IJCI Menofyia University
Faculty of Computers & Information

International Journal of Computers and Information

Volume 2 No 1

2009

Faculty of Computers and Information

IJCI

INTERNATIONAL JOURNAL OF COMPUTERS AND INFORMATION

Faculty of Computers and Information, Menoufia University

ISBN : (1687-8753)
رقم الايداع : ١٢٩١٥ لسنة ٢٠٠٧

Faculty of Computers and Information
IJCI
INTERNATIONAL JOURNAL OF COMPUTERS AND INFORMATION
Faculty of Computers and Information, Menoufia University

Table of Contents:

Faculty of Computers and Information, Menoufia University
http://www.mufic.com

Vol. 2- No.1 **January 2009**

ISBN : (1687-8753)
رقم الإيداع : ١٢٩١٥ لسنة ٢٠٠٧

Faculty of Computers and Information
IJCI
INTERNATIONAL JOURNAL OF COMPUTERS AND INFORMATION
Faculty of Computers and Information, Menoufia University

TITLE OF THE PAPER (Capital, 14Pt Time New Roman, Bold, Centered)

Author address (10Pt, Time New Romen , Centered)
Author address (10pt, Time New Roman , Centered)
E_mail@server.xxx(10Pt, Time New Romen , Centered)

Abstract ***:*** *this is a sample of the format of your full paper. Use word for windows (Microsoft) or equivalent word processor with exactly the same " printing result" by tuning 2cm from right and 2cm from left in the Microsoft word package, or equivalently by keeping 2.5cm, real distance, from right and from left. Use single space. Use one column format paper after the keywords Use 11Pt Time New Roman For Abstract, and KEYWORDS USE Italic*

Keywords : - Leave one blank line after the Abstract and write you Keywords (6-11 words)

1. Introduction

As you can see for the title of paper you must use 14Pt, centered, bold, Time New Roman. Leave one blank line and then type Authors' Name (Capitalize each word,12Pt,Time New Roman, centered). Address (in 12Pt Time New Roman, Centered). THEN you must type your email address e_mail@server.xxx and your website address http://www.yourwebaddress.xxx.xx both in 10pt Time New Roman, Centered).

The heading of each section should be printed in small 14pt, left justified, bold, Time New Roman. You must use numbers 1 , 2, 3, … for the section numbering and not Latin numbering (1.11,111,…).

2. Problem Formulation

Please leave two blank lines between successive sections as here.

Mathematical Equations must be numbered as follow: (1), (2), … , (990.nd not (1.1), (1.2)…, etc. depending on your various sections.

3. Subsection

When including a sub-subsection you must use for its heading, small letters, 12pt, left justified, bold Time New Roman as here.

3.1 sub-subsection

When including a sub-subsection you must use for its heading, small letters,,11pt, left justified, bold Time New Roman as here.

4. Problem Solution

Figures and Tables should be numbered .'S follow: Figure 1, Figure 2… etc. Table 1, Table 2 …etc. if your paper deviates significantly from these specifications, Our Publishing house may not be able to include your paper in the proceedings. When citing references in the text of the abstract, type the corresponding number in square brackets as shown at the end of this sentence (1)

5. Conclusion

Please follow our instructions faithfully, otherwise you have t resubmit your full paper. This will enable us to maintain uniformity in the journal proceedings. Thank you for your cooperation and contribution.

6. References

[1] X1. Author, Title of paper, International Journal of Computers and information, Vol. X, NO. X, 20 XX,pp. XX-XX

[2] X2 Author, Title of paper, Title of the book ,Publishing House,20XX

A Hybrid Evolutionary Algorithm for Solving Flexible Job Shop Scheduling Problem

Mahmoud Riad Mahmoud
Institute of Statistical Studies & Research

Mohamed Sayed Ali Osman,
Higher Technological Institute

Ramadan Abd El-hamed Zean El-Deen
Institute of Statistical Studies & Research

Abd Al-azeem Mohamed Abd Al-azeem
Institute of Statistical Studies & Research

Abstract *This paper presents an Evolutionary Algorithm (EA) to solve the flexible job shop scheduling problem, especially minimizing the makespan. A hybrid algorithm is introduced for solving flexible job shop scheduling problem. The proposed algorithm consists of three sequential stages. The first stage is a new technique for initializing feasible solutions, is used as initial population for the second stage. The second stage uses genetic algorithm to improve the solutions that have been found in the first stage. The final stage uses tabu search to improve the best solution that has been found by genetic algorithm. The Job Shop Scheduling Problem (JSSP) is an NP-hard combinatorial optimization problem that has long challenged researchers. A schedule is a mapping of operations to time slots on the machines. The makespan is the maximum completion time of the jobs. One of the objectives of the JSSP is to find a schedule that minimizes the makespan [12]. Some problems from references are solved using the proposed algorithm and an implementation study is presented. The implementation study shows the efficiency of the proposed algorithm.*

Key words: *flexible job shop scheduling, genetic algorithm, heuristic and tabu search*

1. Introduction

In the classical Job Shop Scheduling Problem (JSSP), n jobs are processed to completion on m unrelated machines. Each job requires processing on each machine exactly once. For each job, technology constraints specify a complete, distinct routing, which is fixed and known in advance. Processing times are sequence independent, fixed, and known in advance. Each machine is continuously available from time zero, and operations are processed without interruption (non-preemption). The common objective is to minimize the maximum completion time (makespan). Each machine is limited to executing one operation (one step of the job route) at a time. Each job has a release-date (the time after which the operations in the job may be executed) and a due-date (the time by which the last activity in the job must finish).

Scheduling has been defined as "the art of assigning resources to tasks in order to insure the termination of these tasks in a reasonable amount of time", or it is allocation of resource over time to perform a collection of tasks [9].

From the area of complexity theory we know that problems can be divided into different classes to enable a differentiation concerning the effort (runtime or memory) that is required to complete a globally best solution [13]:

1. The class of P-problems contains all problems for which the effort in relation to the problem dimension can be dominated by a polynomial function. Consequently the best solution for such problems can be found in most cases due to the computing power available today. It is even possible in the worst case just to try all possible solution candidates iteratively.
2. The class of NP-problems in contrast represents such instances whose effort increases more than polynomial (i.e. exponential or even over-exponential) depending on the problem size. Therefore, problems of that type cannot be solved algorithmically even in rather low dimensions because of the enormous number of potential solution candidates that makes it impossible to find the global optimum in acceptable time.

Job Shop scheduling is a well known scheduling problem. It is NP-hard and one of the most intractable combinatorial problems [3]. There are many methods to solve scheduling problems. These methods are trying to find optimal solution (exact solution) or acceptable solution (approximate solution). In most cases, to find the optimal solution is unpractical (some times impossible). Since it's impossible to find the optimal solution, it's necessary to use methods that find a good solution but no guarantee for how far that solution from the optimal solution. For example, tabu search and genetic algorithms are methods that are used to find an approximate solution.

Tabu search methods have been applied successfully for the scheduling problems and as solvers of mixed integer programming problems [9]. Nowicki and Smutnicki [7] implemented tabu search methods for job shop and flow shop

scheduling problems. Vaessens [7] showed that tabu search methods (in specific job shop scheduling cases) are superior over other approaches such as simulated annealing, genetic algorithms, and neural networks.

The basic idea of tabu search is to explore the search space of all feasible scheduling solutions by a sequence of moves. A move from one schedule to another schedule is made by evaluating all candidates and choosing the best available, just like gradient-based techniques. Some moves are classified as tabu (i.e., they are forbidden) because they either trap the search at a local optimum, or they lead to cycling (repeating part of the search). These moves are put onto something called the tabu List, which is built up from the history of moves used during the search. These tabu moves force examination of the search space until the old solution area (e.g., local optimum) is left behind. Another key element is that of freeing the search by a short-term memory function that provides "strategic forgetting". Tabu search methods have been evolving to more advanced frameworks that include longer term memory mechanisms. These advanced frameworks are sometimes referred as Adaptive Memory Programming (AMP) [9].

Genetic Algorithms (GAs) are a part of evolutionary computing, which is a rapidly growing area of artificial intelligence. As you can guess, genetic algorithms are inspired by Darwin's theory of evolution. When solving a problem using GA, we are usually looking for some solution, which will be the best among others. The space of all feasible solutions (the set of solutions among which the desired solution resides) is called search space (also state space). Each point in the search space represents one possible solution. Each possible solution can be "marked" by its value (or fitness) for the problem. With GAs, we look for the best solution among a number of possible solutions where the best solution represented by one point in the search space.

The next section is assigned to the problem description, and then followed by section three which is assigned to describe the proposed hybrid algorithm. In addition, section four is assigned to give an illustrative example on the proposed algorithm, and in section five, the implementation part is given. Finally, section six represents the conclusion.

2. Description of flexible job shop scheduling

Flexible job shop is a generalization of the job shop and the parallel machine environment, which provides a closer approximation to a wide range of real manufacturing systems [٦].

The flexible job shop problem is more complex than the job shop problem because of the additional need to determine the assignment of operations to machines.

The total flexible job shop scheduling problem can be described as:

1. A set of *n* jobs = $J = \{j_1, j_2, ..., j_n\}$ and these jobs are independent of one another. In addition, each job j_r (r = 1, 2, ..., n) is available from time zero. And, there is no priority for achieving any job earlier than others.
2. Each job consists of a set of *x* operations = $\{o_{i\,r}\}_{i=1}^{x}$ where $o_{i\,r}$ means operation number i of a job j_r.
3. These operations should be achieved in a predetermined order $x_{o\,r}$ which means the order of operation number o of the job j_r
4. Each operation should be assigned for a determined machine from a set of m machines = $M = \{M_1, M_2, ..., M_m\}$
5. An operation which has started can't leave the machine until completion (non-preemption condition).
6. Each machine can perform operations one after another.
7. Each machine has the capability to achieve any operation.
8. The time required to complete the whole jobs constitutes the makespan $C_{\max}$ ($C_{\max} = \max \{C_r\}_{r=1}^{n}$, and C_r = completion time of all operation of job j_r).
9. The objective is to determine the set of completion times for each operation which minimizes $C_{\max}$.

Of course, for simplicity requirements, using some notations are essential for making a mathematical model of the total flexible job shop scheduling problem as follows:

- $o_{i\,r}$ means operation number i of job j_r
- $x_{o\,r}$ means the order of operation number o of job j_r
- $P_{x\,r\,q}$ = processing time of operation with order x of job j_r on machine M_q

- $s_{x\,r\,q}$ = starting time of operation with order x of job j_r on machine M_q
- C_r = completion time of all operation of job j_r
- $I_{x\,q} = \begin{cases} 1, & \text{if machine } M_q \text{ is assigned for} \\ & \text{the operation with order x} \\ 0, & \text{otherwise} \end{cases}$
- $x_{t,r}$ = the total number of operations of a job j_r
- $m_{t,r}$ = the total number of machines available for the job j_r
  - MR_b = the time that machine b (M_b) is available to work.
  - n is the total number of jobs.
  - The objective is to determine the set of completion times for each operation which minimizes $C_{\max}$.

So, formulating a mathematical model for the total flexible job shop scheduling problem can be constructed as follows:

$$\text{Min } [z = \max \{C_r\}_{r=1}^{n}], \text{ where } C_r = \sum_{x=1}^{x=x_{t,r}} \sum_{q=1}^{q=m_{t,r}} I_{x\,q}\ P_{x\,r\,q}$$

Subject to:

$$s_{(x+1)rb} = \max \{ s_{x\,r\,a} + P_{x\,r\,a},\ MR_b \}, \quad \forall M_a, M_b \in M$$

$$s_{1rq} \in [0, \infty[\quad , \forall r, q$$

$$MR_b \in [0, \infty[\quad , \forall b$$

$$C_j > 0$$

$$p_{x\,j\,m} > 0$$

$$I \in \{0, 1\}$$

Bruker and Schlie [٦] were among the first to address the Flexible job shop problem. They developed a polynomial algorithm for solving the flexible job shop scheduling problem with two jobs. Chambers [٦] developed a tabu search algorithm to solve the problem. Mastrolilli and Gambardella [٦] proposed two neighborhood functions for the Flexible job shop problem. Yang [٦] presented a new genetic algorithm (GA)-based discrete dynamic programming approach. Kacem and Borne [٦] proposed the approach by localization to solve the resource assignment problem, and an evolutionary approach controlled by the assignment model for the Flexible job shop problem. Wu and Weng [٦] considered the problem with job earliness and tardiness objectives, and proposed a multiagent scheduling method. Xia and Wu [14] treated this problem with a hybrid of particle swarm optimization and simulated annealing as a local search algorithm. Zhang and Gen [16] proposed a multistage operation-based genetic algorithm to deal with the Flexible job shop problem from a point view of dynamic programming.

3. Description of evolutionary algorithm

Evolutionary Algorithms are in the area of study which uses the principles of Darwin's evolution to search large number of feasible solutions for finding the optimal (best) solution in complex problems [5].

One of the extra benefits of evolutionary algorithms over standard search or optimization techniques is the algorithm's modularity. For each of the stages in an evolutionary algorithm, whether creating the initial sample of candidate solutions, called chromosomes, scoring a generation or creating a new generation, the same (or similar) task is done on every chromosome in the sample. This allows evolutionary algorithms to be multithreaded easily so that they can be run in parallel. The ways that this basic algorithm can be implemented are many. The earliest and most basic algorithms based on the above are the Genetic Algorithm (GA) that was pioneered by John Holland from the 1960s and Genetic Programming in the early 1970s [5].

The proposed evolutionary algorithm is a hybrid algorithm, consists of three algorithms (stages):

1. Dispatching rule (shortest processing time SPT)

2. Genetic algorithm
3. Tabu search

In the beginning, the proposed algorithm starts by creating initial solutions (initial population of genetic algorithm stage) using the principle of SPT. Next, genetic algorithm is trying to improve the result of the SPT stage (initial solutions). Finally, tabu search algorithm stage is used to improve the solution that is found by the genetic algorithm stage.

The big challenge was to find a suitable representation for encoding a solution that make it is easy to move from one stage (algorithm) to another. Thinking that, the most difficult part was in the genetic algorithm stage and how to find that representation that could deal with genetics' steps. So, genetic algorithm and chromosome representation has been focused in for finding a suitable way to encode the solutions of the problem.

3.1. Encoding the problem for genetic algorithm

Generally, chromosomes are encoding as simple binary vectors. Unfortunately, this approach cannot frequently be used for real world engineering problems such as combinatorial ones. Two categories of encoding are used for real world engineering problems [١١]:

- Direct chromosome representation.
- Indirect chromosome representation.

The first, the direct chromosome representation can represent a scheduling problem by using the schedule itself as a chromosome. This method generally requires developing specific genetic operators. The second, the indirect chromosome representation does not directly represent a schedule, and transition from the chromosome representation to a legal schedule builder (decoder) is needed prior to evaluation. In the proposed algorithm, direct representation was chosen to give feasibility and legibility to a chromosome and simplicity of utilization for a user. There are two direct representational chromosomes:

1. Parallel Machines Encoding (PME).
2. Parallel Jobs Encoding (PJE).

The second type of direct chromosomes representation was used. It is a direct encoding which permits to solve some of the problems met in the first encoding (indirect representation) such as illegal schedules (they can't be understood without a decoder) after a crossover operation and the creation of the first population. Indeed, this encoding integrates the precedence constraints.

3.2. Shortest processing time algorithm

Although dispatching rules are not better than the local search methods, they are the more frequently applied heuristics due to their simplicity of implementation and their low time complexity. When a machine is available, a priority-based dispatching rule examines the waiting jobs and selects the job with the highest priority to be processed next. Recently, the introduction of composite dispatching rules (CDRs) has been increasingly investigated by the some researchers. These rules are the heuristic combination of single dispatching rules that aim to take over the advantages of the former. The results show that, with careful combination, the composite dispatching rules do perform better than the single ones in the quality of schedules [١٨].

Depending on the specification of each rule, it can be classified [١٨] into:

- Simple Priority Rules
- CDRs
- Weighted Priority Indexes
- Heuristic Scheduling Rules

Simple Priority Rules (SPRs) are usually based on a single objective function. They usually involve only one model parameter, such as:

- Processing time
- Due date
- Number of operations or arrival time

The Shortest Processing Time (SPT) is an example of a SPR. It orders the jobs on the queue in the order of increasing processing times. When a machine is freed, the next job with the shortest time in the queue will be removed for processing. SPT has been found to be the best rule for minimizing the mean flow time and number of tardy jobs.

The CDRs have been studied to join good features from such SPRs. There are two kinds of CDRs presented in literature:

- The first type involves arranging a select number of SPRs at different machines or work centers. Each machine or work center uses a single rule. When a job enters a specific machine, it is processed by the SPR that is predetermined for that machine.
- The second type involves applying the composition of several SPRs to evaluate the priorities of jobs on the queue. The latter type is executed similarly to SPRs; when a machine is free, this CDR evaluates the queue and then selects the job with the highest priority.

Weighted priority index rules are the linear combination of SPRs described above with computed weights. Depending on specific business domains, the importance of a job determines its weight.

Heuristic rules are rules that depend on the configuration of the system. These rules are usually used together with previous rules, such as SPRs, CDRs or weighted priority index rules.

The proposed algorithm uses the principal of a Simple Priority Rule (shortest processing time) to initialize some feasible solutions. The initialized solutions will be used as the initial population of the genetic algorithm stage. Using SPT as a first stage gives the algorithm a better start than initializing the initial solution randomly.

The first stage, the SPT algorithm is used for getting solutions that will be used as initial population for the genetic algorithm is described as followed:

- Step 1. Scan for unscheduled jobs.
- Step 2. Select randomly one job (suppose the job is b)
- Step 3.
  For j = 1 to x (number of operations of b)
  Assign a machine with the smallest processing time to achieve x_j (select a machine randomly if there are more than one machine have the smallest processing time).
  j = j + 1
  Next
- Step 4. Go to step 1 until all jobs are scheduled.
- If all operations of each job is assigned for a machine, calculate the start time for each operation column by column (by the sequence column 1, column 2, column 3,…….)

3.3. Genetic algorithm

Genetic Algorithms (GAs) are a general methodology for investigating a discrete solutions space in a manner that is similar to process of natural selection procedure in biological systems.

Genetic algorithm is one of the stochastic search algorithms based on biological evolution. In order to solve a clearly defined problem and an offspring represented the candidate of solutions. GA is according to crossover and mutation operators with their probabilities to produce a set of offspring chromosomes. GA likes an over and over process, and tries to find one or more highly fit chromosomes [4].

Genetic algorithms, as the name implies, are a type of algorithms, not a single one. This means that many variants of the basic idea exist, and that individual applications may be highly different. However, every variant should include the following operations [15]:

1. A method for encoding solutions to the problem into a string of characters.

2. An evaluation function which takes a string as an input and returns a fitness value which measures the quality of the solution the strings describes.
3. An adaptive plan, whose purpose is to produce new, improved generation of solutions from the current one.

The advantage of applying GAs to hard optimization problems lies in their ability to scan broader regions of the solution space than heuristic methods based upon neighborhood search do. Nevertheless, also GAs are frequently faced with a problem which, at least in its impact, is quite similar to the problem of fall into a local optimum. This drawback is called *premature convergence* in the terminology of GAs [1].

3.3.1. Crossover operators

In this section, crossover operator is presented to adapt to the parallel jobs encoding. The algorithm of this crossover is presented as follows:

- Step 1. Choose randomly the best two parents (chromosomes) and select randomly one job (a row of the matrix). Suppose that the job J is randomly selected.
- Step 2. The operations of job J in Child 1 received the same machines as assigned to the same job J of Parent 2.
- Step 3. Browse all of the jobs (the rows) R of parent 1
  While ((1 ≤ R ≤ N) and (R ≠ J)) do
  Copy the remainder of the machines assigned to the operations of job R of Parent 2 in the same job R of Child 1
  R = R + 1
  End
- Step 4. To obtain Child 2, go to Step 2 and interchange the roles of Parent 1 and Parent 2.

3.3.2. Mutation operator

In this part, a mutation operator is presented, and is called the controlled mutation operator. This operator is designed for parallel jobs encoding where it can balance the machine loads.

The algorithm of this operator is presented as follows:

- Step 1. Choose randomly one chromosome S and calculates the load of all machines in the shop according to S.
- Step 2. Choose randomly one operation from the set of operations in chromosome S assigned to a machine with a high load.
- Step 3. Assign this operation to another machine with a small load.

3.3.3. Fitness function

Fitness is a measure of how well an algorithm has learnt to predict the outputs from the inputs. The goal of having a fitness evaluation is to give feedback to the learning algorithm regarding which individuals should have a higher probability of being allowed to multiply and reproduce and which of them should have a higher probability of being removed from the population.

Evaluation functions play the same role in EAs as the environment in natural evolution. It must indicate the aspects of schedules which make them seem right or wrong to the users. The objective is to minimize the makespan.

The makespan $C_{\max}$ is calculated, according to the following algorithm:

Select chromosome X = chromosome 1
For X = 1 to popsize (population size)
Select job L = job 1
While ($1 \le L \le N$) do
Calculate C_L = time of completion of the last scheduling operation of job L
L = L + 1
End While

$C_{\max}$ X = Max $\{C_1, C_2, \ldots, C_N\}$, ($N$ is the number of jobs)
X = X +1
Next

For each chromosome the fitness function aims to find the minimum of all $C_{\max}$, and is represented as follows:
Fitness = Min $\{C_{\max}1, C_{\max}2, \ldots, C_{\max}X\}$, ($X$ is the population size)

3.4. Tabu algorithm

The tabu search was proposed by Glover. TS try to keep track of the visited search space to avoid revisiting the same solutions and thus improving the efficiency of search [2]. So, the proposed algorithm used three types of tabu list, in tabu search algorithm stage. It can be described as follows:

1. Machine-tabu-list: the machine is not being allowed to be assigned until trying all machines.
2. Job-tabu-list: the job is not being allowed to be visited until completing all jobs.
3. Operation-tabu-list: the operation is not being allowed to be selected until completing all operations.

Each step of the proposed tabu search algorithm stage is illustrated in the next example.

4. An illustrative example

Three jobs and five machines are considered. The operating sequences of these jobs are as follows:

- Job1: $o_{1,1} \rightarrow o_{2,1} \rightarrow o_{3,1}$
- Job2: $o_{1,2} \rightarrow o_{2,2} \rightarrow o_{3,2}$
- Job3: $o_{1,3} \rightarrow o_{2,3}$

In which $o_{2,3}$ means operation two in job three. This problem is an example of a total flexibility in which any operation can be performed equally well by any machine with different processing times. According to the machine used, the processing time of operations is different as described in table 1. All machines and jobs are available from time zero. Non-preemptive principle is used. Besides, parallel jobs encoding was used to represent any solution. In this case, for each operation, the proposed algorithm determines the following:

$$(\text{machine number, starting time, processing time,completion time})$$

For example, an operation has $(M_3, 4, 2, 6)$ that means the assigned machine is M_3 and the starting time of the operation on M_3 is 4 and the processing time is 2 and the completion time is 6.

Table 1 Processing time of the operations on different machines.

x_j	operation	M_1	M_2	M_3	M_4	M_5
x_1	1, 1	1	8	3	7	5
x_2	2, 1	3	5	2	6	4
x_3	3, 1	6	7	1	4	3
x_1	1, 2	1	4	5	3	8
x_2	2, 2	2	8	4	9	3
x_3	3, 2	9	5	1	2	4
x_1	1, 3	1	8	9	3	2
x_2	2, 3	5	9	2	5	3

4.1. Initial population

Some symbols were used to make it easy to describe each chromosome as the following:

1. GenNo *i* = Generation number *i*
2. Ch *i* = Chromosome number *i*

As an example, two solutions were initialized by using SPT technique as explained before. The next two chromosomes ((GenNo 0, Ch1) and (GenNo 0, Ch2)) represent the two solutions generated by the SPT stage.

(GenNo 0, Ch1)

J_j	Random order	x_1	x_2	x_3	$C_{\max J}$	$\max C_{\max}$
J_1	1	$(M_1,0,1,1)$	$(M_3,1,2,3)$	$(M_3,5,1,6)$	6	
J_2	3	$(M_1,2,1,3)$	$(M_1,3,2,5)$	$(M_3,6,1,7)$	7	7
J_3	2	$(M_1,1,1,2)$	$(M_3,3,2,5)$		5	

(GenNo 0, Ch2)

J_j	Random order	x_1	x_2	x_3	$C_{\max J}$	$\max C_{\max}$
J_1	2	$(M_1,1,1,2)$	$(M_3,2,2,4)$	$(M_3,7,1,8)$	8	
J_2	1	$(M_1,0,1,1)$	$(M_1,3,2,5)$	$(M_3,6,1,7)$	7	8
J_3	3	$(M_1,2,1,3)$	$(M_3,4,2,6)$		6	

4.2. Genetic algorithm

In the proposed genetic algorithm stage, genetic algorithm used the initial solutions that are generated in the first stage as the initial population.

4.2.1. Crossover steps

- Generating (GenNo 1, Ch1) by Selecting randomly a job (as an example J_1) from (GenNo 0, Ch2) and assign its machines for (GenNo 1, Ch1) to get the same machines assigned
- Complete the remaining machines of (GenNo 1, Ch1) from (GenNo 0, Ch1).
- Generating (GenNo 1, Ch2) is the same steps for generating (GenNo 1, Ch1) by switching Ch1 and Ch2.
- Calculate the starting time for each chromosome by using random order for jobs for each chromosome.

The next two solutions ((GenNo 1, Ch1) and (GenNo 1, Ch2)) represent the two chromosomes after crossover steps.

(GenNo 1, Ch1)

J_j	Random order	x_1	x_2	x_3	$C_{\max J}$	$\max C_{\max}$
J_1	1	$(M_1,0,1,1)$	$(M_3,1,2,3)$	$(M_3,5,1,6)$	6	
J_2	2	$(M_1,1,1,2)$	$(M_1,3,2,5)$	$(M_3,6,1,7)$	7	7
J_3	3	$(M_1,2,1,3)$	$(M_3,3,2,5)$		5	

(GenNo 1, Ch2)

J_j	Random order	x_1	x_2	x_3	$C_{\max J}$	$\max C_{\max}$
J_1	3	$(M_1,2,1,3)$	$(M_3,3,2,5)$	$(M_3,6,1,7)$	7	
J_2	2	$(M_1,1,1,2)$	$(M_1,3,2,5)$	$(M_3,5,1,6)$	6	7
J_3	1	$(M_1,0,1,1)$	$(M_3,1,2,3)$		3	

4.2.2. Mutation steps

- Select (GenNo 1, Ch2) for mutation and calculate machine load for the selected chromosome as illustrated in the next table 2.
-

Table 2 Calculation of machine load of (GenNo. 1, Ch2)

M_1	M_2	M_3	M_4	M_5
5	0	6	0	0

- Select M_3 (the highest load machine) to be exchanged by a machine with the smallest load.
  - Select randomly a job where M_3 is assigned to achieve an operation (suppose J_1).
  - Select randomly an operation of the selected job J_1 (suppose x_3).
  - Select randomly from the machines with the smallest load, a machine for completing x_3.
  - Exchange M_3 by M_5 to achieve operation x_3 of job J_1.

The next chromosome ((GenNo 1, Ch3)) represents the solution after mutation steps.

(GenNo 1, Ch3) J_j	Random order	x_1	x_2	x_3	$C_{\max J}$	$\max C_{\max}$
J_1	1	$(M_1,0,1,1)$	$(M_3,1,2,3)$	$(M_5,3,3,6)$	6	
J_2	2	$(M_1,1,1,2)$	$(M_1,3,2,5)$	$(M_3,5,1,6)$	6	6
J_3	3	$(M_1,2,1,3)$	$(M_3,3,2,5)$		5	

4.2.3. Fitness

In fitness stage, the makespan is calculated for each chromosome to determine the chromosomes that will be survive for the next population as illustrated in the next table 3.

Table 3 Calculate fitness of each chromosome

	(GenNo 0, Ch1)	(GenNo 0, Ch2)	(GenNo 1, Ch1)	(GenNo 1, Ch2)	(GenNo 1, Ch3)
$\max C_{\max}$	7	8	7	7	6

4.2.4. Selection

The next population (population number 1) has the same size of the previous population. Selection step is completed by selecting chromosomes randomly from offspring (chromosomes) which have been created in the previous step (offspring consist of the original population and two chromosomes that created by crossover step, and finally, one chromosome that created by mutation step) that have the smallest $C_{\max}$ (the best fitness).

4.2.5. Next population

In this step, the algorithm Uses population number l (l = 1, 2, 3, ...) to generate next population number (l+1) and so on until the desired number of populations are get.

4.3. Tabu search

After generating the required number of population using the genetic algorithm, Select randomly the solution (chromosome) with the smallest $C_{\max}$ and let us suppose it's as (GenNo. 1, Ch3).

(GenNo 1, Ch3)

J_j	Random order	x_1	x_2	x_3	$C_{\max J}$	$\max C_{\max}$
J_1	1	$(M_1,0,1,1)$	$(M_3,1,2,3)$	$(M_5,3,3,6)$	6	
J_2	2	$(M_1,1,1,2)$	$(M_1,3,2,5)$	$(M_3,5,1,6)$	6	6
J_3	3	$(M_1,2,1,3)$	$(M_3,3,2,5)$		5	

Tabu search algorithm procedures:

1. Select the first job in the chromosome (J_1).
2. Put the selected job in job-tabu-list
3. Select by ordering an available operation of J_1 (for the first time it will be x_1)
4. Put the selected operation (x_1) in operation-tabu-list
5. Put the machine assigned for completing x_1 in a machine-tabu-list (to not be visited until trying other machines available for x_1).
6. Exchange the machine assigned for achieving x_1 by another available machine (say M_a)
7. Calculate the starting time for the new solution (using the same order assigned for the solution before).
8. Calculate $C_{\max}$ for the new solution = $C_{\max S_i}$
9. If the new ($\max C_{\max}$) is smaller than the old make a move.
10. Add M_a in machine-tabu-list
11. Go to step 6 until completing all available machines.
12. Clear machine-tabu-list.
13. Go to step 3 until completing all operations.
14. Clear operation-tabu-list.
15. Go to step 1 until selecting all jobs.
16. Clear job-tabu-list.

If it is decided to use multi-start of tabu, it is done only by exchange step 9 of the tabu algorithm by the following:

1. If the new ($\max C_{\max}$) is smaller than or equal the old make a move.

5. Implementation

The implementation was done on the system of Microsoft, Windows XP Professional Service Pack 2 and on a computer with Intel Pentium III processor 800 MHz and 256 MB of RAM. A special program was coded using visual basic 6 to implement the proposed hybrid algorithm.

5.1. The First Test (10×6 problem)

This problem presents a partial flexibility. The problem consists of 10 jobs and 6 machines and each job has 5 or 6 operations in its operating sequence. The problem is called "mk01" and the best known makespan is equal to 40 units of time [10].

The following table 4 shows the results of ten runs. Some notations are used in all tables that illustrate the results, as "I" stands for the number of initial solutions, "P" stands for the population size and "G" stands for the number of generations.

Table 4 Results of the hybrid algorithm for each stage.

I = 50, P = 10, G = 10

Run	Makespan found by				Makespan
No.	SPT	GA	TS	TS with multi-start = 3	(units of time)
1	48	48	48	48	48
2	49	49	46	43	43
3	48	48	44	44	44
4	48	48	45	44	44
5	48	47	42	39	39
6	48	48	45	44	44
7	49	49	47	44	44
8	48	48	45	44	44
9	46	46	41	38	38
10	48	47	43	40	40

It's clear from the results that the use of multi-start technique of the tabu algorithm has the capability to force the solution to be better than the solution that it is found without using multi-start strategy.

Table 5 represents the best solution has been found by the proposed algorithm. Some symbols are used to make it easer to represent the solution, as follows:

- S, refers to start time of the machine
- F, refers to finish time of the machine
- J, refers to the job that the machine is assigned to achieve an operation of that job
- O, refers to the operation order that the machine is assigned to achieve it

Table 5 The best solution (makespan = 38)

M1				M2				M3				M4				M5				M6			
S	**F**	**J**	**O**	**S**	**F**	**J**	**O**	**S**	**F**	**J**	**O**	**S**	**F**	**J**	**O**	**S**	**F**	**J**	**O**	**S**	**F**	**J**	**O**
0	**1**	**4**	**1**	**0**	**2**	**2**	**1**	**0**	**4**	**1**	**1**	**3**	**5**	**7**	**2**	**0**	**3**	**5**	**1**	**0**	**2**	**10**	**1**
5	**7**	**6**	**2**	**2**	**8**	**3**	**1**	**4**	**5**	**2**	**2**	**11**	**14**	**9**	**3**	**6**	**11**	**9**	**2**	**2**	**3**	**7**	**1**
7	**8**	**5**	**2**	**8**	**14**	**4**	**2**	**8**	**12**	**8**	**2**	**14**	**20**	**2**	**4**	**11**	**14**	**1**	**2**	**3**	**5**	**6**	**1**
8	**10**	**2**	**3**	**14**	**20**	**5**	**3**	**12**	**16**	**7**	**3**	**20**	**26**	**10**	**5**	**14**	**17**	**10**	**3**	**5**	**6**	**9**	**1**
16	**17**	**3**	**3**	**20**	**26**	**6**	**4**	**16**	**20**	**6**	**3**	**26**	**32**	**5**	**5**	**17**	**18**	**7**	**4**	**6**	**8**	**8**	**1**
17	**18**	**8**	**3**	**26**	**32**	**8**	**4**	**20**	**21**	**4**	**3**	**32**	**38**	**9**	**6**	**21**	**24**	**4**	**4**	**8**	**14**	**10**	**2**
18	**20**	**9**	**4**	**32**	**38**	**8**	**5**	**21**	**25**	**5**	**4**					**25**	**30**	**3**	**5**	**14**	**16**	**3**	**2**
20	**21**	**1**	**4**					**25**	**26**	**7**	**5**									**16**	**18**	**1**	**3**
26	**27**	**6**	**5**					**26**	**30**	**9**	**5**									**18**	**19**	**10**	**4**
27	**30**	**10**	**6**					**30**	**31**	**1**	**5**									**19**	**25**	**3**	**4**
30	**33**	**6**	**6**					**32**	**36**	**5**	**6**									**25**	**30**	**2**	**5**
																				30	**32**	**4**	**5**
																				32	**38**	**1**	**6**

5.2. The Second Test (10 × 10 problem)

Implementation of the hybrid algorithm was done to a problem with 10 jobs and 10 machines. Each job has 3 operations in its operating sequence. This problem presents a total flexibility [14] [11].

Ten runs were performed and the following table 6 shows the results by using the proposed algorithm for illustrating the efficiency of the proposed hybrid algorithm.]

Table 6 Results of our algorithm for each stage.

I = 50, P = 10, G = 10

Run No.	**Makespan found by**				**Makespan (units of time)**
	SPT	**GA**	**TS**	**TS with multi-start = 3**	
1	**8**	**8**	**8**	**8**	**8**
2	**9**	**8**	**8**	**8**	**8**
3	**9**	**9**	**8**	**8**	**8**
4	**8**	**8**	**8**	**8**	**8**
5	**9**	**9**	**8**	**8**	**8**
6	**9**	**8**	**8**	**8**	**8**
7	**8**	**8**	**8**	**7**	**7**

Table 6 (continued) Results of our algorithm for each stage.

Run No.	**Makespan found by**				**Makespan (units of time)**
	SPT	**GA**	**TS**	**TS with multi-start = 3**	
8	**8**	**8**	**8**	**8**	**8**
9	**9**	**8**	**8**	**8**	**8**
10	**9**	**9**	**9**	**8**	**8**

Table 6 shows the efficiency of our hybrid algorithm and make it clear to think about the SPT algorithm as a good solo technique for solving the flexible job shop scheduling in some cases.

Table 7 gives a comparison between different algorithms that have been applied for solving this problem (10 × 10 problem). In table 5.2.2 and other tables of the comparison between different algorithms, the first column is labeled "Temporal decomposition", refers to F. Chetouane's method. The next column labeled "Classic GA" refers to classical genetic algorithm. The third and forth columns are labeled "Approach by localization" and "AL+CGA", respectively, are two algorithms by Kacem et al. The fifth column is labeled "PSO+SA", refers to the proposed algorithm by Xia and

Wu [14] [6]. The sixth column is labeled "hGA", refers to the hybrid genetic algorithm proposed by Gao, J. et al. [6]. Finally, the last column is labeled "Proposed algorithm", refers to the proposed hybrid evolutionary algorithm.

Table 7 Comparison of results on problem 10×10 with 30 operations

Temporal decomposition	Classic GA	Approach by localization	AL + CGA	PSO + SA	hGA	Proposed algorithm
16	7	8	7	7	7	7

Besides, the success of the proposed hybrid algorithm for finding the best known makespan for this problem, it takes about two minutes to get a solution, which makes the proposed algorithm is the fastest. It takes from the proposed algorithm to find the best solution about 2 minutes, while the "hGA" algorithm takes about 17 minutes to find the best solution [6].

5.3. The Third Test (8 × 8 problem)

This is an instance of partial flexibility. In this flexible job shop problem, there are 8 jobs with 27 operations to be performed on 8 machines [14].

Ten runs were performed and table 8 shows the results by using the proposed algorithm for illustrating the efficiency of the proposed hybrid algorithm.

Table 9 gives a comparison between different algorithms that have been applied for solving this problem (8 × 8 problem).

Table 8 Results of our algorithm for each stage.

I = 50, P = 10, G = 10

Run No.	Makespan found by SPT	GA	TS	TS with multi-start = 3	Makespan (units of time)
1	16	16	16	15	15
2	16	16	16	16	16
3	16	16	16	16	16
4	16	16	15	15	15
5	16	16	15	15	15
6	16	16	16	16	16
7	16	16	16	16	16
8	16	16	16	16	16
9	16	16	16	16	16
10	16	16	16	16	16

Table 9 Comparison of results on problem 8×8 with 27 operations

Temporal decomposition	Classic GA	Approach by localization	AL + CGA		PSO + SA		hGA	Proposed algorithm
19	16	16	15	16	15	16	15	15

It takes from the proposed algorithm to find the best solution about 1.5 minutes, while the "hGA" algorithm takes about 5 minutes to find the best solution [6].

5.4. The Forth Test (15 × 10 problem)

A larger-sized problem is chosen to test the performance of the proposed hybrid evolutionary algorithm. This instance has 15 jobs with 56 operations that have to be processed on 10 machines with total flexibility [14].

Ten runs were performed and table 10 shows the results that have been found by using the proposed algorithm.

Table 11 gives a comparison between different algorithms that have been applied for solving this problem (15 × 10 problem).

Table 10 Results of our algorithm for each stage.

I = 50, P = 10, G = 10

Run No.	Makespan found by SPT	GA	TS	TS with multi-start	Makespan (units of time)
1	15	14	14	14	14
2	15	15	15	15	15
3	14	14	14	14	14
4	15	14	13	13	13
5	14	14	14	14	14
6	15	15	15	15	15
7	14	14	14	14	14
8	15	13	13	13	13
9	15	15	15	15	15
10	15	15	15	15	15

Table 11 Comparison of results on problem 15×10 with 56 operations

Classic GA	AL + CGA	PSO + SA	hGA	Proposed algorithm
23	24	12	11	13

According to the data that is illustrated in table 11, the proposed algorithm did not find the best known makespan, only for this problem. But, it found a solution near the best known in a fast way. It takes from the proposed algorithm to find the best solution (by the proposed algorithm) about 3 minutes, while the "hGA" algorithm takes about 135.47 minutes to find the best solution [6].

5.5. The Fifth Test

In the next section, studying the effects of changing the parameters on the behavior of our algorithm was done. The problem of the second test (10 × 10 total flexible job shop scheduling) was chosen to be solved under various values of parameters. In each test, changing the value of only one parameter and making the others unchanging was done. In each test, five runs were made and the best solution is selected to be illustrated in figure 1, figure 2, figure 3, and figure 4.

Figure 1 (Changing in the number of initial solutions)
Best makespan in five runs: G = 10, P = 10, multi-start = 3

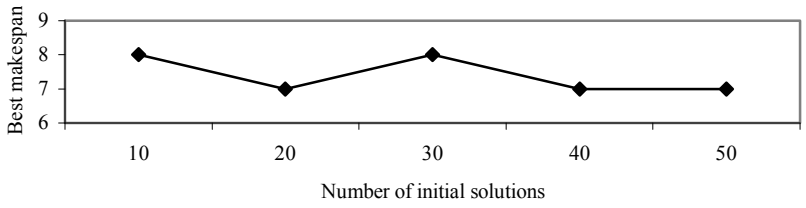


Figure 2 (Changing in the number of generations)
Best makespan in five runs: I = 50, P = 10, multi-start = 3

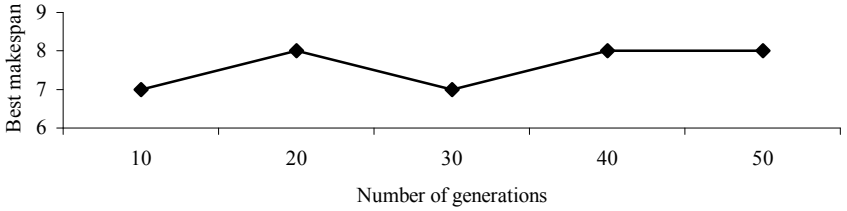


Figure 3 (Changing in the population size)
Best makespan in five runs: I = 50, G = 10, multi-start = 3

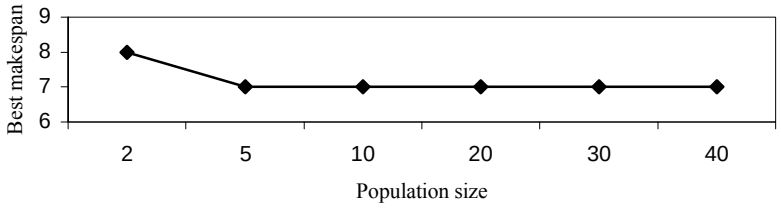
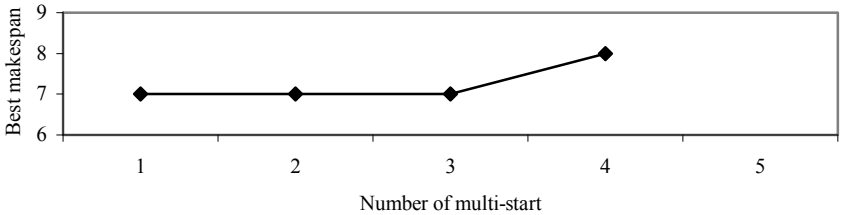


Figure 4 (Changing in the number of multi-start of Tabu)
Best makespan in five runs: I = 50, P = 10, G = 10



6. Conclusion

The application of evolutionary algorithms to a flexible job-shop scheduling problem has been defined. It is demonstrated that choosing a suitable initial solutions and good representation of chromosomes (parallel job encoding) is an important step to get better result in less iterations.

A new method has been developed for generating initial solutions over the chromosome encoding using the concept of SPT which could be considered a solo technique for solving the flexible job shop scheduling problem.

The parameters (number of initial solutions, population size, number of generations, and number of multi-start of tabu algorithm) are selected heuristically.

Implementation results show that the proposed hybrid evolution algorithm is suitable for solving the flexible job shop scheduling problem, confirming the effectiveness of the proposed approach and its fastness.

7. References

[1] Affenzeller, M., Wagner, S. and Winkler, S. (2005), "GA-Selection Revisited from an ES-Driven Point of View". Artificial Intelligence and Knowledge Engineering Applications: A Bioinspired Approach, Lecture Notes in Computer Science 3562, pp. 262-271. Springer-Verlag, 2005.

[2] Caldeira, J., Melicio, F. and Rosa, A. (2004), "Using a Hybrid Evolutionary-Taboo Algorithm to Solve the JobShop Problem". ACM Symposium on Applied Computing, SAC ’04, March 14-17, 2004, Nicosia, Cyprus.

[3] Caldeira, J., Melicio, F. and Rosa, A. (2005), "Improving on Excellence. An Evolutionary Approach", Proceedings of the 6th WSEAS Int. Conf., on Evolutionary Computing, Lisbon, Portugal, June 16-18, 2005, pp. 15-23.

[4] Chiu, H., Hsieh, K., Tang, Y. and Chien, W. (2007), "A Knowledge-Based Genetic Algorithm for the Job Shop Scheduling Problem", Proceedings of the 6th WSEAS Int. Conf. on Artificial Intelligence, Knowledge Engineering and Data Bases, Corfu Island, Greece, February 16-19, 2007.

[5] Crafti, D. (2004). "A Job Shop Scheduler using a Genetic Tree Algorithm". School of Computer Science and Software Engineering, Monash University, Clayton, Victoria, Australia. http://www.csse.monash.edu.au/hons/se-projects/2003/Crafti/thesis.pdf. January 29, 2004.

[6] Gao, J., Gen, M. and Sun, L. (2006), “A Hybrid of Genetic Algorithm and Bottleneck Shifting for Flexible Job Shop Scheduling Problem”, GECCO'06, July 8–12, 2006, Seattle, Washington, USA. Copyright 2006 ACM 1-59593-186-4/06/0007.

[7] Glover, F. (1996), “Tabu search and adaptive memory programming - advances, applications and challenges”. Interfaces in Computer Science and Operations Research. Barr, Helgason and Kennington (eds.), Kluwer Academic Publishers.

[8] Ho, N. and Tay, J. (2005), "Evolving Dispatching Rules for solving the Flexible Job-Shop Problem". Evolutionary Computation, 2005, the 2005 IEEE Congress on 2-5 Sept. 2005, Vol. 3, pp. 2848 – 2855, ISBN: 0-7803-9363-5.

[9] Jones, A. and Rabelo, L. (1998), “Survey of Job Shop Scheduling Techniques”, NISTIR, National Institute of Standards and Technology, Gaithersburg, MD. Journal of Production Research, Vol. 38, pp. 3623-3637.

[10] Mastrolilli, M. and Gambardella, L. (2000), "Effective Neighborhood Functions for the Flexible Job Shop Problem: Appendix". IDSIA - Istituto Dalle Molle di Studi sull'Intelligenza Artificial, C.so Elvezia 36, 6900 Lugano, Switzerland. {monaldo, luca}@idsia.ch, http://www.idsia.ch. February 14, 2000.

[11] Mesghouni, K., Hammadi, S. and Borne, P. (2004), "Evolutionary Algorithms for Job-Shop Scheduling". Int. J. Appl. Math. Comput. Sci., Vol. 14, No. 1, 91–103.

[12] Resende, M. and Ribeiro, C. (2004), “Parallel Greedy Randomized Adaptive Search Procedures”, AT&T Labs Research Technical Report TD-67EKXH. To appear in Parallel Metaheuristics, E. Alba (ed.), John Wiley and Sons, 2005.

[13] Wagner, S., Affenzeller, M. and Schragl, D. (2004), "Traps and Dangers when Modelling Problems for Genetic Algorithms". Austrian Society for Cybernetic Studies. Cybernetics and Systems 2004, pp. 79-84.

[14] Xia, W. and Wu, Z. (2005), "An effective hybrid optimization approach for multi-objective flexible job-shop scheduling problems", Computers & Industrial Engineering Vol. 48, pp. 409–425.

[15] Zdansky, M. and Pozivil, J. (2002), "Combination Genetic/Tabu Search Algorithm for Hybrid Flowshops Optimization". Proceedings of ALGORITMY 2002, Conference on Scientific Computing, pp. 230-236.

[16] Zhang, H. and Gen, M. (2004), “Multistage-Based Genetic Algorithm for Flexible Job-Shop Scheduling Problem”, the 8th Asia Pacific Symposium on Intelligent and Evolutionary Systems 6th - 7th December 2004. Cairns, Australia, pp. 223-232.

IJCI. Vol. 2 – No. 1, January 2009

A Modified Algorithm for Particle Swarm Optimization with Constriction Coefficient

Mahmoud M. El-Sherbiny
Operations Research Dept, Institute of Statistical Studies and Research (ISSR),
Cairo University, Egypt.
Email: m_sherbiny@yahoo.com

Abstract *This paper introduces a modified algorithm to partical swarm based optimization that significantly reduces the number of iterations required to reach good solutions and discusses the results of experimentally comparing its performance with the performance of several variants of the standard particle swarm optimizer. The variants of the modified algorithm and the most common variants of the particle swarm optimizers are tested using a set of multimodal functions commonly used as benchmark optimization problems in evolutionary computation. Results indicate that the algorithm is highly competitive and can be considered as aviable alternative to solve the optimization problems.*

Keywords*: Particle swarm optimization; Convergence; Evolutionary computation; constriction coefficient*

1. Introduction

Particle Swarm Optimization (PSO) is a new evolutionary computation technique motivated from the simulation of social behavior and originally designed and developed by Eberhart and Kennedy [22,23,24]. The population in the PSO is called a swarm and each individual is called a particle [1]. It is inspired by the behavior of bird flocking and fish schooling. A large number of birds or fish flock synchronously, change direction suddenly, and scatter and regroup together. Each particle benefits from the experience of its own and that of the other members of the swarm during the search for food.

PSO algorithm has a number of desirable properties, including simplicity of implementation, scalability in dimension, and good empirical performance. So. It is an attractive choice for solving nonlinear programming problems. PSO algorithm had been applied to evolve weights and structure for artificial neural networks by Shi and Eberhart in 1998 [2], manufacture and milling [3], reactive power and voltage control by Abido, M.A. in 2002 [4,5], to estimate the voltage stability of the electric power distribution systems [6, 7] direct the orbits of discrete chaotic dynamical systems towards desired target region [8] and to solve the permutation flowshop sequencing problem [9]

PSO has been successfully used as an alternative to other evolutionary algorithms in the optimization of D-dimensional real functions. Particles move in a coordinated way through the D-dimensional search space towards the optimum of the function. Their movement is influenced not only by each particle own previous experience, but also by a social compulsion to move towards the best position found by its neighbours. To implement these behaviours, each particle is defined by its position and velocity in the search space. In each iteration, changes resulting from both influences in the particle's trajectory are made to its velocity. The particle's position is then updated accordingly to the calculated velocity. The PSO, its main variants and the structural model behind it are extensively discussed in [10]. Some work has been done that alters basic particle motion with some success, but the possibility for improvement in this area is still open [27].

This paper aims to introduce a modified algorithm for particle swarm optimization (MPSO) and discuss the results of experimentally comparing the performance of its versions with the standard particle swarm optimizer (SPSO)[21] and the combined PSO [25].

The rest of the paper is organized as follows: in section 2, the PSO method is described. In section 3, the modified algorithm and its versions are exposed. Test functions and test conditions are presented in sections 4 and 5. In section 6, optimization test experiments are illustrated. In section 7, the experimental results are reported, and are discussed in section 8. Finally, conclusion is reported in section 9.

2. Particle Swarm Optimization

In PSO, particles evaluate their positions relative to a goal (fitness) at every iteration, and the particles in a local neighborhood share memories of their "best" positions, then use those memories to adjust their own velocities and positions as shown in equations (1) and (2). The PSO formula defines each particle as a potential solution to a problem in the D-dimensional space, with the ith particle represented as Xi = (xi1, xi2, …, xiD). Each particle also remembers its best position, designated as X_{p_i}, and its velocity Vi = (vi1, vi2, …, viD) [11].

In each generation (iteration) t, the velocity of each particle is updated, being pulled in the direction of its own best position (X_{p_i}) and the best of all positions (Xg) reached by all particles until the preceding generation. After finding the two best values, the particle updates its velocity and position according to equations (1) and (2).

$$V_i(t) = aV_i(t-1) + b_1 r_1\left(X_{p_i} - X_i(t)\right) + b_2 r_2\left(X_g - X_i(t)\right) \quad (1)$$

$$X_i(t) = cX_i(t-1) + dV_i(t) \quad (2)$$

At iteration t, the velocity $V_i(t-1)$ is updated based on its current value affected by a momentum factor a and on a term which attracts the particle towards previously found best positions: its own previous best position (X_{p_i}) and globally best position in the whole swarm (Xg). The strength of attraction is given by the average of the own and the social attraction coefficients b1 and b2 . The particle position $X_i(t)$ is updated using its current value and the modified ly computed velocity $V_i(t)$, affected by coefficients c and d, respectively and they can be set to unity without loss of generality [21]. Randomness that useful for good state space exploration is introduced via the vectors of random numbers r1 and r2. They are usually selected as uniform random numbers in the range [0, 1].

The original PSO formula developed by Kennedy and Eberhart [1] were combined by Shi and Eberhart [2] with the introduction of an inertia parameter, ω , that was shown empirically to improve the overall performance of PSO. Clerc and Kennedy provided a theoretical analysis of particle tajectories to ensure convergence to a stable point [26],

$$\chi = \frac{b_1 r_1 X_p + b_2 r_2 X_g}{b_1 r_1 + b_2 r_2}$$

The main result of this work is the introduction of the constriction coefficient and different classes of constriction models. The objective of this theoretically derived constriction coefficient is to prevent the velocity to grow out of bounds, with the advantage that, theoretically, velocity clamping is no longer required. As a result of this study, the velocity equation (1) changes to equation 3.

$$V_i(t) = \chi\left(V_i(t-1) + b_1 r_1\left(X_{p_i} - X_i(t)\right) + b_2 r_2\left(X_g - X_i(t)\right)\right) \quad (3)$$

Several interesting variations of the PSO algorithm have recently been proposed by researchers in [12 - 17]. Many of these PSO improvements are essentially extrinsic to the particle dynamics at the heart of the PSO algorithm. One of these variations is the combined particle swarm optimization (CPSO) algorithm that presented by M. El_Sherbiny [25] and can be applied to augment of the modified algorithm presented in this paper. This paper proposes a modification to the dynamics of particles in PSO, presenting the constriction coefficient to the velocity update equation of the CPSO algorithm.

3. The Modified Algorithm

In the standard PSO algorithm, in each generation t, the velocity of each particle is updated, being pulled in the direction of its own previous best position (X_{p_i}) and the best of all positions (global position) (Xg) reached by all particles until the preceding generation. Whereas in the CPSO algorithm, in each generation t, the velocity $V_i(t-1)$ of particle i is updated based on its own best position (X_{p_i}) and the point (Xc) resulted from the combination of the

global best position (Xg) and the previous global best position (X2g) as illustrated in equation (3). Where R1 and R2 are defied as the combination weights.

$$(X_c) = R_1 (X_g) + R_2 (X_{2g}) \qquad (4)$$

In other words, after finding a new global best position for the (X_g) its old position will be assigned to (X_{2g}) and the particle updates its velocity according to equation (5) and updates its positions according to equation (2).

$$V_i(t) = aV_i(t-1) + b_1 r_1 \left(X_{p_i} - X_i(t)\right) + b_2 r_2 \left(\left(R_1 X_g + R_2 X_{2g}\right) - X_i(t)\right) \qquad (5)$$

In the MPSO algorithm, particle updates its velocity according to equation (6) instead of equation (5) where constartion coefficient χ is presented.

$$V_i(t) = \chi\left(V_i(t-1) + b_1 r_1 \left(X_{p_i} - X_i(t)\right) + b_2 r_2 \left(\left(R_1 X_g + R_2 X_{2g}\right) - X_i(t)\right)\right) \qquad (6)$$

In order to study the effects of constriction coefficiant χ and the parameters R1 and R2 in (6) on the performance of the MPSO algorithm, two variants were used in the experiments denoted as MPSO1, and MPSO2.

In the MPSO1 version, the particle updates its velocity according to equation 6 with equal random weight combination between the global best position (Xg) and the previous global best position (X2g). i.e. R1 = R2 =R.

In the MPSO2 version, the particle updates its velocity according to equation 6 with random weight combination between the global best position (Xg) and the previous global best position (X2g). i.e. R1 and R2 are two different random variables.

4. Test Functions

In order to know how competitive the MPSO algorithm is and the effects of the constriction coefficient χ and combination weights R1 and R2 , two versions of the MPSO algorithm (MPSO1 and MPSO2) will be compared against two counterparts' versions of the CPSO algorithm (CPSO1 and CPSO2) [25] and the SPSO algorithm [21]. Five benchmarking functions were selected to investigate the performance of the above-mentioned algorithms' versions. The considered test functions were used in [6,7, 21]. The functions, the number of dimensions (D), the admissible range of the variable (x), and the goal values are summarized in Table 1.

Table 1. Test functions [21]

Name	Formula	Dim D	Range [xmin, xmax]	Goal for F
Sphere	$F_0(\vec{x}) = \sum_{i=1}^{D} x_i^2$	30	[-100, 100]D	0.01
Rosenbrock	$F_1(\vec{x}) = \sum_{i=1}^{D-1} \left(100\left(x_{i+1} - x_i^2\right)^2 + (x_i - 1)^2\right)$	30	[-30, 30]D	100
Rastrigin	$F_2(\vec{x}) = \sum_{i=1}^{D} \left(x_i^2 - 10\cos(2\pi x_i) + 10\right)$	30	[-5.12, 5.12]D	100
Griewank	$F_3(\vec{x}) = \frac{1}{4000}\sum_{i=1}^{D} x_i^2 - \prod_{i=1}^{D} \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$	30	[-600, 600]D	0.1
Schaffer's	$F_6(\vec{x}) = 0.5 - \frac{\left(\sin\sqrt{x_1^2 + x_2^2}\right)^2 - 0.5}{\left(1 + 0.001\left(x_1^2 + x_2^2\right)\right)^2}$	2	[-100, 100]2	10-5

The optimal solution for all functions is equal to 0

5. Test Conditions

Two parameter sets (Eqs. (1, 2, 5 and 6) $\chi = a$ and $b = b1 = b2$ were selected to be used in the test based on the suggestions in other literature where these values have been found, empirically, to provide good performance [7, 9, 10] and used in testing the SPSO by I.C. Trelea [21].

Parameter set 1 ($a = 0.6$ and $b = 1.7$) was selected by the author [7] in the algorithm convergence domain after a large number of simulation experiments.

Parameter set 2 ($a = 0.729$ and $b = 1.494$) was recommended by Clerc [18] and also tested in [7] giving the best results published so far known to the author. All elements of c and d were set to 1 as used in [21].

A more detailed study of convergence characteristics for different values of these parameters exists in [19].

6. Optimization Test Experiments

In order to test the performance of the three versions of CPSO and SPSO algorithms two sets of experiments were used with the above mentioned test conditions and the two parameter sets.

In the first set of experiments, the maximum iteration number was fixed to 20000. Each optimization experiment was run 20 times with random initial values of x and v in the range [xmin, xmax] indicated in Table 1. Population sizes of N = 15, 30 and 60 particles were tested. The number of iterations required to reach the goal was recorded. Average number, median, minimum, maximum, and success rate of required iterations, and expected number of function evaluations , for each test function are calculated and presented in Tables 2-6.

In the second set of experiments, Each optimization experiment was run 20 times for 1000 iterations with population sizes of N = 30 particles. Averages of the best values in each iteration were calculated and plotted in figures 1- 5.

During the optimization process the particles were allowed to "fly" outside the region defined by [xmin, xmax] and also the velocity was not restricted.

7. The Experimental Results

This section compares the various algorithms to determine their relative rankings using both robustness and convergence speed as criteria. A "robust" algorithm is one that manages to reach the goal consistently (during all runs) in the performed experiments [20]. Tables 2–6 present the following information: Average number, median, minimum, maximum number of iterations required to reach a function value below the goal. Also, success rate of required iterations, and expected number of function evaluations. The "success rate" column lists the number of runs (out of 20) that managed to attain a function value below the goal in less than 2000 iterations, while the "Ex. # of Fn. Evaluation" column presents the expected number of function evaluations needed on average to reach the goal, calculated only for the succeeded runs using the following formula.

Ex. # of Fn. Evaluation = (Average number of iterations) × (number of particls in the swarm) / (success rate)

Table 2 shows that while SPSO algorithm failed to reached the goal during some runs for solving the Sphere function (F0) with parameter set1 and 15 particles, all the algorithms reached the goal during all the runs with both parameter sets.

Table 2. Average number, median, minimum, maximum, and success rate of required iterations, and expected number of function evaluations , for the test function F_0

Fun.	# of Part. N	Algorithm	Number of algorithm iterations to achieve the goal								Success Rate		Ex. # of Fn. Evaluation	
			Average		Median		Minimum		Maximum					
			Set1	Set2	Set1	Set2	Set1	Set2	Set1	Set2	Set1	Set2	Set1	Set2
F_0	15	MPSO1	**56**	**94**	**56**	**96**	**47**	**68**	**63**	**107**	1	1	**844**	**1416**
		MPSO2	107	173	107	173	98	159	117	196	1	1	1605	2595
		CPSO1	125	168	126	173	69	102	161	216	1	1	1874	2516
		CPSO2	320	471	316	457	249	358	373	613	1	1	4804	7065
		SPSO	769	764	722	731	523	586	1377	1275	0.40	1	28838	11460
	30	MPSO1	**53**	**88**	**53**	**89**	**48**	**71**	**58**	**106**	1	1	**1575**	**2650**
		MPSO2	90	146	91	146	84	134	96	158	1	1	2709	4371
		CPSO1	131	180	127	179	103	137	161	221	1	1	3917	5396
		CPSO2	300	404	296	396	259	296	352	528	1	1	9006	12126
		SPSO	344	395	333	395	266	330	457	572	1	1	10320	11850
	60	MPSO1	**49**	**85**	**50**	**85**	**46**	**75**	**52**	**92**	1	1	**2961**	**5080**
		MPSO2	80	129	80	129	76	120	87	145	1	1	4814	7761
		CPSO1	118	157	120	159	90	123	132	185	1	1	7083	9423
		CPSO2	264	346	254	346	221	301	302	389	1	1	15816	20760
		SPSO	252	314	252	313	214	269	309	368	1	1	15120	18840

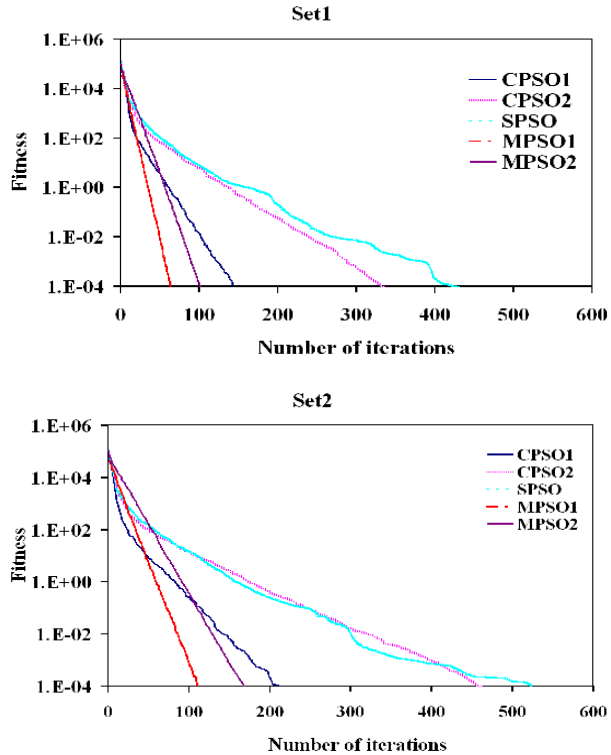


Fig. 1. Average best fitness curves for sphere function (*Fo*)

That means the number of particles affects the convergence of the SPSO algorithm for such problems type while such effect is not obviously clear with the other algorithms for such kind of problems.

Also, as illustrated in fig. 1, all the algorithms have reached a function value below the goal of the Sphere function with both parameter sets and they different are candidate to reach the optimal solution but in number of iterations. The two versions of MPSO algorithm are the candidates to reach the optimal solution in a few number of iteration than its counterpart's versions of the CPSO algorithm. That means the constriction coefficient affects the convergence speed of the versions of MPSO to be faster than its counterpart versions of CPSO algorithm.

Table 3. illustrates the same problem of SPSO algorithm with F1. It can't reach the goal of sphere functions in all runs with parameter set1.

Table 3. Average number, median, minimum, maximum, and success rate of required iterations, and expected number of function evaluations , for the test function F_1

Fun.	**# of Part. N**	**Algorithm**	**Number of algorithm iterations to achieve the goal**								**Success Rate**		**Ex. # of Fn. Evaluation**	
			Average		**Median**		**Minimum**		**Maximum**					
			Set 1	**Set 2**	**Set 1**	**Set 2**	**Set 1**	**Set 2**	**Set 1**	**Set 2**	**Set 1**	**Set 2**	**Set1**	**Set2**
F_1	15	MPSO1	**34**	**59**	**34**	**59**	**29**	**48**	**39**	**68**	1	1	**517**	**888**
		MPSO2	79	124	78	119	65	96	122	166	1	1	1179	1853
		CPSO1	88	112	89	105	62	63	119	160	1	1	1318	1673
		CPSO2	288	484	276	463	199	281	516	760	1	1	4324	7260
		SPSO	531	1430	523	729	413	452	695	9476	0.50	1	15930	21450
	30	MPSO1	**32**	**55**	**32**	**55**	**29**	**42**	**34**	**64**	1	1	**953**	**1645**
		MPSO2	65	109	64	108	57	89	77	134	1	1	1950	3278
		CPSO1	84	105	81	105	67	74	110	127	1	1	2520	3149
		CPSO2	257	392	258	395	204	253	336	771	1	1	7701	11771
		SPSO	614	900	383	408	239	298	3718	4642	1	1	18420	27000
	60	MPSO1	**30**	**53**	**30**	**52**	**26**	**46**	**33**	**62**	1	1	**1789**	**3154**
		MPSO2	57	97	54	96	47	81	73	126	1	1	3423	5820
		CPSO1	77	102	76	101	64	80	89	124	1	1	4623	6138
		CPSO2	227	309	218	298	174	234	468	430	1	1	13590	18561
		SPSO	337	611	284	311	189	219	916	4450	1	1	20220	36660

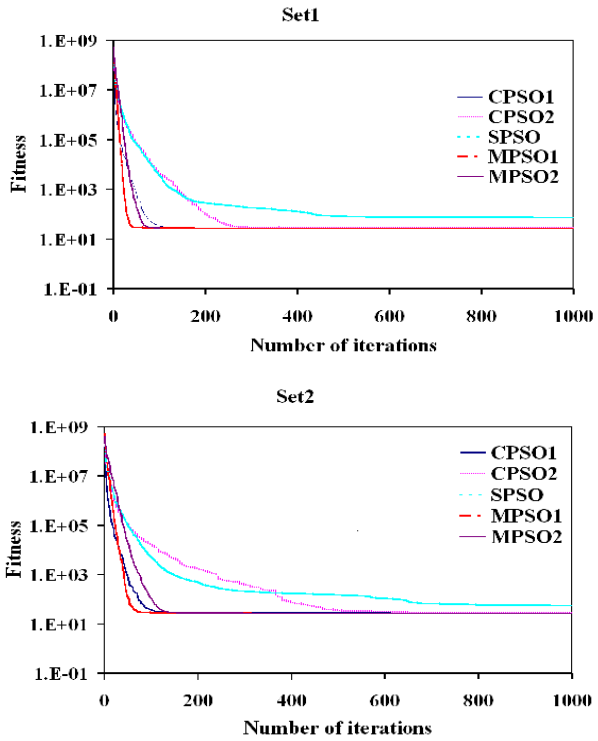


Fig. 2. Average best fitness curves for Rosenbrock function (F_1)

While SPSO with parameter set1 and 15 has some difficulties in reaching the goal of Rosenbrock function (F1), none of the other algorithms had such difficulties with both parameter sets for solving the same function. Also, the expected number of function evaluation needed for the two versions of MPSO algorithm is less than expected number of function evaluation needed for the other algorithms. That means the constriction coefficient accelerates the speed of MPSO algorithm to be faster than the other algorithms tested in this paper in searching for the solution of such kind of problems (see Table 3).

Also, fig. 2 illustrates that the MPSO1 , MPSO2, CPSO1 and CPSO2 reached values below the function goal in different number of iterations and below that value reached by SPSO algorithm. That means solution quality of all algorithms except the SPSO is the same for such kind of problems. But, the versions of MPSO algorithm are reached values below the goal faster than its counterparts of CPSO algorithm. That means, the constriction coefficient accelerates the speed and maintains the solution quality of MPSO the same time.

Table 4 shows that the MPSO1, MPSO2 , and CPSO1 algorithms perform admirably on the Rastrigin function (F2), but the CPSO2 and SPSO algorithms are less robust on the same function for such type of problems where they had some difficulties in reaching the goal in some runs.

Table 4 and fig. 3 illustrate that the MPSO1 and CPSO1 algorithms are doing very well on this problem, delivering the best overall performance for the Rastrigin function (F2), where they reached the goal in approximately 35 and 57 iterations respectively and they are candidates to reach the optimal solution in approximately less than 200 and 400 iterations respectively with both parameter sets. Although the good performance of CPSO1 algorithm the MPSO1 is a superior of it. This superiority because of the effect of constriction coefficient on the performance of MPSO1 algorithm.

Concerning the effects of the parameter sets on the algorithms performance on Rastrigin function (F2), there is no significant difference between the algorithms performance with both the parameter sets except the performance of MPSO2 algorithm with parameter set1 is much better than its performance with parameter set2 as shown in fig. 3.

Griewank's function (F3) proves to be hard to solve for all the algorithms. All the algorithms consistently reached the goal in all runs except SPSO and CPSO2 algorithms had some difficulties in reaching the goal for

Griewank's function with both parameter sets, as can be seen in Table 5. Also, we can realize that the performance of the two versions of MPSO algorithm is best than its counterparts of CPSO algorithm as seen in Table 5.

Table 4. Average number, median, minimum, maximum, and success rate of required iterations, and expected number of function evaluations , for the test function F_2

Fun.	# of Part. N	Algorithm	Number of algorithm iterations to achieve the goal								Success Rate		Ex. # of Fn. Evaluation	
			Average		Median		Minimum		Maximum					
			Set 1	Set 2	Set 1	Set 2	Set 1	Set 2	Set 1	Set2	Set1	Set 2	Set1	Set2
F_2	15	MPSO1	**23**	**38**	**22**	**37**	**16**	**25**	**30**	**55**	1	1	**344**	**576**
		MPSO2	157	177	122	166	66	52	438	351	1	1	2361	2658
		CPSO1	57	81	51	71	23	47	139	147	1	1	857	1217
		CPSO2	264	548	220	517	157	145	698	1357	0.90	1	4402	8214
		SPSO	172	299	147	292	102	123	208	299	0.35	0.8	7371	5606
	30	MPSO1	**22**	**37**	**21**	**37**	**17**	**24**	**30**	**51**	1	1	**667**	**1111**
		MPSO2	102	166	95	149	47	95	174	316	1	1	3072	4988
		CPSO1	48	68	44	63	32	37	83	145	1	1	1451	2054
		CPSO2	281	443	233	348	121	159	857	955	1	1	8862	13283
		SPSO	140	182	128	174	104	123	208	299	0.90	0.95	4667	5747
	60	MPSO1	**21**	**36**	**21**	**35**	**16**	**24**	**30**	**52**	1	1	**1271**	**2180**
		MPSO2	120	127	88	119	35	75	627	223	1	1	7203	7595
		CPSO1	60	69	55	62	39	34	132	127	1	1	3603	4152
		CPSO2	265	517	209	414	144	215	633	1526	1	1	15906	31026
		SPSO	122	166	116	164	84	119	168	214	0.95	1	7705	9960

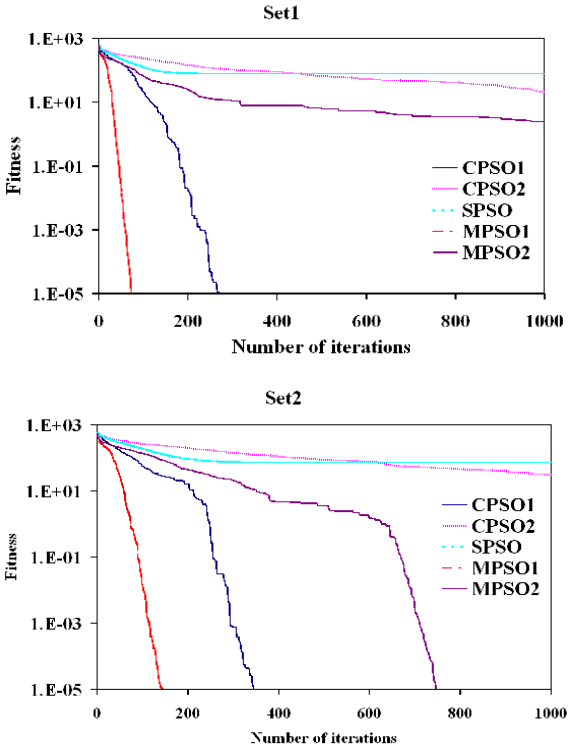


Fig. 3. Average best fitness curves for Rastrigin function F_2

Fig. 4 illustrates that both versions of the MPSO and CPSO1 algorithms with both parameter sets are candidates to reach the optimal solution while SPSO and CPSO2 did not reach the goal during some runs. MPSO1 with parameter set2 is candidate to reach the optimal solution while it is not with parameter set1. (see Fig 4.). Although the good performance of both versions MPSO in solving Griewank function its counterparts of CPSO algorithm are not. That appears the effect of the constriction coefficient on accelerating the speed of MPSO algorithm to be faster than the other algorithms tested in this paper in searching for the solution of such kind of problems.

Fig. 4 illustrates that the MPSO1 and CPSO1 algorithms with both parameter sets are candidates to reach the optimal solution while SPSO and CPSO did not reach the goal during some runs. CPSO1 algorithm with parameter set2 is candidate to reach the optimal while it is not with parameter set1. (see fig. 4).

Table 5. Average number, median, minimum, maximum, and success rate of required iterations, and expected number of function evaluations , for the test function F_3

Fun.	# of Part. N	Algorithm	Number of algorithm iterations to achieve the goal								Success Rate		Ex. # of Fn. Evaluation	
			Average		Median		Minimum		Maximum					
			Set1	Set2	Set1	Set2	Set1	Set2	Set1	Set2	Set1	Set2	Set1	Set2
F_3	15	MPSO1	**53**	**89**	**51**	**86**	**47**	**61**	**71**	**120**	1	1	**790**	**1332**
		MPSO2	129	204	120	192	84	142	208	411	1	1	1930	3059
		CPSO1	132	215	131	205	84	89	191	470	1	1	1982	3218
		CPSO2	364	581	314	522	254	329	731	1403	1	1	5465	8714
		SPSO	689	755	580	608	443	470	1589	1755	0.35	0.6	29529	18875
	30	MPSO1	**47**	**83**	**47**	**85**	**42**	**63**	**54**	**108**	1	1	**1417**	**2479**

		MPSO2	117	165	114	144	77	129	176	317	1	1	3496	4936
		CPSO1	131	168	130	165	79	57	229	293	1	1	3929	5046
		CPSO2	342	440	293	431	227	330	720	667	1	0.95	10257	13910
		SPSO	313	365	304	361	257	319	401	455	0.90	0.90	10433	12167
	60	MPSO1	**45**	**76**	**45**	**76**	**40**	**65**	**50**	**100**	1	1	**2700**	**4554**
		MPSO2	90	140	77	118	67	110	174	221	1	1	5380	8392
		CPSO1	113	152	109	148	72	115	199	198	1	1	6753	9117
		CPSO2	325	421	268	370	188	301	853	718	1	1	19494	25245
		SPSO	226	287	224	280	202	266	250	238	0.95	1	14274	17220

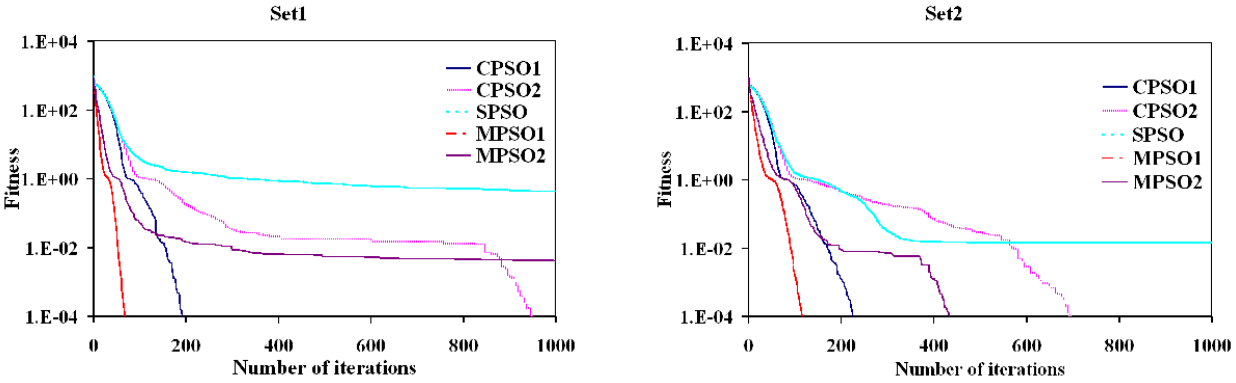


Fig. 4. Average best fitness curves for Griewank function F_3

Concerning the Schaffer's function (F6): Table 6 illustrates that while all versions of the MPOS and CPSO algorithms reached the goal in all runs with both parameter sets, SPSO algorithm had some difficulties in reaching the goal.

Fig. 5 illustrates that: using the average of 30 runs for 1000 iterations of each algorithm, all the algorithms reached a solution value (10-6) below the goal in different number of iterations and be candidate to reach the goal accept the MPSO2 and SPSO algorithms. MPSO2 with parameter set1 stacked at value 10-3 greater than the goal while SPSO algorithm with both parameter sets stacked at value 10-3 greater than the goal. MPSO1 reached a value below the goal in 170 iterations on average while CPSO1 and CPSO2 needed more than 200 iterations.

Table 6. Average number, median, minimum, maximum, and success rate of required iterations, and expected number of function evaluations , for the test function F6

Fun.	# of Part. N	Algorithm	Number of algorithm iterations to achieve the goal								Success Rate		Ex. # of Fn. Evaluation	
			Average		Median		Minimum		Maximum					
			Set 1	Set2	Set 1	Set 2	Set 1	Set 2	Set1	Set2	Set1	Set2	Set1	Set2
F6	15	MPSO1	**148**	**142**	**129**	**110**	**40**	**49**	**344**	**669**	1	1	**2216**	**2132**
		MPSO2	242	254	120	197	63	54	961	773	1	1	3635	3805
		CPSO1	198	232	183	201	57	63	473	498	1	1	2971	3476
		CPSO2	231	254	207	225	72	99	522	631	1	1	3461	3813
		SPSO	583	1203	138	126	63	91	3706	5853	0.45	0.4	19433	45113
	30	MPSO1	**93**	**128**	**72**	**115**	**45**	**60**	**210**	**250**	1	1	**2791**	**3826**
		MPSO2	159	154	78	118	34	53	639	383	1	1	4784	4617
		CPSO1	122	148	98	134	55	57	307	397	1	1	3668	4434
		CPSO2	144	163	126	163	57	87	439	298	1	1	4322	4893
		SPSO	161	350	120	157	74	102	595	1264	0.75	0.60	6440	17500
	60	MPSO1	**60**	**83**	**57**	**77**	**35**	**54**	**108**	**181**	1	1	**3616**	**4954**
		MPSO2	89	104	76	94	35	65	195	190	1	1	5345	6227
		CPSO1	93	112	76	110	42	73	206	193	1	1	5559	6708
		CPSO2	112	147	94	128	52	53	232	422	1	1	6726	8841
		SPSO	169	319	91	119	40	83	854	2361	0.90	0.95	11267	20147

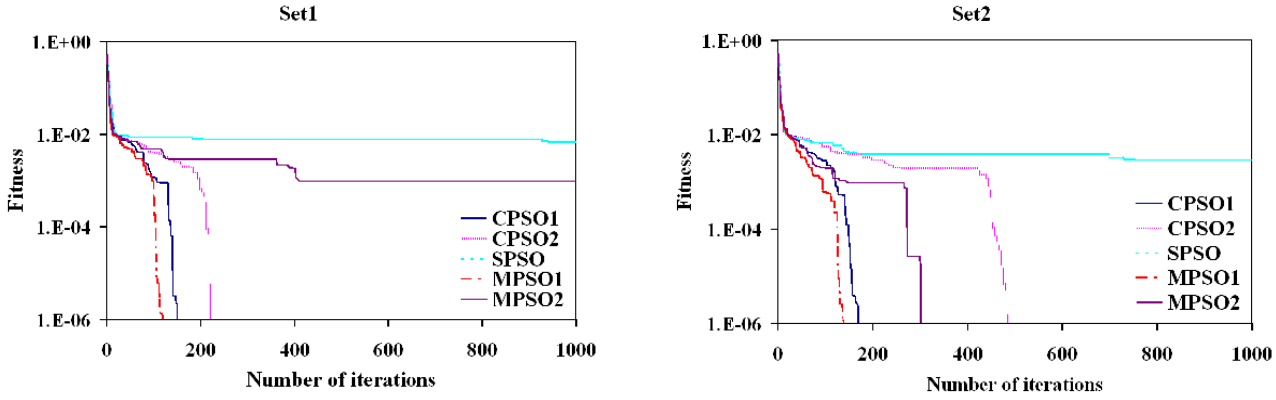


Fig. 5. Average best fitness curves for Schaffer's function F_6

8. Discussion

Overall, as far as robustness is concerned, the MPSO algorithm appears to be the winner, since its two versions (MPSO1 and MPSO2) achieved perfect scores in all the test cases as represented in boldface (see Tables 2-6). So, we can conclued that the constriction coefficient accelerates the convergence and the solution quality of MPSO algorithm. The CPSO1, algorithm is less robust, followed closely by the CPSO2 algorithm. The standard SPSO algorithm was fairly unreliable on this set of problems.

As a result, in many cases the SPSO must be executed several times to ensure good results, whereas one run of MPSO1, MPSO2, CPSO1 and the CPSO2 usually is sufficient.

Regarding convergence speed, MPSO1 is always the fastest followed by MPSO2, and CPSO1 whereas the CPSO2 or SPSO are always the slowest. MPSO1 has very fast convergence (4-6 times faster than SPSO). This may be of practical relevance for some real-world problems where the evaluation is computationally expensive and the search space is relatively simple and of low dimensionality. Overall, MPSO1 and MPSO2 are clearly the best performing algorithms in this study. It finds the lowest fitness value for all the tested problems, see fig. 1-5.

Regarding the parameter sets: SPSO is more sensitive to parameter changes than the other algorithms. When changing the problem, one probably needs to change parameters as well to sustain optimal performance. In general, the performance of all algorithms are best with parameter set1 than the performance with parameter set2, while all of them need less number of iterations to reach the specified goal with parameter set1. That means parameter set2 slows the algorithms and don't make a bridging phenomena while parameter set1 accelerates the algorithms but some while makes a bridging phenomena.

Looking at the number of function evaluations, the MPSO1 was in the lead, followed by the MPSO2 algorithm, as shown in boldface (see Tables 2-6). That means the constriction coefficient speed the algorithm convergence.

Considering, the above-mentioned points the two versions of MPSO algorithm had no difficulty in reaching the functions' goals and all its solutions are below their corresponding goals more than the other algorithms (see fig. 1-5). So, we can conclude that Modified algorithm is more superior to the other algorithms and can be considered as a viable alternative algorithm for solving optimization problems.

9. Conclusion

This paper has proposed a new variation of the particle swarm optimization algorithm called a modified PSO, introducing a constriction coefficient into the velocity and update equation of the combined particle swarm algorithm where a modified term has introduced into the velocity component and update equation. The implementation of this idea is simple, based on storing the previous positions. The MPSO algorithm outperforms the CPSO algorithm [25] and SPSO algorithm [21] on many benchmark functions, being less susceptible to premature convergence, and less likely to be stuck in local optima.

In this study, the MPSO1 and MPSO2 have shown its worth on real-world problems, and it outperformed CPSO and SPSO on all the numerical benchmark problems as well. Among the tested algorithms, the MPSO1 can rightfully be regarded as an excellent first choice, when faced with an optimization problem to solve.

To conclude, the constriction coefficient in MPSO algorithm accelerates the convergence to a stable point in comparison to the other algorithms tested in this paper. The algorithm is simple, robust, converges fast, and finds the optimum in every run. In addition, it has few parameters to set, and the same settings can be used for many different problems.

Future work includes further experimentation with parameters of MPSO, testing the algorithm on other benchmark problems, and evaluating its performance relative to evolutionary algorithms. Investigate the affect of different values of the constriction coefficient on the MPSO algorithm.

10. References

[1] R. Eberhart and J. Kennedy, A modified optimizer using particle swarm theory, in Proc. 6th Int. Symp. Micro Machine and Human Science, Nagoya, Japan, (1995) 39–43.

[2] Y. Shi and R. Eberhart, A Combined Particle Swarm Optimizer. In: Proceedings of IEEE World Congress on Computational Intelligence, (1998) 69–73.

[3] V. Tandon, Closing The Gap Between CAD/CAM and Optimized CNC and Milling, Master thesis, Purdue School of Engineering and Technology, Indiana University Purdue University Indianapolis, 2000.

[4] Abido, M.A. Optimal Power Flow Using Particle Swarm Optimization. Electric Power and Energy Sys. 2002;24:563–71.

[5] Jiang Chuanwen, Etorre Bompard, A Hybrid Method Of Chaotic Particle Swarm Optimization And Linear Interior For Reactive Power Optimization, Mathematics and Computers in Simulation 68 (2005) 57–65.

[6] N. Shigenori, G. Takamu, Y. Toshiku, F. Yoshikazu, A Hybrid Particle Swarm Optimization For Distribution State Estimation, IEEE Transactions on Power Systems 18 (2003) 60– 68.

[7] Amgad A. EL-Dib, Hosam M. Youssef, M.M. EL-Metwally, Z. Osman, Maximum loadability of power systems using hybrid particle swarm optimization, Electric Power Systems Research 76 (2006) 485–492.

[8] Bo Liu, Ling Wang, Yi-Hui Jin, Fang Tang, De-Xian Huang, Directing orbits of chaotic systems by particle swarm optimization, Chaos, Solitons and Fractals 29 (2006) 454–461.

[9] M. Tasgetiren, Yun-Chia Liang, Mehmet Sevkli, Gunes Gencyilmaz , A particle swarm optimization algorithm for makespan and total flowtime minimization in the permutation flowshop sequencing problem, European Journal of Operational Research, (2006) in process.

[10] M. Clerc and J. Kennedy, The Particle Swarm: Explosion, Stability, And Convergence In A Multi-Dimensional

Complex Space, IEEE Trans. Evol. Comput. 6,(2002) 58–73.
[11] A. Carlisle and G. Dozier, Adapting Particle Swarm Optimization to Dynamic Environments, Proceedings of International Conference on Artificial Intelligence, Las Vegas, Nevada, USA, (2000) 429-434.
[12] S. Tsumoto et al. (Eds.), A Guaranteed Global Convergence Particle Swarm Optimizer, RSCTC, (2004) 762–767. Springer-Verlag Berlin 2004.
[13] Bo Liu, Ling Wang, Yi-Hui Jin, Fang Tang, De-Xian Huang, Improved particle swarm optimization combined with chaos, Chaos, Solitons and Fractals 25 (2005) 1261–1271
[14] M. Lovbjerg and T. Krink, Extending Particle Swarm Optimizers with Self-Organized Criticality, Proceedings of Fourth Congress on Evolutionary Computation, (2002) 1588-1593.
[15] Xiao-Feng Xie, Wen-Jun Zhang, Zhi-Lian Yang, Hybrid Particle Swarm Optimizer with Mass Extinction, International Conf. on Communication, Circuits and Systems (ICCCAS), Chengdu, China, 2002.
[16] Xiao-Feng Xie, Wen-Jun Zhang and Zhi-Lian Yang, A Dissipative Particle Swarm Optimization, IEEE Congress on Evolutionary Computation, Honolulu, Hawaii, USA, 2002.
[17] F. van den Bergh, A.P. Engelbrecht, A study of particle swarm optimization particle trajectories, Information Sciences 176 (2006) 937–971.
[18] M. Clerc, The Swarm And The Queen: Towards A Deterministic And Adaptive Particle Swarm Optimization, in: Proc. ICEC, Washington, DC, (1999) 1951–1957.
[19] F. van den Bergh, "An Analysis Of Particle Swarm Optimizers," Ph.D. dissertation, Dept. Comput. Sci., Univ. Pretoria, Pretoria, South Africa, 2002.
[20] Frans van den Bergh and Andries P. Engelbrecht, A Cooperative Algorithm To Particle Swarm Optimization, IEEE Transactions on Evolutionary Computation, 8(3), (2004) 225-239.
[21] Ioan Cristian Trelea, The Particle Swarm Optimization Algorithm: Convergence Analysis And Parameter Selection, Information Processing Letters 85 (2003) 317–325.
[22] Eberhart, R. C., and Kennedy, J. (1995). A New Optimizer Using Particles Swarm Theory, Proc. Sixth International Symposium on Micro Machine and Human Science (Nagoya, Japan), IEEE Service Center, Piscataway, NJ, 39-43.
[23] Kennedy, J., and Eberhart, R. C. (1995). Particle Swarm Optimization, Proc. IEEE International Conference on Neural Networks (Perth, Australia), IEEE Service Center, Piscataway, NJ, IV: 1942-1948.
[24] Kennedy, J. (1997), The Particle Swarm: Social Adaptation of Knowledge, Proc. IEEE International Conference on Evolutionary Computation (Indianapolis, Indiana), IEEE Service Center, Piscataway, NJ, 303-308.
[25] M. M. El_Sherbiny, A Combined Particle Swarm Optimization Algorithm Based on the Previous Global Best and the Global Best Positions, International Journal of Computers and Information, 1,(2007) 13-26.
[26] M. Clerc, J. Kennedy, The Particle Swarm-Explosion, Stability, and Convergence In A Multidimensional Complex Space, IEEE Transactions on Evolutionary Computation 6 (1) (2002) 58–73.
[27] Kennedy, J.: Bare Bones Particle Swarms. In: Proceedings of the IEEE Swarm Intelligence Symposium 2003 (SIS 2003), Indianapolis, Indiana (2003) 80–87.

The ARIMA versus Artificial Neural Network Modeling

Motaz Khorshid
Faculty of Computer & Information. Cairo University, Egypt

Assem Tharwat,
Faculty of Computer & Information. Cairo University, Egypt

Amer Bader
Faculty of Computer & Information. Cairo University, Egypt

Ahmed Omran
Central Lab For Agricultural Expert Systems. Agriculture Research Center, Egypt

Abstract *Linear models almost reach their limitations with non-linearity in the data. This paper provides a new empirical evidence on the relative macroeconomic forecasting performance of linear and nonlinear models. The well-established and widely used univariate Auto-Regressive Integrated Moving Average (ARIMA) models are used as linear forecasting models whereas Artificial Neural Networks (ANN) are used as nonlinear forecasting models. The neural network paradigm that was selected for developing the proposed model is a Multi-layer Feedforward network based upon the Backpropagation training algorithm. ANN has been proven to be successful in handling nonlinear problem optimization and prediction. The forecasting models used to identify whether action is needed to alter the future, when such action should be taken by the decision maker in order to change the future of the bank or its environment to improve the bank's chance of achieving its targets. We applied the proposed model on a Financial Balance Sheet's data of a commercial bank in Egypt. The Results show that, the proposed model (which dependent on the ANN) is more accurate than the other models, which depend on the ARIMA model with accuracy between 8 % and 10.4 %.*

Keywords*: Economic Forecast, Neural Networks, ARIMA, Back-propagation, and Time Series Models.*

1. Introduction

This paper is mainly an empirical research effort intended to compare the forecasting performance of ANN and traditional time series methods, such as ARIMA. Recent work that has used ANN models for forecasting purposes in different applications such as Inflation forecasting, forecasting daily foreign exchange rates, etc [4, 5, 8, 9, 11, 15, 16, 20, 21, 25, 26]. In most of these experiments ANN has indicated a good forecasting accuracy.

Balance Sheet is a list that shows the financial condition of the firm at a particular date. It consists of two sides one side is called Assets and the other side is called Liabilities and Owner's Equity (the total liabilities)[17, 18,24].

This paper is concerning on the most important five items of our case study commercial bank's balance sheet, which are Cash Balances with C.B.E, banks & foreign correspond, Total Loans, Total Assets, Customer Deposits and Total Deposits. The data collected for those items was represented by five different time series (TS1, TS2, TS3, TS4, TS5) to be analysis in this paper to predict the future values for these items and perform a comparative exercise and access the relative performance of the different forecasting methods.

Commercial relations with the five previous items can calculate the all values of other items in the Balance Sheet [17, 18].

Neurosolution, commercially available neural networks simulator, was used in the training of the neural networks models and MINITAB12 was used as statistical software.

This paper includes seven sections after introduction. The next two sections after the introduction briefly discuss methodology and mathematical formulation of ARIMA models and ANN. Section four includes an overview of the software that used for applying both ARIMA and ANN. ARIMA models and the ANN models designs are given in sections five and six. In section seven the comparison between the ARIMA and ANN forecasting models is discussed, and finally some conclusions and further points for research are in the last section.

2. ARIMA Models

ARIMA models were originally proposed by Box and Jenkins (1976), and it is considered today a quite popular tool in economic forecasting. The basic idea is that a stationary time series can be modeled as having both autoregressive (AR) and moving average (MA) components. AR models are based on the application of regression analysis to lagged values of the Yt series. The Autoregressive model of order p, AR (p) is:

$$Y_t = \phi_0 + \phi_1 Y_{t-1} + \phi_2 Y_{t-2} + ... + \phi_p Y_{t-p} + \varepsilon_t \quad (1)$$

In the context of Box-Jenkins Yt = the actual value of the series at time t, Yt-1 = the actual value of the series at time t-i, ϕi = the Autoregressive parameter for Yt-i , εt =the irregular fluctuation at time t, not correlated with past values of the Yt's [19, 22].

MA models are based on the past levels of the series, Yt may be influenced by the recent"stocks" (i.e., random errors) to the series that is, the current value of a series may be the best explained by looking at the most recent q error [19, 22]. The Moving Average model of Order q, MA (q) is:

$$Y_t = \theta_0 + \theta_1 \varepsilon_{t-1} + \theta_2 \varepsilon_{t-2} + ... + \theta_q \varepsilon_{t-q} + \varepsilon_t \quad (2)$$

θi= The Moving Average parameters for εt-i, εt-i = the error term at time t-i, εt = the error term at time t, and εt, εt-1, εt-2, …are uncorrelated with one another.

Mixing autoregressive (AR) and moving average (MA) terms in the same model called ARMA model .The forecasted values of Yt in ARMA model of order (p) and (q), ARMA (p, q) is given by:

$$Y_T = \theta_0 - \theta_1 \varepsilon_{t-1} - \theta_2 \varepsilon_{t-2} - .... - \theta_q \varepsilon_{t-q} + \varepsilon_t + \phi_0 + \phi_1 Y_{t-1} + \phi_2 Y_{t-2} + ... + \phi_P Y_{t-P} \quad (3)$$

εt, εt-1, εt-2, …are uncorrelated with one another [19, 22]. ARIMA model equations are the same ARMA equations except Yt is replaced by the differenced series Wt: Wt=Δd Yt = the Yt series differenced d times. The forecasted values of Yt in ARIMA model of order (p), (d) and (q), ARIMA (p d, q) is given by:

$$W_t = \theta_0 - \theta_1 \varepsilon_{t-1} - \theta_2 \varepsilon_{t-2} - .... - \theta_q \varepsilon_{t-q} + \varepsilon_t + \phi_0 + \phi_1 W_{t-1} + \phi_2 W_{t-2} + ... + \phi_P W_{t-P} \quad (4)$$

εt, εt-1, εt-2, …are uncorrelated with one another [19, 22].

Non-stationary integrated series can also be handled in the ARIMA framework, but it has to be reduced to be stationary.

Fig1 shows schematic representation of Box-Jenkins methodology for time series modeling that includes three phases that are Identification, Estimation, Diagnostic and Forecasting [22].

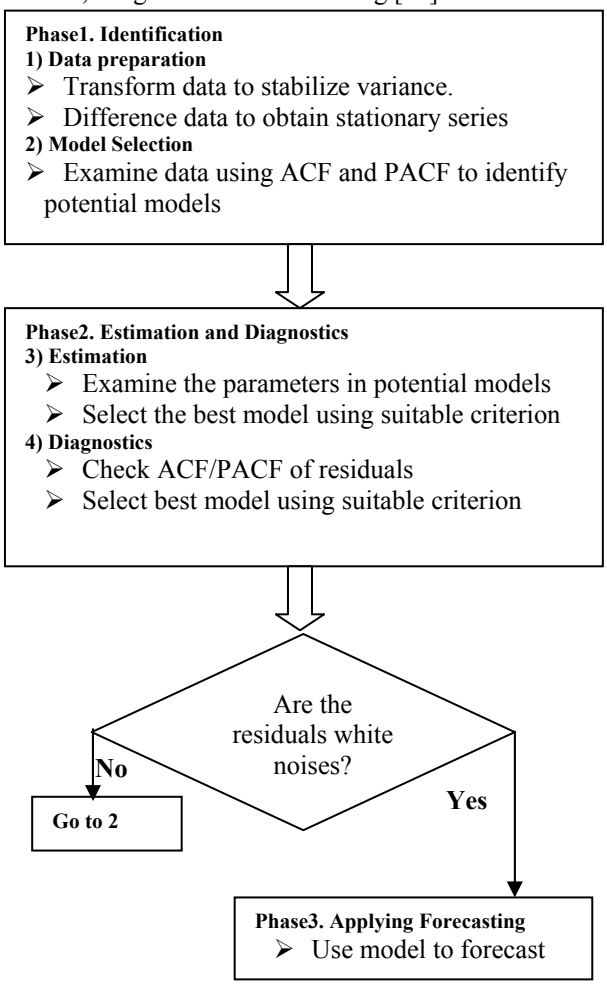


Figure 1 Schematic representation of Box-Jenkins methodology for time series modeling

Table 1 characteristic patterns in the ACF and PACF of ARIMA models

Model	Autocorrelation Function	Partial Autocorrelation Function
AR (p)	Die Out	Cuts off after lag p
MA (p)	Cuts off after lag q	Die Out
ARIMA	Die out after lag q-p	Die out after lag p-q

- Identification: the series is differenced, if necessary, to make it stationary. The data may be stationary in the viewpoint of trend through one or two differences but the time series may be non-stationary. To stabilize the variance, a nonlinear transformation such as a logarithmic or square root transformation is often performed. Then the sample ACF and PACF are calculated; the behavior of both the ACF and PACF determines the number of AR (p) and /or MA (q). Table1 shows the characteristic patterns in the ACF and PACF of ARIMA models [19, 22].
- Estimation and Diagnostics
- In Estimation, least squares estimates of the process parameters are generated. In diagnostic, checking the residuals from the estimated model should look like a random series; failing that, further analysis of the residuals leads to a re-specification of the model [19, 22].

Forecasting: - the fitted model, having first been "integrated "if necessary, is used to forecast the Yt's. (In practice, each of these stages requires the use of a scientific computer program) [19, 22].

3. Artificial Neural Networks

Neural Network is a branch of artificial intelligence. ANN act like a human brain, trying to recognize regularities and patterns in the data. They can learn from experience and generalize based on their previous knowledge. Neural networks are composed of highly interconnected processing elements (nodes) that work simultaneously to solve specific problems. In time series analysis ANN models were used as nonlinear function approximations. ANN takes in a set of inputs and produces one/a set of outputs according to some mapping rules predetermined in their structure [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 13].

This paper considers the most popular form of ANN, which called the feed-forward network. The selected feed-forward neural network model can fit the financial analysis problem because of their adaptively owing to their structure [20, 23, 24]. The existence of hidden layer and nonlinear activation function models the nonlinearity of the data. This important property of feed-forward neural network models enables modeling multi-attribute, nonlinear mapping for the financial analysis problem [3, 4, 5, 6, 7, 8, 9, 14, 15, 16, 20, 24, 25].

Fig.2 depicts such a network that consists of layers of nodes; the input layer and output layer represent the input and output variables of the model. Between the input and output layers there are one or more hidden layers that progressively transform the original input stimuli to final output and enables ANN to learn nonlinear relationships [6, 10, 13].

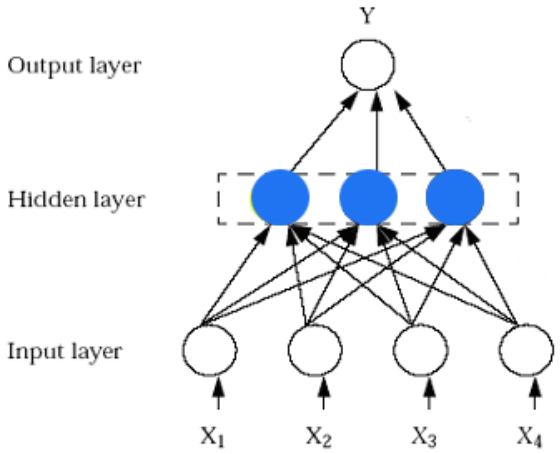


Figure 2 The feed forward neural network with a single hidden layer

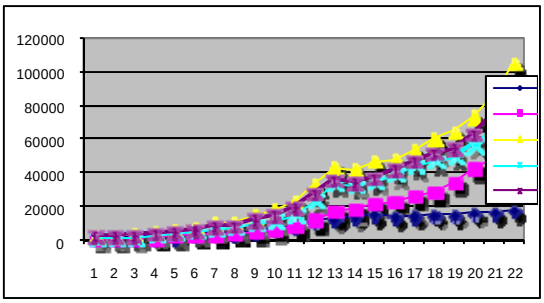


Figure 3 The plot of all time series

ANN has to be trained in order to be used to perform certain tasks like predicting a response corresponding to a new input pattern. The training procedure involves iteratively modifying the randomly initialized weights of the ANN to minimize some kind of error function usually the mean square error (MSE) [6, 10, 13].

Various standard optimization techniques such as the conjugate gradient and quasi-Newton methods exist for minimizing the error function, however, in application studies, the Back-propagation Algorithm developed by the neural network community is the most popular training algorithm used [6, 10,13, 20,24,26].

4. Implementation Tools

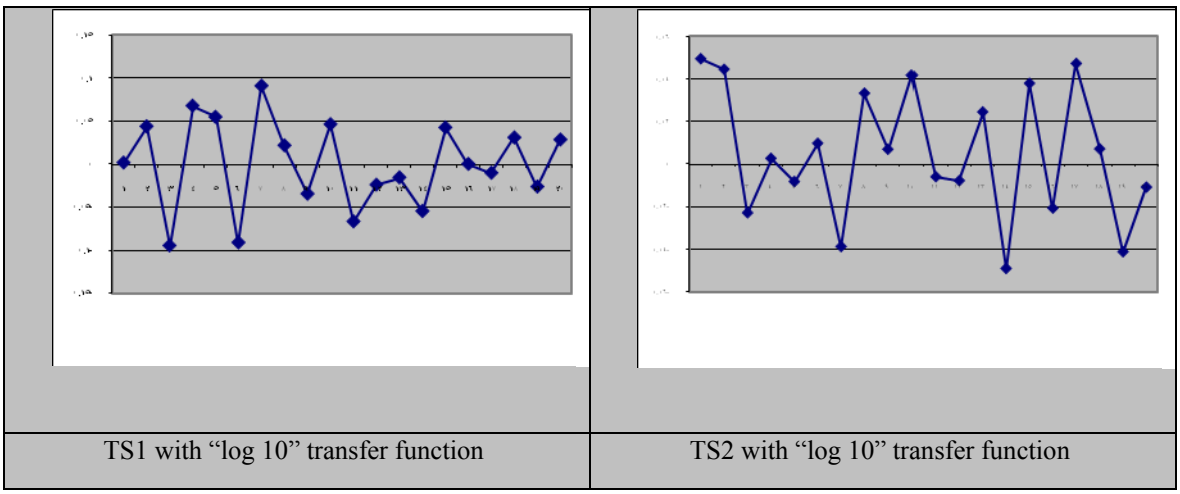
The Network simulator such as NeuroSoltion requires no programming skills and often come with special hardware to minimize the training time. The commercially available neural network simulator NeuroSoltion (NeuroDimension incorporated) was used for development of the proposed neural network application. Use of a shell program of this type is attractive for forecasting financial data environment. MINITAB is the statistical software that offers the methods that we need to implement ARIMA model.

5. Designing ARIMA Models

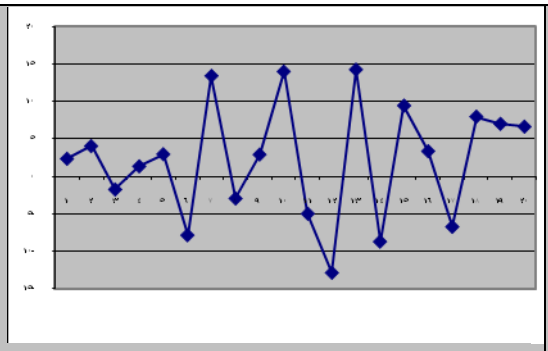
First, each time series should be plotting to discover whether it is stationary or not. Fig3 shows that there is a trend and no seasonality in the given time series. When we plot every time series after two differences the un-stabilization in variances was appeared. To stabilize variances in every time series transformation functions are used [19, 22].

We tried two types of transformation functions, logarithmic and square root, followed by taken two differences for each time series to be stationary. In Fig4 Every time series "looks" stationary, since the time plot of the series appears "similar" at different points along the time axis [12, 14, 19, 22].

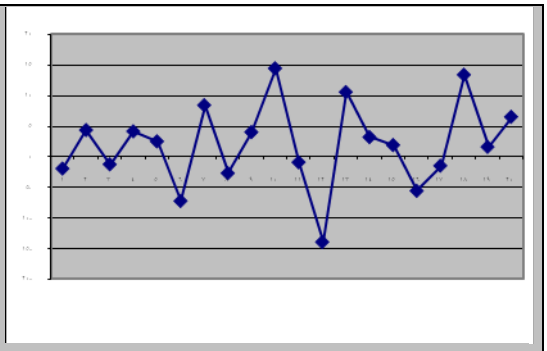
Autocorrelation and partial autocorrelation functions are used to identify an acceptable model for each time series; Fig5 shows the autocorrelation and partial autocorrelation functions for every time series. From the values of ACF & PACF we can deduce the acceptable models for each time series. Table2 states their models.



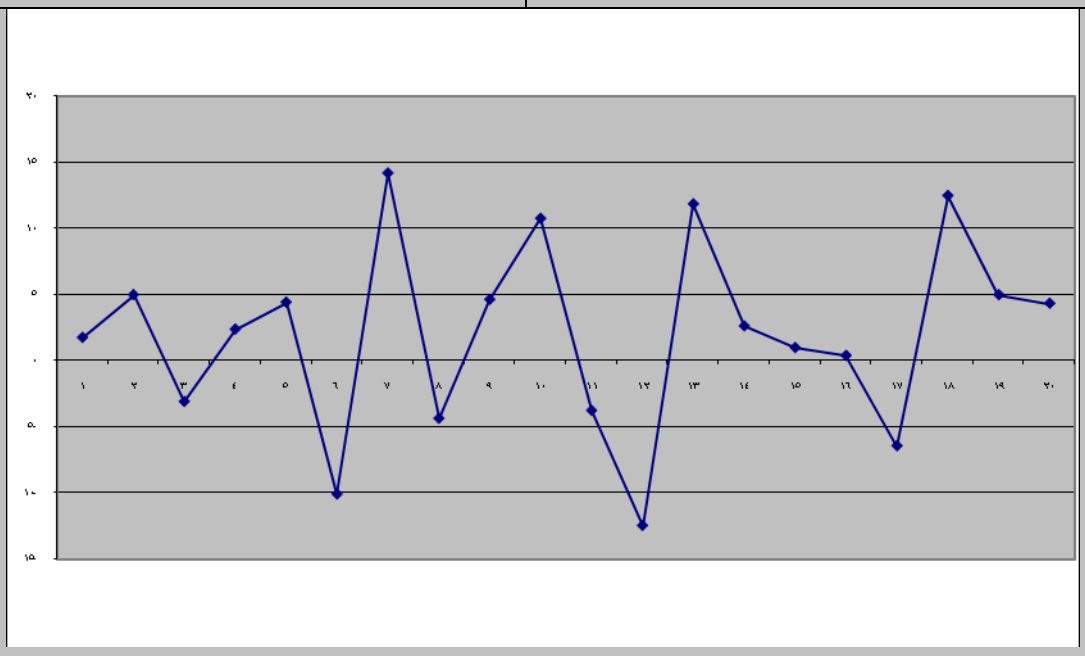
TS1 with “log 10” transfer function	TS2 with “log 10” transfer function



TS3 with “log 10” transfer function and two

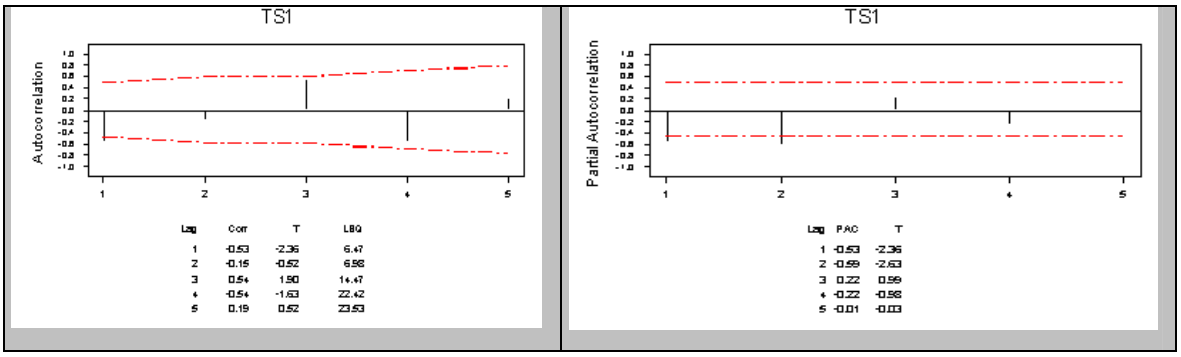


TS4 with “square root” transformation function



TS5 with “square root” transformation function and two differences

Fig4. Time series plots after be stationary



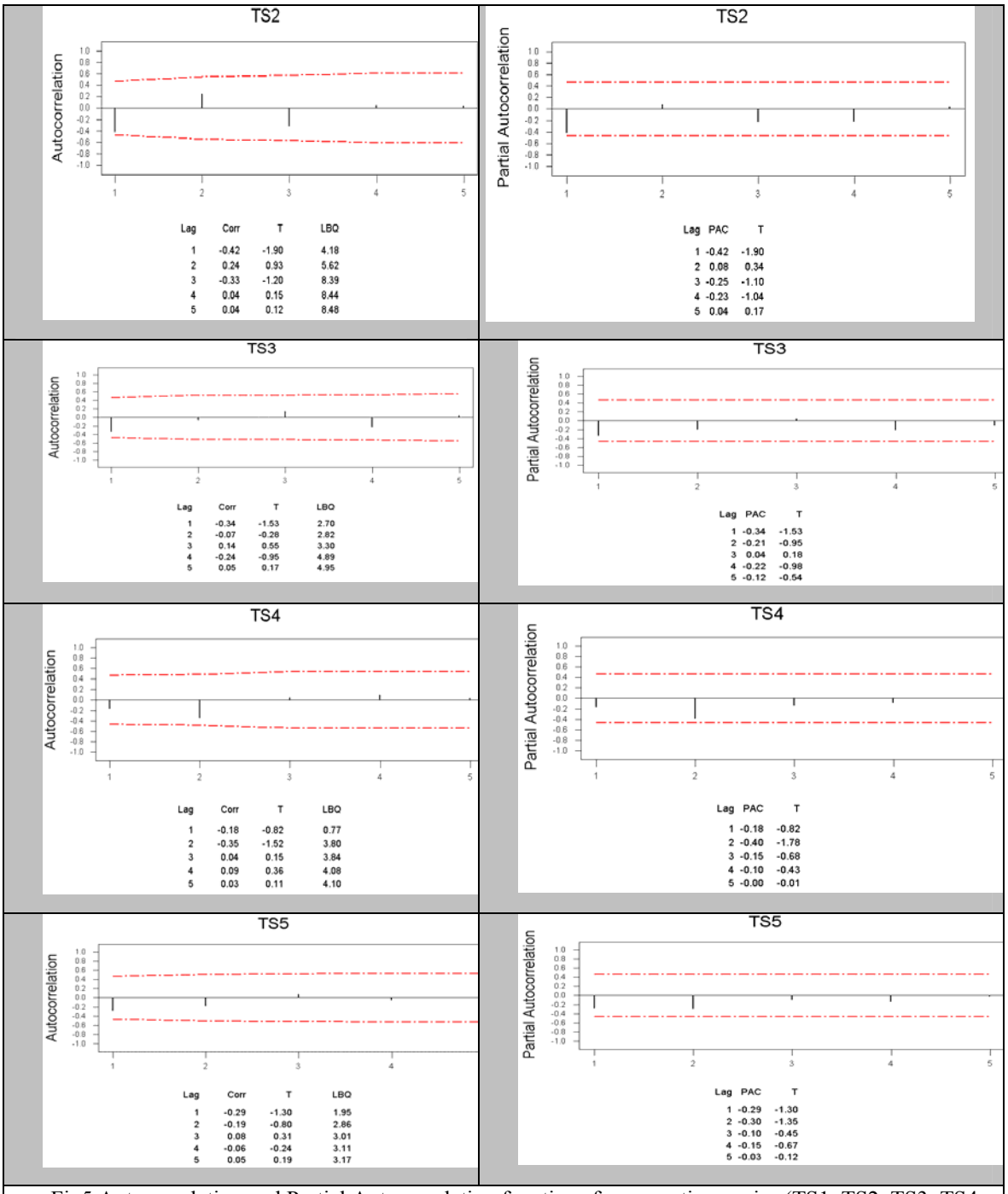


Fig5.Autocorrelation and Partial Autocorrelation functions for every time series (TS1, TS2, TS3, TS4, T5).

Time Series	ACF &PACF	Model
1	The autocorrelations die out in a damped sine-wave manner and that there are exactly **two** significant partial autocorrelation.	AR (2)
2	**One** non-zero autocorrelation at lag1 and that the partial autocorrelation decay exponential.	MA (1)

3	**One** non-zero autocorrelation at lag1 and that the partial autocorrelation decay exponential.	MA (1)
4	At the autocorrelation decay exponential after the interval [q-p] and in the partial autocorrelation decay exponential after the interval [p-q].	ARIMA (1, 2, 1)
5	A tremendous variety of patterns in the ACF and PACF.	ARIMA (1, 2, 1)

Table2. Analysis of ACF &PACF and the acceptable model for each time series

The MINITAB software is used to examine the parameters in potential models and select best models. Tables 3,4 summaries the results to select the best model.

Time Series	Model	Parameters	Corresponding Equation of the best model
1	AR (2)	ϕ_1= -0.4399, ϕ_2= -0.297 □□ ϕ_0 = Not significant.	$\Delta^2 \mathbf{Log}\ Y_t = \log [-0.4399 Y_{t-1} - 0.297 \square\square Y_{t-2}] + \varepsilon_t$
2	MA (1)	θ_1= 0.9518 θ_0= Not significant.	$\Delta^2 \mathbf{Log}\ Y_t = \log [-0.9518 \varepsilon_{t-1}] + \varepsilon_t$
3	MA (1)	θ_1= 0.9510, θ_0= 0.693	$\Delta^2 \mathbf{Log}\ Y_t = 0.693 + \log [-0.9510 \varepsilon_{t-1}] + \varepsilon_t$
4	ARIMA (1,2,1)	ϕ_1= -0.624, θ_1= 0.9214 Constant is not significant.	$W_t = \Delta^2 \sqrt{Y_t}$ $W_t = [-0.9214 \varepsilon_{t-1} - 0.624\ W_{t-1}] + \varepsilon_t$
5	ARIMA (1,2,1)	ϕ_1=-0.474, θ_1= 0.9118 Constant is not significant	$W_t = \Delta^2 \sqrt{Y_t}$ $W_t = [-0.9118\ \varepsilon_{t-1} - 0.474\ W_{t-1}] + \varepsilon_t$

Table3.The best models for TS1, TS2, TS3, TS4 and TS5

Where ε_t was generated from a standard normal distribution.

Diagnostic checking was applied and indicated that all the pervious models are good forecasting models. The ACF &PACF spicks were outside the limits, which suggesting the residual series were white noise. A portmanteaus test was applied to the residuals as an additional test, in this case the value Q* was not significant, that mean the residuals was considered white noise [19, 22].

6. Designing the Neural Networks

A feed-forward neural network model was employed to mimic the complex mapping function between the inputs and output. The Back-propagation learning algorithm is used to perform the training requirements. The determination of the network structure is a very difficult task as involves many variables including learning rate parameter, number of hidden layers, and the number of hidden units per hidden layer [1, 3, 6, 10, 13, 20, 23, 25].

The learning rate parameter plays a critical role in the convergence of the network to the true solution. For a given network and an infinitesimal learning rate, the weights that yield the minimum error can be found. However, they may not be found in a reasonable span of time. Use of a large learning rate proceeds much faster but may also produces oscillations between relatively poor solutions. However, high-dimensional spaces (with many weights) have relatively few local minima [6, 10, 13].

The number of the inputs variables (number of lags) in this work was between 2 and 4 inputs, the number of hidden layers was selected as one hidden layer and the number of neurodes per hidden layer depends on the number of nodes in the input layer. With too few connections, the network cannot learn much. The net result of poor parameter settings will be slow convergence and poor performance of the model [1, 2, 3, 4, 5]. Table5 describes the different neural network structures for the different time series.

Network parameters	**The Model Number**				
	M1	**M 2**	**M 3**	**M4**	**M 5**
1.No. Of input units	2	3	4	3	3
3.No.Of hidden units	3	7	9	7	7
2.No. Of middle layers	1				
4.No.Of output units	1				
5.Learning rate parameter	0.7				
6.Transfer Functions	Hyperbolic tangent (*tanh*) function				

Table5. Description of the ANN Structure.

The logistic function is the most popular activation function among researchers for the hidden layer. However, we use the hyperbolic tangent (tanh) function, as it has been used very successfully in forecasting experiments. It is also generally held that (tanh) gives rise to faster convergence of training algorithms than logistic functions. For the output layer, we follow the recommendation of who suggest the use of the linear function for time series prediction with continuous output [1, 2, 6, 10].

In this paper we use one of the most common forms of preprocessing which consists of rescaling the data in the range [-1, 1] so that they have similar values [6, 10, 13].

- Training of the Neural Network Models
  - The following general guidelines were considered during the training:
  - A fully connected neural network.
  - Pattern (training examples) was presented sequentially during each training session.
  - Updating network weights was performed after each training pattern was presented to the network.
  - The stopping criteria were set such that the total number of iterations does not exceed 50,000 epochs, or a 0.0001 root-mean-square (RMS).
  - Each train – and – test experiment is repeated three times with different random starting state to make sure that the solution obtained is not a local minimum [6, 10, 13].
- **The Results of the ANN Forecasting Models**

The final results of these experiments revealed that the network with between 3-9 hidden units had a better ability to decrease the system error rate (error rate due to training is between 3.5 and 4.0%) and provided a good prediction (average error rate due to test is between 8.3 to10.6 %). Figures from Fig (6-10) show the plots of the final results of the ANN forecasting models.

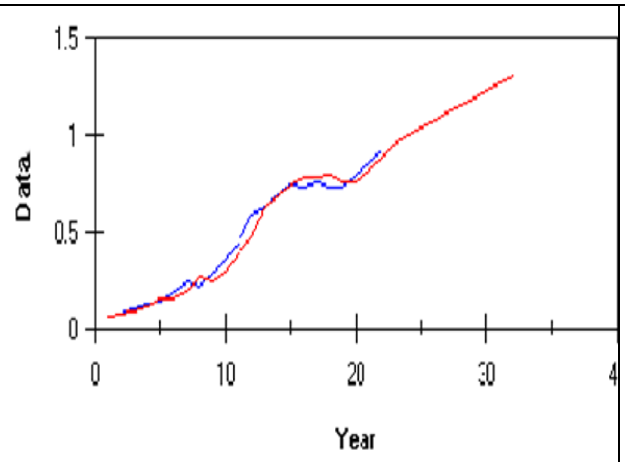


Fig6.Model1 for TS1

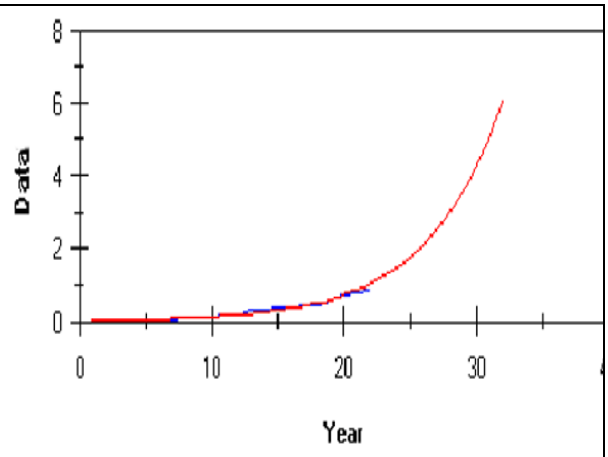


Fig7.Model2 for TS2

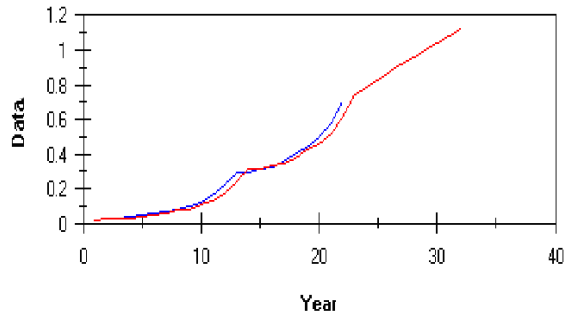


Fig8.Model3 for TS3

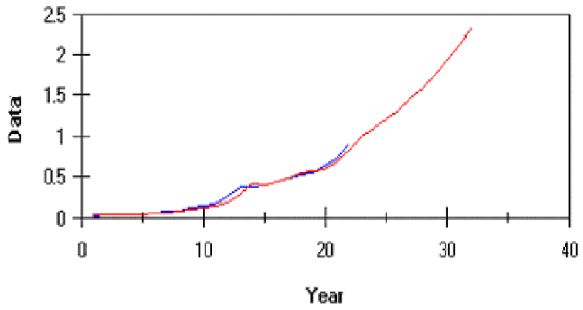


Fig9.Model4 for TS4

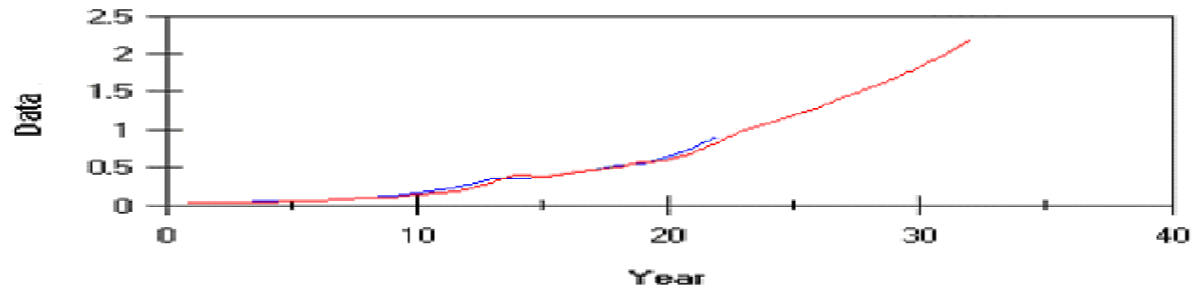


Fig10.Model5 for TS5

7. The Comparison between the Accuracy of the ARIMA and Neural Networks Forecasting Models

There are several measures of accuracy but each of them has its advantages and limitations. For this reason none of them has been accepted universally as the optimum measure of accuracy [10, 13, 19, 22]. In this study we shall use two popular types of performance measures, which are the absolute percentage error (APE) and the Root mean square error (RMSE):-

These measures can be calculated by the following formulas:

$$\mathbf{APE} = \sum_{i=0}^{n-1} |(ti - oi)| / \sum_{i=0}^{n-1} |(oi)| \quad \textbf{(5)}$$

$$\mathbf{RMSE} = \sqrt{[\frac{1}{n} \sum_{i=0}^{n-1} (ti - oi)2]} \quad \textbf{(6)}$$

In Eq5, 6 ti is the target output and oi is the actual output, and n is the number of test samples. The accuracy of the model calculated by: - the accuracy = 1 – (APE or RMSE) %.

The numerical results of (APE) and (RMSE) measures for both ANN and ARIMA models show in table6a and table6b.

Models	**APE values of ANN**	APE values of ARIMA
Model1	**8.30**%	**16.90**%
Model2	**8.90**%	**17.88**%
Model3	**9.30**%	**18.90**%
Model4	**10.30**%	**20.70**%
Model5	**10.60**%	**20.97**%

Table6a. The values of APE for both ANN and ARIMA models

APE values that were described in Table6a deduce that absolute percentage error of the ARIMA models in is between 16.90 % & 20.97 % and the absolute percentage error of the ANN models is between 8.30 % & 10.60%.

RMSE measure that was described in Table6b also confirmed that neural networks outperform ARIMA models.

Models	**RMSE values of ANN**	RMSE values of ARIMA
Model1	**7.60**%	**17.30**%
Model2	**8.20**%	**18.00**%
Model3	**8.60**%	**19.10**%
Mode4	**10.90**%	**20.80**%
Model5	**11.30**%	**21.00**%

Table6b. The values of RMSE for both ANN and ARIMA models

8. Conclusions and Further Points for Research

There is growing evidence that macroeconomic series contain non-linearities but linear models such as the ARIMA models are widely used for forecasting such series, despite the inability of linear models to cope with non-linearities. In this paper we provide a new empirical evidence on the relative macroeconomic forecasting performance of linear ARIMA models and the nonlinear ANN.

Results show that neural networks outperform ARIMA models in our forecasting problem. As an extension of this work, we hope to further refine the best neural network models in this paper by considering additional layers and different training periods to exploit readily available monetary and financial data in order to gauge future macroeconomic activity.

Comparison with more complicated forecasting models to prove the quality of our model is one of our main future research points. Another point is the possibility to build another approach with combing different forecasting models such as ARIMA and neural network models.

9. References

[1] Adya, M. and Calopy.(1998). How effective are neural networks at forecasting and prediction? A review and evaluation, Journal of Forecasting, 17, 481-95.

[2] Balkin, S.D. and Ord, J.K. (2000). Automatic neural networks modeling for univariate time series, International Journal of Forecasting, 16, 509-15.

[3] Barron, A.R. (1994). A comment on neural networks: a review from a statistical perspective, Statistical Science, 9, 33-5.

[4] Binner, J.M, Gazely, A.M. (1999). A neural network approach to inflation forecasting: the case of Italy, Global Business and Economics Review, 1, 76-92.

[5] Binner, J.M, Gazely. (2002). Financial innovation and Divisia indices in Taiwan: A neural network approach, European Journal of Finance, 8, 238- 47.
[6] Bishop, C.M. (1995). Neural Networks for Pattern Recognition, Oxford University, Press, NewYork.
[7] Boger.Z, Weber.R. (2000). Finding an Optimal Artificial Neural Network Topology in Real-Life Modeling. In: Proceedings of the ICSC Symposium on Neural Computation, Article No. 1403/109.
[8] Church, K.B, Curram.S. (1996). Forecasting consumers' expenditure: a comparison between econometric and neural network models, International Journal of Forecasting, 12, 255-67.
[9] Dorsey, R.E. (2000). Neural networks with Divisia money: better forecasts of future inflation?, in Divisia Monetary Aggregates: Theory and Practice (Ed.) M.T. Belongia and J.M. Binner, Palgrave, New York, pp. 28-43.
[10] Gately, E.(1996). Neural Networks for Financial Forecasting, John Wiley, and NewYork.
[11] Gazely, Binner, J.M. (2000). The application of neural networks to the Divisia index debate: evidence from three countries, Applied Economics, 32, 1607-15.
[12] Gorr, W.L, Szcypula. (1994). Comparative study of artificial neural network and statistical models, International Journal of Forecasting, 10, 17-34
[13] Hertz.J, Krogh, Palmer. (1991). Introduction to the Theory of Neural Computation. Addison-Wesley: Redwood City, California.
[14] Mareleo.S, Portugal. (1995). Neural network versus time series methods a forecasting exercise, 14th international symposium on forecasting.
[15] Moshiri, Cameron.(2000). Neural networks versus econometric models in inflation forecasting, Journal of Forecasting, 19, 201-17.
[16] Nag, A.K, Mitra. (2002). Forecasting daily foreign exchange rates using genetically optimized neural networks, Journal of Forecasting, 21, 501-11.
[17] National Bank of Egypt. (1980 to 2002). Economic Bulletins, Vol.No.4.
[18] Naylor, T. H. (1980). Corporate Planning Models, Addison & Wisley.
[19] Nicholas.R, Farnuum, LaVerne.W. (1989). Quantitative-forecasting methods, PWS-KENT publishing bank
[20] Plasmans, Verkooijen, Daniels. (1998). Estimating structural exchange rate models by artificial neural networks, Applied Financial Economics, 8, 541-51.
[21] Shazly, M.R.E, Shazly.(1999). Forecasting daily foreign exchange rates using genetically evolved neural network architecture, International Review of Financial Analysis, 8, 67-82.
[22] Spyros.M, Stephen.C, Wheelwright. (1998). Forecasting methods and applications, third Edition, John Wiley and sons Inc.
[23] Tiao, G.C, Tsay. (1994). Some advances in non-linear and adaptive modeling in time series, Journal of Forecasting, 13, 109-31.
[24] Thomas.H, Naylor. (1985). The Age of Corporate Planning Models, Addison & Wisley.
[25] Tkacz.G. (2001). Neural Network forecasting of Canadian GDP growth, International Journal of Forecasting, 17, 57-69.
[26] Zhang, H.U, Y.H. (1998). Forecasting with artificial neural networks: the state of the art, International Journal of Forecasting, 14, 17-34.

Development an Analytical Performance Models for Matrix Multiplication on Distributing Systems

Arabi Keshk
Computer Science Dept., Faculty of Computers and information, Menoufiya University
arabikeshk@yahoo.com

Abstract *In this paper, we suggest a mechanism for implementing a distributed application using RMI based on JAVA threads. The application is parallel matrices multiplication depending on distributed the products block of rows and columns on different machines. One server and seven clients are run to find the product of matrix multiplication. The server distributes the determine blocks of rows and columns on the registered clients. The clients return their product blocks to a server, which calculate the final product of matrix multiplication. Applications of this type will allow loaded servers to transfer part of the load to clients to exploit the computing power available at client side. Experimental result shows that the speed up ratio is equals 9 or the computation time of matrix multiplication with size of 2048 X 2048 is reduced by 89 % by using 7-client.*

Keywords*: Distributed systems, Performance prediction model, Matrix multiplication*

1. Introduction and related works

Matrix multiplication (MM) is an important linear algebra operation. A number of scientific and engineering applications include this operation as a building block. Due to its fundamental importance, much effort has been devoted to studying and implementing MM. MM has been included in several libraries. Many MM algorithms have been developed for parallel systems [1-4].

Traditional methods for distributed application are decomposed the entire task, which introduces various overheads. The most important are the communication and load balancing overheads. For example, partitioning MM into sub-matrix blocks and decomposing the MM operation are often technology dependent. Applying special implementations for sub-matrix blocks may improve performance since the workload of sub-matrix operations may vary. The information about the workload of a task for matrix operation may not be available at compile time or even at the time of initiating subroutines. It may be available only after these routines have been executed. Since this increases the complexity of load balancing, it is often ignored.

Most parallel algorithms are optimized based on the characteristics of the targeting platform. The PC cluster computing platform has recently emerged as a viable alternative for high-performance and low-cost computing [2]. Generally, the PCs in a cluster have a lot of resources that can be used simultaneously. They have relatively weak communication capabilities. They lack high performance implementation support for data communications compared to supercomputers. They only support some communication channels implemented by software that capitalizes on Ethernet connections. MM operations are embedded in many host programs.

The main reason for using parallel processing is to reduce the computation time required for what would otherwise be very long-running programs. Because poorly parallelized code tends to offer little performance benefit, there is great incentive to ensure that parallel programs are highly optimized. Unfortunately, a lack of sufficiently accurate and easy-to-use performance prediction methods for parallel programs has necessitated resort to a very time-consuming, which modifies design cycle to achieve this [5].

The biggest price we had to pay for the use of a PC cluster was the conversion of an existing serial code to a parallel code based on the message-passing philosophy. The main difficulty with the message passing philosophy is that one needs to ensure that a control node (or master node) is distributing the workload evenly between all the other nodes (the compute nodes). Because all the nodes have to synchronize at each time step, each PC should finish its calculations in about the same amount of time. If the load is uneven (or if the load balancing is poor), the PCs are going to synchronize on the slowest node, leading to a worst-case scenario. Another obstacle is the possibility of communication patterns

that can deadlock [5]. A typical example is if PC A is waiting to receive information from PC B, while B is also waiting to receive information from A.

JAVA provides Remote Method Invocation (RMI) to allow one JAVA Virtual Machine (VM) to invoke methods running on another VM. RMI applications are often comprised of two separate programs, which are server and client. A typical server application creates some remote objects, makes references to them accessible, and waits for clients to invoke methods on these remote objects. A typical client application gets a remote reference to one or more remote objects in the server and then invokes methods on them. RMI provides the mechanism by which the server and the client communicate and pass information back and forth. Such an application is sometimes referred to as a distributed object application [6-8].

The related work [1– 4] deals with moving application from one machine to another in a set of machines. In the pervious distributed modes it is not very easy for a programmer to write an application a part of which can be executed on remote side and result of computation can be combined in original program to compute the final result. i.e. there is no method level distributed available. We have applied object mobility to LAN architecture. This allows development of applications that can be easily load balanced.

In this paper we study implementation for matrix multiplication on distributed systems using RMI based on JAVA threads, which distribute the load between the server and the clients. This system provides the user a level of control over the distribution of the program. The rest of this paper is organized as follow. Section 2 presents the mathematical model of distributed tasks. Section 3 introduces the techniques of matrix multiplication. Section 4 proposes matrix implementation performance models. Section 5 presents our implementation architecture. Section 6 shows the experiment results. Finally the conclusion is provided in Section 7.

2. Distributed tasks assignment mathematical modeling

In [9], generally a mathematical model to the distributed task assignment problem involves the following two steps:

(1) Formulate a cost function to represent the main purpose of the task assignment process,
(2) Formulate a set of constraints/inequalities in terms of both the application requirements and the availability of the system resources.

This section presents the main components of the cost function in general, and it describes the most important constraints that may be considered with the assignment problem. Finally, it presents an example for modeling the assignment problem mathematically.

2.1. Components of the cost function

As mentioned previously, the assignment problem is usually handled based on the optimization of a cost/objective function. Depending on the context, many components of the cost function may be defined. To do so, let X be an M x N binary matrix corresponding to an assignment of M tasks onto N processors (PCs) such that

$$X_{ip} = \begin{cases} 1 & if\ a\ task\ i\ is\ assigned\ to\ a\ processor\ p, \\ 0 & otherwise \end{cases}$$

Where, Xip is binary variables such that i is ranged over the set of tasks and p ranged over the set of processors. The main components of the cost function may be described as in the following.

2.1.1. Accumulative execution cost/time:

The cost of processing tasks assigned to a processor (EXECp) is the total execution time incurred by tasks running on the processor p. This cost depends on the size of the tasks residing at the processor p and on the speed of the processor p. Define TCp as the set of tasks that are assigned to the processor p under a task assignment X, such that TCp={i | Xip=1, 1≤i≤M, 1≤p≤N}, and let Cip denotes the cost of processing a task i on a processor p, then the actual execution cost EXECp may be formulated as:

$$EXEC_p = \sum_{i \in TC_p} C_{ip} = \sum_{i \in TC_p} d_i * e_p$$

Where, di is the size of the task i expressed in size of execution code, i.e., number of instructions to be processed, or in execution time on some normalized processor, and ep is the average processing time of one instruction on the processor p.

For an assignment X, the Accumulative Execution Time (AET) defines the total execution/processing time incurred for running all tasks at all processors in the distributed system. Hence, the AET may be formulated as:

$$AET = \sum_p EXEC_p = \sum_p \sum_i C_{ip}\ X_{ip}$$

2.1.2. Accumulative Communication Time

The actual communication cost/time at a processor COMMp is the total time of communicating data between tasks resident at the processor p with other tasks resident at other processors in the system. This cost depends on the quantity of information to be exchanged between tasks, the interconnection topology between computers of the distributed system, the propagation delay through the transmission media and the speed/bandwidth of the communication media. Let TCq be the set of tasks that are assigned to a processor q, where TCq = {j | Xjq=1, 1≤j≤M, 1≤q≤N}, and define Cijpq as the cost of sending a data between a task i residing at the processor p and a task j residing at other processor q through a communication path/link pq, then the actual communication cost/load COMMp may be formulated as:

$$COMM_p = \sum_{q \neq p} \sum_{i \in TC_p} \sum_{j \neq i, j \in TC_q} C_{ijpq} = \sum_{q \neq p} \sum_{i \in TC_p} \sum_{j \neq i, j \in TC_q} (S_{pq} + d_{ij}\ c_{pq})$$

Where, Spq is the time necessary for a processor p to set up the communication with other processor q, dij is the average quantity of information/data to be transferred between the processors p and q, and cpq is the average time of transferring a data unit from the processor p to the processor q through the path/route pq after the set up is completed.

For an assignment X, the Accumulative Communication Time (ACT) is the total time incurred for communicating/exchanging data between tasks residing at separate computers of the distributed system. Hence, the ACT may be formulated as:

$$ACT = \sum_p COMM_p = \sum_p \sum_{q \neq p} \sum_i \sum_{j \neq i} C_{ijpq}\ X_{ip}\ X_{jq}$$

The ACT is often considered to be one of the most important factors which need to be minimized by the task assignment.

It is worth noting that if two communicating tasks are assigned to different processors, the communication cost contributes to the load at the two processors, i.e., bilateral communication is assumed. Indeed, if two communicating tasks are assigned to the same processor, the communication cost is assumed to be zero as they use the local system memory for data exchange.

2.2. Allocation/Assignment Constraints

To meet the application requirements and not violate the availability of the system resources, the assignment should be done taking into account various kinds of constraints. These constraints depend on the characteristics of the involved application tasks, such as processing load requirements, memory requirements, amount of inter-task communication capacity requirements and precedence relation between tasks, and on the availability of the system resources including the available computation speed of processors, the available memory capacity and the available communication capacity of the communication network resources.

These constraints may be classified into two broad categories, namely, locality and devices constraints. The locality constraints concern the location of tasks on different processors. For instance, a task might require to be allocated onto a specific processor or two exclusive tasks must not be allocated on the same processor. On the other hand, the devices constraints define the relation between

the tasks requirements from a physical device, used by the tasks during execution such as memory, CPU, and transmission media. Devices constraints might be the case that only a limited number of tasks can access a certain device at the same time to ensure that the amount of the resources used, by concurrent tasks, never exceeds a given limit at any instant. Hence, the devices constraints may be classified as: memory constraints, processing load constraints, and communication media capacity constraints. To describe these constraints, define the following parameters:

m_i memory requirements of task i,
p_i computation load requirements of task i,
d_{ij} data to be transferred between tasks i and j.
b_{ij} communication capacity requirements of edge (i,j).
M_p available memory capacity of computer p,
P_p available computation capacity of computer p,
A_s available communication capacity of resource s,
C_{ip} cost of processing task i on computer p.
C_{ijpq} cost of transferring data d_{ij} between two tasks i and j if they are assigned to different computers p and q respectively. We assume that $C_{ijpq} = 0$ if the two tasks i and j are assigned to the same computer.

2.2.1. Location constraints:

The location constraints guarantee that each task is assigned to one and only one processor on which it is entirely executed without preemption, i.e., no software/task redundancy is considered. To do so, the following equality must hold at each task:

$$\sum_{p} X_{ip} = 1$$

2.2.2. Memory Constraints

For an assignment X, the total memory required by all tasks assigned to a processor p must be less than or equal to the available memory capacity of the processor p. Let mi denotes the amount of memory required for processing a task i and Mp defines the available memory capacity at the processor p, then the following inequality must hold at each processor p in the system:

$$\sum_{i \in TC_p} m_i \leq M_p$$

2.2.3. Processing Load Constraints

For an assignment X, the total processing load required by all tasks assigned to a processor p must be less than or equal to the available computational load of the processor p. Let pi denotes the processing load requirements of a task i and Pp denotes the available processing load of the processor p, then the following inequality must hold at each processor p in the system:

$$\sum_{i \in TC_p} p_i \leq P_p$$

2.2.4. Communication Capacity Constraints

For an assignment X, the total communication capacity required by all edges mapped to a communication path/link pq must be less than or equal to the available communication capacity of the path pq. Let bij denotes the amount of communication capacity required to communicate data between tasks i and j residing at different processors p and q respectively, and Apq denotes the available communication capacity of the path/link pq. Then, the following inequality must hold at each communication path/link pq.

$$\sum_{i \in TC_p} \sum_{j \neq i,\ j \in TC_q} b_{ij} \leq A_{pq}$$

Note that, the available communication capacity Apq of the path/link pq is the minimum available communication capacity over all the communication resources of the route from p to q. Let As be the available communication capacity of a communication resource s, then the available communication capacity of the path/link pq may be defined as Apq = min {As | s in resources (path pq)}.

In general, the above constraints are very important to be considered with the assignment problem. If such constraints are not considered, a task may be allocated to a machine that cannot process the task. Indeed, the communication capacity constraints influence the performance of the distributed system and should be considered with the task assignment.

2.3. Mathematical Modeling

Based on the different components of the cost function and the different types of constraints, several versions of the task assignment problem may be formulated. For example, for a given application of M tasks and a distributed system of N computers, if the objective is to find an assignment that minimizes the total sum of execution and communication costs such that each task i must be assigned to exactly one processor, then the problem may be formulated as follows:

$$min \sum_i \sum_p C_{ip}\ X_{ip} + \sum_{(i,j) \in E} \sum_p \sum_{q \neq p} C_{ijpq}\ X_{ip}\ X_{jq}$$

$$subject\ to$$

$$\sum_p\ X_{ip} = 1 \qquad \forall\ tasks\ i$$

$$\sum_i\ p_i\ X_{ip} \leq P_p \qquad \forall\ computers\ p$$

$$\sum_i\ m_i\ X_{ip} \leq M_p \qquad \forall\ computers\ p$$

$$\sum_i \sum_{j \neq i}\ b_{ij}\ X_{ip}\ X_{jq} \leq A_{pq} \qquad \forall\ paths\ pq$$

3. Matrix Multiplication Techniques

Consider the matrix multiplication product C = A×B where A, B, C are matrices of size n X n as shown in Fig. 1 where n = 4. Next subsections present the various methods that used to find the matrix multiplication.

3.1. Sequential Method

The matrix operation derives a resultant matrix by multiplying two input matrices a & b, where matrix a is a matrix of N rows by P columns and matrix b is of P rows by M columns. The resultant matrix c is of N rows by M columns. The serial realization of this operation is quite straightforward as listed in the following:

```
for(k=0; k<M; k++)
  for(i=0; i<N; i++){
    c[i][k]=0.0;
    for(j=0; j<P; j++)
      c[i][k]+=a[i][j]*b[j][k];
  }
```

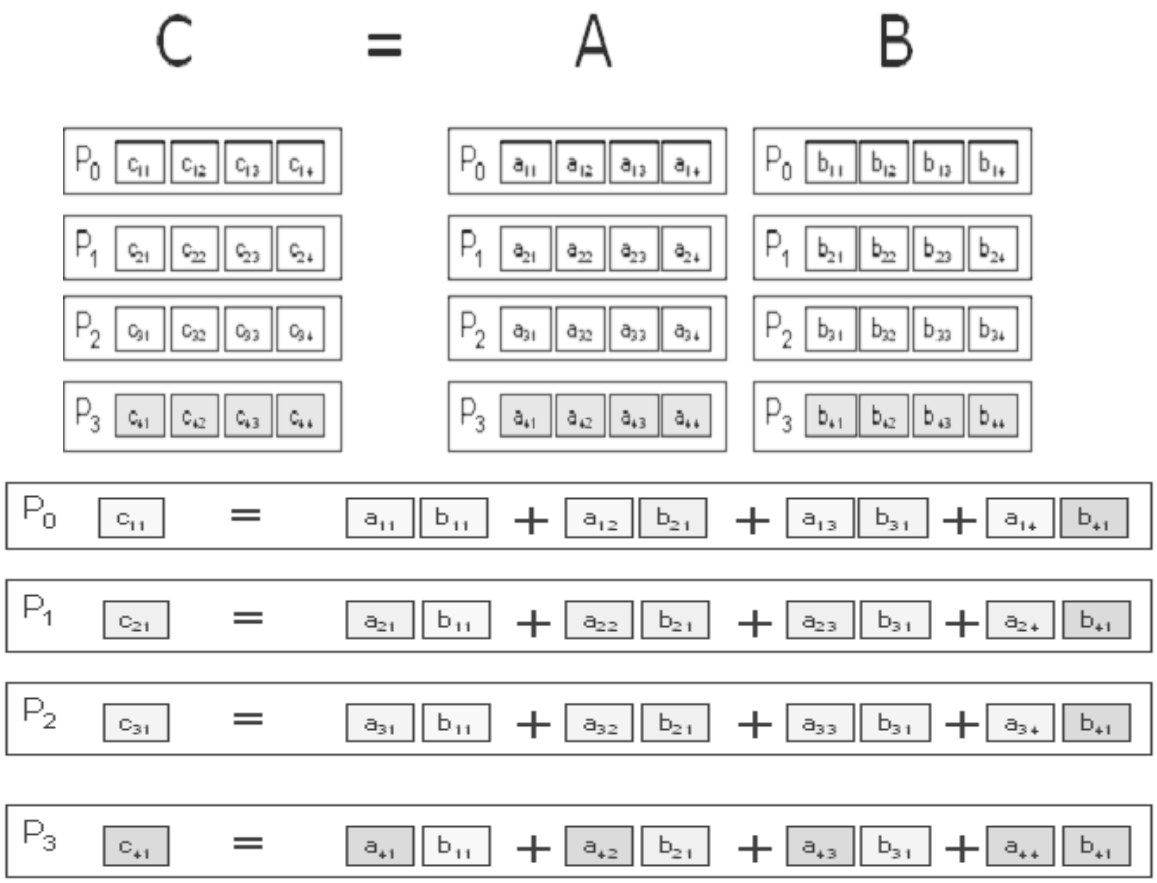


Fig. 1 Matrix multiplication

The above algorithm requires n^3 multiplications and n^3 additions, leading to a sequential time complexity of O (n^3).

3.2. Circular pipeline method

The slaves are peer processes that interact by means of circular pipeline as shown in Fig. 2. Assume that a, b, and c are n × n matrices. Each slave has one row and the first slave have b matrix. Thus each slave execute a series of rounds, where in each round it sends its column of b to the next slave and receives a different column of b from the previous slave.

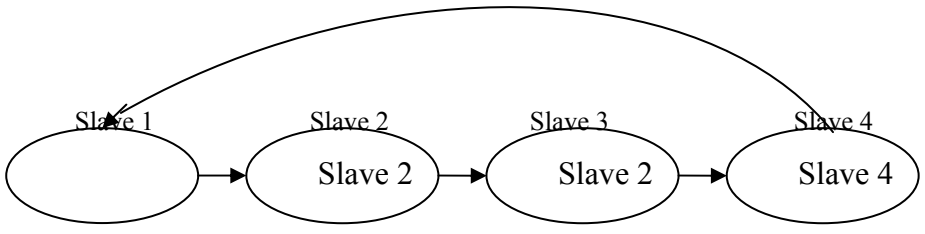


Fig.2 A Circular Pipeline

In Fig. 2, the next slave is the one with the next higher index, and the previous slave is the one with the next lower index (for the slave n, the next slave is 1, for the slave 1, the previous worker is n). The columns of matrix b are passed circularly among the slaves so that each slave sees every column. It is

assumed that master process sends rows of A and columns of B to the slaves and then receives rows of C from the slaves.

3.3 Server Client Model

The matrix multiplication algorithm is implemented in MPI using the straight forward algorithm based on the Server Client paradigm [8]. Server Client computing paradigm, which also called replicated slave computing is consists of broken many computational problems into smaller pieces that can be computed by one or more processes in parallel. The computations are fairly simple, which usually compute-intensive, region of code. The size of loop is quite long.

Fig. 3 shows a Server Client computing paradigm, a Server process takes the work performed in the computationally intensive loop and divides it up into a number of tasks that it deposits into a task bag. One or more processes, known as Client grab these tasks, compute them and place the results back into a result bag. The Server process collects the results as they are computed and combines them into something meaningful such as a vector product.

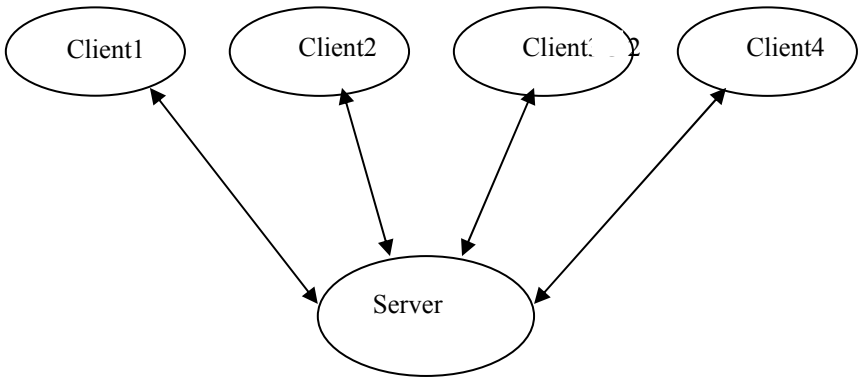


Fig. 3 Server Client computing paradigm

Message Passing Interface is a widely used standard for writing message-passing programs to establish a practical, portable, efficient, and flexible standard for message passing. The Server creates a set of random matrices. Each matrix multiplication job consists of pair of matrices to be multiplied. For each job the Server sends one entire matrix to each slave and distributes the rows of the matrix among the clients. In this way matrix multiplication jobs are computed in a parallel fashion as follow;

(1) The server process for each job, which sends the first matrix from the pair of matrices multiplication joined with a certain number of rows of the other matrix depending on the number of clients.
(2) Each client process receives one entire matrix and a certain number of rows of the other matrix based on the number of clients. Thus it computes the rows of the resulting matrix and sends it back to the server.
(3) The Server process collects the rows of resulting matrix from the clients.

4. Matrix implementation performance models

We develop an analytical performance model to describe the computational behavior of 3- matrix multiplication implementations. We consider the matrix multiplication product C = A×B where the size of matrices A, B, and C are n*n. The system is consists of one server machine and N clients, and the performance modeling of the three implementations is presented in next subsections.

4.1. Allocation block distribution of the matrix B columns

For the analysis, we assume that the entire matrix B stored in the local disk of the server. The basic idea of this implementation is as follows: The server broadcasts the matrix A to all clients. It partitions the matrix B into blocks of columns b and each block is distributed dynamically to a client. Each client executes a parallel matrix – vector multiplication algorithm between the matrix A and the corresponding block of b columns. Finally, each client sends back a block of size b columns of the matrix C.

The execution time of the dynamic implementation that is called matrix multiplication for distributed B columns block (MM-DBCB) can be broken up into five terms:

T1: It includes the communication time for broadcasting of the matrix A to all clients involved in processing of the matrix multiplication by using RMI. The size of each row of the matrix A is n*sizeof(int) bytes. Therefore, the total time T1 is given by:

$$T1= b* n *sizeof(int) / S$$

Where S is the communication speed

T2: It is the total time to read the columns of the matrix B into several blocks of size b*n*sizeof(int) bytes from the local disk of the server. The b is the number of columns. Therefore, the server reads n2*sizeof(int) bytes totally of the matrix. Then, the time T2 is given by:

$$T2= n2 *sizeof(int) / R$$

Where R is the I/O read time of the server

T3: It is the total communication time to send all blocks of the matrix B to all clients. The size of each block is b*n*sizeof(int) bytes. Therefore, the server sends n*sizeof(int) bytes totally. Then, the time T3 is given by:

$$T3 = n *sizeof(int) / S$$

T4: It is the average computation time across the systems. Each client performs a matrix – vector multiplication between the matrix A and the block of the matrix B with size n*(n/b)*sizeof(int) bytes. Then, the time T4 is given by:

$$T4 = n *(n/b)*sizeof(int)$$

T5: It includes the communication time to receive n / b results from all clients. Each client sends back b*n*sizeof(int) bytes. Therefore, the server will receive n 2*sizeof(int) bytes totally. Therefore, the time T5 is given by:

$$T5 = n 2*sizeof(int) / S$$

Therefore, the total execution time of our dynamic implementation, TN, using N clients, is given by:

$$TN = T1+T2+T3+T4+T5$$

4.2. Allocation blocks distribution of the matrix A rows and B columns

For the analysis, we assume that the entire matrices A and B stored in the local disk of the server. The basic idea of this implementation is as follows: The server broadcasts the blocks of matrix A by rows to all clients and. Also, it sends the corresponding blocks of columns from matrix B to the same clients. The server reads a block of size b rows and columns of the matrix A and B from the local disk of the server. Also, each client executes a parallel matrix – vector multiplication algorithm between the rows block of matrix A and the corresponding columns block of matrix B. Finally, each client sends back a block of size b columns of the matrix C. The execution time of the dynamic implementation that is called Matrix multiplication by distributed rows block of matrix A and columns block of matrix B (MM-DAR&BCB), can be broken up into five terms:

T1: It includes the communication time for broadcasting of the matrix A by rows and matrix B by columns to all clients involved. The amount of this time is equal to T3 of the previous dynamic implementation.

T2: It is the same time of T2 of the previous implementation.

T4: It includes the average computation time across the client. The amount of this time is similar to the time T4 of the previous implementation.

T5: It includes the communication time to receive the results of the matrix - vector multiplication from all clients. The amount of this time is same with the time T4 of the previous implementation.

Therefore, the total execution time of our dynamic implementation is reduced from the previous model by the value of T1. TN, using N clients, is given by:

$$TN = T1+T2+T4+T5$$

4.3. Allocation block distribution of the matrix A rows

This algorithm will give the same amount of TN of the first model as in section of 4.1 and is called matrix multiplication for distributed A rows block (MM-DARB).

5. Implementation architecture

In our work we develop server client model to calculate MM by using RMI Java threads for the above models in section 4 especially with detailed discussion on the 4.2 model. In our proposed model, the server determines the distributed numbers of rows from the first matrix and the corresponding columns of the second matrix depending on the balance of workload on registered clients. In the heterogeneous matrix, which mean the number of rows is not the same number of columns, the multiplication will done because the load distribution depend on the number of rows in the first matrix with its corresponding columns of the second matrix. One server and 7-clients are used to implement MM as shown in Fig. 4. RMI implementation algorithm is shown as follow;

Step 1 Client discovery;
Client will register itself with the server to take a task from it

Step 2 Generate Matrices;
Server generates two matrices randomly or getting them as inputs

Step 3 Data distribution
Server will distribute number of rows from first matrix and its corresponding columns of the second matrix on clients that it has been registered using Java threads.

Step 4 Sever waits for result;
Server will waits results from clients and append it in result matrix

Step 5 Results collecting;
Server will collect the results that sent by each client and compute the time that taken by each client and compute all time taken in this process

Step 6 Shutdown;
Finally, server will send shutdown to all clients.

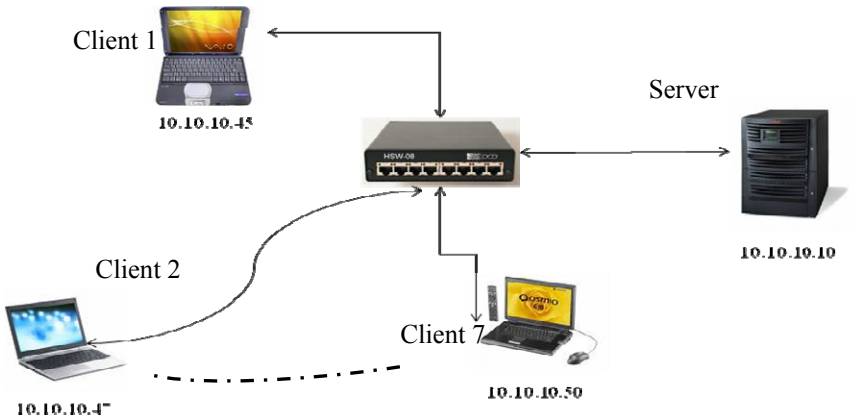


Fig. 4 RMI Server-Client architecture

Fig. 5 shows the flow diagram of the above algorithm. In our work we addressed the following issues:

(4) Performance: As the number of clients increases the time for computation will decreases.
(5) Load Distribution: the work will be distributed among the free clients. Rows and columns of MM are distributed uniformly on all registered clients. In RMI the load of MM was distributed by allowing each process (Threads) to compute a certain number of rows in the resulting matrix.
(6) Scalability: Performance of our model increases as the number of registered clients increases.

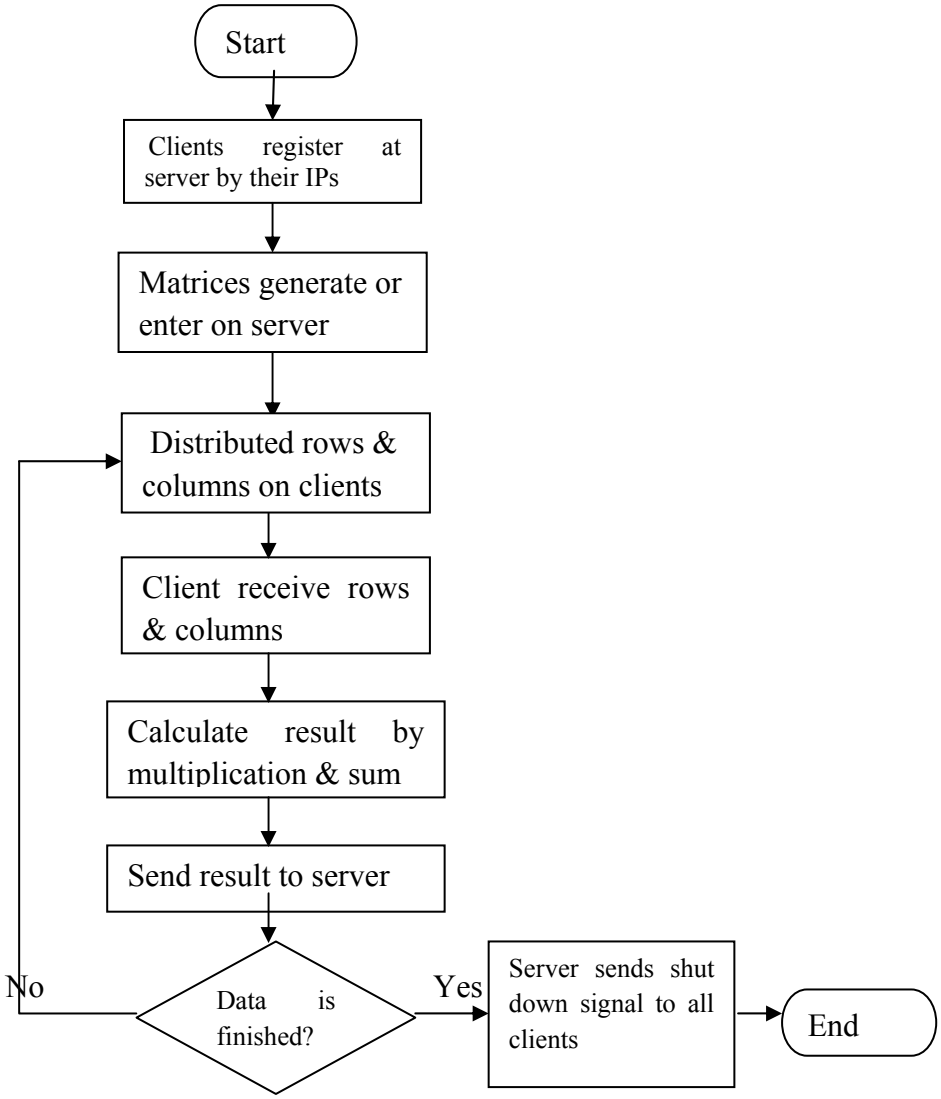


Fig. 5 flow diagram of MM algorithm

6. Experiment results

We develop an analytical performance model to describe the computational behavior of matrix multiplication implementations. We consider the matrix multiplication product C = A×B where the size of matrices A, B, and C are n X n. We implement MM by using JDK 1.6 on LAN of 8-PC 2.66 GHz with 512 MB Ram. Also, the matrix size does not exceed the memory bound of any machine in the system.

Measuring the performance of a parallel program is a way to assess how well and efficient our development, which have been divided the big application into small modules cooperating with each other in parallel.

The most visible and easily recorded metric of performance is the execution time. By measuring the time consumed in execution of parallel program, we can directly measure its effectiveness. To find out how much better for our proposed does on the parallel machine, which it compared with running an application on only one processor. The ratio of execution time is taken into the account, which is called the speedup.

Speedup = (Serial Execution Time)/(Parallel Execution Time)

Speedup = T(1)/T(N)

Where T(N) represents the execution time taken by the program running on N processors, and T(1) represents the time taken by the best serial implementation of the application measured on one processor. Figure 6 shows speedup is equal to 9 for MM with size 2048 X 2048 or reduced the

computation time by 89 %. Also, Figure 7 shows speedup is reduced and equal to 6 for MM with size 2048 X 2048 or reduced the computation time by 84 %.

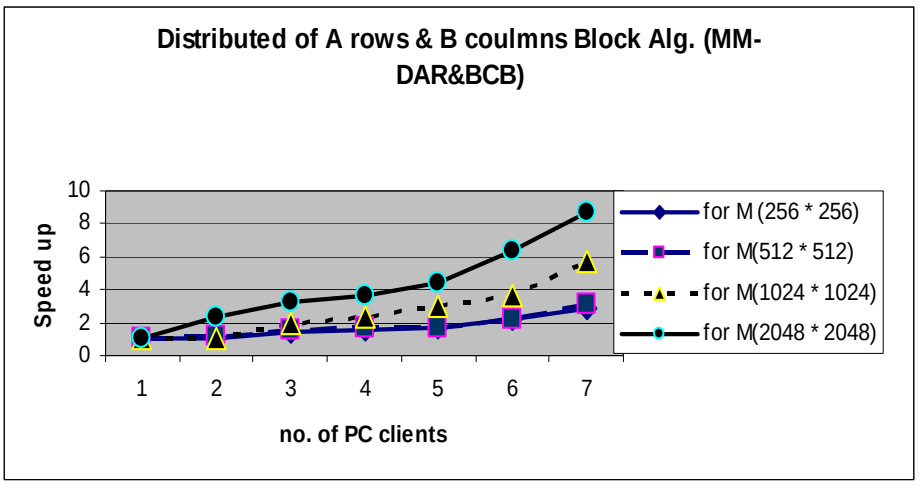


Fig. 6 Experiment results of (MM-DAR&BCB) model of section 4.2

7. Conclusion and Future Work

In this paper we implemented and analyzed the parallel matrix multiplication on distributed systems. Our mechanism will make it easier to automatic migrate the computation load to client. It has been shown that the execution time decreases by 89% for MM with size 2048 X 2048. Future work will apply this implementation on any practical application like weather prediction, databases systems, data compression and others with increasing the numbers of clients.

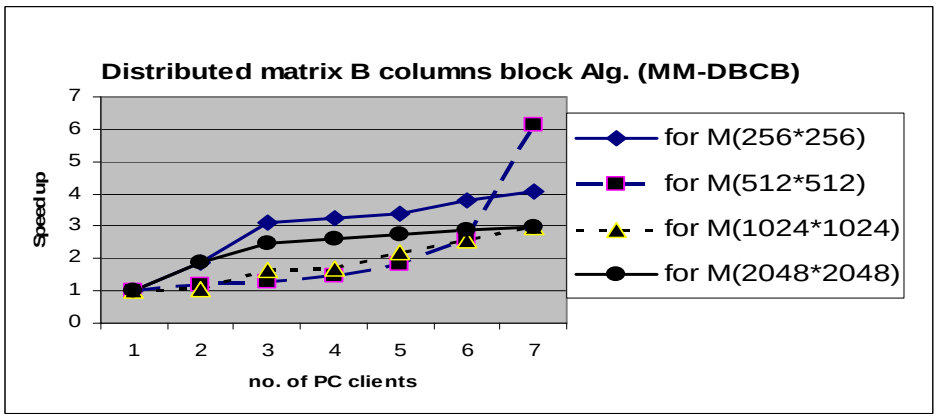


Fig. 7 Experiment results of (MM-DBCB) model of section 4.1

8. Refferences

[1] Tinetti, F., Quijano, A., Giusti, A., Luque, E. “Heterogeneous networks of workstations and the parallel matrix multiplication”, Proceedings of the Euro PVM/MPI 2001, Springer-Verlag, Berlin, pp. 296-303. 2001.

[2] T. Typou, Vasilis stefanids, P. Michailidis, and K. Margaritis,“Implementing Matrix Multiplication on An Cluster of Workstation”, 1st IC-SCCE, Athens, 8-10 Sep., 2004.

[3] Ju-wook Jang, Seonil Choi and Viktor K. Prasanna, “Energy-Efficient Matrix Multiplication on FPGAs”, IEEE Transactions on VLSI (TVLSI), Vol. 13, No. 11, pp. 1305-1319, 2005.

[4] Carrio and Geleertner “How to write Parallel Programs, A Guide to the Perplexed” ACM Computing Serveys, Vol. 21, No. 3, Sep.1999.

[5] Coulouris, et al., “Distributed Systems Concepts and Design”, 3rd Edition, Addision Wesely, Person Education 2001.

[6] Maassen, J., Nieuwpoort, R. V., Veldema, R., Bal, H., and Kielmann, T., “Wide-Area Parallel Computing in Java”, Proceedings of the ACM Conference on Java Grande, San Francisco, CA,(1999), pp. 8-14, 1999.

[7] Grama, A., Gupta, A., Karypis, G., and Kumar, V., “Introduction to Parallel Computing” (Second Edition), Pearson Education Limited, Harlow, England, (2003), pp. 345-349, 2003.

[8] Wilkinson, B., Allen, “Parallel Programming: Techniques and Applications Using Networking Workstations”, Prentice-Hall, Inc. 1999.

[9] G. Attiya and Y. Hamam, “Task allocation for maximizing reliability of distributed systems”, Elsevier journal of parallel and distributed computing, 66, pp. 1259-1266, 2006

Fast Texture Synthesis And Image Completion Method

Mohiy M. Hadhoud
Menoufia University, Faculty of Computers and Information, Information Technology Dept., Egypt,
mmhadhoud@yahoo.com

K. A. Mostafa,
Menoufia University, Faculty of Computers and Information, Information Technology Dept., Egypt
kaaly2006@ yahoo.com

Sameh. Z. Shenoda
Menoufia University, Faculty of Computers and Information, Information Technology Dept., Egypt
sameh_37@hotmail.com

Abstract: *recently image restoration is an important field used extensively in artwork restoration. So in this paper, we present a fast texture synthesis and image completion method of natural scenery, where the removal of a foreground object creates a hole in the image. A fast and good quality resulted from the proposed method compaired with others are introduced. Finally the computation time of our method is decreasing from (days, hours and minutes for compaired methods) to less than one second.*

Keywords-: *image restoration, inpainting, texture synthesis and image completion.*

1. Introduction

Traditionally, skilled artists have performed the restoration of image manually. now digital techniques are used for automatic images restoration to modify the damage area in a non-detectable way for an observer not familiar with the original images.

The restoration can be done by using three related approaches, variational image inpainting, texture synthesis and image completion, whereas the meaning of the first approach is restoring the missed and damaged parts of image in a way that the observer who doesn't know the original image can't detect the difference between the original and the restored image by filling unknown area on the image by using surrounding structure information [1].

The second approach is classified into two methods, texture synthesis and constrained texture synthesis. The texture synthesis method is an input sample of a given texture, and the goal is to produce more of that texture**,** but the constrained texture synthesis method is filling unknown area on the image by using surrounding texture information. Texture synthesis approaches could be employed to restore digitized photographs especially if a damaged area needs to be filled with some pattern or structure. Texture synthesis does a good job if the sample area in the same image is large enough. However texture synthesis usually fails if the area to be reconstructed contains an additional color or intensity gradient [2].

In recent years, for two dimensional digital images, the third approach combines both texture synthesis and inpainting approaches to restore large gaps in the image by filling unknown area on the image by using both texture and structure information .

The difference between the three related filed is the size of the region or hole to be restored and the type of data to fill the region . And the common requirements of all the three related approaches are the region to be restored manually selected by the user, because there is no mathematical equation capable of detecting or knowing the region to be filled without taking desired area [3,4].

There are many applications for each one of the three related approaches: restoration of photographs, films, removal of occlusions such as text, subtitle, logos, stamps and scratches are applied by the first. 3D covering and hole filling are applied by the second. finally removing unwanted objects from images and filling-in the image blocks that are lost in transmission are applied by the third [1, 2].

In this paper we proposed a contribution for image completion and texture synthesis applications, where we introduce a fast method for synthesising texture and filling holes caused by removing object without any search or compution compaired to any related work.

This paper is organized as follows. Section 2 gives an overview on the previous texture synthesis and image completion methods. Section 3 presents the proposed algorithm. section 4 illustrate examples of the tested results. Finally the conclusion is drawn in section 5.

2. Previous work

From the later section, its clear that the texture synthesis and image completion are important approaches for restoration of large damaged area. So, in the following section we introduce the most related methods used in these approaches.

Efros and Leung [5] pioneered a nonparametric approach for producing texture from an initial seed instead of applying filters to the sample texture, the model copied the pixels from the sample image itself in synthesis they modeled the texture as a markov random field (MRF). This model produces the new synthesis pixels from a square window around that pixel by indicating the probability distribution of brightness values for a pixel given the brightness values of its spatial neighborhood. The size of the window is a free parameter for the user. This method has few problems such as slipping in the wrong part of search space and growing garbage.

Wei and Levoy [6] extended to Efros and Leung using multi-resolution neighborhood search to reduce the computation time. This algorithm is based on tree structured vector quantization. Texture is synthesized in a coarse to fine manner by using search for the best neighborhood match and multi level resolution pyramid. This method reduces the quality of the result but decreases the computational time.

Drori et al [7] proposed a fragment based algorithm for image completion that could preserve both structure and texture, this method iteratively fills the missing region from the remaining image as the training set using the principle of self similarity, a confidence map is used to determine which pixels have more surrounding information available. The completion starts from more confident pixels, and proceeds in a multi scale fashion from coarse to fine. In each step, a similar image fragment is found and copied to current unknown location, until image is completed. Most results of this method are good in quality but very slow in time more than hour to complete the image .

Peng Tang [2] proposed a non parametric technique based on a markov random field model. The non parametric sampling procedure for filling the image utilizes a series of binary mask Gaussian pyramid level texture to synthesized the texture in a coarse to fine order. The sampling process randomly and uniquely chooses a pixel from the initial synthesis guess level

Criminisi et al [8] proposed exemplar based method similar to fragment based completion in [8] but simple and fast. It selects the patch located at the gap boundary and search for similar patch in the known region in the whole image with special priority to the isophotes (line of equal gray value) so as to preserve the linear structure of filling then copy the matched patch or block from the known region to fill the gap.

Komodakis and Tziritas [9] proposed a new exemplar based method for image completion and texture synthesis as a discrete global optimization problem with a well defined objective function. The optimal solution founded by priority-BP algorithm that which carries 2 major improvements over standard belief propagation first "label pruning" that accelerate the process by allowing less number of source locations that can be copied to more confident point. Second "priority based message scheduling" that also speeds up by giving high priority to belief propagation from more confident points. This method presented better result and the computation lasted only few seconds up to 2 minutes for an image .

Norihiko, Tomokazu and Naokazu [10] proposed new approach that is different from conventional ones in the objective function. This method extended the conventional objective function that based on pattern similarity SSD (sum of square difference) by allowing brightness change of sample and introducing spatial locality. This extension improves the quality but increases the computational time.

The work done by drori[7] and Criminisi [8] comes closest to the work of our paper. The missing area is iteratively filled using the remaining image but our paper remove the search process of the whole image and find the appropriate texture from the principles of a strong horizontal orientation of texture/color distribution and the two sides scanning.

3. Proposed Method

In figure1, the block diagram of the image restoration with any one of the three related approach are shown, the difference between all the image inpainting, texture synthesis and image completion algorithms are the method of filling. All the three related filed must select the region to be filled manually and initializing the selected region by clearing its color. the time of filling is linear to the size of region not the size of image.

Now we come to the block of the proposed filling method. Our proposed filling method is based on two main principles in built it. The first is a strong horizontal orientation and the second is a two side scanning method. By using these principles, we can decrease the time of filling from a few days to less than one second by canceling the search process which takes more time in all previous techniques. Also, by using the same principles for proposed method the quality of restored image increase by filling the region from the two side.

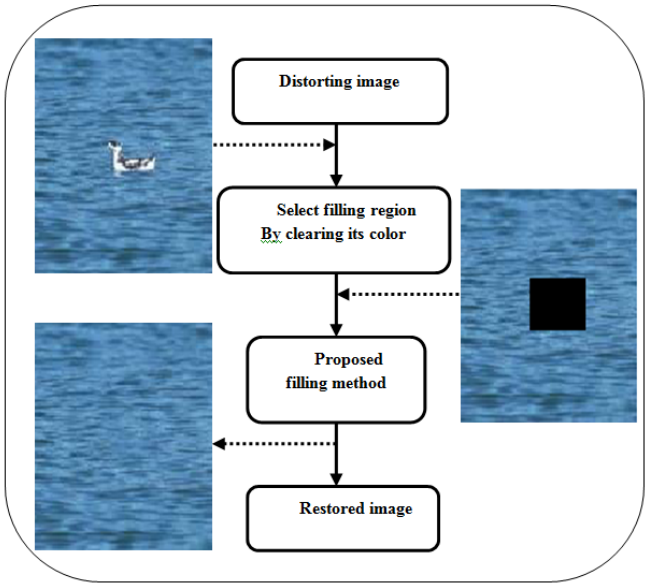


Figure 1 image restoration block diagram

From the natural appearance of the image it is observed that the first principle in images with natural scenery have a strong horizontal orientation of texture/color distribution. We have tried to take the advantage of this fact in our proposed algorithm to fill the missing region .

The assumption of horizontal orientation in natural images is also supported by the Fourier transform of such image, Fourier transform of such image is shown in Figure 2. we observed from the transform, that there is a distinct vertical line at the center. This indicates that the color/texture in the image is horizontally oriented.

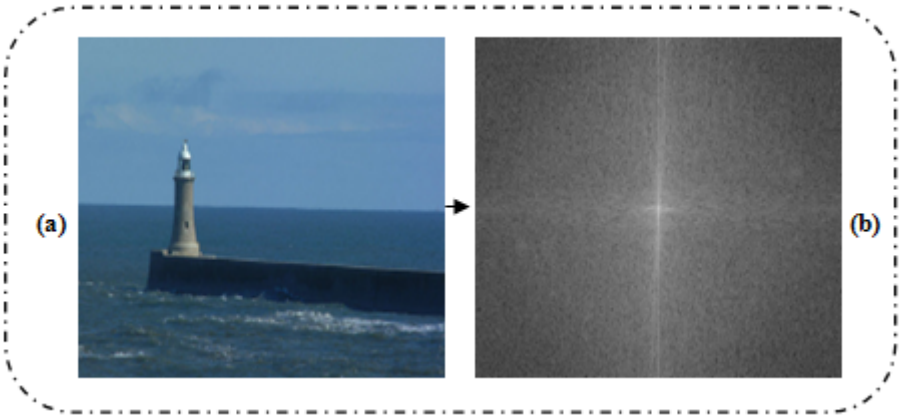


Figure 2 Fourier transform of natural image

a) RGB Image, b) Fourier transform of RGB

We use the image as our training set to complete the image from the horizontal neighborhoods without any search computations.

From the natural appearance of the image, it is also observed the second principle in images with natural scenery or texture image. We noted before removing any object or part from image that the data in the left and right the object is the acceptable data or background that if we fill the region with this data the image appears as well as normal as we show in figure 3 .

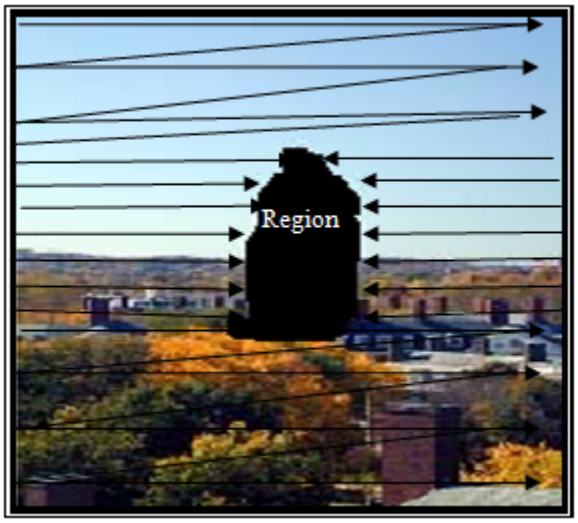


Figure 3 Proposed scanning and filling method

The filling with two side as shown above is better than one side because two reasons. The first reason is scanning and filling with two side decreases the time. The second reason is with one side may fill region with unacceptable data if the object located in non homogeneous texture and spans different homogeneous region. Because of these reasons the algorithm maintains figural similarity by filling the left half of the region with the left side of the known image data and filling the right half of the region with the right part of the known image data .

After explaining the two main principles of our proposed work, now we explain the detailed algorithm.

Instead of explaining the algorithm in a text form we show in figure 4 the detailed flowchart of the proposed method explaining the algorithm step by step in a graph form.

The flowchart is shown in brief. The algorithm makes a raster scan until reaching the first damage pixel, after reaching the first damage pixel the algorithm starts the two side scanning and filling until damage region complete. The detailed steps are shown in the flowchart.

3.1 Concave problem

We fill-up the unknown region with pixels from the source image. In most of the images the removed foreground portion is one single area with a convex shape. Considering a grid row, the unknown region is continuous in such shapes. In some images, the shape of the missing region could be concave. This may cause a small part of the source image to be available between two limbs of the concave region. Due to lack of sufficient number of pixels along the horizontal row, the algorithm would attempt to select some pixels from the missing area itself (self-replacement), causing figural discontinuities in the missing area along one or more rows.

When we come across a row where there are such discontinuities, the result of filling for the source pixels would recover some pixels that are themselves unknown. This means that the pixel in the missing area is being filled in by another pixel from the missing area itself. As we proceed further, the error gets propagated along the row.

The algorithm uses the following method to handle such images with the concave region, when the algorithm fills the row between the two saved region points pixel by pixel it is checked first before filling if the pixel is known between the two limbs of the concave region the algorithm skips this pixel with no change or if the pixel is region pixel (unknown) in one of the two limbs the algorithm continues as the same way as explained later.

4. Dissection of Results

The algorithm is applied with C++ Builder on 1.7 GHz PC processor and 256 of RAM, after experimented the results they are produced in less than one second . To demonstrate the effectiveness of our method we have applied a comparison between our method and latest four conventional methods peng [2], Drori [7], Komodakis [9] and Norihiko [10] as shown in the next comparison tables.

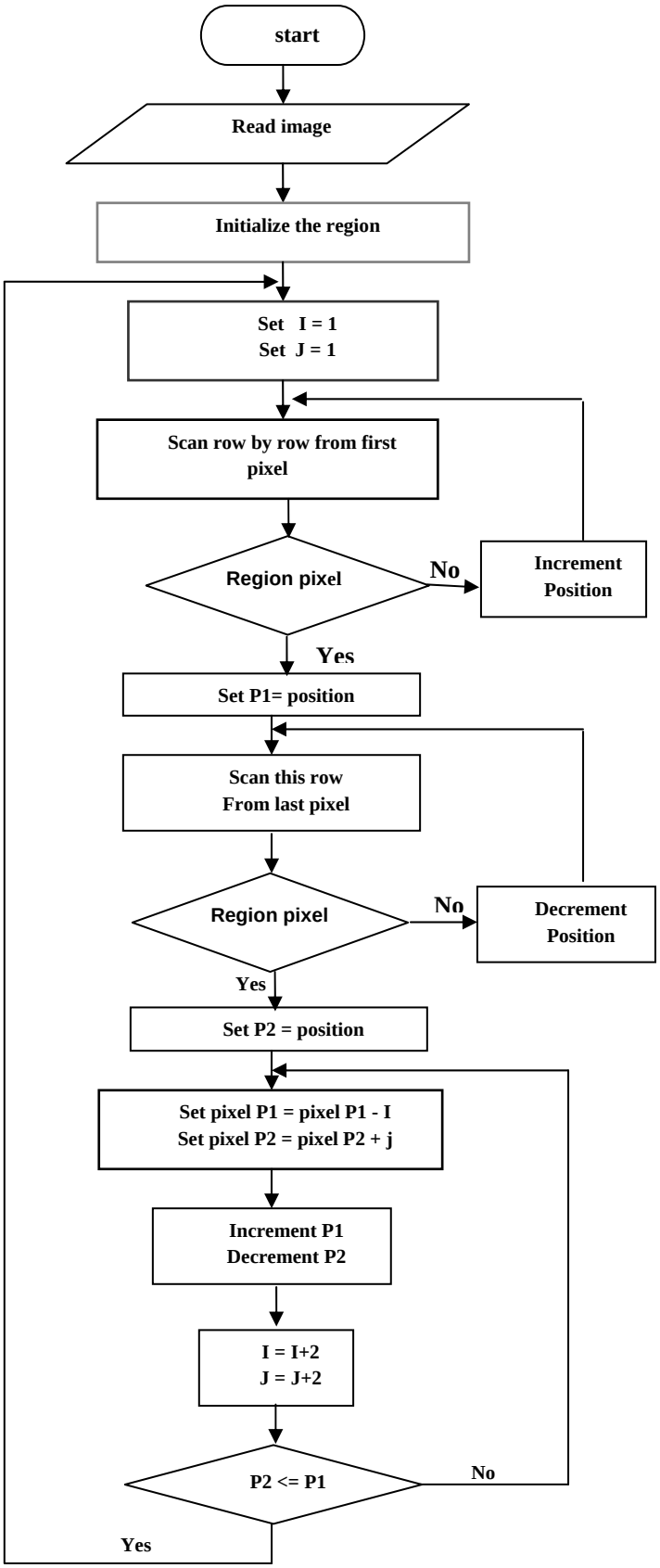


Figure 4 Proposed scanning and filling method

Table 1 processing time of proposed and peng methods

image	Proposed By CPU speed 1.7 GHz	Ref.[2] By CPU speed 2.0 GHz
Fig 5 Tower	Less than 1 sec	days
Fig 5 Horse	Less than 1 sec	days
Fig 5 Texture	Less than 1 sec	days

Table 2 processing time of proposed and Drori methods

image	Proposed By CPU speed 1.7 GHz	Ref. [7] By CPU speed 2.4 GHz
Fig 6 Dolphin	Less than 1 sec	184 sec
Fig 6 Cup	Less than 1 sec	149 sec
Fig 6 Elephant	Less than 1 sec	83 minutes

Table 3 processing time of proposed and Komodakis methods

image	Proposed By CPU speed 1.7 GHz	Ref. [9] By CPU speed 2.4 GHz
Fig 7Golf	Less than 1 sec	up to 2 minutes
Fig 7 Giraffe	Less than 1 sec	up to 2 minutes
Fig 7 Baseball	Less than 1 sec	up to 2 minutes

Table 4 processing time of proposed and Norihiko methods

image	Proposed By CPU speed 1.7 GHz	Ref. [10] By CPU speed 3.2 GHz
Fig 8 Man	Less than 1 sec	8.45 hours
Fig 8 Man & Boy	Less than 1 sec	18.59 hours
Fig 8 Mountain	Less than 1 sec	5 hours

Figure 5 and table 1 illustrate the comparison between the proposed method and Ref.[2] depend on three different images. In the first image in this figure the object is located in non homogeneous texture and spans different homogeneous region. In the second image the object is located in one homogeneous region. In the third image it's a stochastic texture image. our algorithm gives a very good quality with very short time compaired with Ref. [2] in the same image and size.

Figure 6 and table 2 show another images comparison between the proposed method and Ref.[8] . All of the images have a concave region and our algorithm gives a good result in that type also with fewest time. All result produced in [7] range from 120 to 419 seconds for 192 by 128 images and range from 83 to 158 minutes for 384 by 256 images .

Figure 7 shows the proposed method results and the method of Ref [9] with the same image and size. This method is the closest method to our method in time as shown in table 3.

Figure 8 illustrate the proposed method results and the method of Ref [10] with the same image and size, this method give a better quality than our method in the first image but our method gives a good quality than [10] in the second and third image , table 4 show a comparison in processing time.

At the end our algorithm work well in image completion and texture synthesis with very good result in quality and smallest time than other methods.

5. Conclusions

We have offered a fast image completion and texture synthesis algorithm with a very good quality than other methods, and with no search computational time to give smallest method in time.

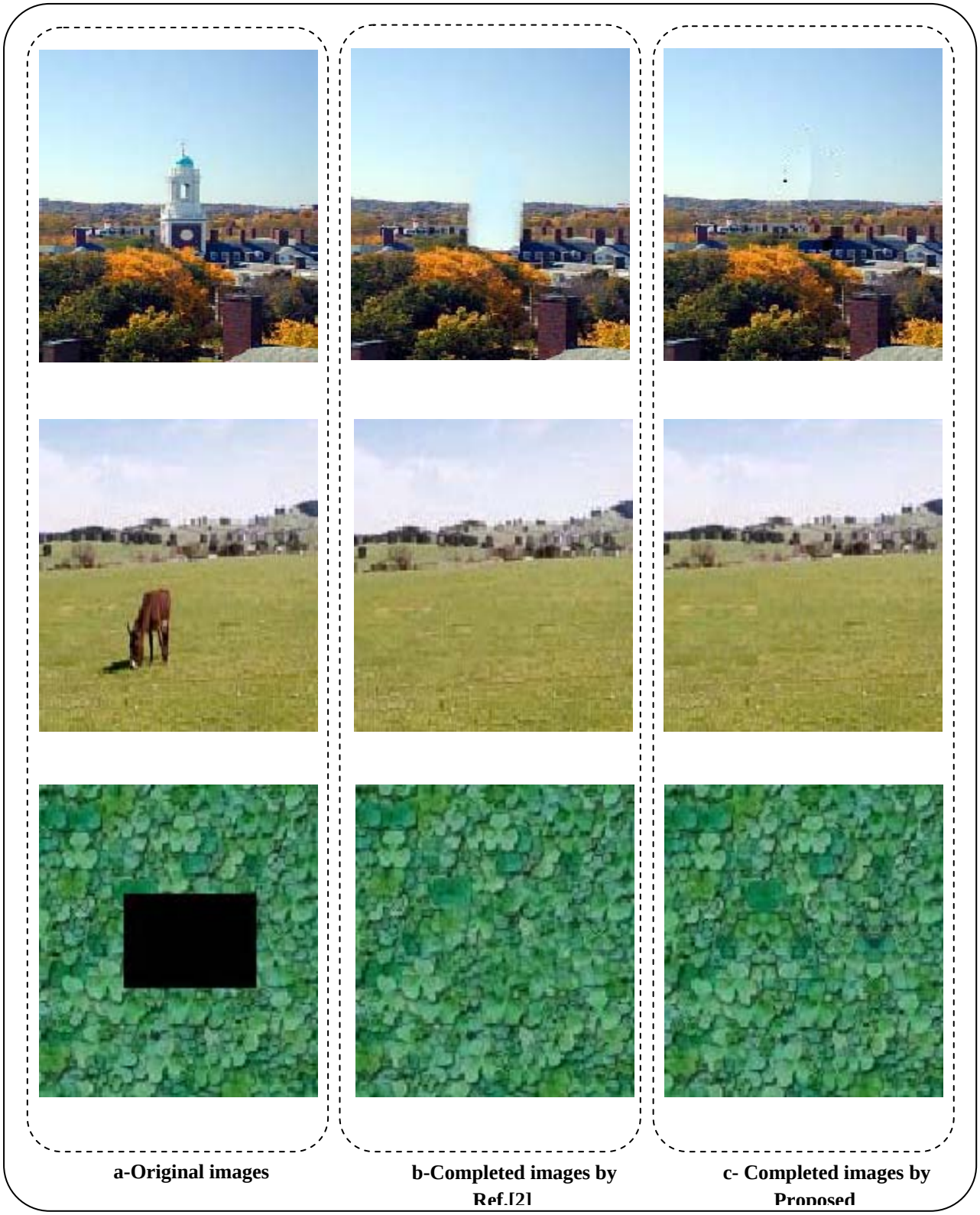


Figure 5 Comparison result of Proposed method with Ref.[2]

a-Original images **b- Completed images by Ref.[7]** **c- Completed images by Proposed**

Figure 6 Comparison result of Proposed method with Ref.[7]

Figure 7 Comparison result of Proposed method with Ref.[9]

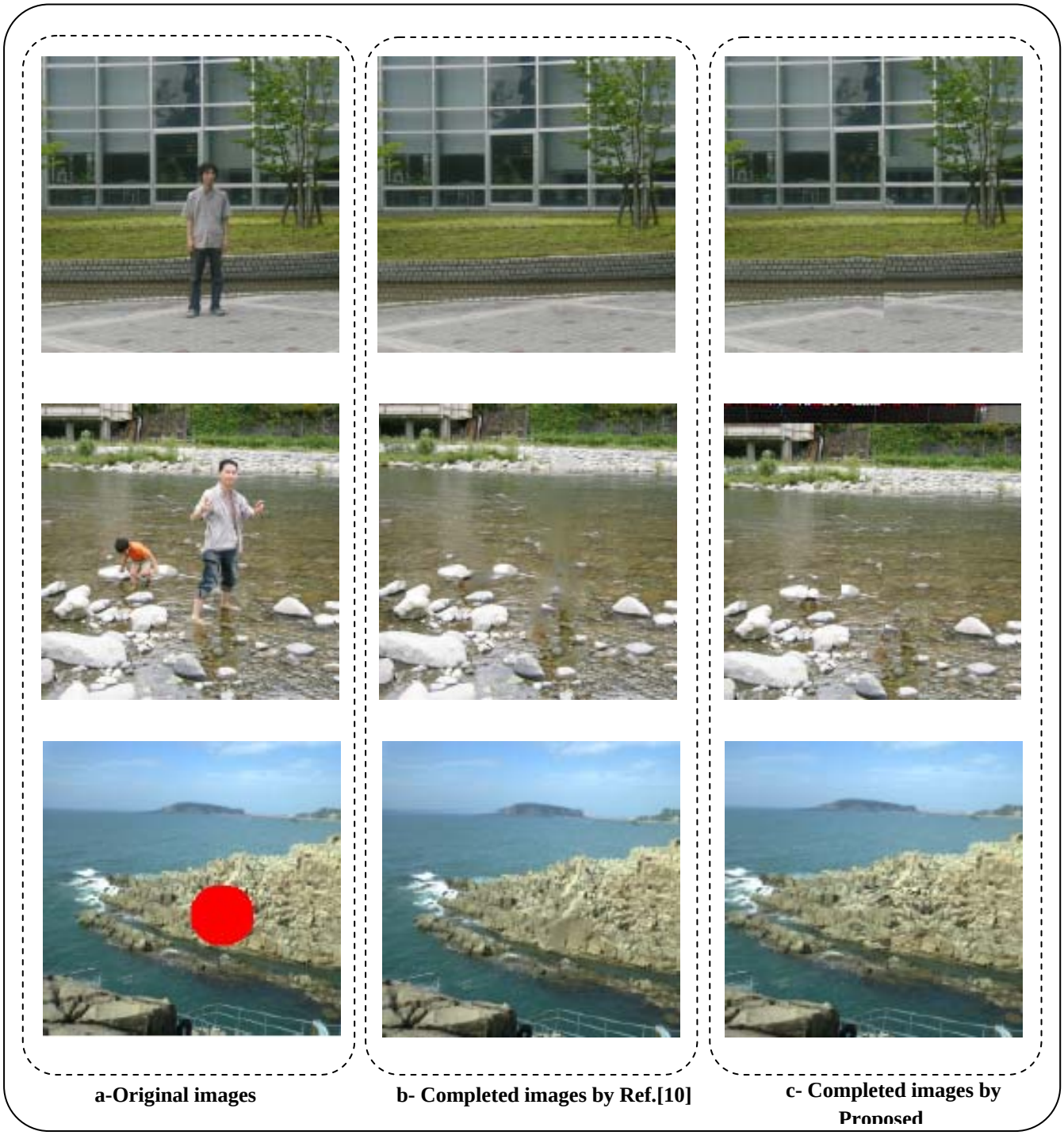


Figure 8 Comparison result of Proposed method with Ref.[10]

6. References

[1] A.telea " An image inpainting technique based on the fast marching method" journal of graphics tools, vol. 9, no. 1, acm press 2004.

[2] Peng Tang "Application of non parametric texture synthesis to image inpainting" thesis, Albuquerue, New Mexico 2004.

[3] S. Zelinka and M. Garland "jump map based interactive texture synthesis " ACM Trans. Graph.,vol. 23, no 4, pp 930-962, 2004.

[4] Işık Barış Fidaner "A Survey on variational image inpainting,texture synthesis and image completion " Bo_gazic¸i University 2007.

[5] A. Efros and T. Leung, "Texture synthesis by non-parametric sampling" in ICCV (2), pp. 1033-1038. 1999

[6] L.-W. Wey and M. Levoy, "Fast texture synthesis using tree-structured vector quantization," in Siggraph 2000, Computer Graphics Proceedings ,K. Akeley, Ed. ACM Press / ACM SIGGRAPH / Addison Wesley Longman, pp. 479-488. 2000.

[7] I. Drori, D. Cohen-Or, and H. Yeshurun, "Fragment-based image completion" in SIGGRAPH ACM SIGGRAPH 2003 Papers. New York, NY, USA: ACM Press, pp. 303-312, 2003.

[8] P. P. A. Criminisi and K. Toyama, "Region filling and object removal by exemplar-based inpainting," IEEE Trans. Image Processing, vol. 13, no. 9, pp. 1200-1212, 2004.

[9] N. Komodakis, "Image completion using global optimization" in CVPR '06: Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. Washington DC,USA IEEE Computer Society, pp. 442-452, 2006.

[10] Norihiko Kawai, Tomokzu and Naokazu "image inpainting considering brightness change and spatial locality of textures" graduate school of information science, nara institute of science and technology ,japan 2008 .

Performance of Encryption Techniques for Real Time Video Streaming

W.S. Elkilani and H.M. Abdul-Kader
Faculty of computers and Information- Menufiya University- Shabin ElKom-Egypt
welkilani@gawab.com , hatem6803@yahoo.com

Abstract *Recently, multimedia security is becoming more important with the continuous increase of digital communications on the internet. Moreover, special and reliable security is needed in many digital applications (such as video conferencing and medical imaging systems). The classical techniques for data security are not appropriate for the current multimedia usage. As a result, we need to develop new security protocols or adapt the available security protocols to be applicable for securing the multimedia applications. Encryption of MPEG-4 video streaming using AES has not been studied. In this paper, the performance of AES in encrypting MPEG-4 video is considered. The performance of AES is compared to two symmetric encryption techniques namely; RC4 and XOR. Three data types (text, audio and video) are used to test the effectiveness of AES in encrypting MPEG-4 . Simulations showed the efficiency of the AES encryption technique in such application for the different data type given.*

Keywords*: Multimedia Security, Video Streaming, AES, MPEG-4*

1. Introduction

The advent of networked multimedia system systems will make continuous media stream. It is very important to secure networked continuous media from potential threats such as hackers , eavesdroppers , etc . The applications for streaming are endless. Streaming can be delivered as a complete video package of linear programming, as a subscription service, or as pay-per-view (PPV). It can form part of an interactive web site or it can be a tool, in its own right, for video preview and film dailies. Some applications are Internet broadcasting (corporate communications), education (viewing lectures and distance learning), web-based channels (IP-TV, Internet radio), Video-on-demand (VOD) and Internet and intranet browsing of content (asset management). Such systems use different types of encryption techniques to increase the security precautions for networked multimedia applications [1][2].

Playing video streams over a network in a real time requires that the transmitted frames are sent in a limited delay. Also, video frames need to be displayed at a certain rate; therefore, sending and receiving encrypted packets must be sent in a certain amount of time in order to utilize the admissible delay. For example: Video on-Demand requires that the video stream needs to be played whenever the receiver asks for it. So, there are no buffer or playback concepts for the video stream (i.e. it runs in real time). Hence, there are many challenges for multimedia security such as:

- The natural size of multimedia data after compression is usually very large, even if using the best available compression techniques. The size of a two-hour MPEG-1 video is about 1 GB.
- Future applications of multimedia need to be run in real time on processes such as video on demand.
- Performance of processing multimedia streams should be acceptable (i.e. bounded by certain value of delay).
- The encryption techniques should be fast enough and require a small overhead in comparison to compression techniques.

The goal of this research is to focus on the following points. First, implementing AES for MPEG-4 in a real time secure video transmitting system. Second, comparing the performance of the AES with respect to two major encryption techniques over a peer to peer channel. Third, evaluating the difference between the overhead resulting from different data types in multimedia (text, audio, and video) due to the three encryption techniques (Xor, RC4, AES).

This paper is organized as follows. In section 2, the basic concepts of video encryption techniques are given. Then in section 3 a brief overview of the previous video encryption methods is discussed. Video streaming quality of services is shown in section 4. In section 5 results of using different encryption techniques are shown. Finally the conclusion is drawn in section 6.

2. Basic Concepts of Video Encryption

The encryption and decryption of a plain text or a video stream can be done in two ways. The first technique is the secret key encryption. The second technique is the public key encryption [3][4]. Public key cryptography is not applicable for secure real time video conferencing because its operations require an amount of time, which is not suitable for video conferencing.

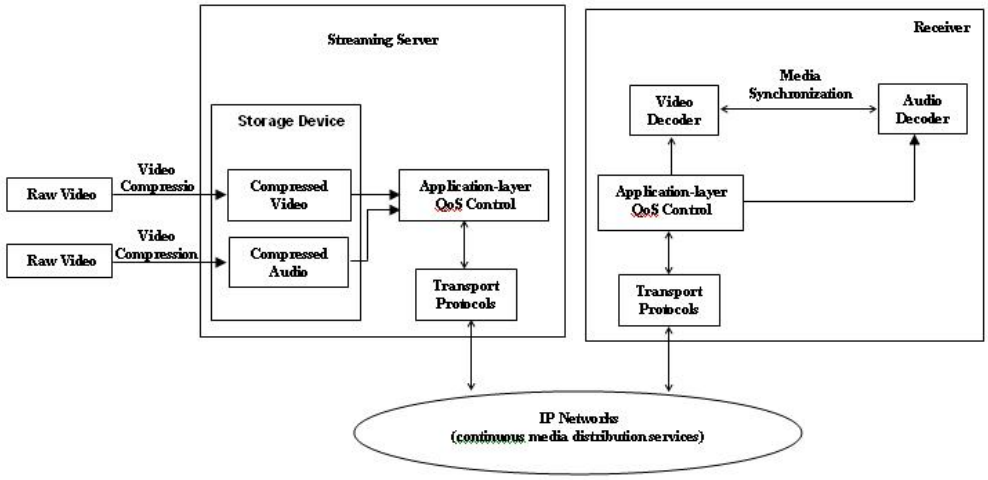


Figure 1. One way data flow block diagram for captured multimedia devices

A video streaming system typically consists of seven building blocks, as illustrated in Figure1. In this figure, raw video and audio data are pre-compressed by video compression and audio compression algorithms and then saved in storage devices. Upon the client's request, a streaming server retrieves compressed video/audio data from storage devices and then the application-layer QoS control module adapts the video/audio bit-streams according to the network status and QoS requirements. After the adaptation, the transport protocols packetize the compressed bit-streams and send the video/audio packets to the Internet IP networks. Packets may be dropped or experience excessive delay inside the Internet due to congestion; on wireless IP segments, packets may be damaged by bit errors. To improve the quality of video/audio transmission, continuous media distribution services are deployed in the Internet. For packets that are successfully delivered to the receiver, they are first pass through the transport layers and then are processed by the application layer before being decoded at the video/audio decoder. To achieve synchronization between video and audio presentations, media synchronization mechanisms are required [5].

There are many video encryption algorithms. Such encryption techniques can be classified as follows: naive algorithm, selective algorithm, Zig-Zag algorithm, RC4 and AES [6]. A review for each one is briefly given. The idea of naive encryption [3] is to deal with the video streams as text data. The simplest way to encrypt video streams is to encrypt every byte. Naive algorithm encrypts every byte in the whole video stream. Naive algorithms guarantee the most security level. However, it is not an applicable solution if the size of the data is large. Due to encryption operations, the time delay increases and the overhead will not be satisfactory for the real time video encryptions.

In selective algorithm [4], four levels of selective algorithms are suggested. These four levels are encrypting all headers, encrypting all headers and I (initial) frames, encrypting all I frames and all I blocks in P and B frames, and finally encrypting all frames as in Naïve algorithm to guarantee the highest security. The idea of ZIG-ZAG algorithm [4] is basically encrypting the video streams before compressing them. Explicitly, when mapping the 8x8 block to a 1x64 vector each time in the same order. We can use a random permutation to map this transformation of the 8x8 block to the 1x64 vector. Therefore, the concept of the encryption key does not exist in the ZIG-ZAG permutation algorithms. Once the permutation list is known, the algorithm will not be secure any longer.

A new video encryption algorithm called VEA that depends on dividing the video streams into chunks. These chunks are separated into two different lists (odd and even lists). Afterward, applying an encryption algorithm like DES to the even list and the final cipher is concatenation of output of encryption algorithm XOR with the odd list streams [5-6]. RC4 is Stream cipher structure in which it encrypts plain text one byte at a time with variable length key size from 1 to 256 bytes (8 to 2048) . It is a symmetric encryption algorithm in which the same key is

used for encryption and decryption .The algorithm is based on the use of random permutation. RC4 is the most widely used stream cipher. It is used in the SSL/TLS (Secure Socket Layer/Transport Layer Security) standards that have been defined for communication between web browsers and servers. It consists basically of two main operations, namely key Setup operation and ciphering operation. In the first operation RC4 generates a pseudorandom stream of bits (a "keystream") then applying some kind of operation on key such as permutation and expansion so as to be more randomized [3]. While in the second operation the plaintext is Xored with the key. The basic operation and sequence of RC4 is shown in figure 2. Further details about this algorithm can be found in [6].

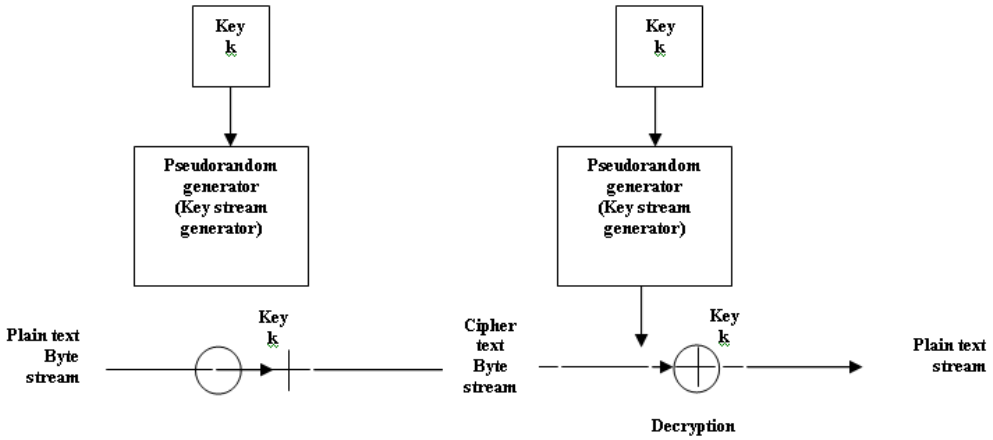


Figure 2. Basic architecture of RC4

The AES algorithm is essentially Rijndael [7] symmetric key cryptosystem that processes 128-bit data blocks using cipher keys with lengths of 128, 192, or 256 bits. Rijndael is more scalable and can handle different key sizes and data block sizes, however they are not included in the standard. Also the basic blocks of AES operation is shown in figure 3. Further details about this algorithm can be found in [8].

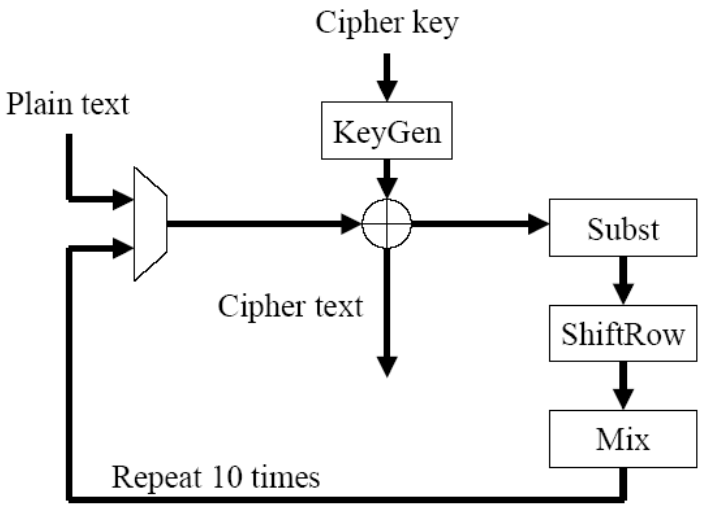


Figure 3. Basic architecture of AES

3. Previous Work in Video Encryption

Some proposed attempts to secure MPEG streams have been reported. The most straightforward method is to encrypt the entire MPEG stream using standard encryption methods. In fact, they have used the naive algorithm approach [8]. The greatest concern about this approach is the speed of processing due to the large size of MPEG files. Another method to secure MPEG streams is the selective encryption algorithm which encrypts only the I-frame of MPEG streams [9,10]. Meyer and Gadegast [11] have designed a new MPEG-like bit-stream SECMPEG that

incorporates selective encryption and additional header information, and has high-speed software execution. SECMPEG can use both DES and RSA and implements four levels of security: 1st level— encrypts all headers. 2nd level — encrypts all headers plus the DC and lower AC terms of the I-blocks. 3rd level — encrypts I frames and all I-blocks in P and B frames. 4th level — encrypts all data. SECMPEG is not compatible with standard MPEG. A special encoder/decoder would be required to view unencrypted SECMPEG streams. A proposal targeting at integration of compression and encryption of MPEG streams into one step is presented in [12] using the "ZigZag-Permutation Algorithm", where the basic idea is to use a random permutation list to replace the zig-zag order to map the individual 8x8 block to a 1x64 vector.

- Salah [13] studied performance of encryption and decryption algorithms such as AES for real time video streams. He adapted AES and XOR algorithms to be used with JPEG, H261, CellB, and MPEG-1/2 video encoders and decoders. He attempted to select specific frames to encrypt. The encrypted video streams are combinations of I, P, and B frames. In [14], four fast MPEG video encryption algorithms are presented. These algorithms are based on the DES [3] by using a secret key to randomly change the sign bits of Discrete Cosine Transform (DCT) coefficients and/or the sign bits of motion vectors. The encryption is accomplished by the inverse DCT (IDCT) during the MPEG video decompression processing. These algorithms add a small overhead to the MPEG codec. As can be noticed that the previous authors haven't studied the performance of the AES in encrypting MPEG-4 video streaming. Moreover, most studies haven't used peer to peer platforms to transfer video stream which has gained more interest in recent decayed due its wide application spectrum.

4. Video Streaming Quality of Service

In this section, we will show the parameters used to measure the quality of an encryption technique. QoS refers to the ability of a network to provide better service to selected network traffic over various underlying technologies. The main QoS features that provide better and more predictable network service can be summarized in the following:

- Algorithm setup time (Ts): Similar to key setup time, the algorithm setup time reports the minimum amount of time before an algorithm is ready to process data. Time to create look-up tables, etc. will fall in this category. None of the evaluated algorithms contained an algorithm setup time greater than zero.
- Time to encrypt one block (Te): This parameter will address minimum latency times for each of the algorithm submissions. The time to encrypt one block , measured in nanoseconds, is a function of two parameters: the worst-case path delay between any two registers, and the number of rounds in the algorithm.
- Time to decrypt one block (Td): As above, this parameter will address minimum latency times for each of the algorithm submissions. Decryption does not always require identical processing as encryption. Therefore, the time required to decrypt one block is reported.
- Time to switch keys (Ts): Originally, this parameter was included as a measure to encompass both key setup time and algorithm setup time overhead. However, since none of the evaluated algorithms contained an algorithm setup time, this parameter is identical to key setup time. Therefore, it will not be reported further in this document.

We can assume that the time delay T represents the summation of the previous time delays (T=Ts+ Te + Td +Ts)

5. Results

We have used the Windows machines with Intel® Celeron CPU 3 GHz, 2 MB of RAM in our experiments .In addition, we used 300SC-Y web camera to capture the video frames. For the video transmission, we used UDP transmission protocol to send and receive the video packets through the network channel. Visual C++ 6.0 programming language has been used since it has many advantages with the network programming. In addition, we modified the standard AES and XOR codes to encrypt different lengths of video streams. We developed our final code in some functions that handle the encryption operations. We selected a fixed key length for AES, RC4 and XOR encryption algorithms. Fortunately, AES helps us to encrypt directly 128 bits of a video stream, which makes the computation fast in comparison to their work and finally, we examined the effect of encrypting the whole length of multimedia packets.

Since the QoS is very important in multimedia networking, we have measured our system performance based on the delay where it is visible slightly in the transmission and reception of data. We will measure the delay in encrypting number of text, audio and video MPEG-4 packets for the three algorithms. The calculations are based on the difference between the start of transmission and the reception time for 100000 packets (15 byte each packet). Figure 4 shows the time for different encryption algorithms (XOR, RC4 and AES) for the MPEG-4 text data type.

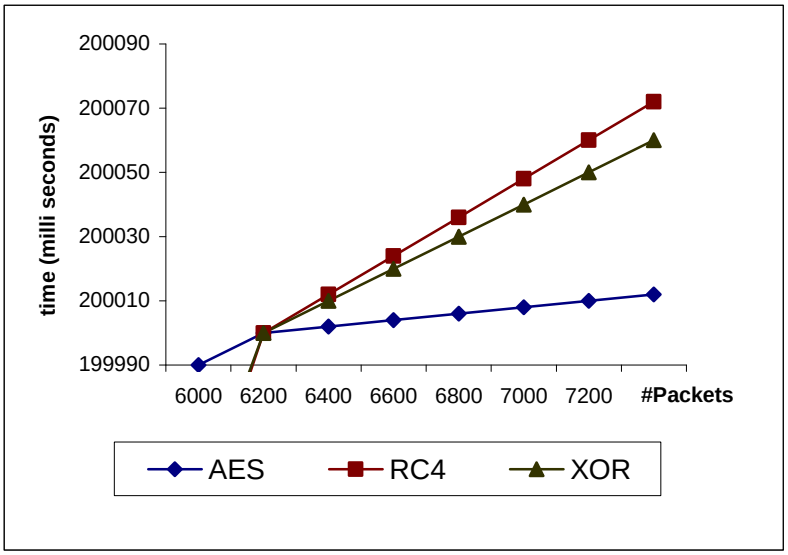


Figure 4. Time delay T for TEXT using XOR, RC4 and AES Encryption Algorithms

For audio, we have measured Te to encrypt 500 packets, 600 byte each. Figure 5 shows the time for different encryption algorithms (XOR, RC4 and AES) for the MPEG-4 audio.

.

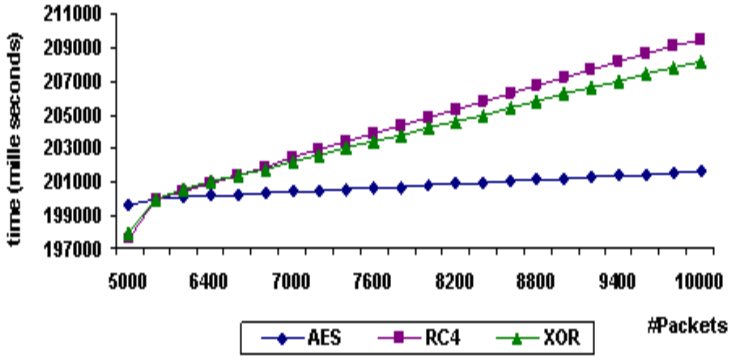


Figure 5. Time delay T for AUDIO using XOR, RC4 and AES Encryption Algorithms

For video, we have measured the time (Te) to encrypt 500 packets, 2464 byte each. Figure 6 shows the time for different encryption algorithms (XOR, RC4 and AES) for the MPEG-4 video

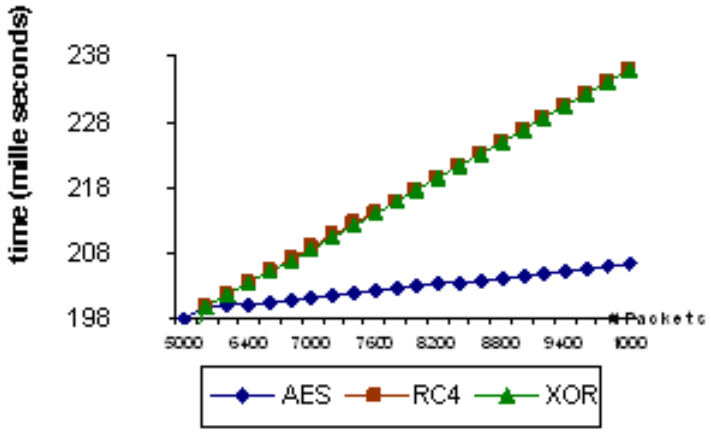


Figure 6. Time delay T for VIDEO XOR, RC4 and AES Encryption Algorithms

As shown in figures 4, 5 and 6, the overhead time of encrypted packets using AES is less than the overhead time of the encrypted packet using RC4 and XOR. Also the overhead time of encrypted packets of type text is less than the overhead time of the encrypted packet of type audio then video. So we use 100000 packet of type text each of size data 15 byte to be sensible of low rate transmission, then audio 500 packet of size 600 byte, then video 500 packet of size 2464 byte and very high rate to avoid video flickering which cause very high overload. From these Figures, the relative time spent for the encryption operation using AES does not negatively affect video stream transmission. Second this is acceptable for video transmissions. In conclusion, encrypting MPEG-4 video streams using AES is an applicable solution to secure real time video transmission. Based on the above results and the criteria which differentiate between different encryption techniques we can summarize the result as shown in table 1

Algorithm	Cost	I/O	T to Encrypt B	T to Decrypt B
XOR	Low	Fixed	Stream Cipher	Stream Cipher
RC4	Medium	Fixed	Stream Cipher	Stream Cipher
AES	Very High	Fixed	0.01ms (depend on the block size)	0.01ms (depend on the block size)

Finally, a comparison between the selected encryption algorithms is conducted from the view of safe time. The result of this comparison is shown in figure 7. This figure indicates the great difference between AES and other algorithms. This implies that AES can be consider the best one from the point of safe time.

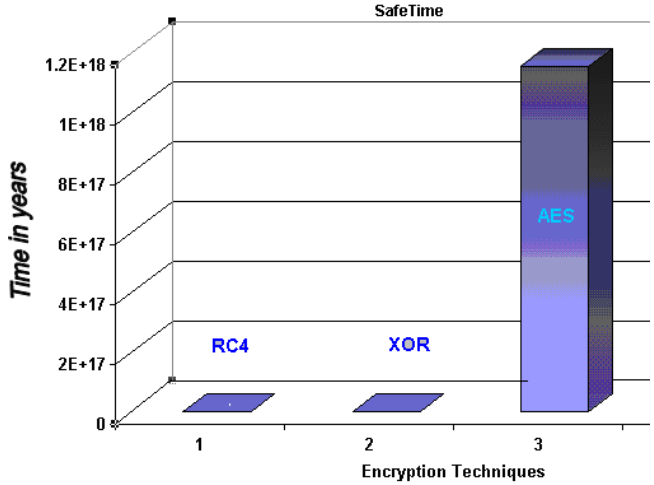


Figure 7. Encryption techniques Safe time for

6. Conclusions

Our study showed that the AES encryption algorithm can be used effectively to encrypt MPEG-4. The performance of AES encryption frames is sufficient to display the received frames on time. The encryptions delay overhead using AES is less than the overhead using RC4 and XOR algorithm. In addition, AES can achieve satisfactory encryption results with little overhead. Therefore, we conclude that using AES in encrypting MPEG-4 is a feasible solution to secure real time video transmissions.

7. References

[1] IEEE Transactions on Circuits and Systems for Video Technology: Special Issue on Authentication, Copyright Protection, and Information Hiding, Vol. 13, No. 8, August 2003.

[2] X. Liu and A.M. Eskicioglu, "Selective Encryption of Multimedia Content in Distribution Networks: Challenges and New Directions", IASTED International Conference on Communications, Internet and Information Technology (CIIT 2003), Scottsdale, AZ, November 17-19, 2003.

[3] D. R. Stinson, "Cryptography Theory and Practice," CRC Press, Inc., 2002.

[4] William Stallings, "Cryptography and Network Security, Principles and Practice", Pearson education, Third Edition, 2005.

[5] Chun-Shien L, "Multimedia Security Steganography and Digital Watermarking Techniques for Protection of Intellectual Property", Idea Group Publishing 2005.

[6] T. Seidel, D. Socek, and M. Sramka, "Cryptanalysis of Video Encryption Algorithms" , Proceedings of The 3rd Central European Conference on Cryptology TATRACRYPT 2003,

[7] B. Gladman, "A Specification for Rijndael, the AES Algorithm," (http://fp.gladman.plus.com/, 2001).

[8] I. Agi and L. Gong, "An Empirical Study of Mpeg Video Transmissions," In Proceedings of the Internet Society Symposium on Network and Distributed System Security, pages 137-144, San Diego, CA, February 1996.

[9] Y. Li, Z. Chen, S. Tan, and R. Campbell, "Security enhanced mpeg player", In Proceedings of IEEE First International Workshop on Multimedia Software Development (MMSD'96), Berlin, Germany, March 1996.

[10] T. B. Maples and G.A. Spanos, "Performance Study of a Selective Encryption Scheme for the Security of Networked Real-time Video", In Proceedings of lath International Conference on Computer Communications and Networks, Las Vegas, Nevada, September 1995.

[11] J. Meyer and F. Gadegast, "Security Mechanisms for Multimedia Data with the Example mpeg-1 Video", Available on WWW via http://www.powerweb.de/ phade/phade.htm/, 1995.

[12] L. Tang, "Methods for Encrypting and Decrypting MPEG Video Data Efficiently", In Proceedings of The Fourth ACM International Multimedia Conference (ACM Multimedia'96), pages 219-230, Boston, MA, November 1996.

[13] Salah Aly, "A Light-Weight Encrypting For Real Time Video Transmission", TR04-002, College of computing and digital media, Depaul university, August 2004 (http://facweb.cs.depaul.edu/research/TechReports/ TR04-002.pdf)

[14] B. Shi, W. Changgui and S. Wang, "MPEG Video Encryption Algorithms", Multimedia Tools and Applications, Vol. 24, Issue: 1, pp. 57-79, September 2004.

MODSBR: An Enhanced Methodology for Defending Byzantine Attacks on Mobile Ad Hoc Networks

Safaa Saad-Eldeen Ahmed
Faculty of computers and Information, Information Technology Department
safee_2000@hotmail.com

Wail Shawky Elkilani
Faculty of computers and Information, Information Technology Department
welkilani@gawab.com

Mohiy Mohammed Hadhoud
Faculty of computers and Information, Information Technology Department
mmhadhoud@yahoo.com

Abstract: *Byzantine attacks signify a risk to Mobile Ad Hoc Networks (MANETs). One of the efficient routing protocols used to mitigate such attacks in MANETs is the On-Demand Secure Byzantine Routing protocol (ODSBR). In this paper, we present the Modified ODSBR (MODSBR) where several enhancements for the performance and security of ODSBR are implemented. The first of these enhancements is enabling route caching in an elegant way that causes a notable decrease in the routing overhead. The central clustered link weight management (CLWM) is a secure technique utilized by MODSBR to detect malicious nodes through a set of management nodes (MN). The effect of the proposed enhancements is evaluated with respect to different types of Byzantine attacks. Simulation experiments show that MODSBR outperforms ODSBR by an average 23% decrease in routing overhead and a 10 % increase in the delivery ratio for different mobility values.*

Keywords: Secure routing; Caching; Mobile Ad Hoc Networks.

1. Introduction

In ad hoc networks, devices rely on each other to keep the network connected. Thus, unlike traditional wireless solutions, such networks do not require any pre-existent (fixed) infrastructure, which minimize their cost and deployment time. Routing protocols enable multi-hop communications in ad hoc networks. To achieve availability, routing protocols should be robust against both topology changes and malicious attacks.

Existing protocol specifications cope well with the change of network topologies. However, defense against malicious attacks has remained optional. Nowadays, the trend is changing and there is an increasing interest on research focused on the provision of proposals for securing ad hoc routing protocols.

Attacks where the adversary has full control of an authenticated device and can perform arbitrary behavior to disrupt the system is called, Byzantine attacks. From a more general perspective, a Byzantine attack is any attack that involves the leaking of authentication secrets so that an adversarial device is indistinguishable from a legitimate one. Significant research in securing wired [7, 4, 13] or ad hoc wireless [10, 9, 17, 16] routing protocols focused on this aspect. In this work, only Byzantine attacks were considered. It is believed that they represent type of attacks that are likely to be mounted against ad hoc wireless routing protocols. And they cover a wide range of adversarial strengths. Individual techniques are proposed [14, 11, 12, 1, 10] to mitigate each type of these attacks. A few research efforts are done for a global prevention solution.

Black hole Attack: where the adversary stops forwarding data packets, but still participates in the routing protocol correctly. As a result, whenever the adversarial node is selected as part of a path by the routing protocol, it prevents communication on that path from taking place. Several techniques exist which attempt to mitigate the effect of black hole attacks on network performance. One of these

methods is Watchdog and Pathrater [14]. The approach has two components, watchdog, a service that is run by each node and monitors the node's neighbors, and pathrater, a service that ensures that adversarial nodes are avoided when selecting future routes. An alternate method for avoiding black hole attacks is the Secure Data Transmission (SDT) protocol [15]. SDT uses authenticated end-to-end acknowledgments from the final destination, providing proof that the packets reached their destination.

Flood Rushing Attack: exploits the flood duplicate suppression technique used by many routing protocols. This attack takes place during the propagation of a legitimate flood and can be seen as a "race" between the legitimate flood and the adversarial variant of it. If an adversary successfully reaches some of its neighbors with its own version of the flood packet before they receive a version through a legitimate route, then those nodes will ignore the legitimate version and will propagate the adversarial version. This may result in the continual inability to establish an adversarial-free route, even when authentication techniques are used. Previous work in addressing the rushing attack is scarce, we are only aware of Rushing Attack Prevention (RAP) [12]. The intuition in this work is that the rushing attack can be prevented by waiting (up to a time limit □) to receive up to □□requests (flood re-broadcasts) and then randomly selecting one to forward rather than always forwarding only the first one.

Byzantine Wormhole Attack: where two adversaries collude by tunneling packets between each other in order to create a shortcut (or wormhole) in the network. This tunnel can be created either using a private communication channel, such as a pair of radios and directional antennas, or by using the existing ad hoc network infrastructure. The adversaries can send a route request and discover a route across the ad hoc network, then tunnel packets through the non-adversarial nodes to execute the attack. . A mechanism called Packet Leashes for preventing wormholes by limiting the transmission distance of a link is proposed in [11]. And also Directional Antenna is a more recent method for preventing wormholes by using the angle of arrival information available when using directional antennas [8].

Byzantine Overlay Network Wormhole Attack: A more general variant of the previous attack occurs when several nodes are compromised and form an overlay network. By tunneling packets through the overlay network, the adversaries make it appear to the routing protocol that they are all neighbors, which considerably increases their chances of being selected on routes. The same prevention techniques of preventing the wormhole attack are utilized.

ODSBR [1-3] routing protocol is a very effective secure on-demand routing protocol that is resilient to Byzantine failures caused by individuals or colluding nodes. An adaptive probing technique is used that detects a malicious link after log n faults occurred, where n is the length of the path. These links are then avoided by multiplicatively increasing their weights and by using an on-demand route discovery protocol that finds a least weight path to the destination. Disabling the route caching property is a factor that causes decrease in the performance of the ODSBR protocol. We believe that techniques taking advantage of route caching will enhance the performance. Also, one of the disadvantages of this protocol is that when any node detects a link that caused failure, it doesn't tell other nodes about this unreliable link. So a technique taking care of this aspect would increase the security. Implementation details are also discussed, as well as changes to the original protocol motivated by practical considerations. The rest of the paper is organized as follows. Section 2 describes the caching technique and its implementation. The central clustered link weight management technique is presented in section 3. Section 4 presents analysis of the simulation results. We conclude the work in Section 5.

2. Applying Route Caching Mechanism to ODSBR Routing Protocol

Due to power limitation each station has a fixed range. It also acts as a router, relaying data packets for other stations to their final destination. One of the main challenges in the design of ad hoc networks is the routing protocol upon a dynamically changing topology, node energy constraints and the properties of the wireless channel. On-demand routing protocol that is generally used in ad hoc networks is one that searches for a route to a destination node when a sending node originates a data packet addressed to the destination node. Every on-demand routing protocol has to maintain some form of routing cache with the intention of avoiding route re-discoveries for each separate data packet and reduction of routing overhead. Additionally, the route cache is not only used to cache routes for the purpose of originating packets, but also for the purpose of allowing nodes to answer Route Requests targeted at other node. Therefore, caching is an essential component of on-demand routing protocol for wireless ad hoc mobile networks.

ODSBR routing protocol is a very effective secure on-demand routing protocol that is resilient to Byzantine failures. But disabling the route caching property is a factor that causes degradation in the performance of the ODSBR protocol. In order to enhance the performance, we investigated ways of taking advantage of route caching.

In the original ODSBR protocol, as shown in Figure 1, when a node receives a route_request it will check if this is a new request or repeated one. If the request is repeated, it will be discarded. Otherwise the signature will be verified. The node will broadcast a route reply for verified signatures if the requested destination ID is the same ID of this node, otherwise the route request will be broadcasted.

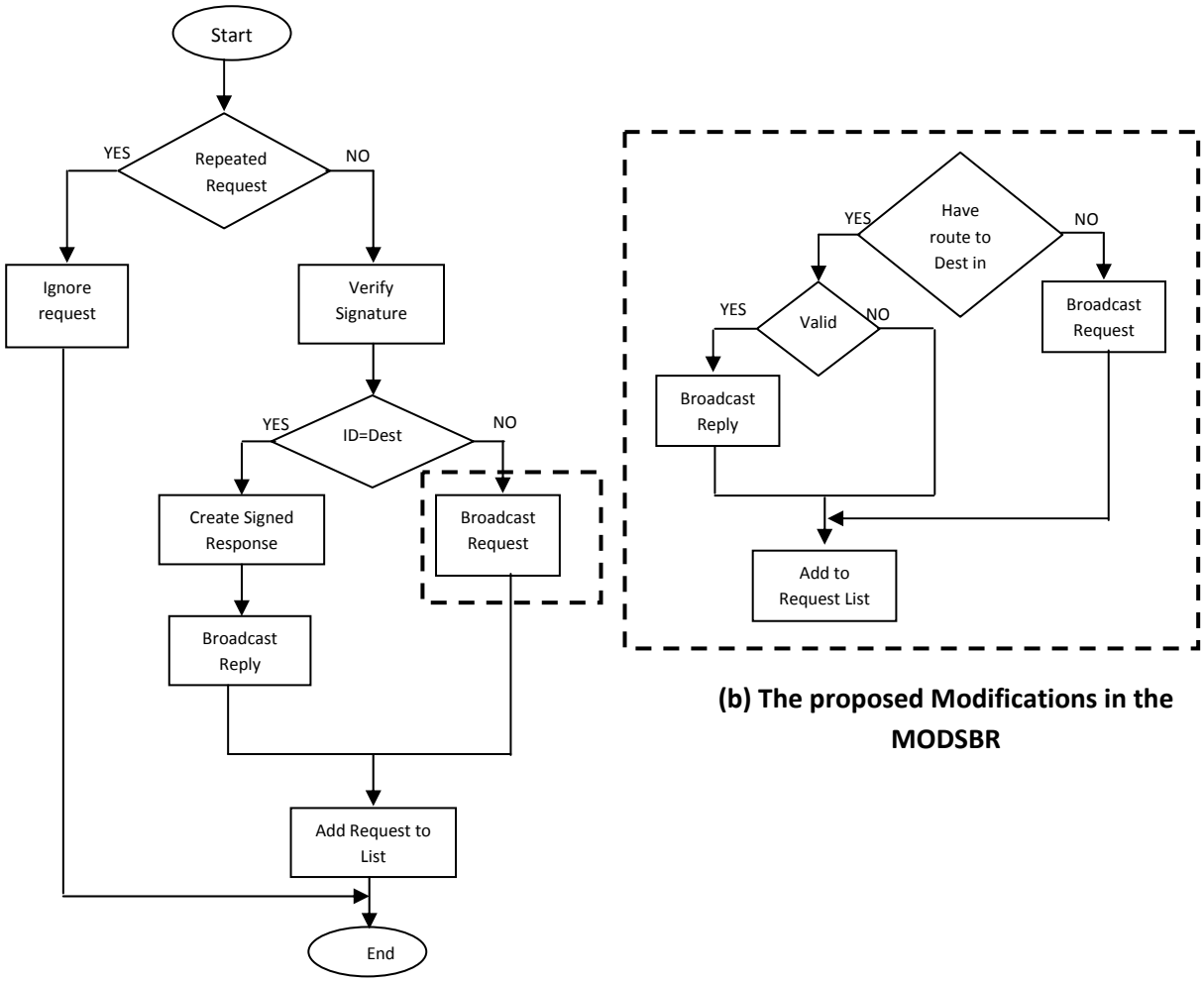
After applying the caching technique the ODSBR will behave as follows: when a node receives a route_request it will check if this is new request or repeated. If the request is repeated, it will be discarded. Otherwise the signature will be verified. Then if the requested destination ID is the same ID of this node, request will be broadcasted for absent routes in the cache. On the other hand, the present route will be validated. Reply will be broadcasted for valid routes.

3. Central Clustered Link Weight Management

In the original ODSBR, every node has its own link weight table. When any node wants to select the best route to a specific destination after a route discovery operation, it will compute the total weight for each path received in a route reply message and then select the path with the smallest weight value. When the value of the loss rate at any node becomes greater than 10%, the node will start the binary search to detect the link that caused loss in packets. When the node finds the lossy link, it will increment the weight for this link in the weight table specialized for this node.

On the other hand, instead of making each node to have a link weight table and when any table has been updated, no way to any node to know about this update. So, a set of nodes called Management nodes (MN) had been specified to store a general weight table, where any node can use this table during computing the total weight to any path. Also, when any node detects a link that caused loss of data, it will increment the weight value of this link in the nearest general weight table. And so on, each node makes updates for links` weights in the general link weight table and also uses this table for selecting the best path.

Normally the number of management nodes is significantly less than the total number of nodes (n). We have found that number of MN giving best performance will be about 20% of the nodes. They are basically normal nodes except that they carry the general management tables and able to be accessed by more than one node in the same time.



(a) ODSBR Behavior when node receives a request

(b) The proposed Modifications in the MODSBR

Figure 1. The Route Request Behavior

We have slowed down the movement of the management nodes such that the same set of nodes are served during simulation. Figure 2 shows the algorithm of the central clustered link weight management technique.

It's assumed that every node has an attribute defined as management flag. For nodes that work as management nodes this flag is set to "1" and nodes that work as non management nodes is set to "0". Each node configures this flag offline before entering the network. Also it's not allowed that all nodes work as non Management nodes, there must be some of nodes to work as Management nodes.

Initialize all weight of nodes with zero in all Management nodes

-*When any node wants to compute the total weight for a specific path*

- Define **list** as an array.
- Add all nodes in the path to **list**.
- Request the weight of the nodes involved in the **list** by sending **weight_request** packet attached with the **list** array to the nearest MN.

-*When any node detects a malicious node that caused failure*

- Increment the weight value of this node causing failure.
- Send an **update_packet** attached with the link new weight value to the nearest Management node, notice that nearest MN is detected by hello packets.

-*When any MN receives an **update_packet***

- Clear the old weight of this link.
- Set the weight of this link to the new weight.

-*When any MN receives a weight request for nodes in the list array*

- Receive the list of links.
- Define weight_list as an array.
- For each item in list array
  - Search for the weight value.
  - Add this weight to the **weight_list** array.
- Send a **weight_reply** packet attached with the **weight_list** array.

Notice: - that detection will be fulfilled as implemented in the ODSBR

Figure 2. An algorithm for central link weight management technique

4. Results

Simulations were conducted using the NS2 [18] network simulator. Nodes in the network were configured to use 802.11 radios with a bandwidth of 2 Mbps and a nominal range of 250 m. All the simulated routing protocols were configured with their default parameters. The simulations were conducted by randomly placing 50 nodes within a 1000 by 1000 meter square area. In addition to these 50 nodes, 0 to 10 adversarial nodes were added to the simulations, depending on the considered attack configuration. A traffic load of 10 constant bit rate (CBR) flows was used to simulate data communication through the ad hoc network. An aggregate load of 0.1 Mbps was offered to the network by having each flow send 256 byte packets at approximately 4.9 packets per second. The simulation time was 300 seconds for each simulation and the results were averaged over 30 random seeds. The next subsections will show the result of applying the MODSBR to defend different Byzantine attacks.

5. The Byzantine Black hole Attack

The delivery ratio was evaluated by using as a baseline the case where no black holes exist in the network. The number of adversarial nodes was then increased in the network and the effect on the delivery ratio was evaluated. The adversarial nodes were placed randomly within the simulation area. Figure 3 shows the delivery ratio of the AODV [5, 6], the ODSBR, and modified ODSBR protocols as a function of the number of adversarial nodes, for different levels of mobility. It can be noticed that at 0 m/s and 1 m/s speeds the MODSBR outperform ODSBR and AODV. Also the MODSBR increased the delivery ratio over 90% at high mobility.

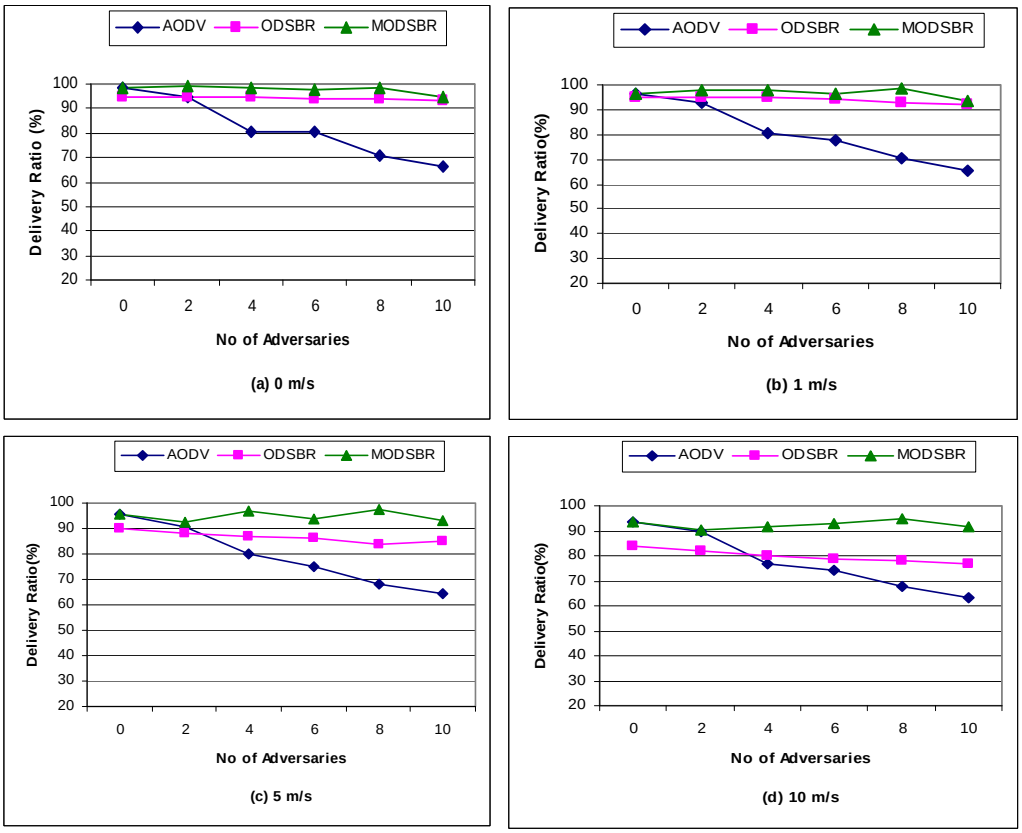


Figure 3. The effect of the Black Hole Attack on the delivery ratio of AODV, ODSBR, and MODSBR for different speeds

To compare between the performance of the MODSBR and ODSBR, we have used % delivery ratio error defined as %DRE = (DRn – DRo) / DRo where DRn is the delivery ratio for the ODSBR or the MODSBR. And DRo is the delivery ratio for the AODV. Table 1 compares between the ODSBR & MODSBR using the %DRE after applying the black hole attack. It can be noticed that ODSBR couldn't improve delivery ratio in some cases due to high mobility while the MODSBR improved the delivery ratio more than the ODSBR for high and low mobility.

	ODSBR				MODSBR			
No of adversaries	**0 m/s**	**1 m/s**	**5 m/s**	**10 m/s**	**0 m/s**	**1 m/s**	**5 m/s**	**10 m/s**
2	0.00	0.03	-0.03	-0.08	0.05	0.05	0.03	0.01
4	0.18	0.18	0.09	0.04	0.23	0.22	0.21	0.19
6	0.16	0.21	0.15	0.07	0.21	0.24	0.25	0.26
8	0.32	0.32	0.23	0.15	0.39	0.40	0.43	0.39
10	0.40	0.40	0.32	0.22	0.43	0.42	0.44	0.45

Table 1. %DRE for ODSBR & MODSBR in the presence of the Black Hole Attack

Simulations were conducted to evaluate the impact of flood rushing on the effectiveness of a black hole attack. Figure 4 shows the delivery ratio of the AODV, ODSBR and the MODSBR protocols as a function of the number of adversarial nodes, for different mobility values in the presence of both the black hole attack and the flood rushing attack. As shown in the figures the impact of flood rushing on MODSBR is almost unnoticeable.

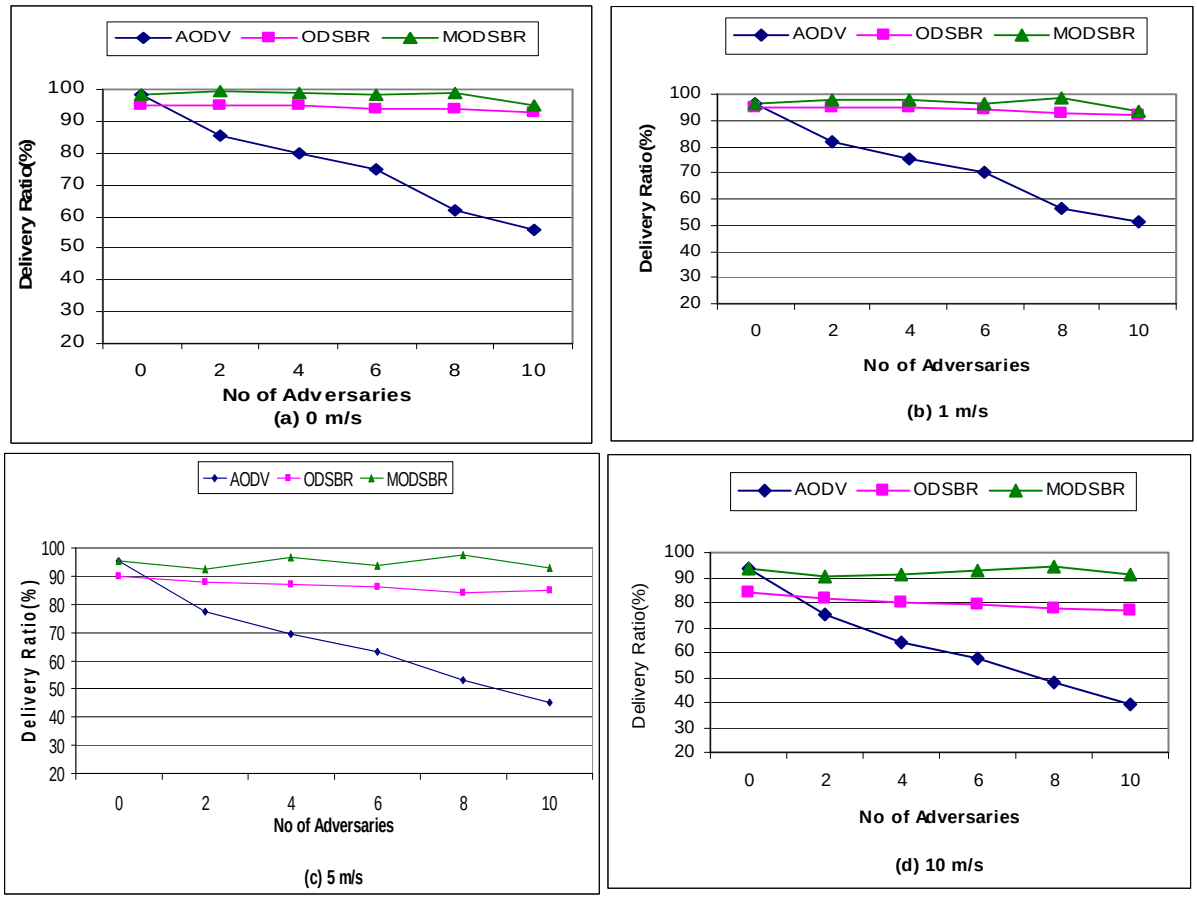


Figure 4. The effect of the Black Hole Attack combined with flood rushing Attack on the delivery ratio of AODV, ODSBR, and MODSBR for different speeds

Table 2 compares between the ODSBR & MODSBR using the %DRE after applying the black hole attack in combination with flood rushing attack. We can notice that in spite of increasing the no of adversaries the MODSBR could improve the delivery ratio.

	ODSBR				MODSBR			
No of adversaries	**0 m/s**	**1 m/s**	**5 m/s**	**10 m/s**	**0 m/s**	**1 m/s**	**5 m/s**	**10 m/s**
2	0.11	0.16	0.14	0.09	0.16	0.20	0.20	0.20
4	0.19	0.27	0.25	0.24	0.24	0.30	0.39	0.42
6	0.26	0.34	0.36	0.37	0.31	0.38	0.48	0.61
8	0.52	0.64	0.58	0.63	0.59	0.74	0.84	0.97
10	0.66	0.80	0.89	0.95	0.70	0.83	1.06	1.32

Table 2. %DRE for ODSBR & MODSBR in the presence of the Black Hole Attack & Flood Rushing.

5.1. Byzantine Wormhole Attacks

The wormhole attack can be implemented by three forms central wormhole, cross of death wormhole, and the random placement attack. In the case of central wormhole configuration only two adversaries placed at coordinates (300,500) and (700,500) in the 1000 x 1000 simulation area as shown in figure 5. In case of cross of death configuration four adversaries are placed at coordinates (200,500), (800,500), (500,200), (500,800). They form two wormholes, in the shape of a cross (see figure 6). In the last configuration a set of wormholes is randomly placed in the network. In all cases, we have first evaluated the effect of the wormhole attack on the delivery ratio. We then combined the wormhole with flood rushing and examined the impact of the combined attack.

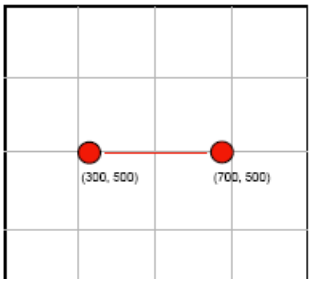


Figure 5. Central Wormhole

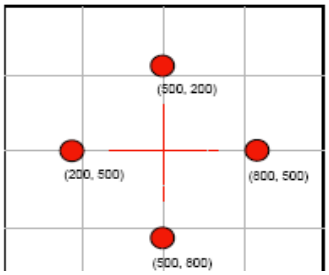


Figure 6. Cross of Death

Figure 7 shows the Delivery ratio for AODV, ODSBR, and MODSBR after applying each form of the wormhole attack. As noticed in case of central wormhole (Figure 7.a) the delivery ratio decreased in low mobility in case of the MODSBR more than the ODSBR. As shown in Figure 7.a that for low mobility the delivery ratio in case of the MODSBR is less than the ODSBR. For high mobility, the DR of the ODSBR has decreased about 20%, while that of the MODSR is nearly constant and stable. The same discussion can be drawn for the cross of death attack as shown in Figure 7.b where the constant value of the MODSBR can be noticed while deterioration in the performance of the ODSBR can be observed. This can be explained by that because the node still under the effect of the attackers in low mobility and the attacker may prevent it from connecting to the management nodes around it. But in high mobility it can leave the area of attacker. For the random placement wormhole (Figure 7.c), the MODSBR performs slightly better than the ODSBR.

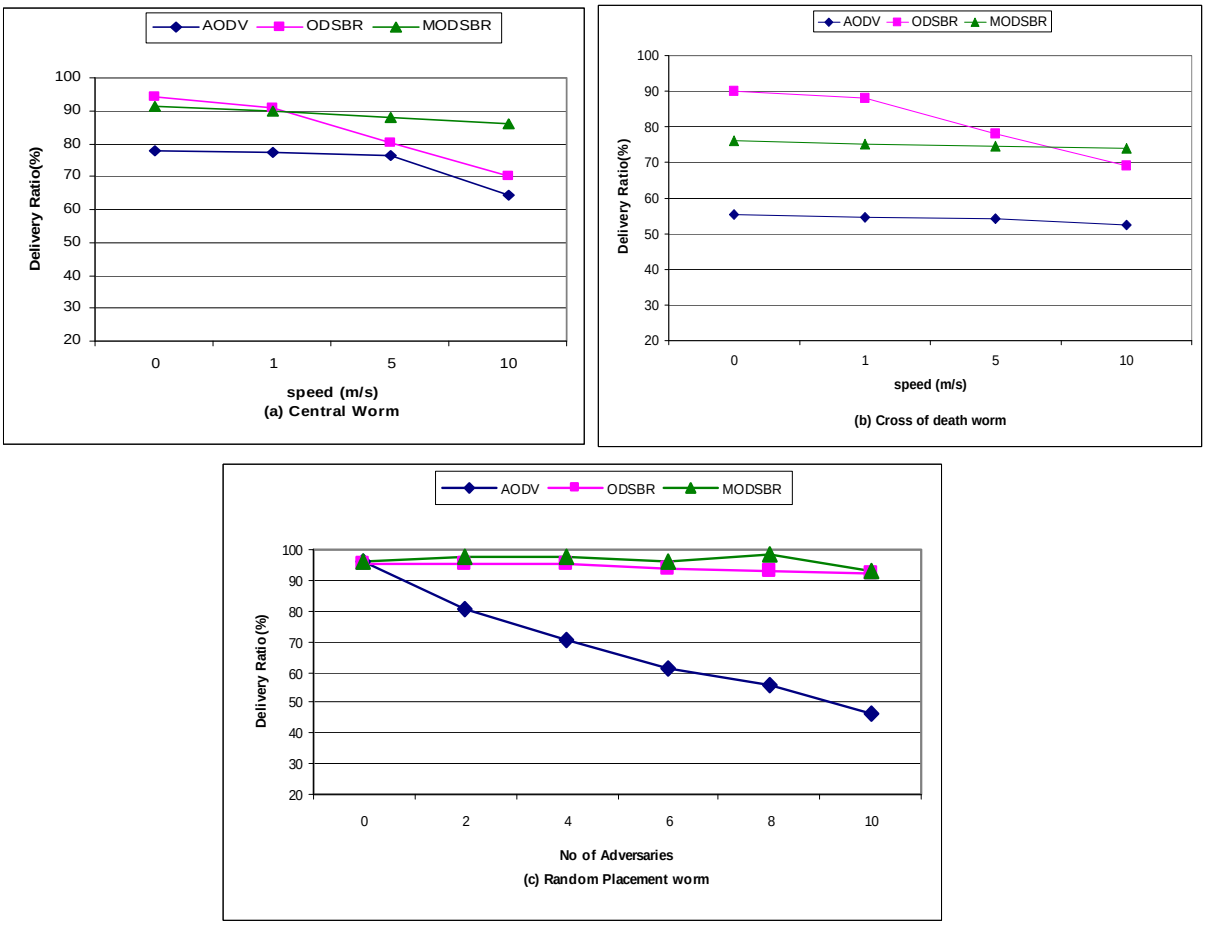


Figure 7. The effect of the Wormhole Attack in different forms on the delivery ratio of AODV, ODSBR, and MODSBR

Table 3 compares between the ODSBR & MODSBR using the %DRE for the central wormhole and cross of death wormhole. In case of the central wormhole, we can notice that MODSBR could improve the delivery ratio for high mobility while the ODSBR performs better in low mobility. Also in the case of cross of death it's clear that ODSBR give good delivery ratio compared to the MODSBR.

	ODSBR		MODSBR	
Speed	**Center**	**Cross of Death**	**Center**	**Cross of Death**
0 m/s	0.21	0.63	0.17	0.38
1 m/s	0.18	0.61	0.16	0.38
5 m/s	0.05	0.44	0.15	0.38
10 m/s	0.09	0.32	0.34	0.41

Table 3. %DRE for ODSBR & MODSBR in the presence of the central wormhole & the cross of death worm

Table 4 compares between the ODSBR & MODSBR using the %DRE for the random placement wormhole. It's clear that MODSBR improved the delivery ratio values more than the ODSBR with any no of adversaries for allmobility levels.

	ODSBR				MODSBR			
	0 m/s	**1 m/s**	**5 m/s**	**10 m/s**	**0 m/s**	**1 m/s**	**5 m/s**	**10 m/s**
2	0.16	0.18	0.14	0.15	0.21	0.21	0.20	0.27
4	0.34	0.35	0.25	0.21	0.40	0.394	0.39	0.38
6	0.51	0.54	0.42	0.32	0.58	0.58	0.55	0.56
8	0.70	0.68	0.52	0.48	0.78	0.78	0.76	0.80
10	0.91	0.97	0.85	0.67	0.95	1.00	1.02	0.99

Table 4. %DRE for ODSBR & MODSBR in the presence of the random placement wormhole

Figure 8 shows the delivery ratio of the AODV, ODSBR, and MODSBR after applying the wormhole attack combined with the flood rushing attack. It can be noticed that the performance of the MODSBR in the presence of flood rushing gets worse. In fact it gives a worse response compared to the ODSBR.

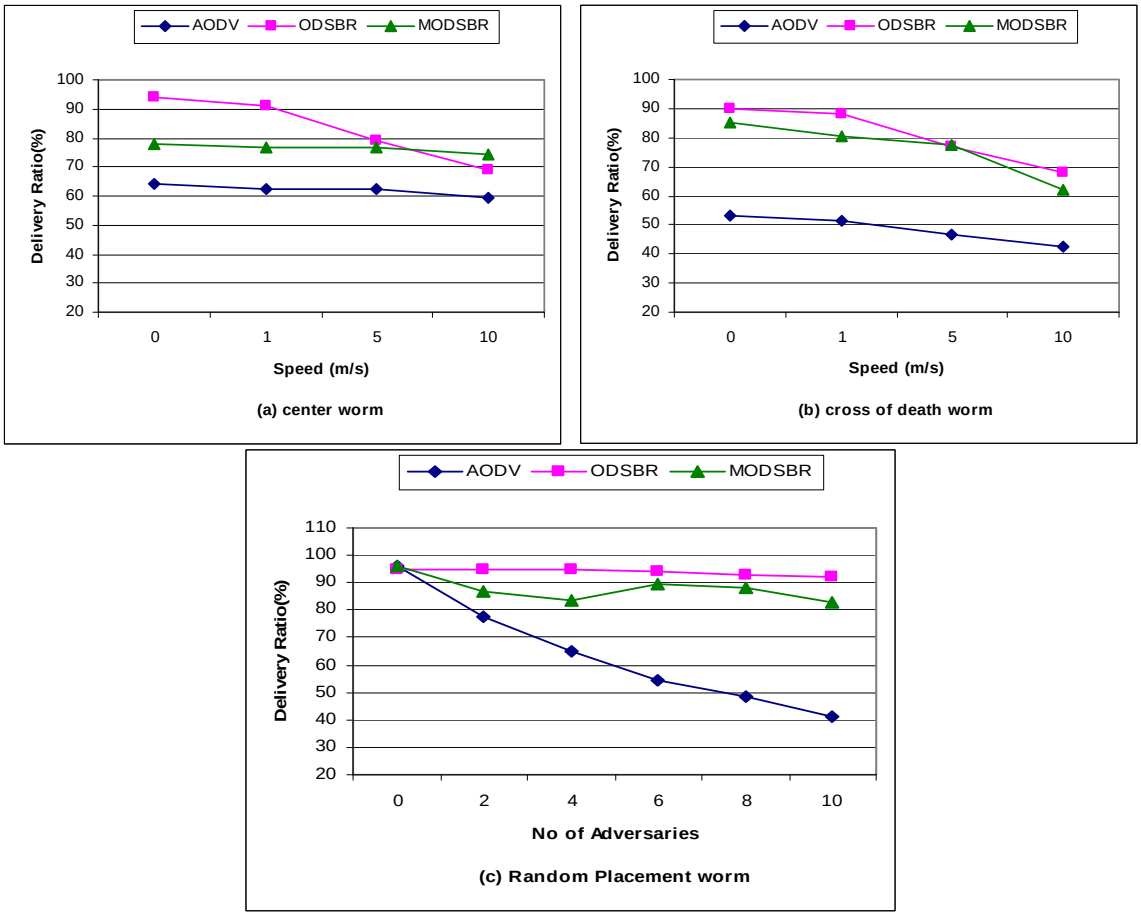


Figure 8. The effect of the Wormhole Attack in different forms combined with the flood rushing on the delivery ratio of AODV, ODSBR, and MODSBR

Table 5 compares between the ODSBR & MODSBR using the %DRE with the presence of central wormhole or cross of death wormhole in combination with the flood rushing. As shown the MODSBR doesn't perform well with the presence of the central wormhole & flood rushing but the ODSBR improves the delivery ratio more in this case. In the case of cross of death with flood rushing the ODSBR and MODSBR have values close to each other.

	ODSBR		MODSBR	
	Center	**Cross of Death**	**Center**	**Cross of Death**
0 m/s	0.47	0.69	0.22	0.60
1 m/s	0.45	0.71	0.23	0.56
5 m/s	0.27	0.65	0.23	0.66
10 m/s	0.16	0.59	0.25	0.46

Table 5. %DRE for ODSBR & MODSBR in the presence of each central wormhole & cross of death worm in combination with Flood Rushing

Table 6 compares between the ODSBR & MODSBR using the %DRE with the presence of random placement wormhole in combination with the flood rushing. As shown the MODSBR improved the delivery ratio values in a noticeable way but when compared to the ODSBR, the ODSBR is a little better.

	ODSBR				MODSBR			
	0 m/s	**1 m/s**	**5 m/s**	**10 m/s**	**0 m/s**	**1 m/s**	**5 m/s**	**10 m/s**
2	0.21	0.23	0.19	0.20	0.14	0.12	0.16	0.19
4	0.45	0.47	0.36	0.31	0.38	0.29	0.30	0.34
6	0.70	0.72	0.59	0.48	0.66	0.63	0.65	0.55
8	0.93	0.91	0.73	0.70	0.84	0.80	0.73	0.82
10	1.26	1.22	1.12	0.93	1.04	0.99	1.03	0.98

Table 6. %DRE for ODSBR & MODSBR in the presence of the random placement wormhole & flood rushing

5.2. Byzantine Overlay Network Wormhole Attacks

In this section, we will evaluate the damage caused to AODV by a set of attackers performing a coordinated super-wormhole attack, and demonstrate the effectiveness of the MODSBR and ODSBR

protocol in mitigating this attack. Similar to the wormhole attack we investigated three configurations namely the cross of death, random placement and complete coverage. The same configurations of the cross of death and random placement were applied. In the complete coverage the adversaries attempt to arrange themselves so that their combined communication areas completely cover the full ad hoc network. This means that if any transmission takes place in the network, an adversary will hear it. We simulated the configuration shown in Figure 9, with five adversarial nodes placed at coordinates (250,250), (250,750), (500,500), (750,250), (750,750).

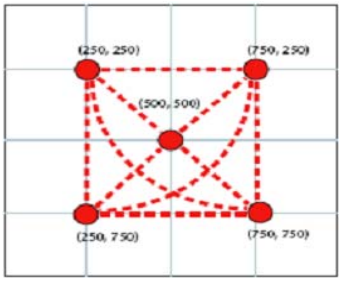


Figure 9 Complete Coverage Configuration

Figure 10 shows the delivery ratio of the AODV, ODSBR, and MODSBR after applying the three configurations of the overlay wormhole attack. In the case of cross of death and complete coverage it can be noticed that the delivery ratio of the ODSBR is better than the delivery ratio of the MODSBR except for high mobility. On the other hand, the delivery ratio of the ODSBR had decreased as the mobility increased, while for high mobility the MODSBR is approximately constant as with low mobility. In the case of random placement the delivery ratio has increased over than 90% even if the mobility increased.

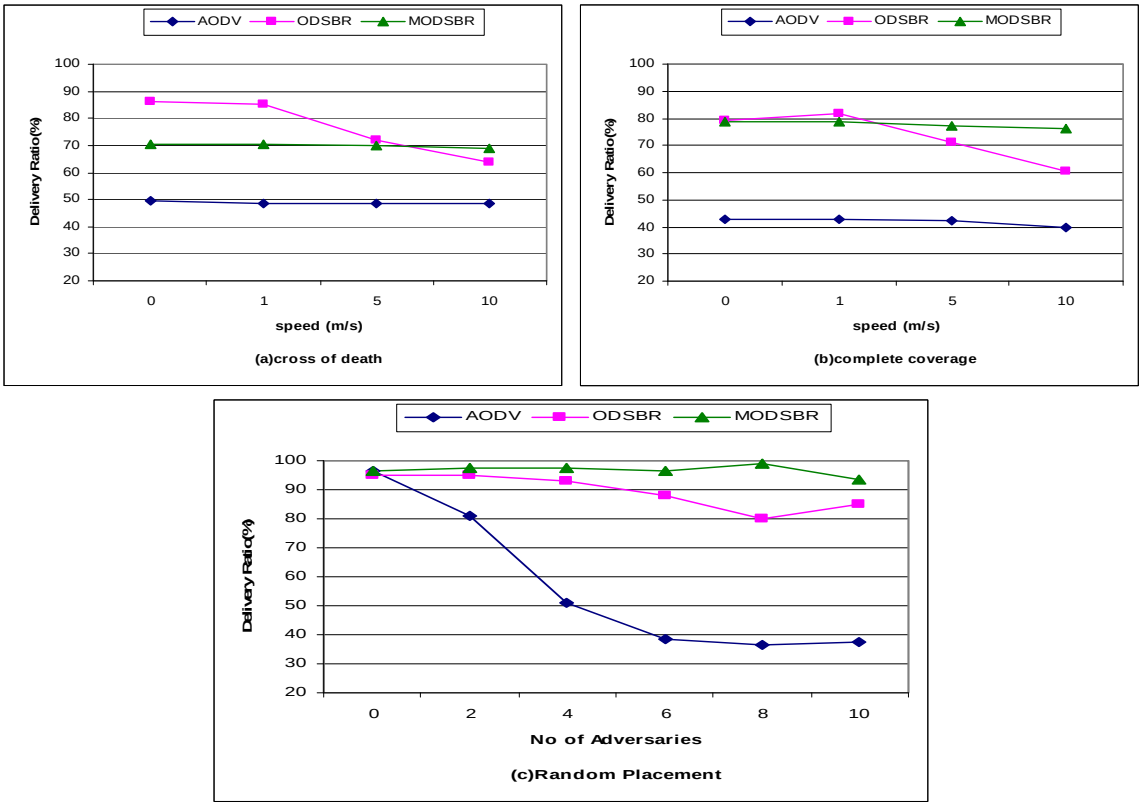


Figure 10. The effect of the overlay wormhole attack in different forms on the delivery ratio of AODV, ODSBR, and MODSBR

Table 7 compares between the ODSBR & MODSBR using the %DRE in the presence of complete coverage or cross of death. We can notice that MODSBR has improved the delivery ratio only for high mobility in these two cases.

	ODSBR		MODSBR	
Speed	**Cross of Death**	**Complete Coverage**	**Cross of Death**	**Complete Coverage**
0 m/s	0.73	0.84	0.42	0.84
1 m/s	0.75	0.92	0.44	0.85
5 m/s	0.48	0.68	0.44	0.82
10 m/s	0.32	0.52	0.43	0.91

Table 7. %DRE for ODSBR & MODSBR in the presence of each complete coverage & cross of death overlay wormhole

Table 8 compares between the ODSBR & MODSBR using the %DRE in the presence of random placement overlay wormhole. We can notice that MODSBR has improved the delivery ratio even for high and low mobility with respect to all no of adversaries.

	ODSBR				MODSBR			
No of adversaries	**0 m/s**	**1 m/s**	**5 m/s**	**10 m/s**	**0 m/s**	**1 m/s**	**5 m/s**	**10 m/s**
2	0.14	0.17	0.19	0.09	0.19	0.20	0.25	0.24
4	0.79	0.82	0.63	0.53	0.91	0.91	0.89	0.86
6	1.27	1.30	1.15	0.99	1.47	1.52	1.55	1.57
8	1.20	1.18	1.32	1.24	1.47	1.70	2.10	2.07
10	1.15	1.26	0.94	0.78	1.49	1.48	1.50	1.50

Table 8. %DRE for ODSBR & MODSBR in the presence of random placement overlay wormhole

Figure 11 shows the delivery ratio of the AODV, ODSBR, and MODSBR after applying each configuration of the overlay wormhole attack combined with the flood rushing attack. In the case of cross of death and complete coverage it can be noticed that the delivery ratio of the ODSBR is better than the delivery ratio of the MODSBR except for high mobility. On the other hand, the delivery ratio of the ODSBR has decreased as the mobility increased, but the MODSBR gives approximately the same DR for high mobility. In the case of random placement the delivery ratio has increased over than 90% even if the mobility increased. It can be noticed that performance for this case is similar to the overlay wormhole results.

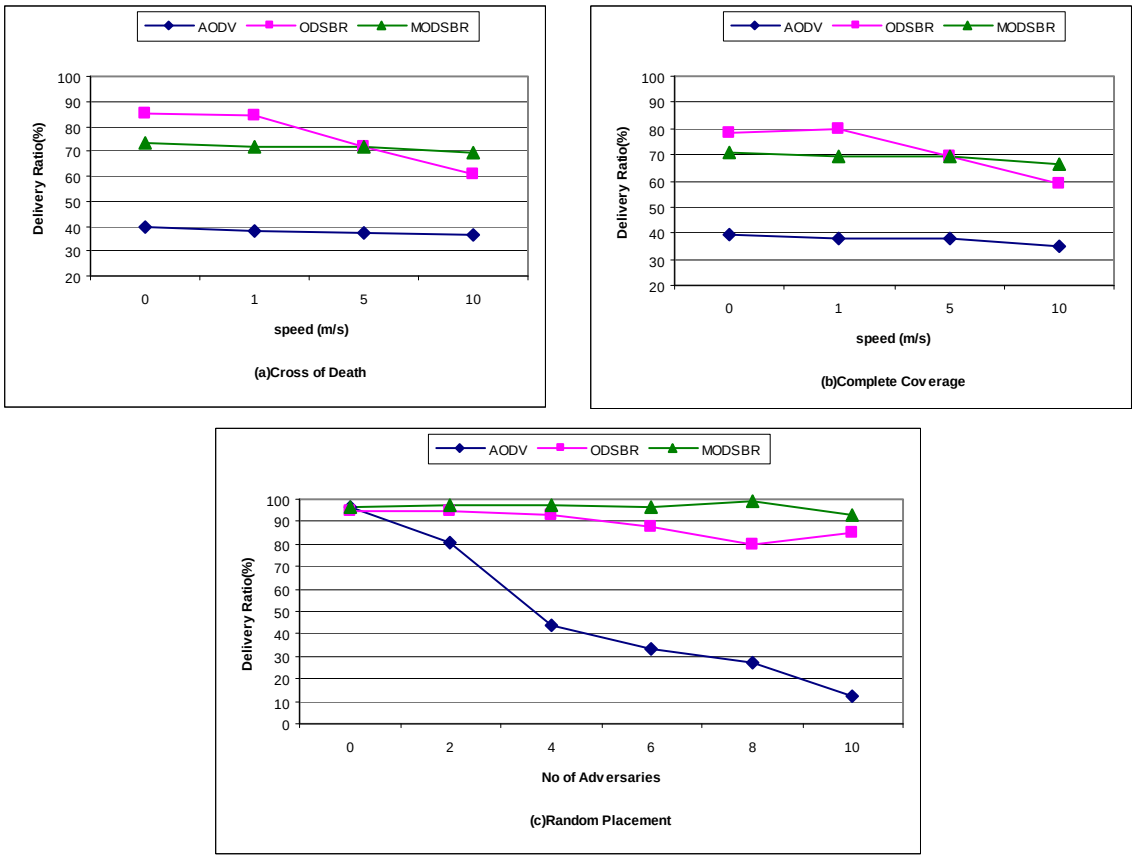


Figure 11. The effect of the overlay Wormhole Attack in different forms combined with the flood rushing on the delivery ratio of AODV, ODSBR, and MODSBR

Table 9 compares between the ODSBR and MODSBR using the %DRE when applying each of the complete coverage overlay wormhole and cross of death wormhole combined with flood rushing. Also here the MODSBR performs well for high mobility but in low mobility the ODSBR is better.

	ODSBR		MODSBR	
	Cross of Death	Complete Coverage	Cross of Death	Complete Coverage
0 m/s	0.73	0.84	0.87	0.80
1 m/s	0.75	0.92	0.87	0.82
5 m/s	0.48	0.68	0.92	0.82
10 m/s	0.32	0.52	0.88	0.87

Table 9. %DRE for ODSBR & MODSBR in the presence of each complete coverage & cross of death overlay wormhole with flood rushing

Table 10 compares between the ODSBR and MODSBR using the %DRE when applying the random placement overlay wormhole in combination with flood rushing. It can be noticed that the MODSBR improved the delivery ratio values more than the ODSBR in spite of presence of the flood rushing attack.

	ODSBR				MODSBR			
	0 m/s	**1 m/s**	**5 m/s**	**10 m/s**	**0 m/s**	**1 m/s**	**5 m/s**	**10 m/s**
2	0.16	0.17	0.27	0.18	0.22	0.20	0.33	0.33
4	0.81	1.12	1.10	1.12	0.92	1.23	1.44	1.59
6	1.62	1.66	1.41	2.04	1.85	1.91	1.86	2.93
8	2.16	1.91	1.82	2.33	2.56	2.59	2.77	3.57
10	5.00	6.08	8.11	7.71	5.96	6.78	10.75	11.25

Table 10. %DRE for ODSBR & MODSBR in the presence of random placement overlay wormhole & flood rushing

To conclude the previous results of all attacks, we can see the average of the delivery ratio values in Table 11. From the values we can say that the MODSBR has improved the delivery ratio more than the ODSBR by approximately 10% on the average. To be noted %average delivery ratio is calculated according to [(DRMODSBR - DRODSBR) / DRODSBR].

Attacks	**ODSBR**	**MODSBR**	**% average delivery ratio**
Black hole	86.75	95.4951	% 6.745102
Black hole Rushing	87.75	95.4951	% 6.7451
Wormhole center	83.75	89.7791	% 5.029147
Wormhole random	90.666	95.495	% 3.828435
Overlay Worm Complete Coverage	72.125	78.77	% 4.652532
Wormhole random	81.654	96.4951	% 11.82843
Wormhole random rush	82.666	96.6849	**% 13.01827**
Averages	86.75	95.4955	**% 9.6**

Table 11. Average delivery ratio for all experimented attacks

5.3. Routing Overhead

Simulations were conducted to compare the overhead of ODSBR with that of AODV, in order to evaluate the cost of security. In addition to route discovery overhead before and after applying route caching to ODSBR, ODSBR requires a protocol acknowledgment for each successfully delivered data packet. In real implementations, ODSBR acknowledgments can be piggy-backed on TCP acknowledgment packets, thus we only consider routing packets in the overhead measurements.

5.3.1. Simulation Results

Figure 12 illustrates the overhead of ODSBR and MODSBR in a non-adversarial scenario at all levels of mobility.

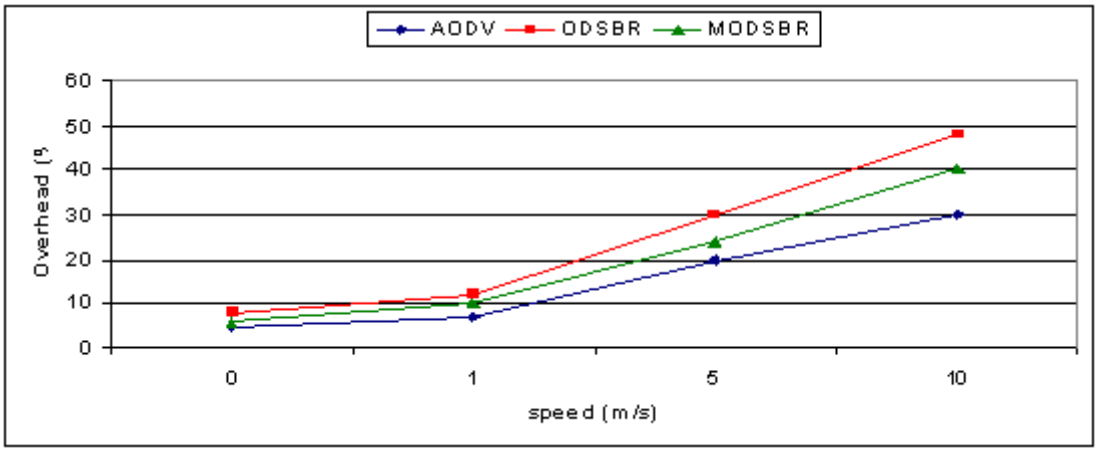


Figure 12. The overhead for the ODSBR and MODSBR without any attacks

Figure 13 depicts the overhead of the routing protocols as a function of the number of adversaries, when the adversaries execute a black hole attack. The nodes are under random way-point mobility with a maximum speed of 1 m/s. Observe that the routing overhead of MODSBR increases with the number of adversaries. This occurs as a result of the protocol activity in detecting faults and readjusting the path to avoid them. The overhead of MODSBR increases proportionally to the number of faulty links in the network.

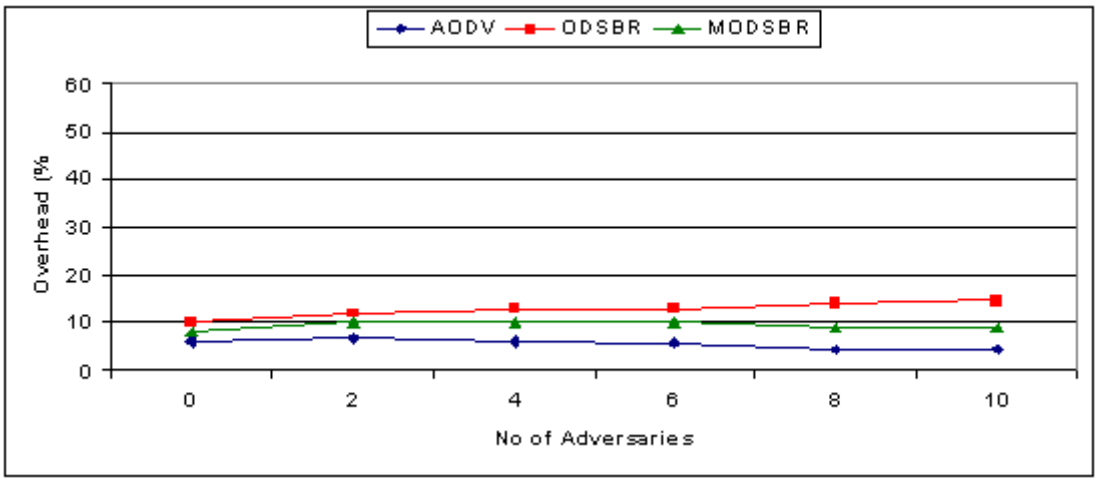


Figure 13. The overhead for the ODSBR and MODSBR in the presence of the black hole attack attacks

Figure 14 shows the overhead of ODSBR, MODSBR, and AODV with the presence of super wormhole attack. As shown the overhead of the MODSBR has decreased less than the ODSBR, but still greater than AODV. This is because disabling route caching is not the only reason for the high overhead.

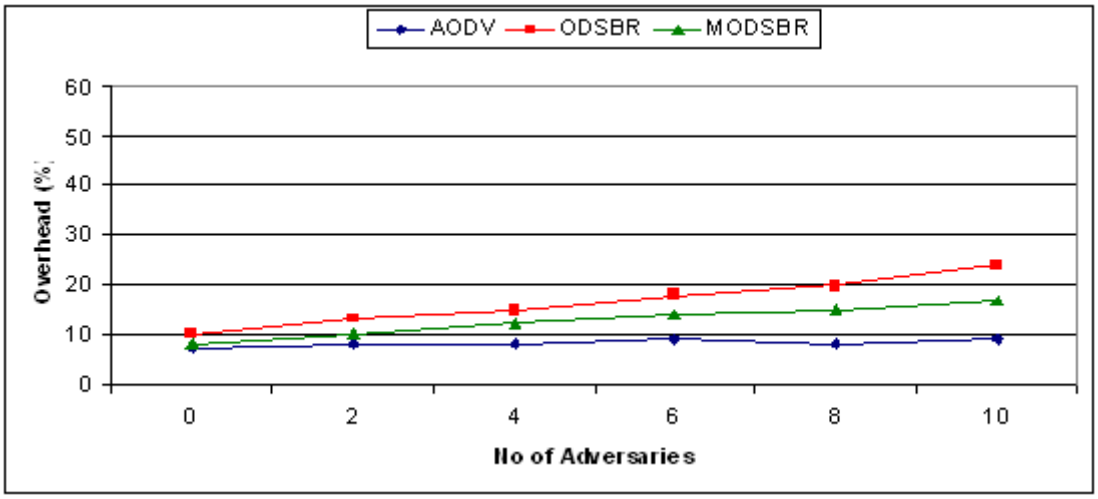


Figure 14. The overhead for the ODSBR and MODSBR in the presence of the super wormhole attack attacks

To conclude the previous results, we have listed the average of the overhead values in Table 12. Simulation experiments showed that the MODSBR outperforms ODSBR by an average of 23% decrease in routing overhead. To be noted that %average overhead in table 12 is calculated according to %average = (AVGODSBR - AVGMODSBR) / AVGODSBR.

	ODSBR	MODSBR	% Average overhead
Normal	24.5	20	% 18.3
Black hole	12.8	9.3	% 27.27
Wormhole	16.7	12.7	% 24.
Averages	18	14	**% 23.216**

Table 12. Average Overhead Values

6. Conclusions

ODSBR routing protocol is a very effective secure on-demand routing protocol that is resilient to Byzantine failures. But disabling the route caching property is a factor that causes decrease in the performance of the ODSBR protocol. So In order to enhance the performance of the ODSBR routing protocol, ways of taking advantage of route caching were investigated. The performance of the MODSBR protocol was analyzed in the presence of adversarial scenarios and in the normal behavior (non-adversarial scenarios). Our experiments showed that MODSBR outperforms ODSBR by an average 23% decrease in the overhead.

Another technique was implemented to MODSBR to improve its ability for security called "Central Clustered Link Weight Management". This technique aims to make nodes know about links

that caused network failure faster than previously. This was achieved by specifying a set of Supervisory nodes to store a general weight table, where any node can use this table during computing the total weight to any path or to update the weight for any link. The impact of the Byzantine attacks on the MODSBR was evaluated by computing the percentage delivery ratio (%DRE). The results were compared with the original ODSBR. The results showed that the MODSBR protocol was able to increase the delivery ratio in the presence of the black hole attack with and without flood rushing. Also it has increased the delivery ratio in the presence of the wormhole attack for the random placement case and the super wormhole attack for the complete coverage case. But in the presence of the central wormhole attack or the cross of death attack the delivery ratio was increased only for high mobility while for low or no mobility the original ODSBR is better to be used. On the average, we can say that MODSBR had improved the delivery by approximately 10% on the average.

7. References

[1] Awerbuch, B. Holmer, D. Nita-Rotaru, C., and Rubens, H. "An on-demand secure routing protocol resilient to Byzantine failures," in ACM Workshop on Wireless Security (WiSe), September 2002.

[2] Awerbuch, B. Holmer, D. Curtmola, R. Nita-Rotaru, C., and Rubens, H. "Mitigating Byzantine Attacks in Ad Hoc Wireless Networks," Technical Report Version 1, March 2004.

[3] Awerbuch, B. Holmer, D. Curtmola, R. Nita-Rotaru, C., and Rubens, H. "On the Survivability of Routing Protocols in Ad HocWireless Networks," in the First International Conference on Security and Privacy for Emerging Areas in Communication Networks (SecureComm), 2005.

[4] B. R. Smith, S. Murthy, and J. Garcia-Luna-Aceves, "Securing distance-vector routing protocols," in Symposium on Networks and Distributed Systems Security, 1997.

[5] C. E. Perkins and E. M. Royer, Ad hoc Networking, ch. Ad hoc On-Demand Distance Vector Routing. Addison-Wesley, 2000.

[6] C. Perkins, E. Belding-Royer, and S. Das, Ad hoc On-Demand Distance Vector (AODV) Routing. IETF – Network Working Group, The Internet Society, July 2003. RFC3561.

[7] Hauser, R. Przygienda, T., and Tsudik, G. "Reducing the cost of security in link-state routing," in Symposium of Network and Distributed Systems Security, 1997.

[8] Hu, L., and Evans, D. "Using directional antennas to prevent wormhole attacks," in NDSS 2004, 2004.

[9] Hu, Y, -C. Johnson, D, B., and Perrig, A. "SEAD: Secure efficient distance vector routing for mobile wireless ad hoc networks," in The 4th IEEE Workshop on Mobile Computing Systems and Applications, June 2002.

[10] Hu, Y, -C. Perrig, A., and Johnson, D, B. "Ariadne: A secure on-demand routing protocol for ad hoc networks," in The 8th ACM International Conference on Mobile Computing and Networking, September 2002.

[11] Hu, Y, -C. Perrig, A., and Johnson, D, B. "Packet leashes: A defense against wormhole attacks in wireless ad hoc networks," in Proceedings of the 22nd Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM 2003), April 2003.

[12] Hu, Y, -C. Perrig, A., and Johnson, D, B. “Rushing attacks and defense in wireless ad hoc network routing protocols,” in ACM Workshop on Wireless Security (WiSe), 2003.

[13] Kent, S. Lynn, C., and Seo, K. “Secure border gateway protocol (s-bgp),” IEEE Journal on Selected Areas in Communication, vol. 18, no. 4, 2000.

[14] Marti, S. Giuli, T. Lai, K., and Baker, M. “Mitigating routing misbehavior in mobile ad hoc networks,” in The 6th ACM International Conference on Mobile Computing and Networking, August 2000.

[15] Papadimitratos, P., and Haas, Z. “Secure data transmission in mobile ad hoc networks,” in 2nd ACM Workshop on Wireless Security (WiSe), 2003.

[16] Papadimitratos, P., and Haas, Z. “Secure routing for mobile ad hoc networks,” in SCS Communication Networks and Distributed Systems Modeling and Simulation Conference, pp. 27–31, January 2002.

[17] Sanzgiri, K. Dahill, B. Levine, B, N. Shields, C., and Belding-Royer, E. “A secure routing protocol for ad hoc networks,” in 10th IEEE International Conference on Network Protocols (ICNP’02), November 2002.

[18] “The network simulator - ns2.” http://www.isi.edu/nsnam/ns/

Developing an Intelligent System for Medical Diagnosis

Samir M. Abd El-razek
Department Of Information System
Faculty of Computers & Information
Mansoura University
Mansoura , Egypt
E-mail: samirabdelrazek@yahoo.com
http://csimu.mans.edu.eg

Alaa M. Riad
Department Of Information System
Faculty of Computers & Information
Mansoura University
Mansoura , Egypt
E-mail: amriad2000@yahoo.com
http://csimu.mans.edu.eg

Waeil Fathi Abd El-wahd
Department Of Operation Research
Faculty of Computers & Information
El-Menoufya University
Shiben El-Kom , Egypt
E-mail: waeilf@yahoo.com
http://mufic.com

Abstract *The clinicsoft system provides a decision support system that helps the physician to take a correct decision in diagnosis then treatment process. It is a Web based system that can help the physician in decision making through the diagnosis process, and knowledgebase user interface for reviewing and updating the knowledge. The system's features are: The importance of easy access to patient data essential for decision support, the commitment to continued measurement and revision of both the logic and the interventional strategy in a decision support application, and the role of clinical reports in supporting the decision making process.*

1. 1. Introduction

Physicians have to keep track of Medicine details all the time, including the new diseases, special cases and new medicine and methodology of treatment. Also they have to keep track of patients' information and medical history to make the right decision of diagnosis and treatment. This information needs to be integrated all the time to reduce the possibility of faults which is very vital to diagnosis and treatment process. The main challenge is to provide a decision support system that helps the physician to take a correct decision in diagnosis then treatment. This needs to be very accurate and flexible to the Medicine updates. As the knowledge of Medicine increases by the time and new methodologies of treatment take place of the old ones. Patients' medical history also is so important in the diagnosis process and treatments, which is very hard and complicated to physicians to handle [3].

There is a range of problems facing Physicians such as

(1) The usual clinical applications is very attached to patients records and patient's history, It helps the physician in the paper work only. But they still have to keep track of All Medicine details all the time.
(2) Specialized physicians don't keep track of other fields all the time, which is very dangerous in treatment process if the patient has some treatment cautions due to a specific medical condition.
(3) Interlaces between diseases' complaints, Medical investigations, and Treatments have a very high factor on the diagnosis accuracy.
(4) On the other hand, the next lines show the work of others in medical intelligent applications.
(5) Many intelligent systems have been developed for the purpose of enhancing health-care and provide better health care facilities, reduce cost and etc. As express by many studies such as
(6) [4] intelligent system was developed to assist users (particularly doctors and patients) and provide early diagnosis and prediction to prevent serious illness. Even though the system is equipped with "human" knowledge, the system will never replace human expertise as human are required to frequently monitor and update the system's knowledge. Therefore, the role of medical specialist and doctors (or medical practitioner) is important to ensure system validity[5-6].
(7) Early studies in intelligent medical system such as MYCIN, CASNET, PIP and Internist-I have shown to out perform manual practice of diagnosis in several disease domains [7]. MYCIN was developed in the early 1970s

to diagnose certain antimicrobial infections and recommends drug treatment. It has several facilities such as explanation facilities, knowledge acquisition facilities, teaching facilities and system building facilities. CASNET (Causal ASsociational NETworks) was developed in early 1960s is a general tool for building expert system for the diagnosis and treatment of diseases. CASNET major application was the diagnosis and recommendation of treatment for glaucoma. PIP an abbreviation for Present Illness Program was developed in 1970s to simulate the behaviour of an expert nephrologist in taking the history of the present illness of a patient with underlying renal disease. The work on Internist-I in early 1982s was concentrated on the investigation of heuristic methods for imposing differential diagnostic task structures on clinical decision making. It was applied in diagnoses of internal medicine[2].

(8) In 1990s, the studies in intelligent system was enhanced to utilize the system based on current needs. In several studies two or more techniques were combined and utilized the function of the system to ensure system performance. ICHT (An Intelligent Referral System for Primary Child Health Care) developed to reduce children mortality especially in rural areas [1]. The system success in catering common paediatric complaints, taking into consideration the important risk factors such as weight monitoring, immunization, development milestones and nutrition. ICHT utilized expert system in the process of taking the history data from patients. Other expert system have been developed such as HERMES (HEpathology Rule-based Medical Expert System) an expert system for prognosis of chronic liver diseases [8], Neo-Dat an expert system for clinical trails [9], SETH an expert system for the management on acute drug poisoning [10], PROVANES a hybrid expert system for critical patients in Anesthesiology [11]and ISS (Interactive STD Station) for diagnosis of sexually transmitted diseases [12].

Case Based Reasoning (CBR) was employed to utilize the specific knowledge of previously experienced and concrete problem or cases. The system can be used by patients to diagnose themselves without having to make frequent visit to doctors and as well as medical practitioner to extend their knowledge in domain cases (breast cancer).

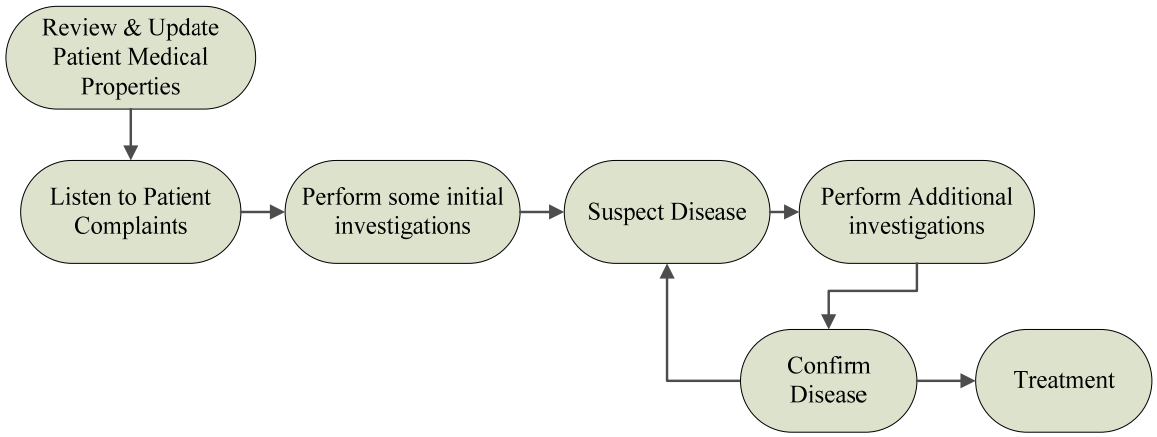
The paper is organized such that, section 2 points out brief description of the system workflow and implementation methods. Section 3 points out description of the system (clinicsoft) functionality through liver disease case study .section 4 points out achievements of the system.

1.1. 2. Cycle of implementation

ClinicSoft is a web based active knowledge system which integrates medical knowledge and patient data to generate case specific advice. It's an open system that can be applied on different medical domains. The web based architecture of ClinicSoft enables its usage remotely on the internet, in advance it will enable physicians to share and update their medical experience within its knowledgebase. ClinicSoft can help physicians to manage and make use of patient data by integrating it with the medical knowledge to generate some specific advices in diagnosis like suggesting medical investigations to perform, possibilities of infection, and the best treatment of the case. Simplifying the knowledge of medicine through some rule based abstraction is the basis of decision making in the system. It has a dynamic modeling of relating medical knowledge, using a simple web based interface that enables users to manage and update knowledge and relations.

22.1 Medical diagnosis workflow

The medical diagnosis workflow can be simplified as shown in Figure(1).



Figure(1). Workflow of clinical diagnosis

In this model the physician has to make more than decision depending on the case attributes (clinical variables) like:

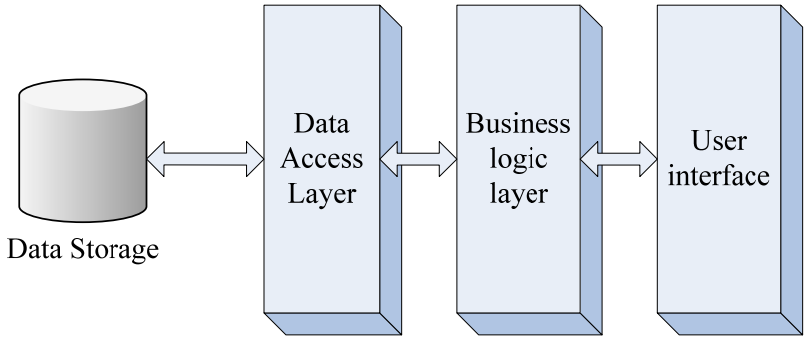
- What investigations to be performed.
- Choose a disease from a set of possible options.
- Selecting the appropriate treatment for the case.

In the early stages of diagnosis, identifying the disease is not an easy task. The set of known diseases can interlace from the domain variables perspective. Some disease are so similar in their properties, they may differ in one or two property. So the doctor has to narrow the area of search by performing some initial investigations depending on the set complaints that the patient have. The investigations values can lead the physician to few possibilities depending on his experience. If the physician suspected the existence of a certain disease, he should confirm its existence and its level to give the appropriate treatment. This confirmation may lead to another disease suspicion. Finally the physician has to choose the appropriate treatments for the case depending on its attributes, some treatments has cautions of use.

2.2 Methods

ClinicSoft is designed using 'N-tier' model as shown in which the user interface, functional process logic ("medical rules"), computer data storage and data access are developed and maintained as independent modules. As shown in Figure (2) ClinicSoft consists of four tiers.

Apart from the usual advantages of modular software with well defined interfaces, the N-tier architecture is intended to allow any of the N tiers to be upgraded or replaced independently as requirements or technology change.



Figure(2). N-tier architecture of *ClinicSoft*

3. Analysis

ClinicSoft is composed from two main modules:

(9) 1. Diagnosis wizard, included most of the decision support functions. It consists of a series of ordered steps that defines the diagnosis process on patient from reviewing the medical information of the patient, then identifying his complaints. The system starts to suggest some primary investigations to be performed on the patient to help the system and the physician to identify the disease. The outcome of the performed investigations on the patient will lead the system to suspect the existence of a certain disease. Then the physician is about to choose the appropriate treatment for the patient, the system will suggest the set of treatments needed.

(10) 2. Knowledgebase manager, this module defines all the information, relations, and rules used in the diagnosis process. The physician will be able to view update all the set of diseases and their treatments, the set of known complaints and their relation with diseases, and the set of medical investigations and medical lab analysis.

(11) As shown previously that ClinicSoft can be applied on different medical domains, it has been applied on Liver diseases domain. After supplying the domain information to the system as shown in figure (3) , it had provide valuable medical advices like :

(12) the system can suggest performing some medical investigations depending on the correlation between patient's information and his complaints as shown in figure (4) .

(13) the system analyzes the case attributes to give graphical report of suspected diseases and its infection rates as shown in figure (5) .

(14) the system can suggest the set of appropriate treatments taking into consideration patient's medical conditions and can define the set of forbidden treatments that cautions examined on the patient as shown in figure (6) .

Figure (3). An example of Knowledgebase manager – Diseases page

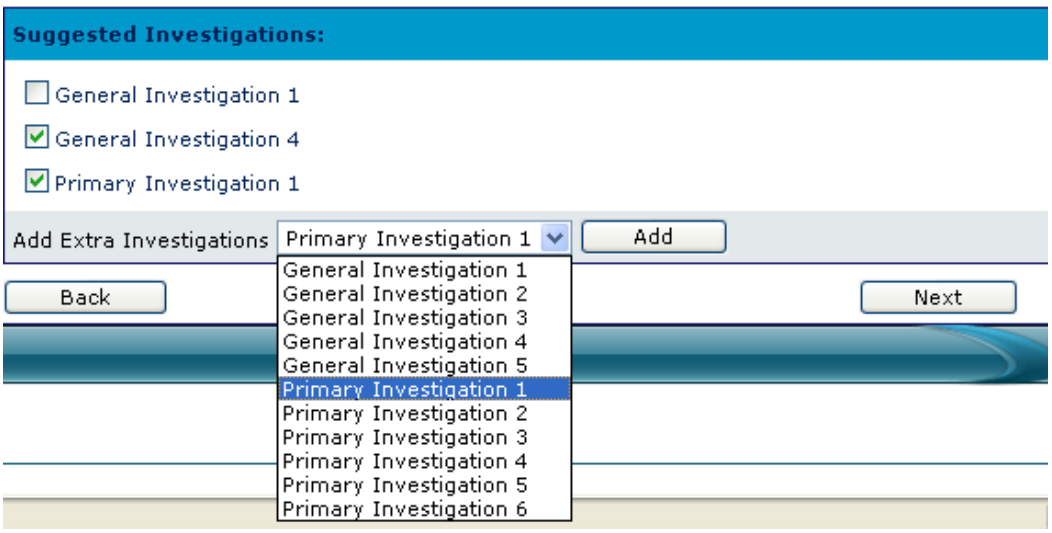


Figure (4). An example Suggested investigations

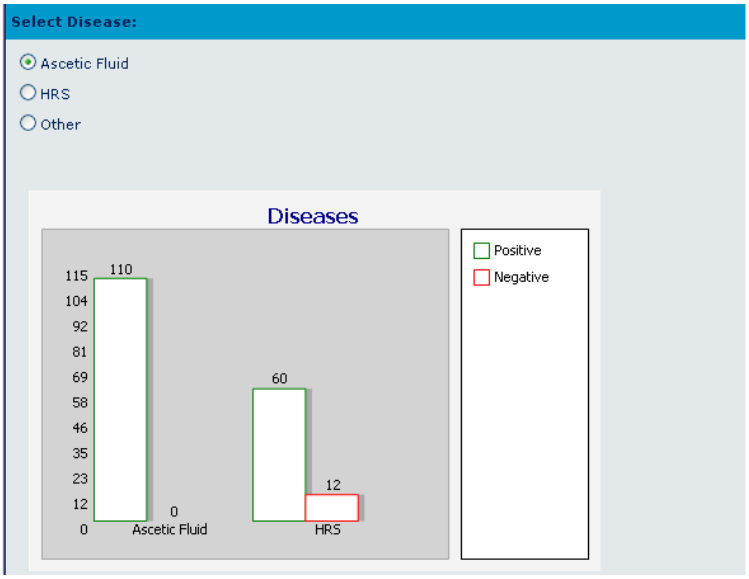


Figure (5). An example Bar chart displaying the suspected diseases and the infection rates

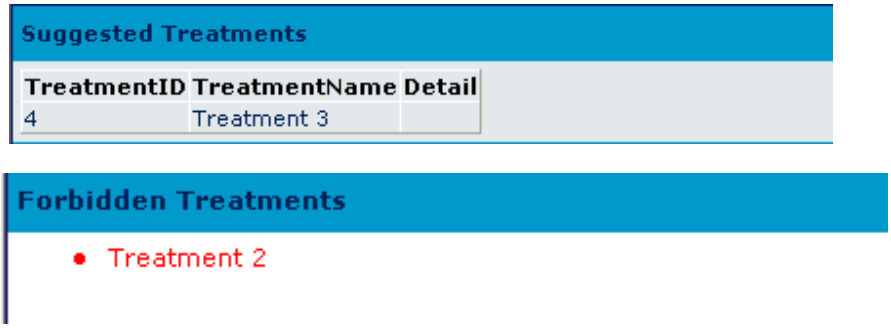


Figure (6). Examples of suggested treatments & forbidden treatments page

4. Conclusions

ClinicSoft system could accomplish the four key functions of electronic clinical decision support systems are outlined in [13]:

1- Supporting clinical coding and documentation, authorization of procedures, and referrals.

ClinicSoft provides a powerful administration tool that enables the physician to control the knowledgebase of the system. In addition to the authorization techniques built to prevent unauthorized access to the system or the patient records.

2- Managing clinical complexity and details, Keeping patients on research and chemotherapy protocols; tracking orders, referrals follow-up, and preventive care.

The design of ClinicSoft has simplified the workflow of medical diagnosis into some rule based techniques, In addition to the friendly user interface in the diagnosis wizard.

3- Monitoring medication orders; avoiding duplicate or unnecessary tests.

On the other hand , In the diagnosis wizard, The investigations performed can't be duplicated, as they grouped in one unit called the case investigations. Even the duplications that may happen in lab tests are avoided.

4- Supporting clinical diagnosis and treatment plan processes; and promoting use of best practices, condition-specific guidelines, and population-based management.

The developed system provides decision support in multi phases of diagnosis, like the primary investigations to be performed, the disease identification, and treatment selection.

References

[1] Blois & Shortliffe, from http://dmi- ww.mc.duke.edu/dukemi/essent/whatis.html, 10 January 2008.
[2] D. Georgios, "Hybrid Computational Intelligence in Medicine" University of the Aegean, Dept. of Business Administration, Greece, 2006.
[3] H. S. Edward, J. C. James, "Biomedical Informatics, Computer Applications in Health Care and Biomedicine", Springer Science+Business Media, LLC, third edition, 2006.
[4] A. Ramesh , C. Kambhampati2, J. Monson and P. Drew," Artificial intelligence in medicine ", Ann R Coll Surg Engl, 2004
[5] B. Kaplan, N. T. Shaw " Future directions in evaluation research: people, organizational, and social issues ", Methods Inf. Med. 43, pp. 231-251, 2004.
[6] R. Haux, A. Winter, E. Ammenwerth and B. Brigl," Strategic Information Management in Hospitals. An Introduction to Hospital Information Systems," Springer, New York, 2004.
[7] M. T. Andrew, "Computer decision support systems in general practice", International Journal of Information Management 21, 2003.
[8] E. Turban, "Decision support and expert systems. Management support systems." Macmillan Publishing Company, USA, 2001.
[9] H. W. I. Wan, S. Fadzilah, " ARTIFICIAL INTELLIGENCE IN MEDICAL APPLICATION: AN EXPLORATION " School of Information Technology, Universiti Utara Malaysia, 06010 Sintok, Kedah, MALAYSIA, 2001.

[10] O. Moore, "Decision support based on laboratory data", Methods of Information in medicine, 27, 2000.

[11] Passold, Fs., Ojeda, R. G., and Mur. Hybrid Expert System in Anesthesiology for Critical Patients, In Proceedings of the 8th IEEE Mediterranean Electrotecnical Conference - MELECON'96 (ITALIA), Vol. III, pp. 1486-1489 , 1999 .

[12] Walker, N. J., and Kwon, O. ISS: An Expert System for the Diagnosis of Sexually Transmitted Diseases, 11th Annual Midwest Computer Conference (MCC'97) March 21, Springfield, Illinois , 1999.

[13] Wyatt J, Spiegelhalter D,. From http://www.openclinical.org/dss.html#perreault , 2001.