A Parallel Chemical Reaction Optimization Algorithm for MaxFlow Problem

Mohammed Y. Alkhanafseh[1], Mohammad Qatawneh[1], Hussein A. al Ofeishat[2]

[1]University of Jordan, King Abdullah II School for Information Technology Computer Science Computer, Amman -Jordan, [2]Al-Balqa applied university Jordan.
{mkhanafsa@gmail.com, mohd.qat@ju.edu.jo, ofeishat@bau.edu.jo}

Abstract

This paper presents an algorithm for Maxflow problem using Parallel Chemical Reaction Optimization (CRO). The main purpose of maxflow is to find the max value that can be transferred from the source node to the sink node in different paths without violating the capacity value for each edge. The proposed parallel CRO algorithm has been evaluated and compared using IMAN1 with sequential version of CRO algorithm in terms running time for calculation the max flow value and speedup. The simulation results show that less run time and high speed up have been achieved applying parallel CRO algorithm when compared with a sequential version of maxflow.

Keywords: Chemical Reaction Optimization, MaxFlow problem, Molecule, Synthesis, Optimization, augmenting path, Multi Core, Parallel.

1. Introduction

The movement of material from the source point where it was produced, to the sink node where it will be consumed will done based on transfer rate through channel or edge. The flow network is a weighted graph, where each edge has a capacity, and can't transfer data through it, if it's greater than its capacity value [2]. Many real-life applications can represent Maxflow problem like traffic on road, currents in electronic circuits, and water flow through pipes where the diameter of the pipe will control the amount of water that can be send from source to sink which represent the capacity of edge needed to send and must be less than edge capacity value. Maxflow problem become one of most well knows problem for its optimization in the weighted directed graph [3], it can be implemented on different applications on engineering, networking and transportation fields [3].
The results were conducted using IMAN1 supercomputer which is Jordan's first and fastest supercomputer. It is available for use by academia and industry in Jordan and the region and provides multiple resources and clusters to run and test High Performance Computing (HPC) codes [13,14,15,16]

In this paper, a new parallel CRO algorithm for maxflow problem is implemented by the library Message Passing Interface MPI, where MPI processes are assigned to the cores. Experimentation of the proposed algorithm was conducted using IMAN1 supercomputer which is Jordan's first supercomputer. The IMAN1 is available for use by academia and industry in Jordan and the region, where the graph is divided into subgraphs with different augmenting paths.

https://sites.google.com/site/ijcsis/
ISSN 1947-5500

The rest of this paper is organized as follow. Section 2 presents the related work. Section 3 illustrates the maximum flow problem. Section 4 introduces the sequential and parallel Chemical Reaction Optimization. Section 5 shows the experimental results. Section 6 presents the conclusion.

2. Related Work

Maxflow problem was widely studies and different researchers was work on this problem to get best results, first pseudo polynomial algorithm to solve Maxflow problem is augmenting path algorithm which implemented by Ford Fulkerson, it refer to most famous solution for Maxflow problem(1956)[3], other solution proposed for Maxflow problem suggested by Edmond Karp[4], where independent polynomial version of augmenting path algorithm for Ford Fulkerson, other work on Maxflow problem was produced by Edmonds Karp(1972) and Dink(1970), proved that if augmenting path is shortest one, then the algorithm will performs in O(mn) augmentation steps, where n is number of vertices and m is number of edges on the graph. several other efficient algorithm was developed as Ahuja and Orlin improved the shortest path augmenting path algorithm in (1987)[5], the push and re-label method is introduced by GoldBerg [6], and gobergtargan [7] do other development for MaxFlow problem by introducing relabel operation to perform fine-grain update of the vertices distance, oriln [8], which do other development on Maxflow problem by improving polynomial time algorithm for Maxflow defined on the network with n nodes and m arcs, and show how to solve MF problem with O(mn), and other development on the time need for execution was done by King, Rao, and Tarjan [15] who solved the max flow problem in O(nm logm/(n log n) n) time.

all above related work on the field of original problem Maxflow problem, on the field of CRO solution, different solutions were improved last year on the field of using CRO as heuristic algorithm to solve many problem like Maxflow Problem, where CRO was recently proposed as general-purpose meta-heuristic algorithm, the CRO was proposed by Lam et a1 in 2010 and was developed to solve combinatorial optimization problems. They solved some classical problems, e.g., quadratic assignment problem and channel assignment problem [10]. it can give solutions for problem which was solved by other algorithms such as genetic algorithm, which refer to meta-heuristic algorithm as in [11], where genetic here was used to estimate COCOMO model parameter using genetic algorithm.

3. Maximum Flow Problem

Suppose there is a directed network G = (V, E) defined by a set V of nodes (or vertices) and a set E of arcs (or edges). Each arc (i,j) in E has an associated nonnegative capacity uij-where i, j are nodes in V. Also, there are two distinguished special nodes in G: a *source* (or start) node s and a *sink* (or a target) node t [1]. For each i in V, denote by E(i) all the arcs emanating from node i. Let U = max {uij by (i,j) in E}. Denote the number of vertices by n and the number of edges by m [1].

The purpose is to find the maximum amount of that can be send from source node to sink node which satisfy the arc capacities and mass balanced constraints at all node. Representing the flow on arc (i,j) in E by xij, an optimization model for the maximum flow problem can be obtained as in (1) [1]:

$$\text{Maximize } f(x) = \sum x_{ij} \text{ subject to..... (1)}$$

https://sites.google.com/site/ijcsis/
ISSN 1947-5500

$$\sum Xij - \sum Xjt = 0 \ \forall t \in V\{s,t\}$$
$0<Xij<\forall(ij) \in E$

4. Sequential and parallel Chemical Reaction Optimization (CRO)

4.1 Sequential CRO algorithm

Chemical reaction algorithm refers to multi agent algorithm, and the manipulated agent are molecules. Different attributes are associated with each molecule, some of these attributes was essential on the basic operation of CRO, which are (1) the molecule structure (ω); (2) the potential energy (PE) and (3) the kinetic energy (KE). Other different attributes based on the algorithm operator to construct different CRO variants for particular problem provided that their implementation satisfies the characteristics of elementary reactions, where molecule structure capture the solution for the problem it was not required to be on any specific format, it can be number of vector or even matrix. Potential energy PE is considered as an important part for solution which presents the objective function value for corresponding solution presented by ω. If f denotes the objective function, then PE (ω) = F (ω). Finally, the Kinetic energy KE which is a non-negative number and it quantifies the tolerance of the system accepting a worse solution than the current one.

There are four types of elementary reactions in CRO, each of which takes place in each iteration of CRO. Assume that molecules are in a container, one of the following four reactions will be possible to occur in each of CRO iteration.

I. On-Wall Ineffective Collision

The On-wall Ineffective Collision reaction describes the condition when a molecule collide with the wall of the container and then bounces back, where this molecule ω will be converted to new molecule with different structure ω'

$$\omega \gg \omega'$$

II. Decomposition

This type of reaction is used to mimic the process of hitting the wall and then decomposition into two or more pieces. The original molecules is shown as follows.

$$\omega \gg \omega 1 + \omega 2$$

III. Inter Molecular Ineffective Collision

The Inter Molecular Ineffective Collision the process that two molecular ineffective collision is as follows.

$$\omega 1 \mid \omega 2 \gg \omega 1' \mid \omega 2'$$

IV. Synthesis

The synthesis is the process when more than one molecule collides and combines together. Suppose two molecules $\omega 1 \ and \ \omega 2$ collide with each other, and then a new molecule ω' bounce back.

$$\omega 1 + \omega 2 \gg \omega'$$

https://sites.google.com/site/ijcsis/
ISSN 1947-5500

To find MaxFlow for a graph using CRO, there is a need to explore different molecules, where these molecules present different solutions, and each molecule has its own potential energy, which is the objective function for that solution and a kinetic energy which helps in making decision.

4.2 Maxflow-CRO algorithm

As in [2], CRO algorithm has three stages: the initialization, the iteration, and the final stage. Figure 1 shows the attributes and their meaning related to the MaxFlow problem.

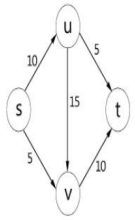
Flow network	Chemical Meaning	CRO Meaning
	MoleculerStrucutre	Flow Matrix
	Potential Energy PE	MaxFlow Value for specific graph
	Kinetic Energy	Measure of having worse solution
	Number of hits	Current total number of itterations
	Minimum Structure	Current Optimal Solution

Figure 1: The CRO attributes.

- Initialization phase

Through first stage initialization stage, parent size was specified based on specific parameter, where this parent presents first population, which will do the reaction with each other or with the wall of the container to generate other molecule or populations, and other parameter will specify the objective function. This parameter can be calculated based on different ways to fine shortest augmenting path from source to sink, where the value of MaxFlow which computed from objective function presents the current MaxFlow value, and this value can be improved gradually, other parameters will be defined through initialization stage is threshold like alphaα and beta β, where α threshold refer to decomposition threshold and β which refer to synthesis threshold and both divide parent size variable value by 2, the purpose from these threshold to make shore that reactions will be happened after specific number of collision like on decomposition with specific number of collision with the wall of the container if this number greater than threshold value then decomposition reaction will be happened.

- Iteration Phase

second stage after initialization stage refer to Iteration phase, where CRO algorithm refer to meta-heuristic algorithm, that it depends on number of iteration and different population to have solution for current population better than what exist in previous one, as iteration work the max flow value will be improved gradually until reach to the best solution, iteration stage on CRO start after Initialization stage, after define different parameters, calculation for potential energy will done through this stage then one or two molecule will be picked randomly based on b value which will be defined as number between 0 and 1, then this value will be compared with the value of molecule, if it greater than molecule value then it will select one molecule and then the value of HIT will be compared with alpha (α), if the value if HIT is greater than α value then decomposition will happened else if HIT value less than α value the inter wall infective collision will be happened, but if the value of B is less than molecule value then randomly two molecule will be

https://sites.google.com/site/ijcsis/
ISSN 1947-5500

selected and the value of Kinetic energy will be compared with (β) threshold and the parent size greater than 2 then synthesis will happened else Inter Molecule Effective Collision will happened.

Final phase

After finishing specific number of iterations, best molecule or solution which have Maxflow value between source node and sink node, and Maxflow value will be returned as best molecule or solution. The pseudo code of the algorithm is listed below.

```
1 //initialization phase
2 Set flow_network_size, C[i][j]: maximum capacity
3 parentSize, iterationNumber, s: source node, t:
4 sink node
5 HIT= 0
6 β = parentSize/2
7 α = parentSize/2
8 KE = parent Size/1.5
9 Generate molecule ∈ [0, 1]
10 parent Generating(C[i][j] , parent Size )
11 for (inti=1 to iterationNumber) // Iteration phase
12 Generate b ∈ [0, 1]
13 if b > Molecule then
4 Randomly select one parent
5 if (HIT > α) then
16 Decomposition( )
17 else
18 OnWallIneffectiveCollision( )
19 end if
20 end if
29 HIT++
30 KE--
 31 Check for any new maximum solution
 32 end for-loop // final stage
 33 return the best solution found
```

Time Complexity

Initial MaxFlow of the first parent the time complexity is O(NEF), where N is number of parent node and E is number of edges in flow network and f is maximum flow in the flow network, for different number of iteration the complexity will be O(IX), where I is number of iteration and X is the complexity for each iteration, which defined upper.

4.3 Parallel Maxflow CRO

The distributed system consists of a collection of independent computers that appear to the end user as single computer. These different computers have its own resources, and the architecture for different component with each other is shared nothing, so each node is independent and self-sufficient and there is no single point contention across all the system. The shared nothing architecture means that none of the computer share any resources like memory or storage with any other computer, for that communication between different computers was happened using message passing over the network.

https://sites.google.com/site/ijcsis/
ISSN 1947-5500

MaxFlow problem based on network size which determined by number of graph nodes, through sequential version of MaxFlow CRO algorithm the maximum number of graph node can be implemented up to 5500 graph node or vertices, this number of node or network size is small when we deal with some real case problem, for that we need to implement MaxFlow CRO algorithm on network or graph with large number of nodes to show how meta-heuristic algorithm CRO can solve MaxFlow problem with time less than time that optimal Ford Fulkerson algorithm needs, with same level of accuracy, this is the main aim of our study. Another goal for implementing this algorithm in parallel to show the amount of enhancement on time needs for finding Maximum flow value with same level of accuracy specially when number of nodes in the graph was increased and to show the relation between number of processor which used for execution in parallel and the amount of enhancement.

Through proposed solution, sequential version of CRO MaxFlow algorithm will be implemented on real parallel system using Message Passing Interaction library using C programming language, different steps were done to implement current sequential algorithm in parallel, the first one is to calculate time needs for each step in sequential CRO MaxFlow algorithm to determine which step of sequential algorithm consume most of the execution time.
through MaxFlow CRO algorithm different augmenting path was founded from source node to sink node, the summation of different augmenting paths from source to sink will produce the Maximum Flow value for current iteration, this value will be enhanced from iteration to other iteration, through implementing MaxFlow CRO in parallel this objective Function will be implemented on parallel by dividing the graph which contain different number of node and two main node source and sink one into number of processor which will solve the problem, this mean that one or more than one augmenting path between source and sink will be computed by one processor and other augmenting paths will compute by other processor.

Parallelism = Number of augmenting paths/ Number of Processor

the process of distribution the matrix which will contain different graph nodes will be done by first processor which contain the main graph, this matrix will divided based on number of processors, then the result from different processor for MaxFlow value will returned to main Processor to calculate total MaxFlow value based on result which come back from different processors, the following pseudo code for the part which implemented in parallel from MaxFlow CRO algorithm to divide the algorithm in parallel.

```
1 for (j=1 to flow_network_size/2)
2 F1'= generate new flow randomly
3 Define N as number of Processor
4 divide generated flow by number of processor
5 calculate summation of MaxFlow value for generated augmenting path.
6 return MaxFlow Value for generated augmenting path for processor x to Main processor.
7 PE1' = objective Function(F1')
8 //Compute PE' objective function for the
8 new molecule
9 if (PE1'> PE1 ) then // solution confirmed
10 destroy F1
11 return F1'
12 for (j=1 to flow_network_size/2)
13 F2'= generate new flow randomly
```


https://sites.google.com/site/ijcsis/
ISSN 1947-5500

14 nd for-loop
15 define number of processor
16 divide number of node over number of processor
17 distribute number of augmenting path based on number of processor
18 calculate the summation of Maximum flow value for different augmenting path for each processor
19 return summation value to main processor
20 distribute generated graph over number of Node
21 PE_2' = objectiveFunction(F_2')
22 //Compute PE' objective function for the
23 new molecule
24 if (PE_2'> PE_2) then // solution confirmed

4.4 Time Complexity

parallel implementation for MaxFlow CRO based on divide graph into different sub graph and distribute it over different number of processor, based on that time complexity for algorithm will be O (NEF * P), where N is number of node for each sub graph and E is related to number of edges for each augmenting path from source to sink and F is Maximum Flow value from source node to sink node, and P refer to number of processor used for execution in parallel.

5. Experimental Results

MaxFlow CRO parallel version using Message passing iteration library in C programming language was implemented in real distributed system using different number of nodes for generated graph and using different number of processors, table1 presents different number of node and time need for calculating Maximum Flow value using single processor and different number of processor as follow:

Table 1: show results for implementing different network size using different number of processor.

NetWork Size	Sequential Using 1 P	Parallel Using 2 P	Parallel Using 4 P	Parallel Using 8 P	Parallel Using 12 P	Parallel Using 16 P	Parallel Using 24P	Parallel Using 32 P
5000	4	3	2	2	3	4	5	8
6000	7	5	4	3	4	6	6	7
7000	9	6	5	3	6	7	8	9
8000	11	7	6	4	7	8	8	9
9000	14	8	6	5	9	9	10	11
10000	19	14	12	10	12	15	16	17
11000	25	16	14	12	16	19	21	21
12000	27	17	15	14	18	17	25	25
13000	29	19	17	15	20	19	23	25
14000	35	22	20	19	24	23	26	26
15000	39	25	24	21	25	26	27	27
20000	71	43	41	38	44	44	44	44
25000	130	84	81	79	81	53	53	54
30000	167	108	104	98	98	105	98	102
35000	208	123	120	115	115	120	118	125

The upper results show that the graph size that can be implemented was increased from 5500 nodes on sequential Maxflow CRO algorithm to more than 35000 nodes in parallel. Also, the amount of enhancement when convert the algorithm from sequence to parallel using 2 processors greater than that when number of processor was increased larger than 2 processors, when we increase number of processors to calculate

https://sites.google.com/site/ijcsis/
ISSN 1947-5500

maximum flow for generated graph other enhancement was achieved on the execution time but not with same level for enhancement when we moved from sequential to parallel using 2 processor, and this enhancement was decreased when moved from using 4 processor to 8 processor, and at some network size there is no enhancement, for that we can note that number of processor which used to calculate Maximum flow value was valid up to specific number, in our proposed solution this number was limited to 8 processors. For large number of processors, the result will be adversely effect on the execution time, this refer to communication needed between different processors to send and receive data. The amount of time which needs for communication will be greater than the time needs for execution through each processor, this mean that the amount of enhancement on calculation which done in each processor will be less than the overhead caused by communication between different processor, this point was clear from the result when convert the algorithm from running on single processor to run in parallel using 2 processor, here in two processor the communication overhead between these two processor less than the enhancement which done on the execution to fine maximum flow value, and when number of processor increased communication overhead will be increased.

Figure 2 shows a comparison between the performances of the different processors.

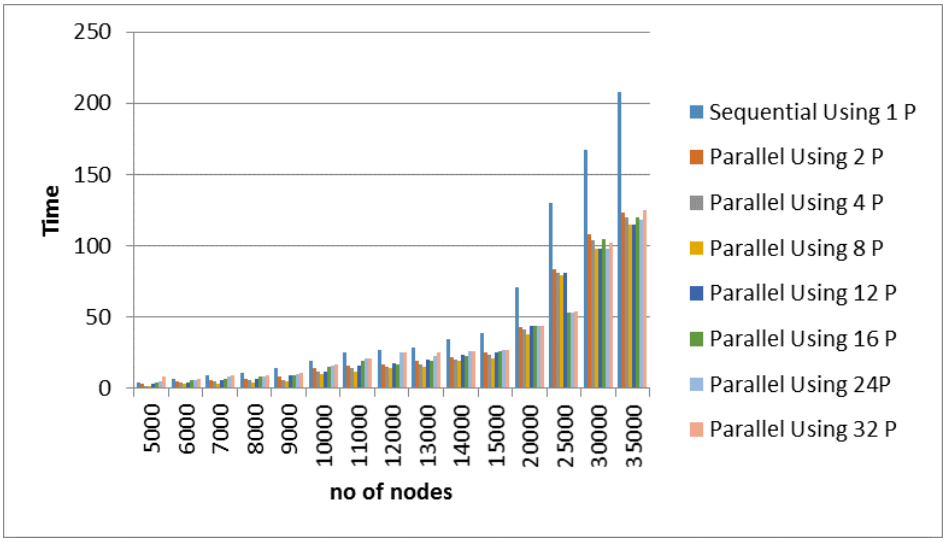


Figure 2: performance of MaxFlow CRO In Parallel using 1,2,4,..,24,32 Processor.

Figure 3 shows the relation between time needs for sequential and parallel using two processors.

We achieved about 39% of time enhancement compared with the time needs on sequential version of algorithm, this will increase the efficiency of algorithm special when run on large graph.

The improvement on increasing number of processor from 2 processors to 4 processors was about 18% of execution time, it is not large but still there is enhancement, as shown in figure 3.

https://sites.google.com/site/ijcsis/
ISSN 1947-5500

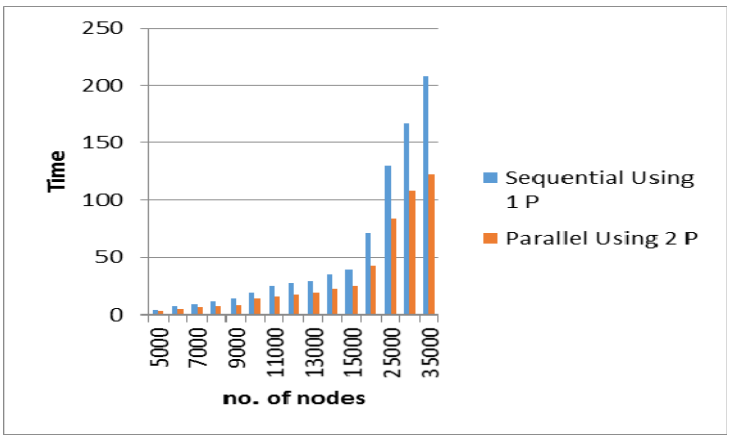


Figure 3: Relation between time needs for Sequential and parallel using 2 processors

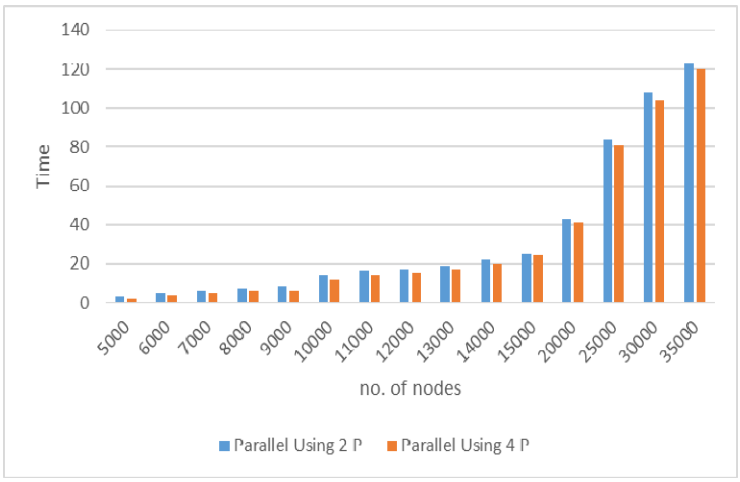


Figure 4: Enhancement Achieved by increasing number of processor used for calculate MaxFlow value by using 2 processors and 4 processors.

When increasing the number of processors the running time do not effect that much, as figure 5 shows that the different between running time using 4 processors and 8 processors is little.

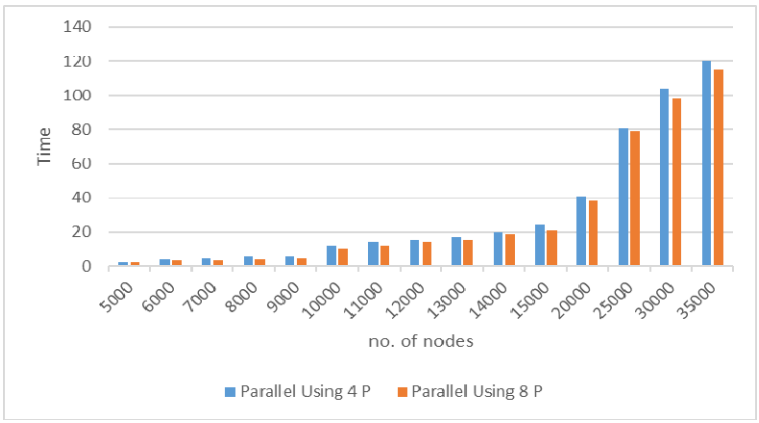


Figure 5: Improvement achieved by increasing number of processor used for calculate MaxFlow from 4 processor to 8 processor.

https://sites.google.com/site/ijcsis/
ISSN 1947-5500

5.1 Speedup Evaluation

The speedup ration refer to relation between sequential running time and parallel running time using different number of processor, through our implementation we use different number of processor start from 2 processor until 32 processor, and each time 2 processor was increased, from figure below we can show great speedup was shown when transfer from sequential execution to parallel using 2 processor.

Speedup = Sequential execution time \ parallel execution time

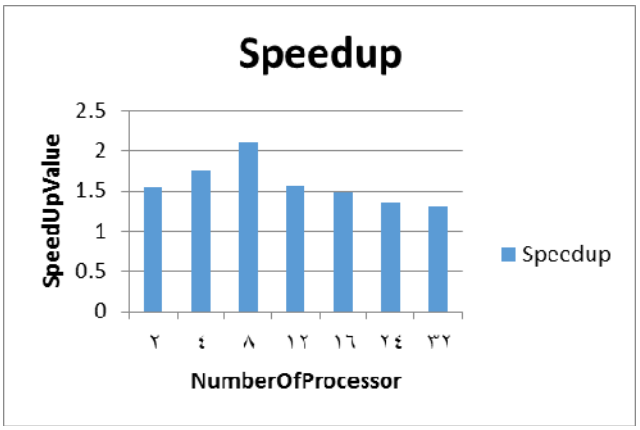


Figure 6: Speedup value based on different number of processor.

5.2 Parallel Implementation for MaxFlow CRO in multi core processor

Parallel implementation for algorithm using different number of CPU cores have other enhancement on time needs for calculating maximum flow value from source node to sink node. The idea of implementing algorithm on parallel on same CPU is the same as idea of parallel implementation using different processors, by dividing graph augmenting paths over different number of threads, each of these threads work on separated core of CPU cores. Through this step overall graph nodes will divided over different number of threads, each thread will calculate maximum flow value for its own nodes, through implementing MaxFlow CRO in parallel on multi core processor, good enhancement achieved on the level of time needs for execution from sequence version, the implementation of parallel over multi core was happened with different number of threads, first test was implemented using 2 threads and second one using 4 threads, result for both ones was compared with sequential version of algorithm, number of nodes which used for testing multi core version of algorithm implemented on different graph size started from graph with 5000 node and each time we increase number of nodes by 1000 until maximum graph size that can run on parallel using multi core processor which refer to 14500 node. parallel version using multi threads over multi core processor was implemented using C programming language, on Intel Core I7-3632 QM CPU@2.20 GH with 8 GB internal memory.

implementing algorithm in parallel using multi core allow to exploitation different percents of CPU resources, the following graphs present the amount of CPU usage based on number of threads and cores used for execution MaxFlow CRO algorithm in parallel, First graph present the percent of CPU usage when implement algorithm sequentially, just amount 12% to 15% of CPU was investigated, as we show in figure 6 from lower screen shoot.

https://sites.google.com/site/ijcsis/
ISSN 1947-5500

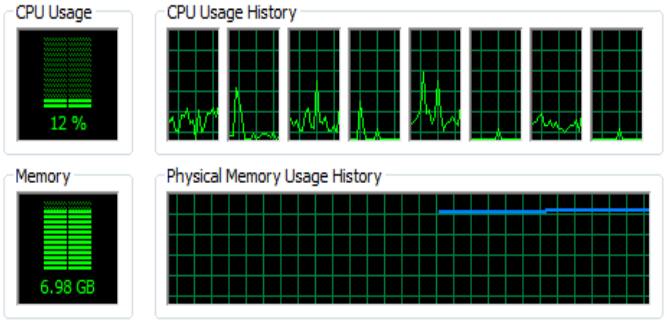


Figure 7: CPU usage for running MaxFlow CRO Sequentially.

when number of threads used for execution MaxFlow CRO was increased to run in parallel using 2 thread, the percent of CPU usage was different to be from 25% to 28%, this will decrease ideal percent of CPU, as shown in figure 7.

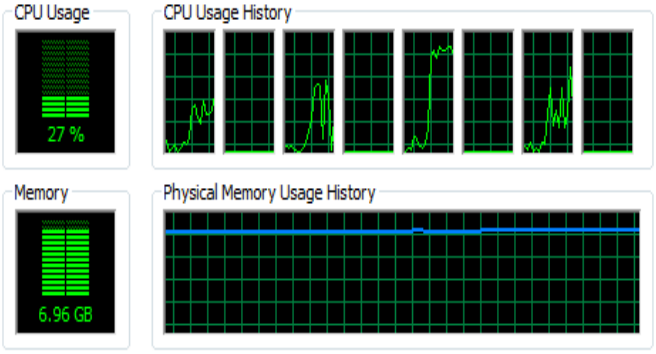


Figure 8: CPU Usage For Running MaxFlow CRO in Parallel using 2 threads.

if the number of thread used for execution MaxFlow CRO in parallel is 4 then percent of CPU usage will be increased to be from 48% to 52%, this will decrease ideal percent of CPU usage as show in figure 8.

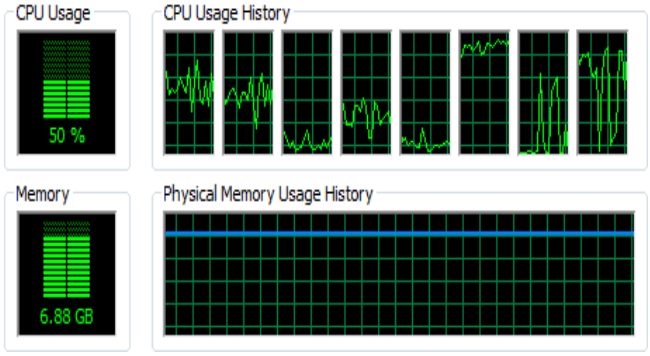


Figure 9: CPU Usage For Running MaxFlow CRO in Parallel Using 4 Threads.

table 2 present results for implementing MaxFlow CRO algorithm in parallel using multi threads on multi core processor, great enhancement was achieved when implement algorithm in parallel using multi threads, especially when transfer from sequential using single thread to parallel using two threads, and 4 thread as shown from table and figure bellow.

https://sites.google.com/site/ijcsis/
ISSN 1947-5500

Table2: Results for implementing algorithm in sequence and parallel using multithreads

Graph Size	SEQ	2TH	4TH	6TH
5000 Node	5	3	2	2
6000 Node	7	6	5	5
7000 Node	10	7	6	6
8000 Node	12	9	8	7
9000 Node	14	10	9	9
10000 Node	23	21	19	19
11000 Node	24	19	16	16
11500 Node	29	19	16	16
12000 Node	61	48	46	48

As we show from upper graph, that there are great enhancement from sequential time need to calculate maximum flow value and parallel time needs, especially when moved from sequential execution to parallel execution using 2 threads, and other enhancement found when moved from parallel using 2 threads to parallel using 4 threads, but the level of enhancement when moved from sequential to parallel using 2 thread is greater than enhancement achieved when moved from parallel using 2 threads to parallel using 4 threads, as we will show in figure 9 which present time need for each execution.

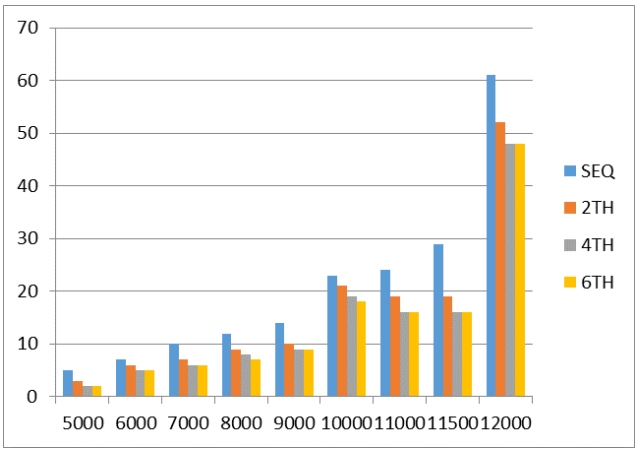


Figure 10: Time Need for execution using different number of threads.

From figure 9 we note that level of enhancement will decreased when number of thread used from implementation was increased, this refer to communication delay that happened when number of threads increased, because data or number of node will be divided over number of threads, this lead to communication time greater than processing time, for that enhancement was related to specific number of threads.

6. Conclusion

In this paper, we present the idea of implementing Maximum Flow using CRO in parallel using Multi processor in real distributed system "Iman1", and parallel using multi threads in multi core processor with shared memory. The sequential and parallel results are compared in multi core system and distributed. The

https://sites.google.com/site/ijcsis/
ISSN 1947-5500

evaluation of different algorithms is based on number of nodes on the graph and number of processors used for execution in parallel system "Iman1", and number of threads on parallel algorithm using multi core processor.

The results show a great improvement in implementing MaxFlow using CRO algorithm in parallel with small number of processers or threads for large number of graph node greater than 5000 vertices, this improvement was related to specific number of processor or threads, after this limited number the result was revered, the amount of enhancement achieved when moved execution from sequential to parallel using 2 processor is about 48% of time needs for execution was decreased , and about 21% of execution time was decreased when moved from execution using 2 process to execution using 4 process, and improvement achieved for execution in parallel using 2 thread have 36% of improvement for time needs for execution.

References

[1] T. H. Cormen, C. E. Leiserson, R. L. Rivest and C. Stein, Introduction to Algorithms, 3rd ed., The MIT Press, 2009.

[2] A.Y.S. Lam, V.O.K. Li, "Chemical reaction optimization: a tutorial", Memetic Computing 4, 2012, pp. 3–17.

[3] Ford jr., L.R., Fulkerson, D.R., "Maximal flow through a network". Can. J. Math. 8(3), 1956, pp. 399-404.

[4] Edmonds, J., Karp, R.M., "Theoretical improvements in algorithmic efficiency for network flow problems". J. Assoc. Compute. Machinery 19(2), 1972, pp. 248-264.

[5] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin., Network Flows: Theory, Algorithms, and Applications, Prentice Hall, 1993.

[6] Goldberg, A.V., "A new max-flow algorithm". Technical Report MIT/LCS/ TM-291, Laboratory for Computer Science, M.I.T, 1985.

[7] Goldberg, A.V., Tarjan, R.E., "A new approach to the maximum flow problem". Proc. 18th ACM STOC, 1986, pp. 136-146.

[8] J. B. Orlin. Max flows in o(nm) time, or better. In STOC'13: Proceedings of the 45th Annual ACM Symposium on the Theory of Computing, 2013, pp.765–774.

[9] V. King, S. Rao, and R. Tarjan, "A faster deterministic maximum flow algorithm", In Proceedings of the 8th Annual ACM–SIAM Symposium on Discrete Algorithms, 1992, pp. 157–164.

[10] Y. S. Lam and V. O. K. Li, "Chemical-reaction-inspired metaheuristic for optimization", IEEE Trans EvolComput vol. 14, no. 3, 2010, pp. 381– 399.

[11]Sheta, A. F. "Estimation of the COCOMO Model Parameters Using Genetic Algorithms for NASA Software Projects". Journal of Computer Science 2 (2), 2006, pp. 118–123.

[12] R.Barham, A.Sharieh, A.Sliet. "Chemical Reaction Optimization for Max Flow Problem", (IJACSA) International Journal of Advanced Computer Science and Applications, Vol. 7, No. 8, 2016.

[13] Qatawneh Mohammad. "Adaptive Fault Tolerant Routing Algorithm for Tree-Hypercube Multicomputer", Journal of Computer Science 2 (2): 124-126, 2006.

[14] Qatawneh Mohammed. "Embedding linear array network into the tree-hypercube network", European Journal of Scientific Research 10(2): 72-76, 2005.

https://sites.google.com/site/ijcsis/
ISSN 1947-5500

[15] Mohammad Qatawneh. "Multilayer Hex-Cells: A New Class of Hex-Cell Interconnection Networks for Massively Parallel Systems", Int. J. Communications, Network and System Sciences, 2011, 4, 704-708.

[16] Mohammad Qatawneh. "New Efficient Algorithm for Mapping Linear Array into Hex-Cell Network", International Journal of Advanced Science and Technology, 2016, 90, 9-14.

https://sites.google.com/site/ijcsis/
ISSN 1947-5500