data transfer protocol for grid would ideally offer all the features currently available from any of the protocols in use. At a minimum, it must offer all of the features that are queried for the types of scientific and engineering applications that are intended to support in the grid. For this the existing FTP standard is selected, by adding some features a common data transfer protocol, 'GridFTP' is developed [59]. GridFTP is used as a data transfer protocol for transferring a large volume of data in grid computing. It adopts parallel data transfer which improves the throughput by creating multiple TCP connections in parallel and automatic negotiation of TCP socket buffer size. GridFTP uses TCP as its transport-level communication protocol [60]. In order to get maximal data transfer throughput, it has to use optimal TCP send and receive socket buffer sizes for the link being used. TCP congestion window never fully opens if the buffer size is too small. If the receiver buffers are too large, TCP flow control breaks, and the sender can overrun the receiver, thereby causing the TCP window to shut. This situation is likely to happen if the sending host is faster than the receiving host. The optimal buffer size is twice the bandwidth delay product (BDP) of the link is $Buffer\ size = 2 * bandwidth\ delay$.

The GridFTP is implemented in Globus [61] and uses multiple TCP streams for transferring file. Using multiple TCP streams improves performance because of these reasons: i) aggregate TCP buffer size which is closer to real size and ii) circumvents the congestion control.

Several experiments were done for analyzing GridFTP [62]. According to a technical report [63] globus_url_copy achieved a throughput very close to 95%. The windows size was set to bandwidth*RTT, when more than one TCP streams are used, then the window size was set to windows size * num streams. However, to achieve high throughput, the number of TCP connections has to be optimized according to network condition. Problems persist in the file sizes, when the end points want to transfer lots of small files, and then the throughput is reduced.

Figure 7 and 8 shows the download performance for various FTP clients with single and multithreaded flow. Figure 9 shows the data transfer performance among various FTP clients. The performance of GridFTP [64, 65] depends on the number of connections used in parallel, the best performance is achieved with 4 connections and when more connections are there, it creates too much control overhead.

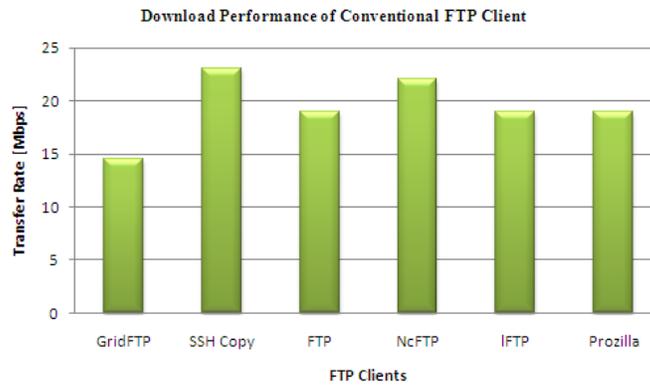


FIGURE 7: Download performance of Conventional FTP Clients [Single Threaded]

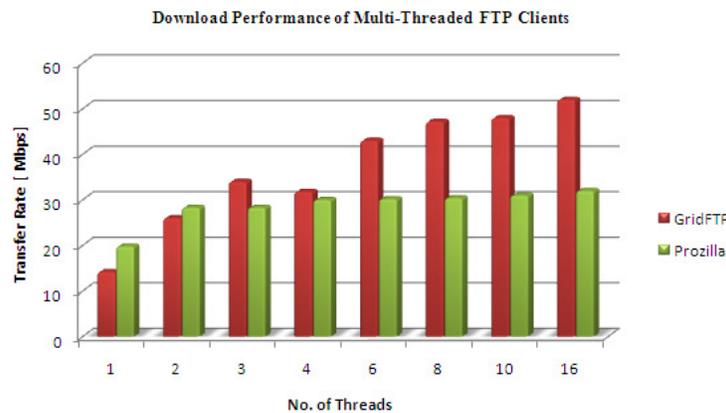


FIGURE 8: Download performance of Multi-threaded FTP Clients

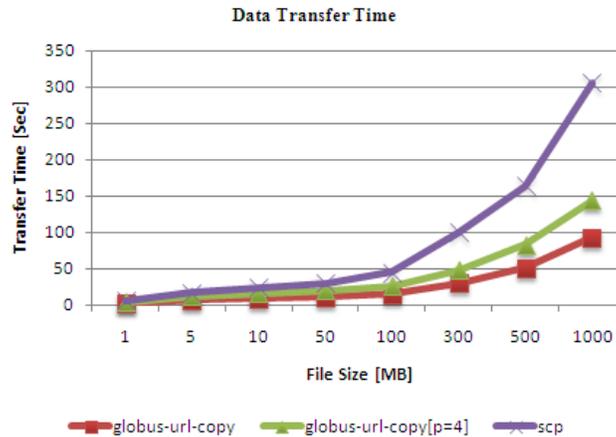


FIGURE 9: Data Transfer Performance of FTP Clients

5.8. GridCopy

GridCopy [66], or GCP, provides a simple user interface to this sophisticated functionality, and takes care of all to get optimal performance for data transfers. GCP accepts scp-style source and destination specifications. If well-connected GridFTP servers can access the source file and/or the destination file, GCP translates the filenames into the corresponding names on the GridFTP servers. In addition to translating the filenames/URLs into GridFTP URLs, GCP adds appropriate protocol parameters such as TCP buffer size and number of parallel streams to attain optimal performance in networks.

Tools such as ping and synack can be used to estimate end-to-end delay; and tools such as IGI [67], abing [68], pathrate [69], and Spruce [70] can be used to estimate end-to-end bandwidth. Latency estimation tools need to be run on one of the two nodes between which the latency needs to be estimated. For data transfers between a client and server, the tools mentioned above can be used to estimate the bandwidth-delay product. However, in Grid environments, users often perform third-party data transfers, in which the client initiates transfers between two servers.

The end-to-end delay and bandwidth estimation tools cited above are not useful for third-party transfers. King [71] developed at the University of Washington at Seattle makes it possible to calculate the round-trip time (RTT) between arbitrary hosts on the Internet. GCP uses King to estimate the RTT between source and destination nodes in a transfer. GCP assumes a fixed one Gbits bandwidth for all source and destination pairs. King estimates RTT between any two hosts in the internet by estimating the RTT between their domain name servers. For example, if King estimates the RTT between the source and the destination to be 50 ms, GCP sets the TCP buffer size to 0.05. GCP caches the source, destination, and buffer size in a configuration file which is available in the home directory of the user running GCP. By default, GCP uses four parallel streams for the first transfer between two sites by a user. GCP calculates the TCP buffer size for each stream by $(BDF/\max(1, streams/L_f))$, where L_f is set to 2 by default to accommodate for the fact that the streams that are hit by congestion would go slower and the streams that are not hit by congestion would go faster.

The primary design goal for GCP are i) to provide a scp-style interface for high performance, reliable, secure data transfers, ii) to calculate the optimal TCP buffer size and optimal number of parallel TCP streams to maximize throughput and iii) to support configurable URL translations to optimize throughput.

5.9. Summary

Table 10 presents the summary of application layer protocols for high performance grid computing environment.

Protocols	Contributors	Year	Changes Done	Remarks
FOBS	Phillip M. Dickens	2003	Object Based	Less functionality
LOBS	Phillip M. Dickens	2003	Object Based & above FOBS	Least Functionality
SABUL	Yunhong Gu et. al.	2008	Change in flow control	Application Level Library
UDT	Yunhong Gu et. al.	2009	Rate control + MAIMD	More functionality
Lambda Stream	Chaoyue Xiong et. al.	2005	bandwidth estimation method [proactive]	more bandwidth utilization +small application level jitter
GTP	Ryan Wu et. al.	2004	rate based flow control	Utilizing available bandwidth fastly.
GridFTP	W. Allcock et. al.	2008	Parallel TCP	No User Level Ease
GridCopy	Rajkumar Kettimuthu	2007	Parallel TCP + SCP	User Level Ease

TABLE 10: Summary of Application Layer Protocols for HPC

6. CONCLUSION

A detailed study of the most recent developments on network protocols for grid computing environment is done in this paper. We reviewed the protocols based on TCP and UDP. Below are some points which has to be considered when developing high performance protocols grid computing environment, i) using TCP in another transport protocol should be avoided, ii) using packet delay as indication of congestion can be hazardous to protocol reliability, iii) processing continuous loss is critical to the performance and iv) a knowledge of how much CPU time each part of the protocol costs helps to make an efficient implementation. And also, concentrating on three inter-related research tasks namely: i) dynamic right-sizing, ii) high-performance IP, iii) rate-adjusting, iv) better congestion control and avoidance algorithm and v) efficient flow control algorithm can lead to efficient transport protocol for grid computing environment. Table 11 and 12 gives the comparison chart for TCP based protocols. Table 13 compares the various UDP protocols and Table 14 and 15 gives the application layer protocols comparison.

	STCP	HS-TCP	TCP-Africa	Fast TCP
Design	TCP	TCP	TCP	TCP
Mode of Operation	D2D	D2D	D2D	D2D
Security	No	No	No	No
Congestion control	MIMD	Modified	Modified + 2 Mode CA rule adopted	Depends on parameters
Fairness	No	Yes	Improved	Proportional fairness
TCP Friendly	Yes	Yes	Yes	NA
Performance	Improved	window size and recovery time less	Two mode congestion increases the performance	Well performed
Throughput	Increases	Ok	Good	Increases
Bandwidth Utilization	Ok	Low	Acquires available bandwidth	More
Multicast	No	No	No	No
Implementation details	Linux Kernel 2.4.19	Linux Kernel 2.4.16	NA	NA
Usage	High Speed data Transfer	High Speed data Transfer	High Bandwidth Networks	High Speed data Transfer

TABLE 11: Comparison chart for TCP Based Protocols – Part 1

	TCP-Illinois	CTCP	CUBIC TCP	XTP
Design	TCP	TCP	TCP	TCP
Mode of Operation	D2D	D2D, M2M	D2D	D2D
Security	No	No	No	No
Congestion control	Uses C-AIMD Algorithm and use loss and delay for CC.	Combination of Loss and Delay	Modified	Modified
Fairness	Maintains fairness	Improved	Enhanced	Enhanced
TCP Friendly	Yes	Yes	Yes	Yes
Performance	More than TCP	Increased because of good network utilization	Increased because of good network utilization	Increased, more when used in FDDI, Fibre Channels.
Throughput	Better Throughput	Good	Good	High
Bandwidth Utilization	High	Good	Good	Good
Multicast	No	No	No	Yes
Implementation details	NA	Microsoft and implemented [XP]	NA	Developed by XTP Forum
Usage	High speed TCP variant.	High Speed Data Network.	High Speed Networks	High Speed N/w and Multimedia Applications.

TABLE 12: Comparison chart for TCP Based Protocols – Part 2

	NETBLT	RBUDP	Tsunami
Design	UDP	UDP data + TCP control.	UDP data + TCP control.
Mode of Operation	D2D	D2D & M2M	D2D
Security	No	No	Yes
Congestion control	No	Optional. Limited congestion control can be turned on.	Limited. Sending rate is reduced when loss rate is more than a threshold.
Fairness	No	NA	NA
TCP Friendly	No	No	No
Performance	Performed extremely well	eliminates TCP's slow-start and uses full bandwidth	relays on the parameters
Throughput	Ok	Good	Good
Bandwidth Utilization	Fair	Good	Good
Multicast	No	No	No
Implementation details	No	Provides C++ API.	www.indiana.edu
Usage	developed at MIT for high throughput bulk data transfer	Aggressive protocol designed for dedicated or QoS enabled high bandwidth networks.	Designed for faster transfer of large file over high-speed networks.

TABLE 13: Comparison chart for UDP Based Protocols

	FOBS	LOBS	SABUL	UDT
Design	OO Based	FOBS	UDP data + TCP control.	UDP
Mode of Operation	D2D	D2D	D2D & M2M	D2D & M2M
Security	NA	NA	No	Yes
Congestion control	Yes	Yes	Rate based algorithm	D- AIMD
Fairness	Yes	Yes	Fairness is independent to network delay	Fairness of UDT is independent of RTT
TCP Friendly	Yes	Yes	Yes.	Yes
Performance	90% bandwidth utilized	rate of 35MB per second	Ok	Ok
Throughput	NA	Ok	Ok	Good
Bandwidth Utilization	NA	Good	Ok	High
Multicast	No	No	No	No
Implementation details	NA	NA	C++ Library on Linux.	udt.sourceforge.net
Usage	High Speed data Transfer	High Speed data Transfer	A general purpose protocol.	High Speed data Transfer

TABLE 14: Comparison chart for Application Layer Based Protocols – Part 1

	LambdaStream	GTP	GridFTP	Grid Copy
Design	UDP data + TCP control	Light weight UDP	FTP	FTP
Mode of Operation	D2D	D2D	D2D & M2M	D2D & M2M
Security	No	No	GSI	scp style
Congestion control	Rate Based + Modified	Same as TCP	Same as TCP	Same as TCP
Fairness	Yes	Yes	Yes	Yes
TCP Friendly	Yes	Yes	Yes	Yes
Performance	Improved, because of loss recovery and rate control	Improved because of Receiver driven and flow control	Problems persist in file sizes and number of connections	Problems persist in file sizes (small size)
Throughput	Good	Good	Good	Good
Bandwidth Utilization	Good	Good	High	High
Multicast	No	Yes	Yes	Yes
Implementation details	NA	NA	Globus Toolkit	NA
Usage	Long Fat Networks	High Bandwidth Networks	High Bandwidth networks	High Bandwidth networks

TABLE 15: Comparison chart for Application Layer Based Protocols – Part 2

7. REFERENCES

- [1] Foster and C. Kesselman, "*The Grid: Blue print for a new computing infrastructure*", Morgan Kaufmann Publications (1999).
- [2] Foster, C. Kesselman, J. M. Nick and S. Tuecke, "*The physiology of the Grid: An open grid services architecture for distributed systems integration*", Grid Forum white paper, 2003.
- [3] Volker Sander, "*Networking Issues for Grid Infrastructure*", GFD-I.037, Nov, 22, 2004.
- [4] T. DeFanti, C. D. Laat, J. Mambretti, K. Neggers and B. Arnaud, "*TransLight: A global scale LambdaGrid for e-science*", Communications of the ACM, 47(11), November, 2003.
- [5] L. Smarr, A. Chien, T. DeFanti, J. Leigh and P. Papadopoulos, "*The OptIPuter*" Communications of the ACM, 47(11), November, 2003.
- [6] "CANARIE." <http://www.canarie.ca/>
- [7] M. Gaynor, M. Welsh, S. Moulton, A. Rowan, E. LaCombe and J.Wynne, "*Integrating wireless sensor networks with the Grid*", In Proceedings of IEEE Internet Computing, Special Issue on Wireless Grids, July/August, 2004.
- [8] J. Postel, "*Transmission control protocol*", RFC 793, September 1981.
- [9] J. Postel, "*User datagram protocol*", RFC 768, September 1980.
- [10] B. Jacobson, "*TCP extensions for high performance*", RFC 1323, May 1992.
- [11] Douglas .J. Leith, Robert N. Shorten, "*Next Generation TCP: Open Questions*", In Proceedings of International Workshop on Protocols for Fast Long-Distance Networks, Manchester, UK, 2008.
- [12] Ryan X. Wu, Andrew A. Chien et.al., "*A High performance configurable transport protocol for grid computing*", In Proceedings of 5th IEEE International Symposium of Cluster Computing and the grid, Vol.2, pp 1117-1125, 2005.
- [13] D. Katabi, M. Hardley, and C. Rohrs, "*Internet Congestion Control for Future High Bandwidth-Delay Product Environments*", ACM SIGCOMM, Pittsburgh, PA, Aug. 19 - 23, pp 89-102, 2002.
- [14] J. Padhye, V. Firoiu, D. Towsley, and J. Kurose, "*Modeling TCP throughput: a simple model and its Empirical validation*", ACM Technical Report, UM-CS-1998-008, 1998.
- [15] Y. Zhang, E. Yan, and S. K. Dao, "*A measurement of TCP over Long-Delay Network*", In Proceedings of 6th International Conference on Telecommunication Systems, Modeling and Analysis, Nashville, TN, March 1998.
- [16] W. Feng et.al., "*The Failure of TCP in High Performance Computational Grids*", Super Computing, ACM/IEEE Conference, November, pp 37, 2000.
- [17] R.N. Shorten and D.J. Leith and J. Foy and R. Kilduff, "*Analysis and design of AIMD congestion control algorithms in communication networks*", Article in Automatica, 41(4) pp:725-730, doi:10.1016/j.automatica.2004.09.017, 2005.
- [18] Jain, R., Chiu, D.M., and Hawe, W., "*A Quantitative Measure of Fairness and Discrimination for Resource Allocation in Shared Systems*", DEC Research Report TR-301, 1984.

- [19] E. Souza, D. Agarwal, "A HighSpeed TCP study: characteristics and deployment issues", Technical Report LBNL-53215, Lawrence Berkeley National Lab, 2003. Available at: <http://www-itg.lbl.gov/evandro/hstcp/hstcp-lbnl-53215.pdf>.
- [20] T.A. Trinh, B. Sonkoly, S. Molnr, "A HighSpeed TCP study: observations and reevaluation", In Proceedings of 10th Eunice Summer School and IFIP Workshop on Advances in Fixed and Mobile Networks, Tampere, Finland, 2004.
- [21] Tom Kelly, "Scalable TCP: Improving performance in high speed wide area networks", ACM SIGCOMM Computing Communications Review, 33(2), April, pp 83-91, 2003.
- [22] C. Jin, D.X. Wei, S.H. Low, "FAST TCP: motivation, architecture, algorithms, Performance", In Proceedings of IEEE Infocom , Vol. 4, Hong Kong, China, pp. 2490-2501, 2004.
- [23] Gurtov, "Effect of delays on TCP performance", In Proceedings of IFIP Personal Wireless Communications 2001 (PWC2001), Lappeenranta, Finland, pp. 810, 2001.
- [24] Sumitha Bhandarkar, Saurabh Jain and A. L. Narasimha Reddy, "LTCP: Improving the Performance of TCP in HighSpeed Networks", ACM SIGCOMM Computer Communications Review, 36(1), January, pp 41-50, 2006.
- [25] M. Gerla, M. Y. Sanadidi, R. Wang, A. Zanella, C. Casetti, and S. Mascolo, "TCP Westwood: Congestion Window Control Using Bandwidth Estimation", IEEE Globecom 2001, vol: 3, pp 1698-1702, 2001.
- [26] Lawrence S. Brakmo, Student Member, IEEE, and Larry L. Peterson, "TCP Vegas: End to End Congestion Avoidance on a Global Internet", IEEE JOURNAL ON Selected Areas in Communications, 13(8), October 1995.
- [27] M. Fisk and W. Feng, "Dynamic right-sizing in TCP", In Proceedings of International Conference on Computer Communication and Networks, October, pp 152-158, 2001.
- [28] S. Floyd, S. Ratnasamy, and S. Shenker, "Modifying TCPs congestion control for high speeds", Preliminary Draft. URL: <http://www.icir.org/floyd/papers/hstcp.pdf>, 2002.
- [29] S. Floyd, "HighSpeed TCP for Large Congestion Windows", RFC 3629, December 2003.
- [30] King R, Baraniuk R, Riedi R, "TCP-Africa: an adaptive and fair rapid increase rule for scalable TCP", In IEEE Proceedings of IEEE Computer and Communications Societies Conference, vol.3, pp. 1838-1848, doi: 10.1109/INFCOM.2005.1498463, 2005.
- [31] Cheng Jin et.al., "FAST TCP: From Theory to Experiments", IEEE Network Communications, 19(1), pp 4-11, 2005.
- [32] Sally Floyd, Mark Handley, Jitendra Padhye, and Joerg Widmer, "Equation-Based Congestion Control for Unicast Applications", SIGCOMM 2000
- [33] Wei Steven H. Low, Cheng Jin David X, "FAST TCP: Motivation, Architecture, Algorithms, Performance", IEEE/ACM Trans Networking, 14(6), pp 1246-1259, 2006.
- [34] Shao Liu, Tamer Basar, and R. Srikant, "TCP-Illinois: a loss and delay based congestion control algorithm for high-speed networks", In Proceedings of the 1st international conference on Performance evaluation methodologies and tools, ACM, Article 55, doi:10.1145/1190095.1190166, 2006.

- [35] Shao Liu, Tamer Basar, and R. Srikant, "TCP-Illinois: A loss- and delay-based congestion control algorithm for high-speed networks", Performance Evaluation, 65(6-7) pp.417-440. doi:10.1016/j.peva.2007.12.007, 2008.
- [36] Tan, K. Song, J. Zhang, Q. Sridharan, M, "A Compound TCP Approach for High-Speed and Long Distance Networks", Proceedings of 25th IEEE International Conference on Computer Communications, pp:1-12, doi:10.1109/INFOCOM.2006.188, 2006.
- [37] K. Tan, J. Song, M. Sridharan, and C.Y. Ho,, "CTCP-TUBE: Improving TCP friendliness over low-buffered network links", Proceedings of 6th International Workshop on Protocols for FAST Long-Distance Networks, March 2008.
- [38] Alberto Blanc, Konstantin Avrachenkov, Denis Collange and Giovanni Neglia, "Compound TCP with Random Losses", Springer Berlin / Heidelberg, Lecture Notes in Computer Science, Networking, pp:482-494, doi: 10.1007/978-3-642-01399-7_38, 2009.
- [39] Sangtae Ha, Injong Rhee, and Lisong Xu, "CUBIC: a new TCP-friendly high-speed TCP variant", SIGOPS Operating Systems Review 42(5) pp: 64-74, doi:10.1145/1400097.1400105, 2008.
- [40] L. Xu, K. Harfoush, and I. Rhee, "Binary Increase Congestion Control (BIC) for Fast Long-Distance Networks" In Proceedings of IEEE INFOCOM, March 2004.
- [41] Diogo R. Viegas, Rodrigo Mario A. R. Dantas, Michael A. Bauer, "SCTP, XTP and TCP as Transport Protocols for High Performance Computing on Multi-cluster Grid Environments", HPCS, pp:230-240, 2009, doi:10.1007/978-3-642-12659-8_17.
- [42] Mark L. Lambert, Lixia Zhang, "NETBLT: A Bulk Data Transfer Protocol", RFC 969, December 1985.
- [43] David D Clark, Mark L Lamberl, Lixia Zhang, "NETBLT: A High Throughput Transport Protocol", ACM SIGCOMM Computer Communications Review, vol:17, no:5, 1987, pp 353-359, 1987.
- [44] Eric He, Jason Leigh, Oliver Yu, Thomas A. DeFanti, "Reliable Blast UDP: Predictable High Performance Bulk Data Transfer", Fourth IEEE International Conference on Cluster Computing (CLUSTER'02), pp 317, 2002.
- [45] <http://dast.nlanr.net/Projects/lperf/>
- [46] <http://netperf.org/netperf/NetperfPage.html>
- [47] Mark R. Meiss (2008), "Tsunami: A High-Speed Rate Controlled Protocol for File Transfer", Indiana University, <http://steinbeck.ucs.indiana.edu/mmeiss/papers>
- [48] Altman, E. Barman, D. Tuffin, B. Vojnovic, M, "Parallel TCP Sockets: Simple Model, Throughput and Validation", INFOCOM 2006. 25th IEEE International Conference on Computer Communications. Proceedings pp:1 - 12, 2006
doi: 10.1109/INFOCOM.2006.104.
- [49] Phillip M. Dickens, "FOBS: A Lightweight Communication Protocol for Grid Computing", Euro-Par 2003 Parallel Processing Lecture Notes in Computer Science, Volume 2790/2003, pp:938-946, 2003. doi: 10.1007/978-3-540-45209-6_130.
- [50] www.umcs.maine.edu/~dickens/pubs/LOBS.pdpta.submitted.doc

- [51] Yunhong Gu and Robert Grossman, "SABUL: A Transport Protocol for Grid Computing", Journal of Grid Computing, Vol.1, No.4, December, pp 377-386, 2003.
- [52] Phoemphun Oothongsap, Yannis Viniotis, and Mladen Vouk, "Improvements of the SABUL Congestion Control Algorithm", Proceedings of 1st International Symposium on Communication Systems Networks and Digital Signal Processing, July, 2008.
- [53] S. Floyd, M.Mathis, M. Podolsky, "An Extension to the Selective Acknowledgement (SACK) Option or TCP", RFC 2883, July 2000.
- [54] Yunhong Gu and Robert L. Grossman, "UDT: An Application Level Transport Protocol for Grid Computing", Second International Workshop on Protocols for Fast Long-Distance Networks, PFLDNet, Feb 16-17, 2004, Argonne, Illinois, USA
- [55] Yunhong Gu and Robert L. Grossman, "UDT: UDP-based Data Transfer for High-Speed Wide Area Networks", International Journal of Computer & Telecommunications Networks, Vol.51, No.7, May, 2007.
- [56] Chaoyue Xiong, Jason Leigh, Eric He, Venkatram Vishwanath, Tadao Murata, Luc Renambot, Thomas A. DeFanti, "LambdaStream: A Data Transport Protocol for Network-Intensive Streaming Applications over Photonic Networks", In Proceedings of the Third International Workshop on Protocols for Fast Long-Distance Networks (PFLDnet 2005), Lyons, France, February 3-4, 2005.
- [57] Xinran (Ryan) Wu and Andrew A. Chien, "Evaluation of Rate-Based Transport Protocols for Lambda-Grids", Proceedings of the 13th IEEE International Symposium on High Performance Distributed Computing (HPDC '04). IEEE Computer Society, pp:87-96, 2004, doi:10.1109/HPDC.2004.13.
- [58] Wu, R.X., Chien, A.A., "GTP: Group Transport Protocol for Lambda-Grids", IEEE International Symposium on Cluster Computing and the Grid, (CCGrid 2004), pp: 228- 238, 19-22 April 2004, doi: 10.1109/CCGrid.2004.1336572.
- [59] W. Allcock, J. Bester, J. Bresnahan, A. Chervenak (2003), "GridFTP: Protocol Extensions to FTP for the Grid", GFD-20, April 2003.
- [60] John Bresnahan, Michael Link, Gaurav Khanna, Zulfikar Imani, Rajkumar Kettimuthu and Ian Foster, "Globus GridFTP: Whats New in 2007", Proceedings of International Conference on Networks for grid applications, Article 19, October, 2007.
- [61] www.globus.org/toolkit/data/gridftp/
- [62] John Bresnahan, Michael Link, Rajkumar Kettimuthu, Dan Fraser and Ian Foster, "GridFTP Pipelining", Proceedings of the 2007 TeraGrid Conference, June, 2007.
- [63] http://www.globus.org/toolkit/docs/latest-stable/data/gridftp/gridftp_performance.doc
- [64] Hiroyuki Ohsaki, Makoto Imase, "Performance Evaluation of Data Transfer Protocol GridFTP for Grid Computing", International Journal of Applied Mathematics and Computer Sciences 3(1), pp:39-44, 2009.
- [65] M. Cannataro, C. Mastroianni, D. Talia, and P. Trunfio, "Evaluating and Enhancing the Use of the GridFTP Protocol for Efficient Data Transfer on the Grid", in Proc. PVM/MPI, pp.619-628, 2003.

- [66] Kettimuthu, R, Allcock, W, Liming, L, Navarro, J.-P, Foster, I., "GridCopy: Moving Data Fast on the Grid", IEEE International Parallel and Distributed Processing Symposium, IPDPS 2007. pp:1 - 6, doi:10.1109/IPDPS.2007.370553
- [67] <http://www.cs.cmu.edu/~hnn/igi/>
- [68] www.wcisd.hpc.mil/tools/
- [69] <http://www.cc.gatech.edu/fac/Constantinos.Dovrolis/bw-est/>
- [70] www.icir.org/models/tools.html/
- [71] K. P. Gummadi, S. Saroiu, S., and S. D. Gribble, "King: Estimating latency between arbitrary internet end hosts", SIGCOMM Computer Communication Review, vol. 32, no. 3, pp. 518, July 2002.

Phishing: a Field Experiment

Danuvasin Charoen, Ph.D.

NIDA Business School

National Institute of Development Administration

Bangkok, Thailand

danuvasin@nida.ac.th

Abstract

Phishing is a method that hackers use to fraudulently acquire sensitive or private information from a victim by impersonating a real entity [1]. Phishing can be defined as the act of soliciting or stealing sensitive information such as usernames, passwords, bank account numbers, credit card numbers, and social security or citizen ID numbers from individuals using the Internet [2]. Phishing often involves some kind of deception. The results from a study of Jagatic et al. (2007) indicate that Internet users are four times more likely to become phishing victims if they receive a request from someone appearing to be a known friend or colleague. The Anti-Phishing Work Group indicates that at least five percent of users responded to phishing scams and about two million users gave away their information to spoofed websites [3]. This results in direct losses of \$1.2 billion for banks and credit card companies (Dhamija, 2006).

In order to understand how phishing can be conducted, the researcher set up a phishing experiment in one of Thailand's higher education institutions. The subjects were MBA students. A phishing email was sent to the subjects, and the message led the subject to visit the phishing website. One hundred seventy students became victims. The data collection included a survey, an interview, and a focus group. The results indicated that phishing could be easily conducted, and the result can have a great impact on the security of an organization. Organizations can use and apply the lessons learned from this study to formulate an effective security policy and security awareness training programs.

Keywords: Phishing, Computer Crime, Data Security

1. INTRODUCTION

Computers and the Internet have become critical parts of daily life in Thailand, where there has been explosive growth in E-Commerce and in the ICT Market. The total market value for e-Commerce in 2007 was about fourteen billion dollars (National Statistic Office, 2007). Computers and the Internet can provide opportunities for a company to develop competitive advantages, for example, through e-Commerce, online customer services, supply chain management, etc. However, one of the obstacles to e-Commerce growth is the growing concern about computer and Internet-related crimes. Seventy point two percent of E-Commerce customers indicate that they are concerned about the security and privacy of their data when they conduct transactions online [4]. Additionally, customers are worried about giving out their private information on the Internet because of growing concern over computer crime.

In Thailand, computer and Internet-related crimes have become a major concern because the financial loss for the damage can be tremendous. There has been especially strong concern by the industries that utilize computers and the Internet, such as banks and hospitals [5].

In the past, the major concern regarding computer security was malicious software, such as viruses, worms, and spyware. In 2005, this malicious software was a top concern; however, since 2007, the most reported incident was phishing [5]. Phishing is the act or method of tricking users into giving away their private information, including credit card numbers, banking accounts, and usernames and passwords [6]. The methods of phishing include sending phony email messages and duplicating legitimate websites. In 2010, there were at least 48,244 phishing attacks worldwide [7], and an attack is defined as a phishing website that targets a specific company or brand [7].

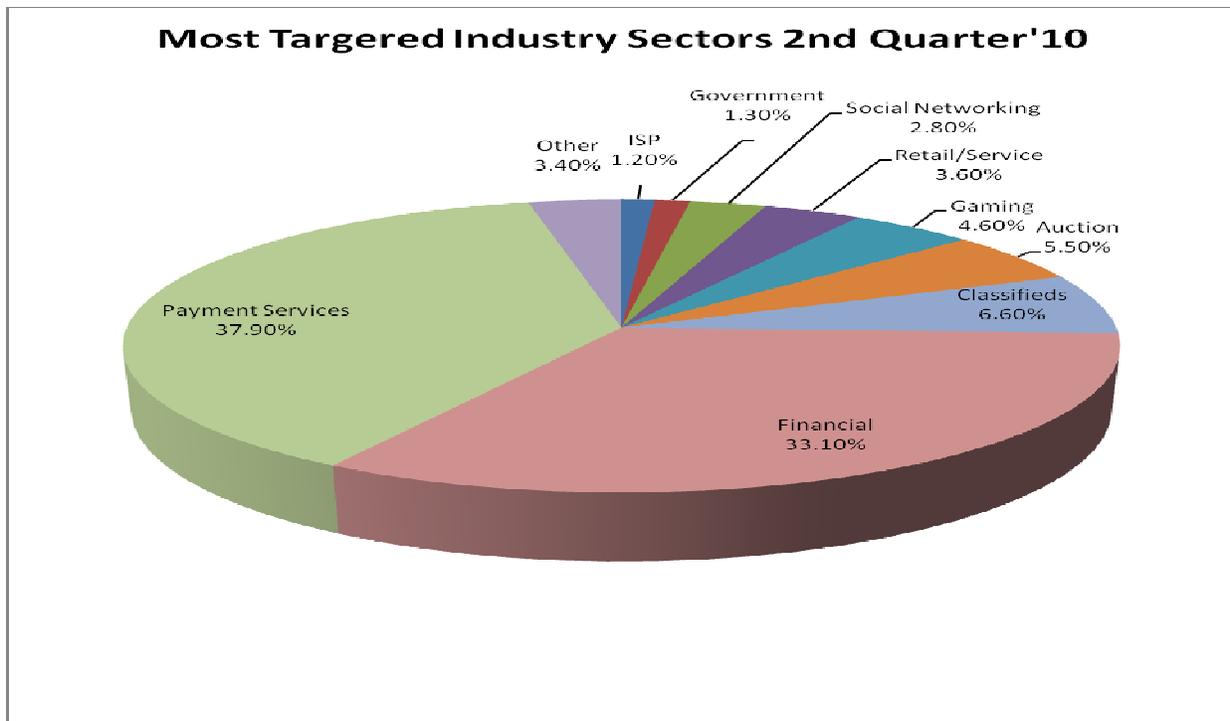


FIGURE 1: Most Targeted Industry Sector [8]

According to the Anti-Phishing Working Group (APWG), the most targeted industry in 2010 was payment services, accounting for 37.9%. The second was financial services, which accounted for 33%, followed by classifieds at 6.6% [8]. Gaming and social networking accounted for 4.6% and 2.8% [8] respectively.

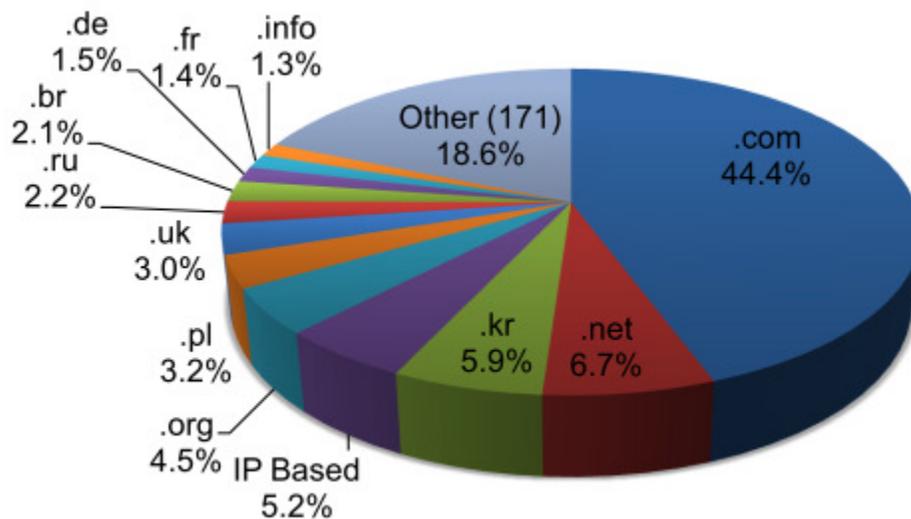


FIGURE 2: Most Targeted Industry Sector [8]

Com was the world largest and most ubiquitous top level domain name. In 2010, .Com contained 44.4% of the phishing domains.

Phishing is now the most common computer crime in Thailand and accounts for 77% of all computer-related crimes [5]. Phishing includes duplicating legitimate Websites such as Ebay, PayPal, and others in order to trick users into revealing their account information.

RANK	TLD	TLD Location	# Unique Phishing attacks 1H2010	Unique Domain Names used for phishing 1H2010	Domains in registry May 2010	Score: Phish per 10,000 domains 1H2010	Score: Attacks per 10,000 domains 1H2010
1	th	Thailand	86	62	49,000	12.7	17.6
2	kr	Korea	2,888	989	1,079,298	9.2	26.8
3	ie	Ireland	102	79	145,724	5.4	7.0
4	pl	Poland	1,582	744	1,805,894	4.1	8.8
5	cl	Chile	159	111	282,526	3.9	5.6
6	my	Malaysia	61	39	99,736	3.9	6.1
7	gr	Greece	130	93	260,000	3.6	5.0
8	ro	Romania	324	156	443,700	3.5	7.3
9	vn	Vietnam	64	48	153,002	3.1	4.2
10	cz	Czech Republic	480	207	689,813	3.0	7.0

TABLE 1: Phishing per 10,000 domain names [7]

The above table indicates that phishing has become a serious crime in Thailand. In 2010, Thailand was ranked first in the world for the second consecutive year in terms of the number of phishing websites per 10,000 domains [7]. .Th (Thailand) was at the top of the phishing list between 2009 and 2010. Phishing in .th took place mostly in academic institutions (.ac.th) and governmental organizations (go.th), as well as in commercial organizations (.co.th). These phishing websites dramatically affect consumers' confidence in conducting transactions online.

In the past, the major concern regarding computer security was malicious software, such as viruses, worms, and spyware. Since 2007, the most reported incident has been phishing [5].

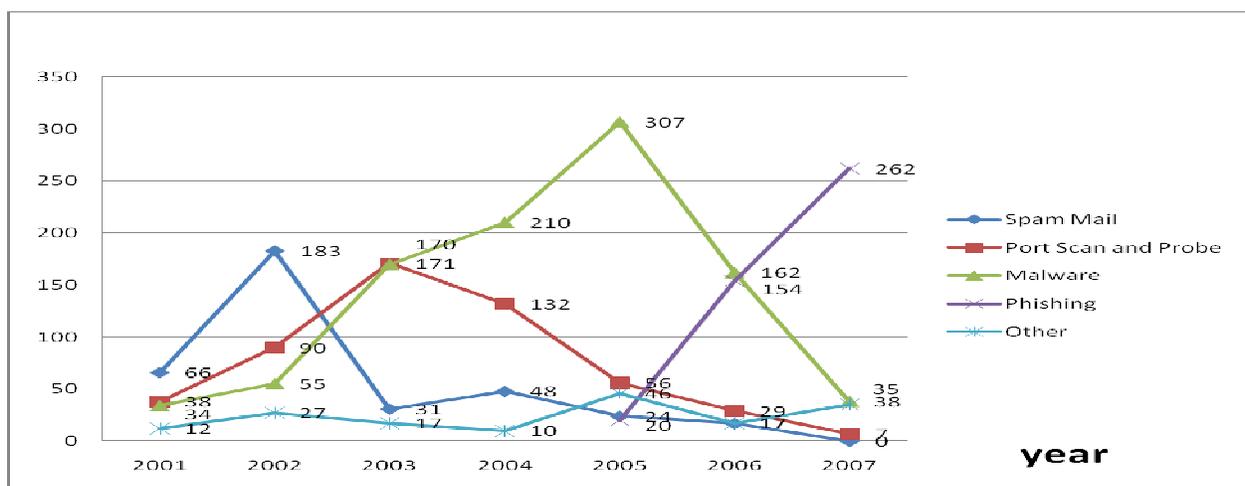


FIGURE 3: Number of Incidents since 2001 categorized based on methods [5]

The phishing problem is now the most frequent computer crime in Thailand and accounts for 77% of all computer-related crimes [5].

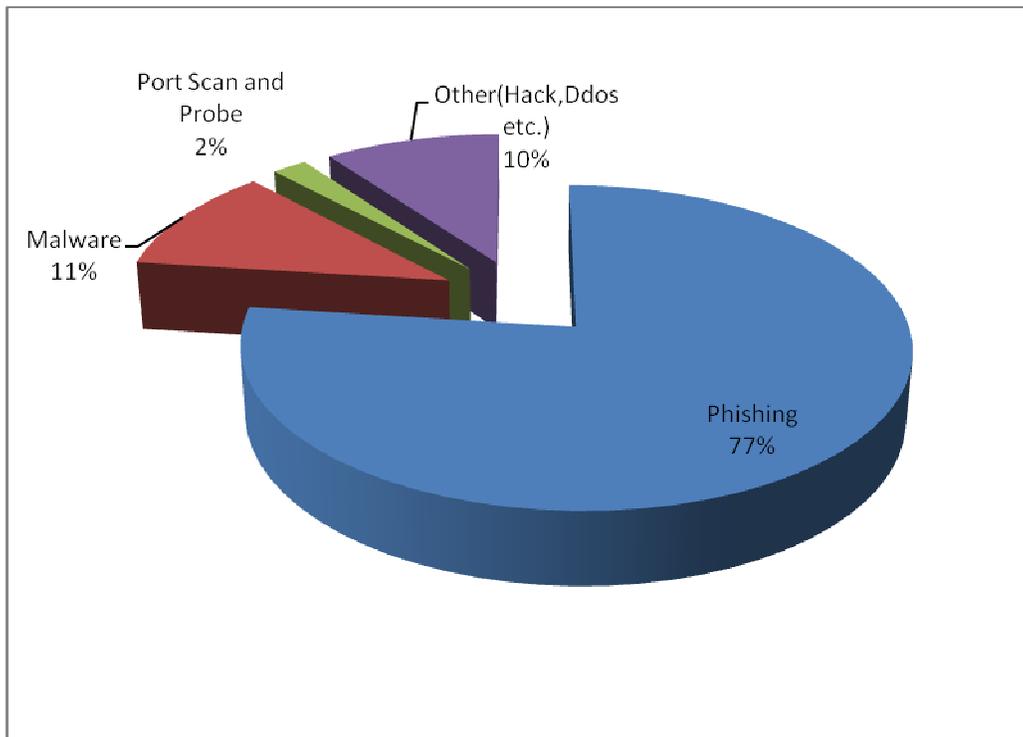


FIGURE 4: Ratio of each type of ThaiCERT- reported incident in the year 2007 [5]

Figure 4 indicates the number of incidents categorized by incident sources. From the figure, it can be seen that phishing is the most frequent computer crime. The survey indicates that government organizations are the main target of computer criminals. In 2007, 170 government organizations in Thailand became victims of computer crimes [5]. This may be because government organizations store a lot of sensitive information, such as citizen information, tax information, etc. This information can be a gold mine for the computer criminal to commit identity theft.

2. RESEARCH OBJECTIVES

The study is intended to investigate how users become victims of phishing attack. To study how phishing work in the real world, the researcher chose a field experiment approach. The results of the study can potentially explain how phishing works and how people become the victims.

3. RESEARCH SETTING

The researcher recently conducted a phishing experiment at one of the graduate institutions in Thailand. For the purpose of this study, we focused on a subset of the target group. The targets consisted of second year MBA students in the School of Business Administration, the National Institute of Development Administration. The main purpose was to evaluate how social context can increase the success of a phishing attack. The researcher worked closely with the school's institutional review board in designing the research protocol for this study, which can be broken down into four phases.

3.1 Phase 1: Preparation

The researcher identified the target group, which consisted of 174 MBA students. All of them had school accounts to log into the registration system. Their account contained private information, including name, address, student ID, citizen ID, courses taken, and grades. The objective of the experiment was to spoof

the student's username and password. In order to do this, a phishing website was created based on the registration website (see figure 5).

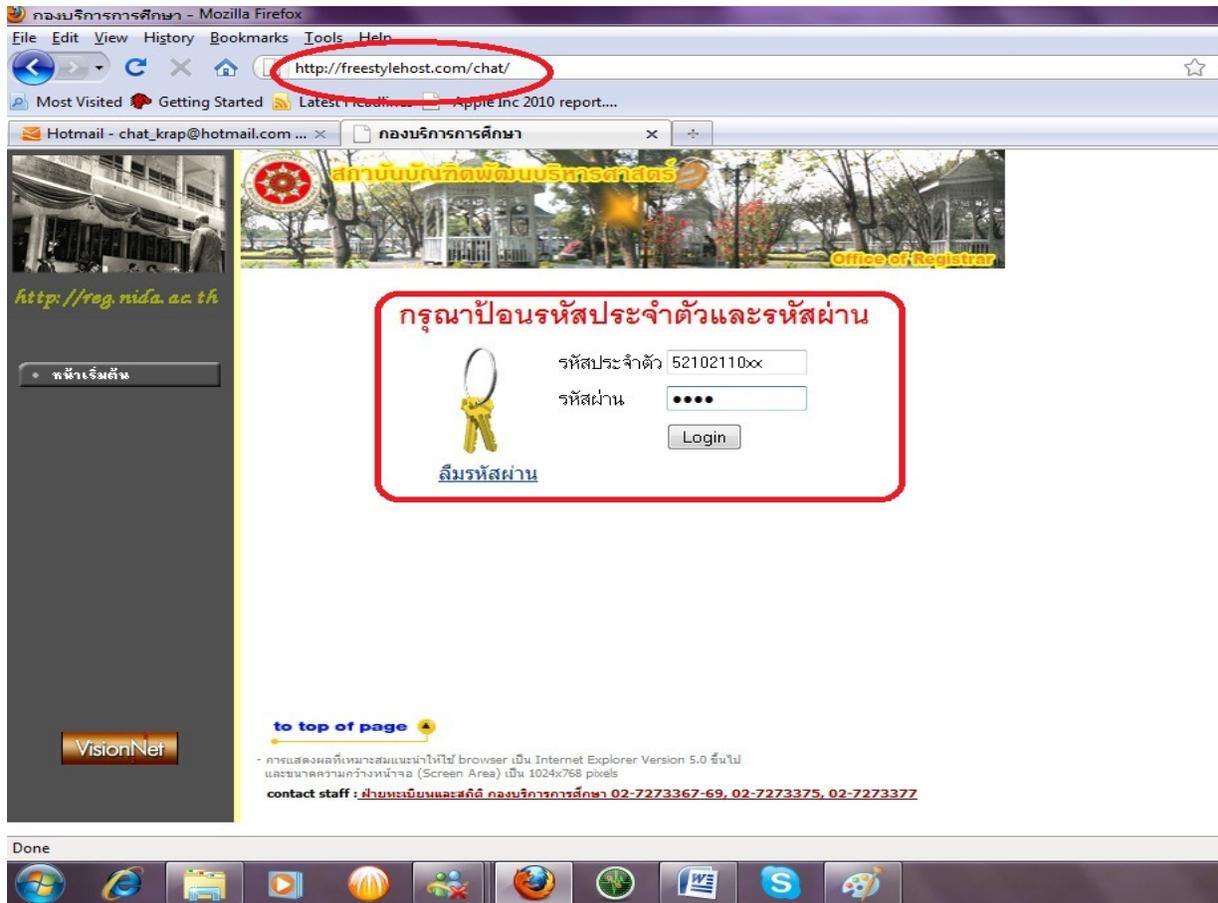


FIGURE 5: The phishing registration website

The experiment created a phishing administrator email account at mba_admin@nida.ac.th by collaborating with the Information Systems Education Center (ISEC) at NIDA (see figure 6). The message indicated that there were some major changes in the registration systems that required students to visit the link (phishing website) and log into it using their student IDs and passwords to update their personal information.

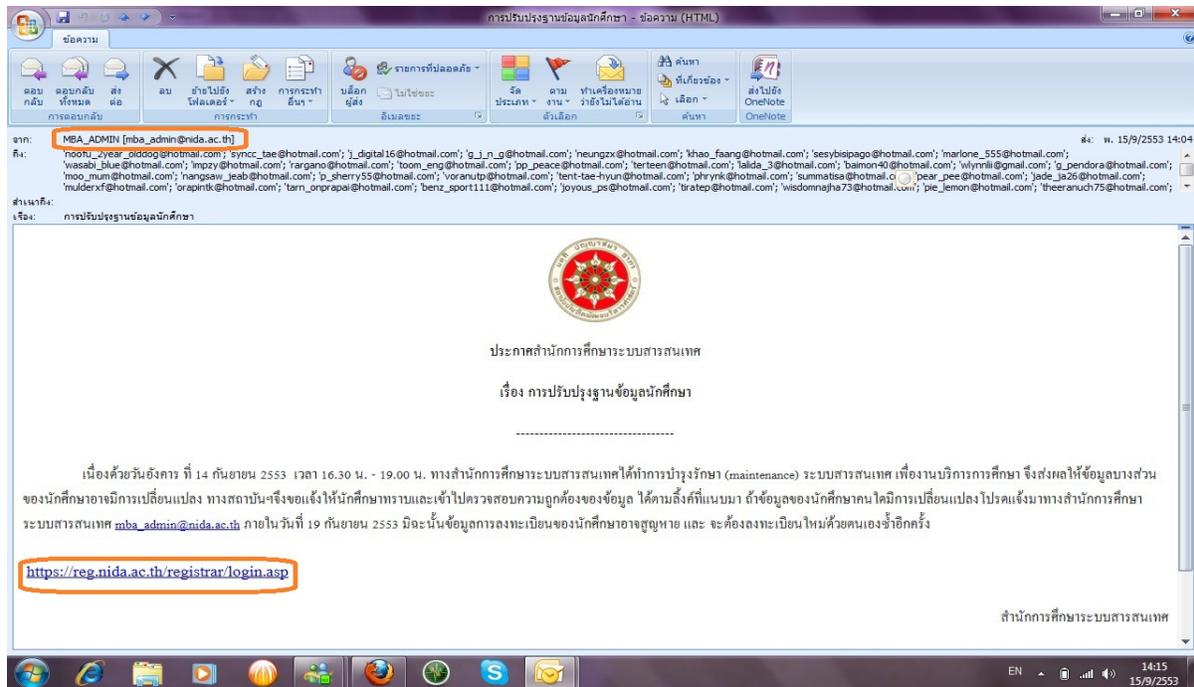


FIGURE 6: The spoofed email message

In order to enhance the credibility, the experiment spoofed an email account from the president of the MBA class using an email-spoofing application and used his email to send the link of the phishing website to all MBA students (see figure7).



FIGURE 7: E-mail spoofer application

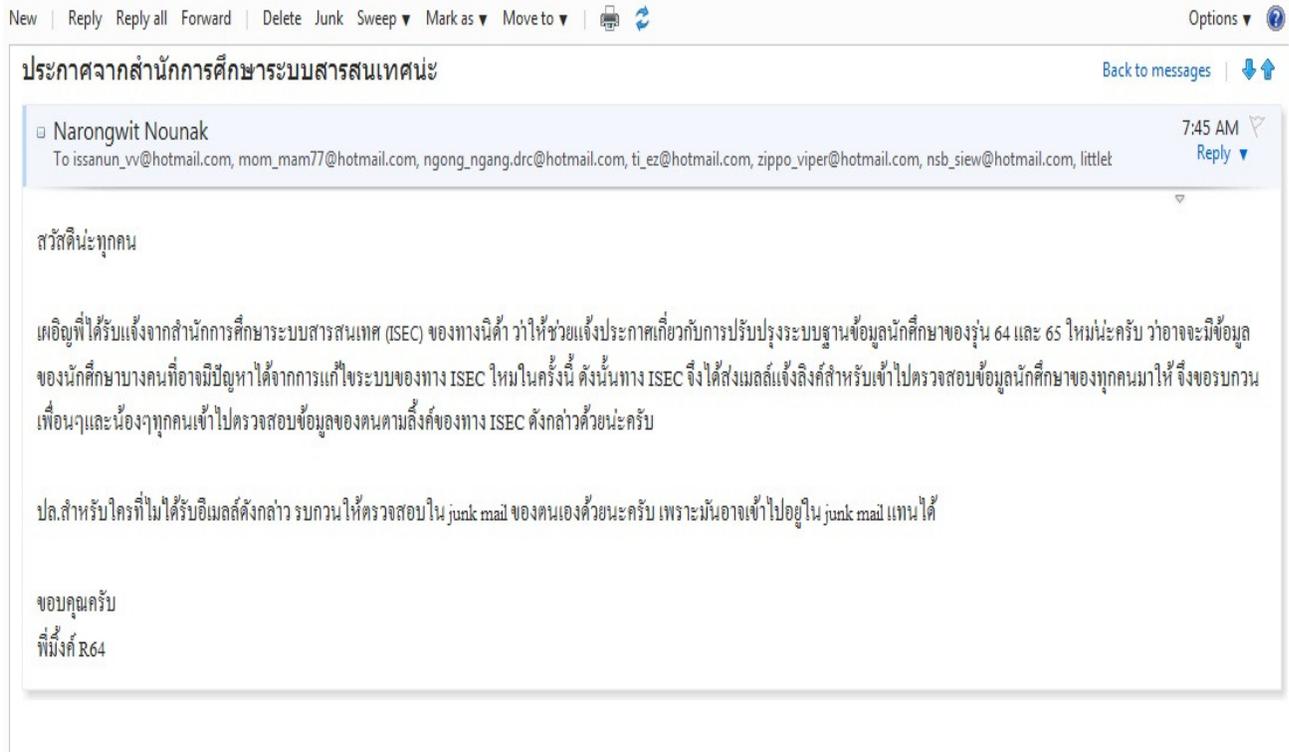


FIGURE 8: The message in the spoofed email

The message indicated that the Information Systems Education Center (ISEC) needed to upgrade the system and it needed students to log into the provided link. It was also stated that those students that do not log in might not be able to register for any class (see figure 8).

3.2 Phase 2: Execution

Once the phishing registration website was set up, the email was sent to 174 MBA students. The URL on the phishing registration website was not a real URL, while the interface of the phishing website was the same as the real registration website (see figure 9).

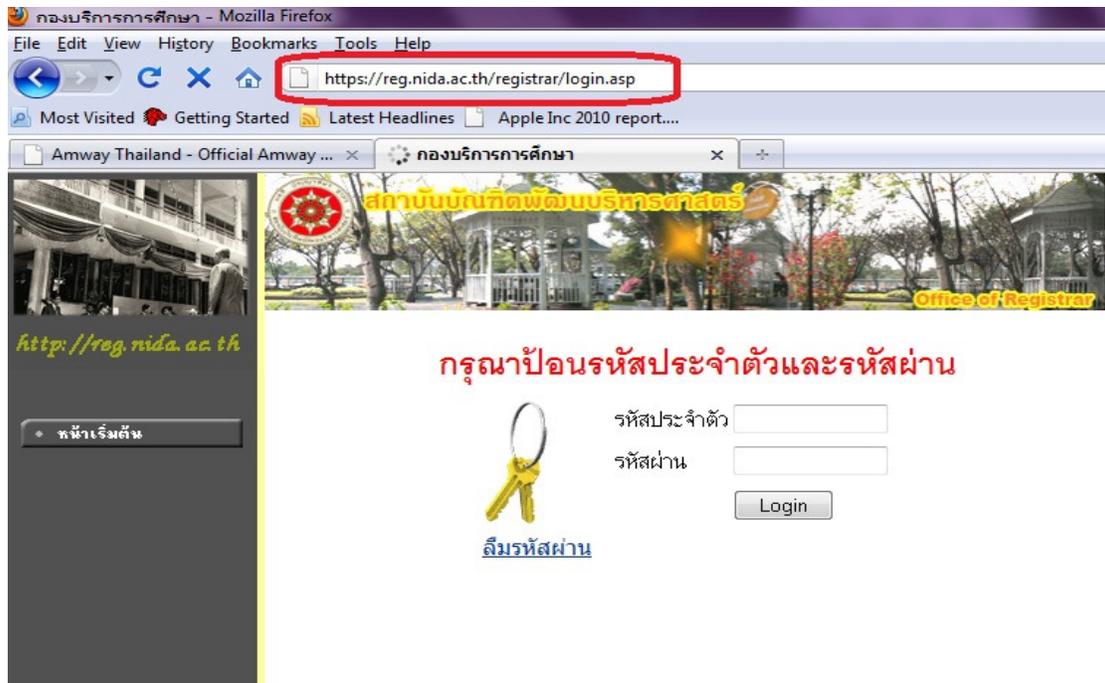


FIGURE 9: The real registration website

When the students received the email from the administrator and the class president, they clicked on the link to the phishing registration website. To log in, they had to provide their student ID and password. When the students clicked login, the page indicated that the system had a problem and asked them to reload the page. Once they reloaded the page, it led them to the real registration website (see figure 9), and once there, they could log into the system. However, the experimenter already had their passwords.

Once they entered their student ID and password, this information was emailed to the researcher. The passwords were encrypted in a way that the researcher cannot see the clear text. All the encrypted passwords were destroyed after the experimentation.

3.3 Phase 3 Results

Out of the 174 MBA students, 170 students logged into the phishing website—79 males (49.3%) and 91 females (50.70%). Although the researcher did not keep the password, the subjects were asked to change their passwords after the experiment. The attack was more successful since it was a spoofed email sent by an influential person, in this case the class president. From the post experimentation survey, 87.3% of the victims indicated that they trusted the content of the email because it was sent from the class president.

3.4 Phase 4 Evaluation

A focus group was conducted after the experiment to evaluate the effectiveness of the phishing attack. The results are the following:

Surprise: all subjects were surprised that they became victims to the phishing website. They believed that the email was actually sent from the class president, and they did not suspect any phishing activity. This might be because the website was made exactly like the real registration website. However, most subjects that became victims did not suspect that it was the wrong URL. Few subjects reported that they saw the wrong URL but still submitted their ID and password because they did not think the website was a fake.

Anger: some subjects were angry to find out that they revealed their ID and password to the phishing website. Some called experiment unethical and illegal. Although no sensitive information about the

victims was retained, most victims reported that they were upset to find out they were being tricked to reveal their password. Just like the study of Jagatic et al. (2007), this proves that a phishing attack can result in a tremendous psychology cost for the victims [9].

Mistrust: all victims indicated that the reason they felt victimized was because the email was sent from the class president; they trusted the content of the email to be authentic. The subjects reported that they would never trust any email from an authority asking them to reveal sensitive information.

Misunderstanding: most victims reported that they believed email accounts could not be spoofed. Many subjects did not understand how the researcher was able to spoof the class president's email and most of them believed that the researcher needed to hack into the class president's email. Many subjects believed that the email must have been actually sent from the class leader—they did not know how easy it was for an email to be spoofed.

4. LESSONS LEARNED

The Credibility of a Spoofed Email Plays a Critical Role in the Success of a Phishing Attack

Phishers sometimes send emails that appear to be coming from a legitimate and influential person (IT department, customer support or CEO) within an organization to its employees or customers to update their account passwords. Employees or customers are then directed to a fake website. Since employees or customers believe they are familiar with the sender of the email and the website they are directed to, they are likely to give away sensitive information, such as username, password, and other private information. Today, phishers are more sophisticated with email spoofing. They can easily identify the email of legitimate and influential people through corporate websites and/or social networks such as Facebook.

Phishers Often Use Visual Deception to Imitate Legitimate Websites by Using Text and Images.

Using fraudulent e-mail attachments with a hyperlink that imitates a legitimate website makes the victim believe that it is an official website; he or she then inputs his or her personal information.

Phishers Often Create a Sense of Emergency

Most Phishers create an emergency where the victim needs to react quickly. In this case study, the subjects were notified that they had to log in immediately; otherwise, they could not register for any class. Most victims of this phishing method feel rushed and that they need to respond right away or risk having a negative consequence.

Phishers Often Pretend to be an Authority

In order to gain the trust of the victim, a lot of phishers pretend to be some kind of authority by spoofing email from real authorities. In many cases, phishers have spoofed emails from email administrators and threaten to close an account if the victims do not reply with their username and password. In this case study, the email was made to appear to have been sent from the administrator (mba_admin@nida.ac.th), and lot of students reported that they were duped by the scam because they were afraid of authority.

IT Security Education is Needed to Prevent Phishing

The most effective way to prevent phishing is through security education or a security awareness program. Employees or people in an organization should know about the different methods of phishing and how they can become a victim. Role play can be used as part of the training so that the trainee will understand what phishing methods are used.

An Organization Should Frequently Conduct a Phishing Audit

A phishing audit is an ultimate method to deter and prevent phishing. An internal and external audit should be implemented. An internal audit can be done through an organizational IT department in order to identify weakness and at the same time educate end users. In this study, the process and results of the phishing experiment were shared with the IT department. Because of the results, the IT department considered revising its security policy and training system. An external audit can be done through an outside security consultant.

5. CONCLUSIONS

Phishing is the worst computer crime in Thailand, both in terms of frequency and in terms of impact. This study illustrates how phishing works and how easy it is for people to become a victim of a phishing attack. The researcher conducted a field experiment. A phishing website was created based on a real registration website. The subjects were MBA students. The spoofed email was sent to the subjects and they were asked to visit the phishing registration website. One hundred seventy subjects became the victims by revealing their student IDs and passwords. After the experiment, the subjects were notified to change their passwords, and a focus group was conducted, where the phishing victims expressed feelings of surprise, mistrust, and misunderstanding. The lessons learned from this study indicate that the credibility of a spoofed email plays a critical role in the accomplishment of a phishing attack. The results of the study also indicate that creating a sense of emergency and pretending to be an authority can be an effective method for deceiving victims. From the study, the author suggests that the best way to prevent phishing is through security awareness programs and security audits.

6. REFERENCES

- [1] Turban, E., et al., Information Technology for Management: Transforming Organizations in the Digital Economy 7th ed. 2010: Wiley.
- [2] Ohaya, C. Managing Phishing Threats in an Organization. in InfoSecCD. 2006. Kennesaw, GA: ACM.
- [3] APWG. (2009). Phishing Activity Trends Report: [www.antiphishing.org](http://www.antiphishing.org/reports/apwg_report_Q4_2009.pdf). Available: http://www.antiphishing.org/reports/apwg_report_Q4_2009.pdf [December 21, 2010].
- [4] Office, N. S. (2008). E-Commerce Report Bangkok, Thailand: National Statistical Office Available: http://service.nso.go.th/nso/nsopublish/pocketBook/electThaiRep_52.pdf [November 7, 2010].
- [5] ThaiCert, Year 2007 ThaiCERT's handled Incident Response Summary, N. Sanglerdinlapachai, Editor. 2007, Thai Computer Emergency Response Team.
- [6] Turban, E., et al., Information Technology for Management: Transforming Organizations in the Digital Economy 6th ed. 2008: Wiley.
- [7] APWG. (2010a). Global Phishing Survey: Trends and Domain Name Use in 1H2010 Available: http://www.antiphishing.org/reports/APWG_GlobalPhishingSurvey_1H2010.pdf [December 25, 2010].
- [8] APWG. (2010b). Phishing Activity Trends Report: APWG. Available: http://www.antiphishing.org/reports/apwg_report_Q1_2010.pdf [December 25, 2010].
- [9] Jagatic, T.N., et al., Social Phishing. Communications of the ACM, 2007. 50(10).

Design and Implementation of a Multi-Agent System for the Job Shop Scheduling Problem

Leila Asadzadeh
Information Technology Department
Payame Noor University
19395-4697 Tehran, I. R. of IRAN

leila_asadzadeh_cs@yahoo.com

Kamran Zamanifar
Computer Engineering Department
University of Isfahan
Isfahan, I. R. of IRAN

zamanifar@eng.ui.ac.ir

Abstract

Job shop scheduling is one of the strongly NP-complete combinatorial optimization problems. Developing effective search methods is always an important and valuable work. Meta-heuristic methods such as genetic algorithms are widely applied to find optimal or near-optimal solutions for the job shop scheduling problem. Parallelizing genetic algorithms is one of the best approaches that can be used to enhance the performance of these algorithms. In this paper, we propose an agent-based parallel genetic algorithm for the job shop scheduling problem. In our approach, initial population is created in an agent-based parallel way then an agent-based method is used to parallelize the genetic algorithm. Experimental results showed that the proposed approach enhances the performance.

Keywords: Job Shop Scheduling Problem, Genetic Algorithms, Parallel Genetic Algorithms, Agents and Multi Agent Systems.

1. INTRODUCTION

Job shop scheduling problem is one of the most important problems in machine scheduling. This problem is considered to be a member of a large class of intractable numerical problems known as NP-hard [1]. High complexity of problem makes it hard and in some cases impossible to find the optimal solution within reasonable time. Hence, searching for approximate solutions in polynomial time instead of exact solutions at high cost is preferred for difficult instances of problem.

Historically job shop scheduling problem has been primarily treated using the branch and bound [2-4], heuristic rules [5-7] and shifting bottleneck procedure [8]. In recent years, meta-heuristic methods are widely applied to this problem. These methods, such as taboo search [9-11], simulated annealing [12-15], genetic algorithms [16-20], neural networks [21] and ant colony optimization [22-25] are well suited to solving complex problems with high costs. A survey on job shop scheduling techniques can be found in [1].

Comparing with other meta-heuristic methods, genetic algorithms are widely used to solve various optimization problems. Many genetic algorithm based approaches are proposed for the job shop scheduling problem. In [26,27] authors introduce an approach that uses load balancing of machines as an important parameter in job assignment. An advantage of these approaches is that maximize machine utilization while minimizing makespan. Ombuki and Ventresca [28] proposed a local search genetic algorithm that uses an efficient solution representation strategy in which both checking of the constraints and repair mechanism can be avoided. In their approach at local search phase a new mutation-like operator is used to improve the solution quality. They also developed a hybrid strategy using the genetic algorithm reinforced with a taboo search for problem. Lin et al. [29] introduced a hybrid model consisting of coarse-grain genetic algorithms connected in a fine-grain style topology. Their method can avoid premature convergence, and it produced excellent results on standard

benchmark job shop scheduling problems. Chen et al. [30] gave a tutorial survey of recent works on various hybrid approaches of the genetic algorithms proposed so far for the job shop scheduling problem. Wang and zheng [20] by combining simulated annealing and genetic algorithms developed a general, parallel and easily implemented hybrid optimization framework, and applied it to job shop scheduling problem. Based on effective encoding scheme and some specific optimization operators, some benchmark job shop scheduling problems are well solved by the hybrid optimization strategy. In [19] a hybrid method is proposed to obtain a near-optimal solution within a reasonable amount of time. This method uses a neural network approach to generate initial feasible solutions and then a simulated annealing algorithm to improve the quality and performance of the initial solutions in order to produce the optimal/near-optimal solution. Chen et al [31] proposed an agent-based genetic algorithm that accelerates the creation of initial population. In this approach, the processing of selection, crossover and mutation can be controlled in an intelligent way.

In this paper, we propose an agent-based parallel genetic algorithm for the job shop scheduling problem. In our approach, initial population is created in an agent-based parallel way then an agent-based method is used to parallelize genetic algorithm.

The remainder of this paper is organized as follow. In section 2, we describe job shop scheduling problem. Details of our proposed agent-based architecture and the parallel genetic algorithm are represented in section 3. In section 4, we discuss implementation and experimental results of proposed approach. Conclusion is represented in section 5.

2. JOB SHOP SCHEDULING PROBLEM

Job Shop Scheduling Problem can be described as follow. A set of n jobs and a set of m machines are given. Each job consists of a sequence of operations that must be processed on a specified order. Each job consists of a chain of operations, each of which needs to be processed during an uninterrupted time period of a given length on a given machine. Each machine can process only one job and each job can be processed by only one machine at a time. Usually we denote the general job shop scheduling problem as $n \times m$, where n is the number of jobs and m is the number of machines. TABLE 1 shows an example 5×4 job shop scheduling problem. The duration in which all operations for all jobs are completed is referred to as the makespan. A schedule determines the execution sequence of all operations for all jobs on machines. The objective is to find optimal schedule. Optimal schedule is the schedule that minimizes makespan. Due to factorial explosion of possible solutions, job shop scheduling problems are considered to be a member of a large class of intractable numerical problems known as NP-hard [1]. It is hard and in some cases impossible to find the optimal solution within reasonable time due to High complexity of problem. Hence, searching for approximate solutions in polynomial time instead of exact solutions at high cost is preferred for difficult instances of problem.

Job	Machine, Processing time			
P_1	2,4	1,3	3,11	4,10
P_2	4,10	1,8	2,5	3,4
P_3	1,5	3,6	2,4	4,3
P_4	1,7	2,3	3,2	4,12
P_5	3,5	4,8	1,9	2,5

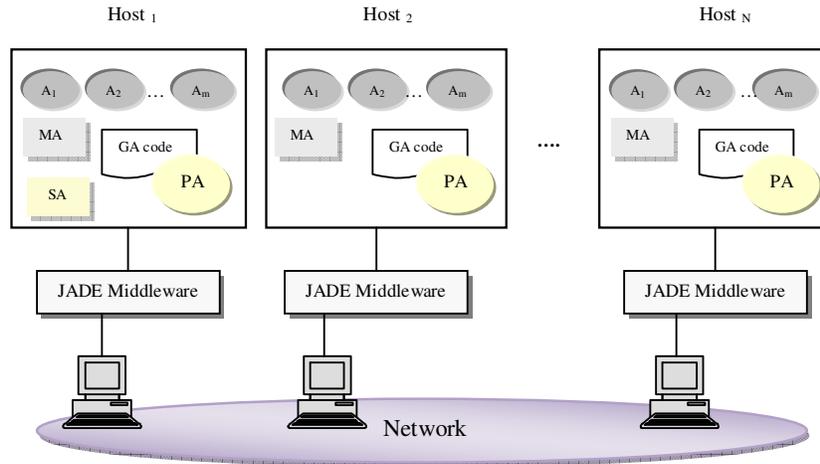


FIGURE 1: Improved Agent-based Architecture for the Job Shop Scheduling Problem

TABLE 1: An Example 5x4 Job Shop Scheduling Problem

3. AGENT-BASED PARALLEL MODEL

In [32] we proposed an agent-based parallel approach for the job shop scheduling problem. In that model, we developed a multi-agent system containing some agents with special actions that are used to parallelize the genetic algorithm and create its population. We used *JADE* middleware [33] to implement our multi-agent system. Agents distributed over various hosts in network and *JADE* provides a secure communication channel for them to communicate. In this model, each agent has been developed for a special purpose. We can describe them as follow [32]:

- **MA (Management Agent):** *MA* and A_i ($i=1,2,\dots,m$) agents have the responsibility of creating the initial population for the genetic algorithm. This agent controls the behaviors of A_i s and coordinates them in creation step.
- **A_i (Execute Agent):** Each machine has an A_i agent to schedule the operations on it.
- **PA (Processor Agent):** Each *PA* locates on a distinct host and executes genetic algorithm on its sub-population.
- **SA (Synchronization Agent):** This agent locates on main host and coordinates migration between sub-populations of *PA* agents.

In that model, the genetic population is created serially by *MA* and A_i ($i=1,2,\dots,m$) agents. The sub-populations of *PA* agents are determined and sent to them by *MA*. One disadvantage of this model is the lack of load balancing on the network hosts. On the other hand, the main host that locates the *MA*, A_i and *SA* agents is the bottleneck of system and if it crash, the whole multi-agent system will be stopped working.

To solve this problem and improve the performance of creating the initial population, we can extend the model to create sub-populations in a parallel manner. An overall architecture of improved agent-based model has represented in FIGURE 1. In this model, each host has one *MA* and m A_i ($i=1,2,\dots,m$) agents. These agents have the responsibility of creating the subpopulation for their host's *PA*.

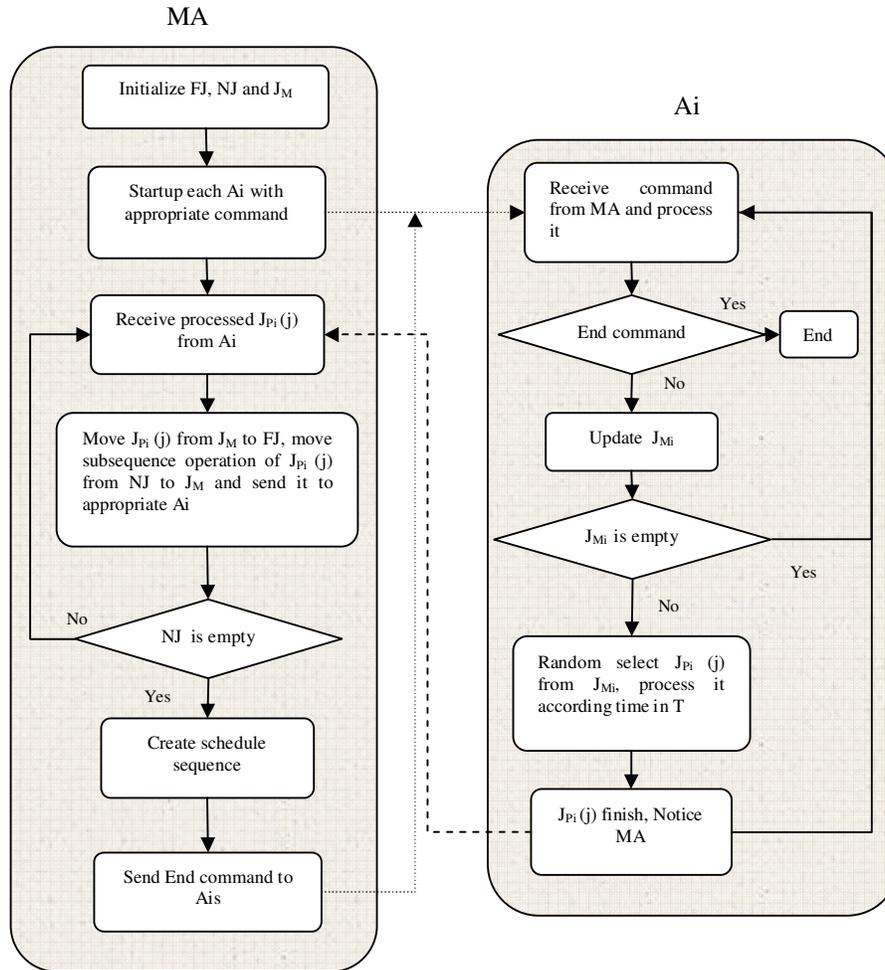


FIGURE 2: Schedule Process of MA and Ai

To synchronize the various processor agents in migration phase, synchronization agent (SA) locates on main host and synchronizes them.

Parallel creation of sub-populations improves the speed and performance. On the other hand, the division of the population into several sub-populations and sending them to PAs can be avoided.

3.1 Initial Population of Genetic Algorithm

Before we describe how initial population can be created by an agent-based method we define the job shop scheduling problem formally with the following definitions [31].

1. $P = \{P_1, P_2, \dots, P_n\}$ is the set of n jobs.
2. $M = \{M_1, M_2, \dots, M_m\}$ is the set of m machines.
3. Job P_i has k_i operations. $J_{P_i} = \{J_{P_i}(1), J_{P_i}(2), \dots, J_{P_i}(k_i)\}$ is the set of operations of job P_i . $J_P = \{J_{P_1}, J_{P_2}, \dots, J_{P_n}\}$ is the matrix of operations of n jobs. The value of $J_{P_i}(j)$ is the machine that operation j of job P_i must be processed on.
4. Let T is the $n \times m$ matrix of processing times of each job in m machines. $T[i, j]$ is the processing time of job P_i on machine j .

Status of each operation is determined by using the following definitions.

5. J_{M_i} is the set of all schedulable operations on machine M_i . A schedulable operation is an operation for which all the foregoing operations have finished. $J_M = \{J_{M_1}, J_{M_2}, \dots, J_{M_m}\}$ is the set of schedulable operations on all machines.
6. NJ is the set of un-schedulable operations, i.e. operations for which at least one of the foregoing operations has not finished.

7. FJ is the set of finished operations.

Chromosomes of the genetic population are created by using method proposed in [31]. In this method, two kinds of agents are used to create chromosomes of the initial population: the management agent (MA) and the execute agent A_i ($i=1,2,\dots,m$). Each machine in a specified problem instance has an execute agent. Each A_i schedules the operations of its machine. MA manages operations of all jobs and controls the execution of A_i s. In the first step, MA initializes J_M , FJ and NJ as follow.

$$FJ = \square$$

$$NJ = \{J_{P_1}, J_{P_2}, \dots, J_{P_n}\}$$

$$J_M = \{J_{P_1}(1), J_{P_2}(1), \dots, J_{P_n}(1)\}$$

Parallel genetic algorithm in our approach has multiple populations. The whole population is divided into several subpopulations which are called islands and each island is evolved independently. To improve the speed we produce various sub-populations concurrently. Each host in the system has one MA agent and m A_i ($i=1,2,\dots,m$) agents and these agents produce chromosomes for one of sub-populations. The schedule process in FIGURE 2 is used to create sub-populations. Each execution of this process creates a chromosome indicating a feasible schedule for a specified problem instance. To create a sub-population with size N we execute the schedule process N times.

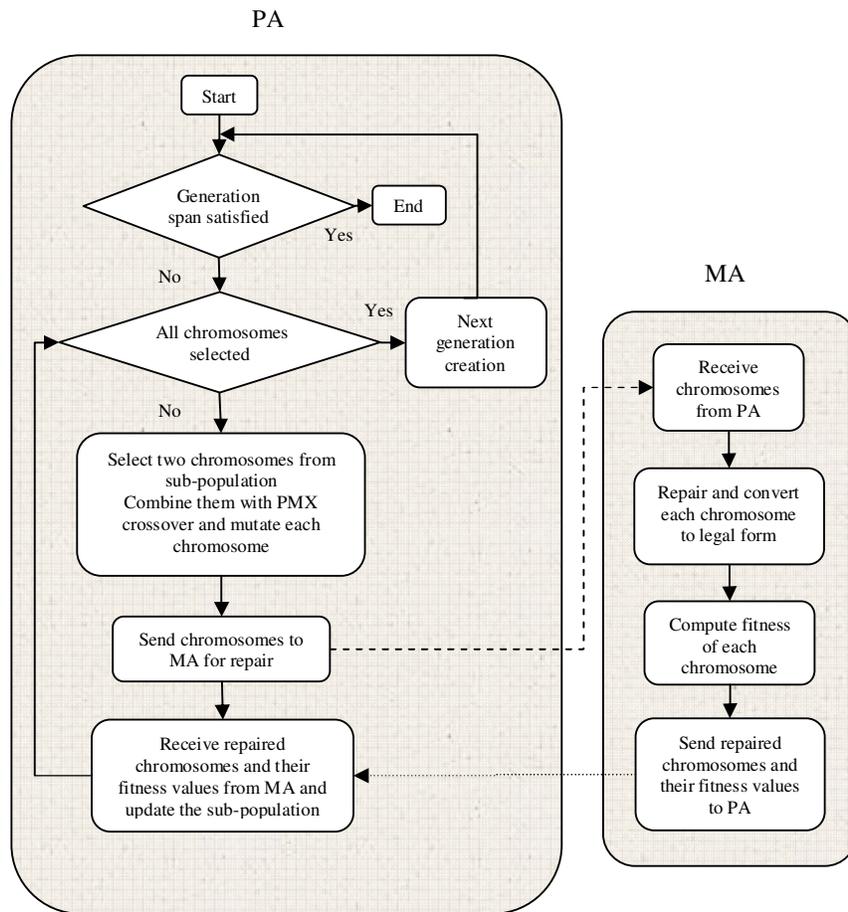


FIGURE 3: PA and MA Communication in the Execution Phase

3.2 Parallel Genetic Algorithm

To parallelize our genetic algorithm we use a coarse-grained model. This model has multiple and smaller populations and exchanges information among the sub-populations. This exchange is performed by moving some individuals from one population to another and is known as migration. Communication between sub-populations restricted to migration of chromosomes.

In our method, various sub-populations are created by *MA* and A_i ($i=1,2,\dots,m$) agents in a parallel way. Each processor agent (*PA*) locates on a distinct host and executes genetic algorithm on its sub-population independently. Different sub-populations communicate with exchanging of migrants. Parallel genetic algorithm consists of two phases: The execution phase and the migration phase. In the execution phase, sub-populations are evolved independently by processor agents and in the migration phase, *PA*s exchange migrants. These two phases run repeatedly for predefined times [32]. Detailed communication between *PA* and *MA* in execution phase is showed in FIGURE 3.

3.2.1 Migration Policy

Communication between various sub-populations is carried out by exchanging of migrants. In our approach we use synchronous migration policy. Each *PA* executes genetic algorithm on its sub-population for a predefined number of generations then it sends a message to *SA* informing end of its execution. The *SA* is a synchronization agent, which coordinates migration between sub-populations of *PA* agents. After receiving message from all the *PA*s, *SA* broadcasts a message to them notifying start of the migration phase. In the migration phase, each *PA* exchanges some of its best chromosomes with its neighbors (see FIGURE 4). Chromosomes with low fitness value in sub-population are replaced with the best chromosomes of neighbors [32].

4. IMPLEMENTATION AND EXPERIMENTAL RESULTS

We showed that the parallel agent-based genetic algorithm for the job shop scheduling problem enhances the speed and performance [32]. In this paper we evaluate our improved approach. Firstly, we explain detailed implementation of the genetic algorithm as follow.

- a. **Chromosome representation:** Operation-based method that each job has a distinct number for indicating its operations.
- b. **Selection:** Roulette wheel selection containing the elite retaining model [34].
- c. **The crossover operator:** Partially matched crossover (*PMX*) [35], two crossover points is chosen from the chromosomes randomly and equably. Then the genes of two parents that are in the area between crossover points are exchanged.
- d. **The mutation operator:** Shift mutation, a point from the chromosome is chosen randomly and the gene that is in this point is exchanged with its subsequent gene.
- e. **Fitness function:** The fitness function is defined as follow [34]:

$$Fitness(C) = P_Time_{max} - P_Time(C) \quad (1)$$

Where $P_Time(C)$ is the maximal processing time of chromosome C and P_Time_{max} is the maximum value of $P_Time(C)$. In our approach, *MA* agents compute fitness value of chromosomes.

Creating new chromosomes by using crossover operator may be lead to illegal schedules. Repair mechanism is used by *MA*s to convert these chromosomes to legal form. When a new chromosome is created, *PA* sends it to *MA* agent of its host. *MA* replaces repeat operations of the new chromosome with absent operations to ensure the appearance times of each job P_i is equal to k_i . Repaired chromosome is sent to *PA*.

We used some benchmark instances for the job shop scheduling problem. These problem instances are available from the OR library web site [36]. We set parameter values for genetic algorithm as follow:

- population size =1000
- generation span =1000
- crossover rate = 0.95

- mutation rate = 0.01

The number of *PA* agents was fixed at eight in our experiments and these agents form a virtual cube among them. Each *PA* has three neighbors. Parameters of migration were set as follow:

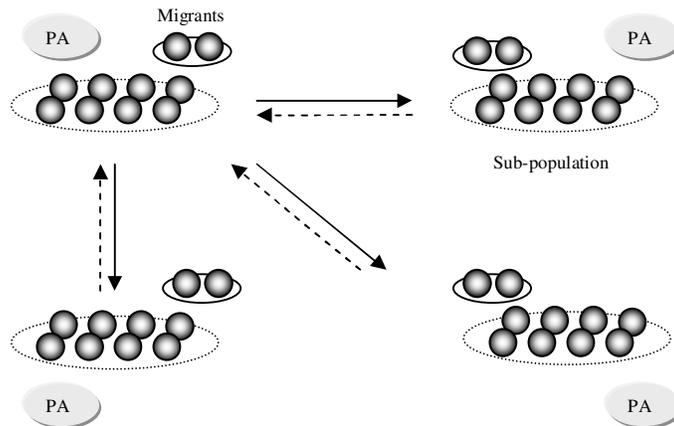


FIGURE 4: Migration Between *PA* Agents

- Migration Frequency : 100 generations
- Migration Rate : 10 chromosomes
- Migration Topology : cube

To evaluate our proposed parallel agent-based genetic algorithm, we compare it with the serial case that we have only one genetic population that evolves by a *PA* agent. We computed average makespan of best schedules obtained by applying algorithms on some problem instances during various generations. Results are shown in FIGURES 5-7. These figures demonstrate the effect of applying parallel and serial approaches on *LA30*, *ORB09* and *FT10* instances. As shown by these figures, convergence to near optimal solution in parallel method is happen rapidly and the solutions that it finds in various generation numbers have shorter lengths than those that are found by serial method.

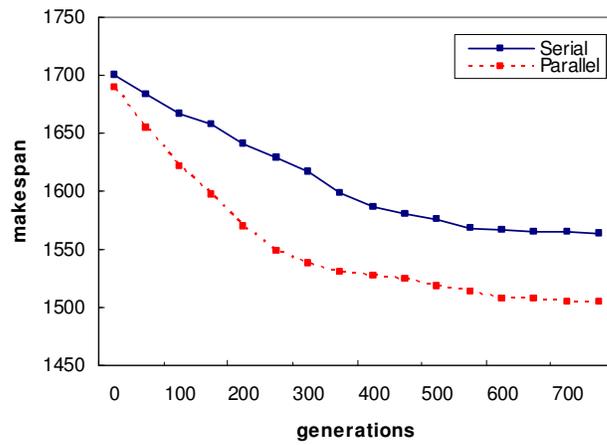
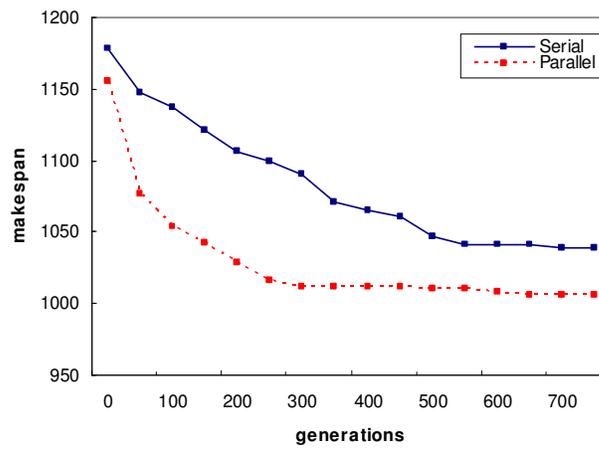


FIGURE 5: Parallel Approach Finds Better Solutions Comparing with Serial Method. Test Problem:



LA30. Results Are Averages of Best Solutions Over 10 Runs.

FIGURE 6: Parallel Approach Finds Better Solutions Comparing With Serial Method. Test Problem: *ORB09*. Results Are Averages of Best Solutions Over 10 Runs.

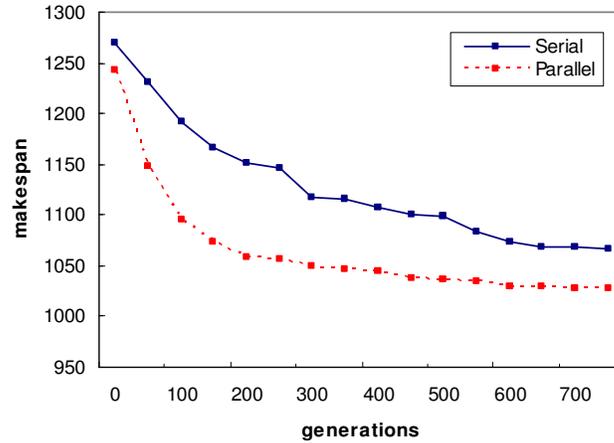


FIGURE 7: Parallel Approach Finds Better Solutions Comparing With Serial Method. Test Problem: *FT10*. Results Are Averages of Best Solutions Over 10 Runs.

To compare the parallel method proposed to create genetic population with the serial approach, we carried out some experiments. The evaluation parameter is the required time for creation of sub-populations for various problem instances. In serial method we have one population with 1000 chromosomes that is divided into 8 sub-populations with 125 chromosomes after that it has been created by MA and $A_i (i=1,2,\dots,m)$ agents. In parallel method, 8 sub-populations each with 125 chromosomes are created in parallel manner. Results are shown in TABLE 2. Times are in second. According to experimental results, parallel creation of sub-populations takes less time comparing with serial method.

5. CONCLUSION

Job shop scheduling problem is one of the most important problems in machine scheduling. This problem is considered to be a member of a large class of intractable numerical problems known as NP-hard. In this paper, we proposed an agent-based parallel genetic algorithm for this problem. To enhance the performance of the creation of the initial population for the genetic algorithm, we parallelized it using agent-based method. We compared performance of the parallel approach with the serial method. The results showed that the parallel method improves the speed of genetic population creation. Future work will concentrate on improving the performance of our method and applying it to similar problems.

Problem	Jobs	Machines	Time/s	
			Serial	Parallel
LA04	10	5	20	6
LA06	15	5	25	6
LA11	20	5	36	5
LA16	10	10	130	9
LA21	15	10	125	15
LA26	20	10	156	17
LA31	30	10	328	25

TABLE 2: Comparisons Between Serial and Parallel Methods for Creating the Genetic Population on *LA* Instances

6. REFERENCES

- [1] A. S. Jain and S. Meeran. "Deterministic job-shop scheduling: past, present and future," Department of Applied Physics and Electronic and Mechanical Engineering, University of Dundee, Dundee, Scotland, UK, 1998.
- [2] J. Carlier and E. Pinson. "An algorithm for solving the job shop problem." *Management Science*, vol. 35, no. 29, pp. 164-176, 1989.
- [3] B. J. Lageweg, J. K. Lenstra, and A. H. G. Rinnooy Kan. "Job shop scheduling by implicit enumeration." *Management Science*, vol. 24, pp. 441-450, 1977.
- [4] P. Brucker, B. Jurisch, and B. Sievers. "A branch and bound algorithm for job-shop scheduling problem." *Discrete Applied Mathematics*, vol. 49, pp. 105-127, 1994.
- [5] V. R. Kannan and S. Ghosh. "Evaluation of the interaction between dispatching rules and truncation procedures in job-shop scheduling." *International journal of production research*, vol. 31, pp. 1637-1654, 1993.
- [6] R. Vancheeswaran and M. A. Townsend. "A two-stage heuristic procedure for scheduling job shops." *Journal of Manufacturing Systems*, vol. 12, pp. 315-325, 1993.
- [7] Z. He, T. Yang, and D. E. Deal. "Multiple-pass heuristic rule for job scheduling with due dates." *International journal of production research*, vol. 31, pp. 2677-2692, 1993.
- [8] J. adams, E. Balas, and D. Zawack. "The shifting bottleneck procedure for job shop scheduling." *Management Science*, vol. 34, pp. 391-401, 1988.
- [9] E. Nowicki and C. Smutnicki. "A fast taboo search algorithm for the job-shop problem." *Management Science*, vol. 42, no. 6, pp. 797-813, June 1996.
- [10] S. G. Ponnambalam, P. Aravindan, and S. V. Rajesh. "A tabu search algorithm for job shop scheduling." *International Journal of Advanced Manufacturing Technology*, vol. 16, pp. 765-771, 2000.
- [11] P. V. Laarhoven, E. Aarts, and J. K. Lenstra. "Job shop scheduling by simulated annealing." *Operations Research*, vol. 40, pp. 113-125, 1992.
- [12] J. B. Chambers. "Classical and flexible job shop scheduling by tabu search," Ph.D. dissertation, University of Texas at Austin, Austin, TX, 1996.
- [13] M. E. Aydin and T. C. Fogarty. "Simulated annealing with evolutionary processes in job shop scheduling," in *Evolutionary Methods for Design, Optimization and Control*, (Proceeding Of EUROGEN 2001), Barcelona, 2002.
- [14] M. Kolonko. "Some new results on simulated annealing applied to job shop scheduling problem." *European Journal of Operational Research*, vol. 113, pp. 123-136, 1999.
- [15] T. Satake, K. Morikawa, K. Takahashi, and N. Nakamura. "Simulated annealing approach for minimizing the makespan of the general job-shop." *International Journal of Production Economics*, vol. 60-61, pp. 515-522, 1999.
- [16] F. D. Croce, R. Tadei, and G. Volta. "A genetic algorithm for the job shop problem." *Computers and Operations Research*, vol. 22, pp. 15-24, 1995.
- [17] J. F. Goncalves, J. J. d. M. Mendes, and M. G. C. Resende. "A hybrid genetic algorithm for the job shop scheduling problem." *European Journal of Operational Research*, vol. 167, pp. 77-95, 2005.
- [18] L. Wang and D. Z. Zheng. "A Modified Genetic Algorithm for Job Shop Scheduling." *International journal of advanced manufacturing technology*, pp. 72-76, 2002.

- [19] R. T. Mogaddam, F. Jolai, F. Vaziri, P. K. Ahmed, and A. Azaron. "A hybrid method for solving stochastic job shop scheduling problems." *Applied Mathematics and Computation*, vol. 170, pp. 185-206, 2005.
- [20] L. Wang and D. Z. Zheng. "An effective hybrid optimization strategy for job-shop scheduling problems." *Computers & Operations Research*, vol. 28, pp. 585-596, 2001.
- [21] S. Y. Foo, Y. Takefuji, and H. Szu. "Scaling properties of neural networks for job shop scheduling." *Neurocomputing*, vol. 8, no.1, pp. 79-91, 1995.
- [22] J. Zhang, X. Hu, X. Tan, J. H. Zhong, and Q. Huang. "Implementation of an Ant Colony Optimization technique for job shop scheduling problem." *Transactions of the Institute of Measurement and Control*, vol. 28, pp. 93-108, 2006.
- [23] K. L. Huang and C. J. Liao. "Ant colony optimization combined with taboo search for the job shop scheduling problem." *Computers & Operations Research*, vol. 35, pp. 1030-1046, 2008.
- [24] J. Montgomery, C. Fayad, and S. Petrovic. "Solution representation for job shop scheduling problems in ant colony optimization," Faculty of Information & Communication Technologies, Swinburne University of Technology, 2006.
- [25] M. Ventresca and B. Ombuki. "Ant Colony Optimization for Job Shop Scheduling Problem," in Proceedings of 8th IASTED International Conference On Artificial Intelligence and Soft Computing, 2004, pp. 451-152.
- [26] S. Petrovic, and C. Fayad. "A genetic algorithm for job shop scheduling with load balancing," School of Computer Science and Information Technology, University of Nottingham, Nottingham, 2005.
- [27] S. Rajakumar, V. P. Arunachalam, and V. Selladurai. "Workflow balancing in parallel machine scheduling with precedence constraints using genetic algorithm." *Journal of Manufacturing Technology Management*, vol. 17, pp. 239-254, 2006.
- [28] B. M. Ombuki, and M. Ventresca. "Local search genetic algorithms for the job shop scheduling problem." *Applied Intelligence*, vol. 21, pp. 99-109, 2004.
- [29] S. C. Lin, E. D. Goodman, and W. F. Punch. "Investigating parallel genetic algorithms on job shop scheduling problems," Genetic algorithm research and applications group, State university of Michigan, Michigan, 1995.
- [30] R. Cheng, M. Gen, and Y. Tsujimura. "A tutorial survey of job-shop scheduling problems using genetic algorithms, part II: Hybrid genetic search strategies." *Computers & Industrial Engineering*, vol. 36, pp. 343-364, 1999.
- [31] Y. Chen, Z. Z. Li, and Z. W. Wang. "Multi-agent-based genetic algorithm for JSSP," in Proceedings of the third international conference on Machine Learning and Cybernetics, 2004, pp. 267-270.
- [32] L. Asadzadeh, K. Zamanifar. "An Agent-based Parallel Approach for the Job Shop Scheduling Problem with Genetic Algorithms." *Mathematical and Computer Modeling*, Vol. 52, pp. 1957-1965, 2010.
- [33] F. Bellifemine, A. Poggi, and G. Rimassa. "Developing multi-agent systems with a FIPA-compliant agent framework." *Software: Practice and Experience*, vol. 31, pp. 103-128, 2001.
- [34] D. E. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning*. MA: Addison-Wesley, 1989.

- [35] D. E. Goldberg and R. Lingle. "Alleles, loci, and the TSP," in Proceeding of 1st International Conference on Genetic Algorithms, 1985, pp. 154-159.
- [36] D. C. Mattfeld and R. J. M. Vaessens. "Job shop scheduling benchmarks." Internet: www.mscmga.ms.ic.ac.uk, [Jul. 10, 2008].

INSTRUCTIONS TO CONTRIBUTORS

The *International Journal of Computer Science and Security (IJCSS)* is a refereed online journal which is a forum for publication of current research in computer science and computer security technologies. It considers any material dealing primarily with the technological aspects of computer science and computer security. The journal is targeted to be read by academics, scholars, advanced students, practitioners, and those seeking an update on current experience and future prospects in relation to all aspects computer science in general but specific to computer security themes. Subjects covered include: access control, computer security, cryptography, communications and data security, databases, electronic commerce, multimedia, bioinformatics, signal processing and image processing etc.

To build its International reputation, we are disseminating the publication information through Google Books, Google Scholar, Directory of Open Access Journals (DOAJ), Open J Gate, ScientificCommons, Docstoc and many more. Our International Editors are working on establishing ISI listing and a good impact factor for IJCSS.

The initial efforts helped to shape the editorial policy and to sharpen the focus of the journal. Starting with volume 5, 2011, IJCSS appears in more focused issues. Besides normal publications, IJCSS intend to organized special issues on more focused topics. Each special issue will have a designated editor (editors) – either member of the editorial board or another recognized specialist in the respective field.

We are open to contributions, proposals for any topic as well as for editors and reviewers. We understand that it is through the effort of volunteers that CSC Journals continues to grow and flourish.

IJCSS LIST OF TOPICS

The realm of International Journal of Computer Science and Security (IJCSS) extends, but not limited, to the following:

- Authentication and authorization models
- Computer Engineering
- Computer Networks
- Cryptography
- Databases
- Image processing
- Operating systems
- Programming languages
- Signal processing
- Theory
- Communications and data security
- Bioinformatics
- Computer graphics
- Computer security
- Data mining
- Electronic commerce
- Object Orientation
- Parallel and distributed processing
- Robotics
- Software engineering

CALL FOR PAPERS

Volume: 5 - **Issue:** 4 - July 2011

i. Paper Submission: July 31, 2011

ii. Author Notification: September 01, 2011

iii. Issue Publication: September / October 2011

CONTACT INFORMATION

Computer Science Journals Sdn Bhd

M-3-19, Plaza Damas Sri Hartamas
50480, Kuala Lumpur MALAYSIA

Phone: 006 03 6207 1607
006 03 2782 6991

Fax: 006 03 6207 1697

Email: cscpress@cscjournals.org

CSC PUBLISHERS © 2011
COMPUTER SCIENCE JOURNALS SDN BHD
M-3-19, PLAZA DAMAS
SRI HARTAMAS
50480, KUALA LUMPUR
MALAYSIA

PHONE: 006 03 6207 1607
006 03 2782 6991

FAX: 006 03 6207 1697
EMAIL: cscpress@cscjournals.org