Modern Applied Science; Vol. 14, No. 3; 2020
ISSN 1913-1844 E-ISSN 1913-1852
Published by Canadian Center of Science and Education

Parallel Whale Optimization Algorithm for Maximum Flow Problem

Raja Masadeh[1], Abdullah Alzaqebah[1], Bushra Smadi[2] & Esra Masadeh[3]

[1]Computer Science Department, the World Islamic Sciences and Education University, Amman, Jordan

[2]Computer Science Department, the University of Jordan, Amman, Jordan

[3]Independent researcher, Amman, Jordan

Correspondence: Raja Masadeh, Faculty of Information Technology, The World Islamic Sciences and Education University, Amman, Jordan.

Received: January 17, 2020 Accepted: February 18, 2020 Online Published: February 19, 2020

doi:10.5539/mas.v14n3p30 URL: https://doi.org/10.5539/mas.v14n3p30

Abstract

Maximum Flow Problem (MFP) is considered as one of several famous problems in directed graphs. Many researchers studied MFP and its applications to solve problems using different techniques. One of the most popular algorithms that are employed to solve MFP is Ford-Fulkerson algorithm. However, this algorithm has long run time when it comes to application with large data size. For this reason, this study presents a parallel whale optimization (PWO) algorithm to get maximum flow in a weighted directed graph. The PWO algorithm is implemented and tested on datasets with different sizes. The PWO algorithm achieved up to 3.79 speedup on a machine with 4 processors.

Keywords: Maximum Flow Problem (MFP), meta-heuristic, optimization, parallel, Whale Optimization Algorithm (WOA)

1. Introduction

In this study, a parallel algorithm for solving the maximum flow problem (MFP) based on a meta-heuristic approach called a whale optimization algorithm (WOA) by (Mirjalili & Lewis, 2016) is proposed. MFP is a well-known issue from various issues of optimization in the weighted directed network (Topcoder - Community - Maximum Flow Section 1&2). It can be applied to many applications such as networks, engineering, and transportations (Munakata & Hashier, 1993). Many solutions were presented and suggested by researchers to solve MFP using different techniques (Tarjan,1986, Ford and Fulkerson, 1956). The problem of MF is to determine the optimum solution for a directed and weighted graph; where the weight indicates to the flow capacity at each edge that communicates two vertices (Topcoder - Community - Maximum Flow Section 1&2). Based on that, the main target is obtaining the maximum value of flow from the source to the sink (Community - Maximum Flow Section 1&2 – Topcoder).

A directed and weighted flow network is defined as G = (V, E), where V indicates to the set of vertices and E points to set of edges (Cormen, 2009). Thus, a flow capacity (C_{uv}) is represented by the weight at each edge which is non-negative C (u, v) ≥0; where *u* and *v* belong to *V* (Cormen, 2009). Moreover, the network has two main vertices: the source *s* and the sink *t*. The source is the inception vertex and the sink is the intention vertex (Cormen, 2009). Assume a graph G (V, E) contains a vertex called *X*, and all edges connected with vertex *X* are called E(X), suppose that $F_{uv} = \max\{C_{uv}: (u,v) \in E\}$. Thus, the main issue is to get the optimal solution for a specific directed graph wherever every edge has a capacity. According to these restrictions, the aim is to find the maximum aggregate flow that directed to the sink (Goldberg & Tarjan, 1988). This scenario is mathematically modeled by Equation (1) (Cormen, 2009).

$$\text{Max } f(X) = \sum_{(u,v)\in E(s)}^{n} X_{uv}$$

Where

$$\sum_{\{v:(u,v)\in E)} X_{uv} - \sum_{\{v:(v,u)\in E)} X_{vu} = 0; \text{For all } u \in V\{s,t) \quad (1)$$

$$0 \le X_{uv} \le F_{uv}; \text{For all } (u,v) \in E.$$

Where, the flow on the edge (u, v) is represented by X_{uv}.

Currently optimization algorithms are gaining significant popularity and perform great significant roles in many various applications, mainly because they are founded on easy notion resulting from their imitation of natural perceptions and ideas which are nature-gained (Mirjalili & Lewis, 2016). Consequently, they are also perceived easy to apply and can be of significance in a variety of problem settings. The Whale Optimization Algorithm (WOA) is one of these optimization algorithms, and currently suggested in (Mirjalili & Lewis, 2016) and inspired from the foraging and hunting mechanism of the humpback whale in nature.

Most meta-heuristic optimization algorithms come up in the previous decades (Kirkpatrick et al., 1983). There are three main categories of these meta-heuristic optimization algorithms: physics-based, evolutionary and swarm intelligence algorithms (Kirkpatrick et al., 1983, Masadeh et al., 2018). The physics-based mechanisms were derivative from physics-based rules (Alatas, 2011, Masadeh et al., 2019). Evolutionary meta-heuristic mechanisms were stimulated from nature (Holland, 1992, Alzaqebah et al, 2018). The swarm intelligence algorithms were derived from the swarms' societal intelligent attitude (Dorigo et al., 2008).

The behavior of humpback whales is simulated in the WOA (Mirjalili & Lewis, 2016), which imitates the hunting behavior of the humpback whales. The main stages of this behavior are the investigative phase of looking for the prey and surrounding it, and the exploitation phase which signifies a bubble-net method and confronting the prey.

Recently, parallel computing comes to be an affecting solution to surpass computation speed. In the sector of supercomputing, there will emerge the need to analyze big networks having billions of vertices and edges consistently. Due to this, there is a need to make MF solvers be faster and be capable of operating on immense networks. The traditional MF algorithm runs on one processor. The parallel algorithms can be applied on several processors to solve MFP faster than in single processor. Therefore, this study is a representative of a probable equivalent resolution to the MFP by using the WOA. This will be by concurrently grouping the search space to establish the MF for the individual cluster (local MF) in the pursuit of finding the general solution (global MF) of the chosen graph. By assuming that the graph is the search space and that the whales are looking to get the prey, the WOA solves MFP. In this setting, the sink in the network indicates to the prey and the vertices represent the whales.

In this research, a parallel WOA was proposed in order to solve the MFP by clustering the search space to collect the MF for each cluster simultaneously for the purpose of finding the global MF of the desired graph. This research is organized as follows: Section 2 presents related works and background on this research. A description of the WOA in details is presented in Section 3. Section 4 describes the proposed Parallel-MaxFlow-WOA. The complexity analysis of proposed algorithm is outlined in Section 5. Section 6 presents an example of the proposed technique. The simulation results are argued out in Section 7. Finally, the conclusion and future work are drawn in Section 8.

2. Related Work

It has come to the attention of many researchers that the MFP involves quite a wide range of concerns all which need to be studied in different ways (Ford & Fulkerson, 1956, McHugh, 1990). Ford and Fulkerson (FF) method was the first to be applied in 1956 (Ford and Fulkerson, 1956). It is significant in getting the MF from the origin to a destination by applying augmenting path algorithm. Dinic (Dinic, 1970) and Cormen (Cormen, 2009) discovered that the run time complexity performance of the algorithm is O (mn) augmentation steps if every augmenting path in the shortest one; where m is the count of arcs while n denotes the number of nodes (Cormen, 2009, Dinic, 1970). Edmonds and Karp presented work which was close to that of Dinic's algorithm using some new algorithm (Dinic, 1970, Edmonds & Karp, 1972). The new algorithm acknowledged that the shortest path was that the dimension of the arc equals one; this kind of results was attained using the Breadth First Search (Eppstein, 1998, Leiserson et al., 2001).

More algorithms have been established with time to solve the MFP. There was an adapted of a new algorithm to work out the shortest augmenting path algorithm (Leiserson et al., 2001, Nemhauser & Wolsey, 1989). Orlin's study (Orlin, 2013) on sparse network exhibited ameliorated polynomial time algorithm (Orlin, 2013). Moreover, the researcher presented ways of finding a solution for the MFP in O (mn); where m = O (n), as well as, resolving the MF in O ($nm+m^{31/16}\log^2 n$) time, where m= O($n^{1.06}$). (King et al., 1994) describe an improvement to this algorithm. They find a solution for the MFP in O ($nm\ log\ m$/ (n log n) n) time.

Genetic algorithm (GA) is seen to be difficult algorithm to be useful in MFP in most of the graph problem because the graph problems are recognized by its multiple unique characteristics (Munakata & Hashier, 1993). However, the GA was applied to establish the MF from the source to the sink in a weighted directed graph by (Munakata & Hashier, 1993). Every answer is executed by a specific flow matrix in the previous study. In the

fitness function, there are two major characteristics, which include balancing nodes and the concentration levels of the flow. The initial generation population is selected randomly. After applying the GA, the resulted generation is then improved. The results will be optimal or near optimal after a specific count of iterations of utilizing the genetic algorithm.

Lam and Li suggested Chemical Reaction Optimization CRO (Lam et al., 2010), which is a meta-heuristic algorithm that was mainly intended for solving combinatorial optimization problems. The run time complexity of CRO algorithm for MFP is O (I E^2) were presented by (Barham et al., 2016). Where, *I* indicates to the iterations number and *E* denotes the edges count in the flow graph. The MaxFlow-CRO algorithm is designed to identify the best MF that can be relocated from the source to the sink in the flowing network with no restraints for the capacity and violation, where the flow in each arc rests in the upper capacity boundary.

Grey Wolf Optimization (GWO) algorithm was suggested by (Mirjalili et al., 2014), which is a meta-heuristics algorithm that is proposed as a resolution to the combinatorial optimization problems too. Masadeh (Masadeh et al., 2017) implemented a GWO algorithm to solve the MFP in O ($|N| + |E|^2$) time.

A parallel genetic algorithm (PGA) was presented by (Surakhi et al., 2017). This would be done by solving each augmenting route in the elected graph from the origin to the sink in equivalent stages during iterations. By using the Message Passing Interface (MPI) library the PGA is implemented. Similarly, the study is realized and shown results from real distributed system IMAN1 supercomputer. Additionally, the study's outcomes present recommendable enhancement in regard to speedup efficiency. In addition, PGA makes the running time faster by realizing up to 50% corresponding efficiency.

A contrast study concerning the performances of GA and CRO algorithms in resolving MFP associated with FF algorithm that is known algorithm for explaining MFP is presented by (Khanafseh et al., 2017). Establishing which algorithm will come up with results roughly closer to FF with the same accuracy is the main aim of the study. Therefore, the outcomes of both algorithms that discussed in their study may be useful in solving MFP with accurateness nearly close to FF results. However, it is better when GA in relation to time and accuracy is applied.

WOA was adapted to solve the MFP by (Masadeh et al., 2018). The MFP is solved by the WOA by assuming the graph is the search space which the whales seek to reach their prey. The prey is the sink in the network, represented by the graph, while other whales are signified in other nodes within the graph. The experimental outcomes of the proposed algorithm indicated that the technique can reinforce its performance while solving the MFP.

According to (Masadeh et al., 2018), the WOA is employed to solve MF problem in sequential manner. This is to make the computation faster and without degrading the accuracy. The parallel MaxFlow WOA is proposed in this research.

3. Whale Optimization Algorithm

Whale Optimization Algorithm (WOA) is a random optimization algorithm that was created by (Mirjalili & Lewis, 2016). The WOA aims to locate global optimum of a search agents (whales) population in a problem. The first step in the search process starts with creating randomly multiple candidate solutions of a specific problem. Then, through numbers of iterations the candidate solutions are improved until reaching a satisfied solution set. The Whales in reality mimic a bubble-net feeding method which is a private hunting technique as shown in Figure 1 (Mirjalili & Lewis, 2016).

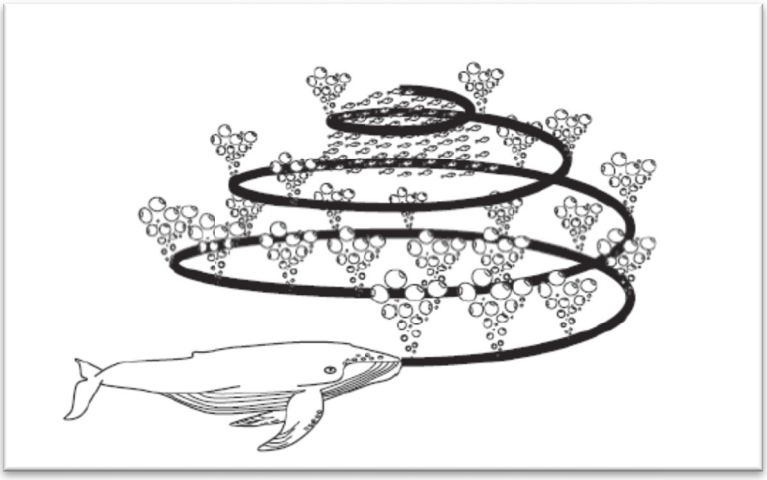

Figure 1. Bubble-net hunting behavior

As shown in Figure 1, two main techniques are used in WOA. The first is a humpback whale produce ambush around the victims in a spiral shape, after that it generates bubbles all the way ahead. This technique is the main idea of the whale optimization algorithm in the search process. The second is the encircling technique that is used in the WOA, whereas the victims are surrounded by the humpback whales so they can start hunting using the bubble- net technique which is foraging mechanism (Mirjalili & Lewis, 2016, Watkins & Schevill, 1979). The mathematical representation of selecting between either the shrinking encircling mechanism or the spiral model in order to update the locations of humpback whales over optimization is shown in Equation (2).

$$\vec{X}(t+1) = \begin{cases} \vec{X} * (t) - \vec{A}.\vec{D} & ,0 \leq p < 0.5 \\ D'.e^{bl}.Cos\,(2\,\pi\,l) + \vec{X} * (t) & ,1 \geq p \geq 0.5 \end{cases} \quad (2)$$

Where, p indicates to a random number between [0, 1], b represents a constant value for defining the shape of the logarithmic spiral, and l refers to a random number between [−1, 1], t is the current iteration, and $D' = |X^*(t) - X(t)|$ which mentions the distance between the ith whale and the prey.

The first stage of WOA Equation is the foraging mechanism that mimics the bubble-net mechanism, and then becomes the second stage which is the encircling technique. The p variable in the two phases runs with a similar probability. The mentioned Equations are shown in Figure 2 (Mirjalili & Lewis, 2016).

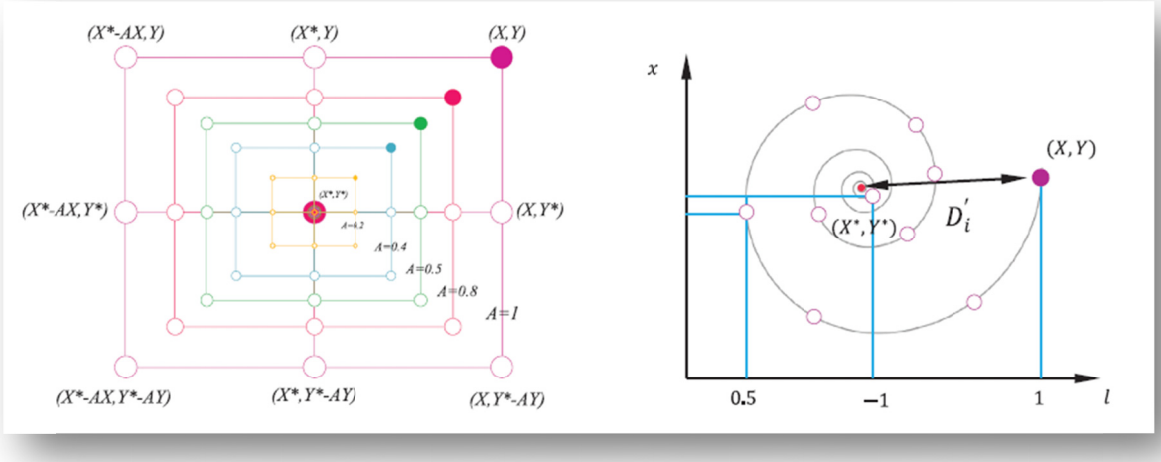

Figure 2. Mathematical models for prey encircling and bubble-net hunting, where (X, Y) is the location of the whale and (X*, Y*) is the location of the prey.

Mainly, there are two optimization algorithm phases used in population based algorithms which are exploitation and exploration phases. WOA uses both phases substituted as r and M in the major Equation.

$$\vec{D} = \left|\vec{M} . \overrightarrow{X^*}(t) - \vec{X}(t)\right| \quad (3)$$

$$\vec{X}(t+1) = \overrightarrow{W^*}(t) - \vec{N} . \vec{D} \quad (4)$$

$$\vec{N} = 2\vec{r} . \vec{n} - \vec{r} \quad (5)$$

$$\vec{M} = 2.\vec{n} \quad (6)$$

For a given problem, random solutions are created by WOA in order to optimize this problem. At each iteration, search agents update their positions rely on randomly selected search agent or the search agents which will get so far. In order to ensure the exploration stage, update the positions of the other search agents which relies on the best solution which mentions the pivot point when |A| is bigger than 1. Whereas, the other case is when |A| is less than 1; where the best solutions plays another role with the pivot point. However, in exploration phase the position of a search agent is updated according to a randomly selected search agent until find the best search agent. This technique allows WOA to proceed a global search. The mathematical model is represented as shown in Equation (7) and Equation (8). Algorithm 1 shows the pseudocode of the WOA [Mirjalili & Lewis, 2016].

$$\vec{D} = \left|\vec{M} . \overrightarrow{X_{rand}}(t) - \vec{X}(t)\right| \quad (7)$$

$$\vec{X}(t+1) = \overrightarrow{X_{rand}}(t) - \vec{N} . \vec{D} \quad (8)$$

Initialize the whales population X_i (i = 1, 2, 3, …, n)
Initialize r, M and N
Calculate the fitness of each search agent
X* = the best search agent
While t ≤ Max_iteration **do**
  For each search agent **do**
    If |N| ≤ 1 **then**
      Update the position of the current search agent by the Equation (2).
    Else if |N| ≥1 **then**
      Select a random search agent X_{rand}
      Update the position of the current agent by Equation (8)
    End if
End for
Update r, M and N
Update X* if there is a better solution
t = t + 1
End while
Return X*

Algorithm 1. Pseudocodes of WOA

4. Proposed Parallel-MaxFlow-WOA

The proposed algorithm was to improve the WOA in solving MFP by using the parallel approach. Thus, when clustering the search spaces alternating each cluster processing sequentially, all the clusters proceed simultaneously, hoping for reduction overall running.

4.1 Initialization Phase

Starting with a random population of whales represents the vertices; one of them is selected at random to represent the source and the prey is selected to denote the sink. In this stage; the distance between the whole

whales and all X_{rand} are computed in order to help the whales to assign to the nearest X_{rand} and to be a member of its cluster.

4.2 Fitness Function

The main mission of X_{rand} is to create bubble-net when it sees the prey. Thus, all the whales that are in the search space and see the bubble-net will associate its cluster. Thereafter, the members that joined the cluster should update their locations toward the X_{rand}. Moreover, the locations of the whales will update depending on the value of *A*. In case A is less than one, the whale is still associating the cluster and updates its location. This means the whale moves towards the prey. When *A* is greater than one, the whale is not associating the cluster and should look for another near X_{rand} to join. This behaviour is modelled mathematically by Equation (9), Equation (10) and Equation (11) (Mirjalili and Lewis, 2016).

$$\vec{X}(t+1) = \vec{D} \cdot e^{bl} \cdot Cos\,(2\pi l) + \overrightarrow{X^*}(t) \tag{9}$$

Where $\vec{D} = |(X^*)\ \vec{t} - \vec{X}\ (t)|$, *b* is a constant for determining the form of the logarithmic spiral and *l* indicates to a random number in [−1, 1] according to (Mirjalili & Lewis, 2016).

$$\vec{X}(t+1) = \overrightarrow{X_{rand}} - \vec{A} \cdot \vec{D} \tag{10}$$

Where ($\vec{X}$) is a random position vector, ($\overrightarrow{X_{rand}}$) is the best search agent.

$$V = h * r * \pi\,(\tfrac{1}{3}) \tag{11}$$

Where *r* indicates the radius of the bubble-net, *h* represents the height of bubble-net that is chosen randomly between 6 and 12 as its inspiration in nature (Watkins & Schevill, 1979) and π is a constant value that is approximated to 3.14.

4.3 Clustering Phase

Each cluster has X_{rand} that is selected randomly at the beginning, then fitness function will be computed for each whale in order to inspect if it catches sight of the bubble-net. Thus, the whole clusters will compute their local maximum flows simultaneously. In other words, all clusters are working at the same time to get their local maximum flow, then get the global maximum flow at them sink.

4.4 Maximum Flow Function

In this stage, each cluster will calculate its maximum flow (MF) by invoking max-flow function which is proposed by Ford-Fulkerson (Ford & Fulkerson, 1993) that is based on augmenting paths in the given residual flow network to find the MF from the source to the sink. In other words, each cluster works as an autonomous subnetwork, and each one computes the MF by using FF algorithm simultaneously. This technique will get back the local MF for each cluster. The MF function will create N numbers of clusters and each one will calculate its local MF. Thus, the algorithm will return the global MF of the given network using the following Equation (12).

$$Global\ Max\ Flow = \sum_{k=1}^{N} (local\ max\ flow)_k \tag{12}$$

5. Complexity Analysis of Parallel-MaxFlow-WOA

The proposed algorithm works by clustering the graph space into N clusters, each cluster represents a subgraph of the original graph, and then the Ford Fulkerson algorithm employs in order to find the maximum flow of the graph. In WOA, the FF algorithm will be run N times, according to the number of clusters, and the time complexity of FF is O ($|V| + |E|^2$) since the |V| indicates to the number of humpback whales (vertices) for clustering function and fitness function. O ($|E|^2$) is the time complexity of max-flow function, showing that the cost of finding the augmenting path is O (|E|). When the graph is complete, the Fm = |E|; where Fm indicates to the maximum flow. Therefore, the run time complexity of Maximum Flow function is O (|E| *Fm) = O ($|E|^2$).

The overall running time for sequential WOA is N* O ($|V| + |E|^2$) where *N* is the number of clusters. The parallel running time for Parallel-MaxFlow-WOA will be the maximum value of O ($|V| + |E|^2$) on all clusters which is MAX (O ($|V| + |E|^2$) ^{N}).

6. Example of Parallel-MaxFlow-WOA

In this subsection, an example is presented in order to clarify how the suggested algorithm is adapted the WOA to solve MFP. The following example is based on the given network shown in Figure 3 that has a set of nodes and a set of edges. Moreover, each edge has a capacity which is selected randomly (Masadeh et al., 2018).

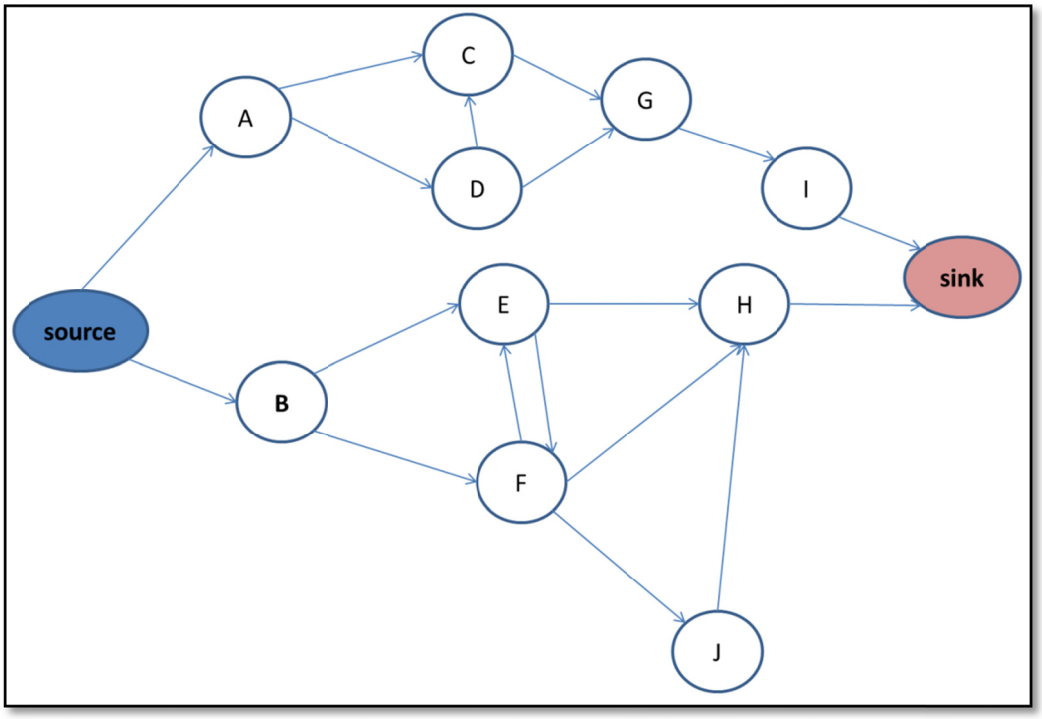

Figure 3. Main given network

6.1 Initialization Phase

As shown in Figure 3, the source and the sink were selected randomly; where the source indicates the whale and the sink represents the prey. In Figure 4, nodes A and B have the minimum distances to the source; thus these nodes represent X_{rands}. Which means each X_{rand} represent a cluster.

Each X_{rand} use the bubble-net mechanism using Equation (11) which displays the cone shape as shown in Figure 5 (Masadeh et al., 2018).

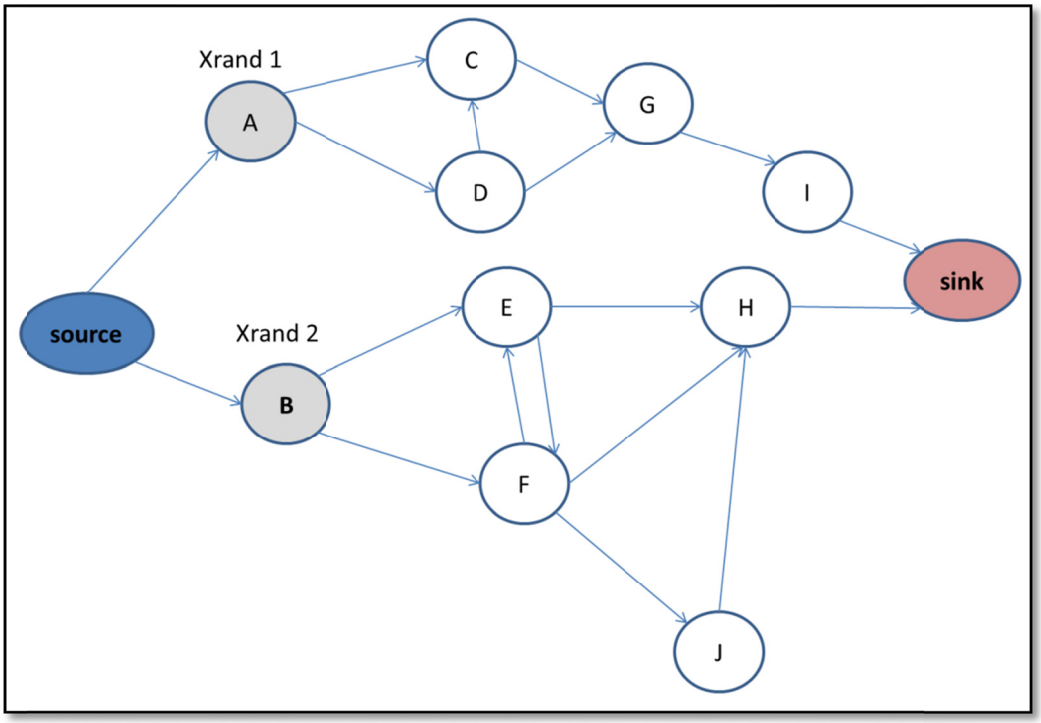

Figure 4. X_{rand} selection to identify cluster

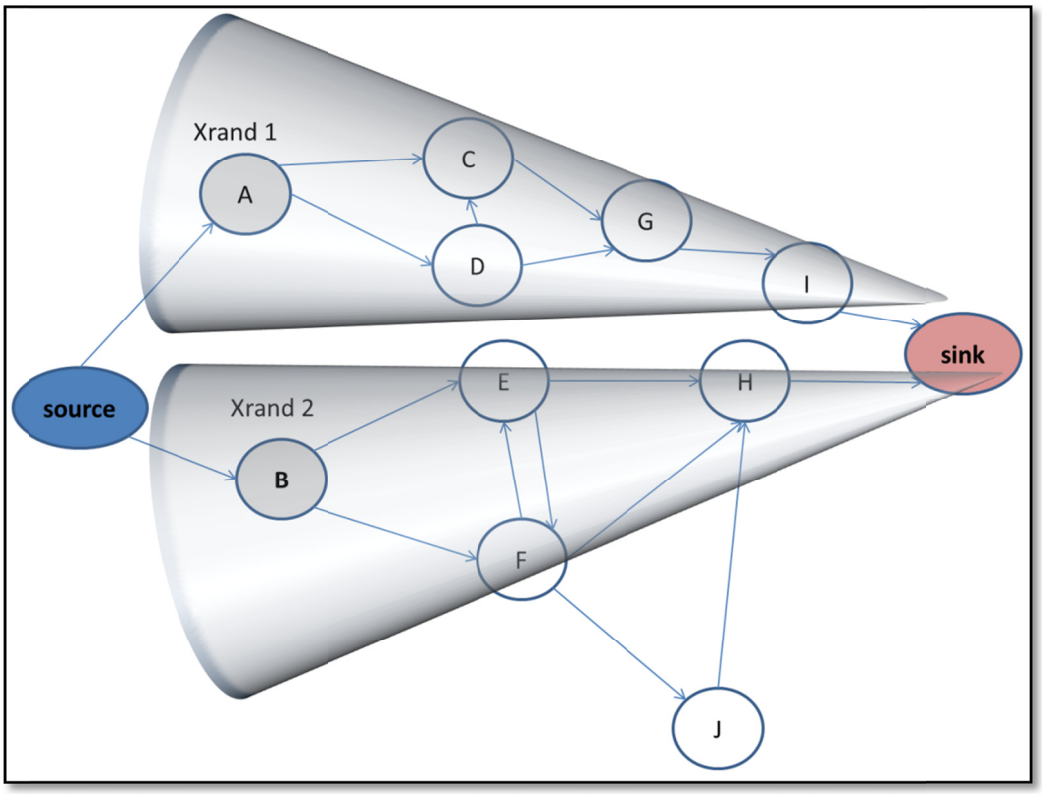

Figure 5. Each X_{rand} creates bubble-net to the sink

6.2 Fitness Function and Clustering Phases

When each cluster is created by X_{rand}, each node will designate the nearest cluster according to the fitness function. In this case many nodes join a cluster while other nodes do not join any cluster such as node J as shown in Figure 6.

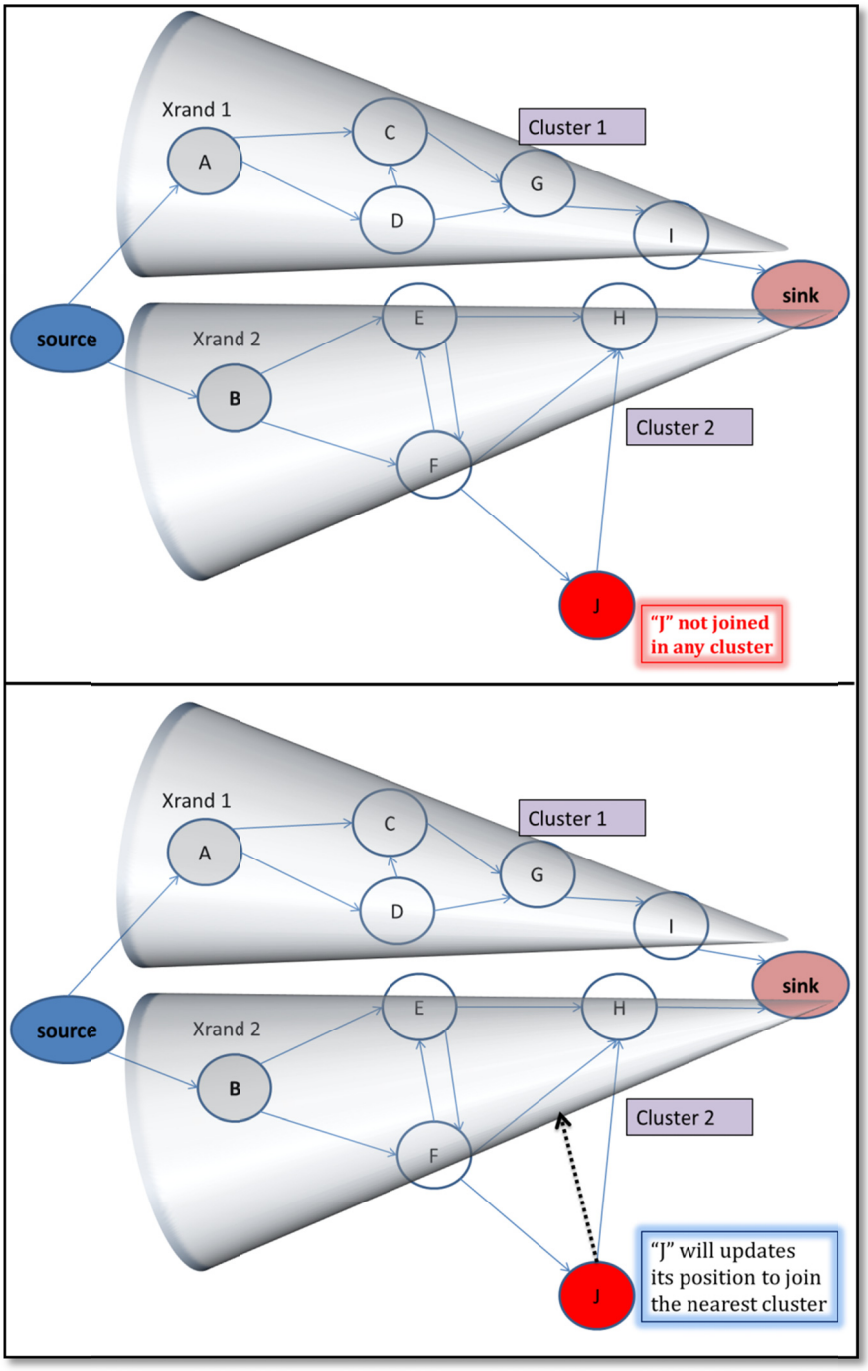

Figure 6. Clustered graph with a node out of bubble-net (cluster)

Each node will update its position based on the X_{rand}, especially each node that does not belong to any cluster in order to designate the nearest cluster such as node J in Figure 6.

As clearly shown in Figure7, node J has updated its position to join cluster 2 which is considered the nearest cluster to it.

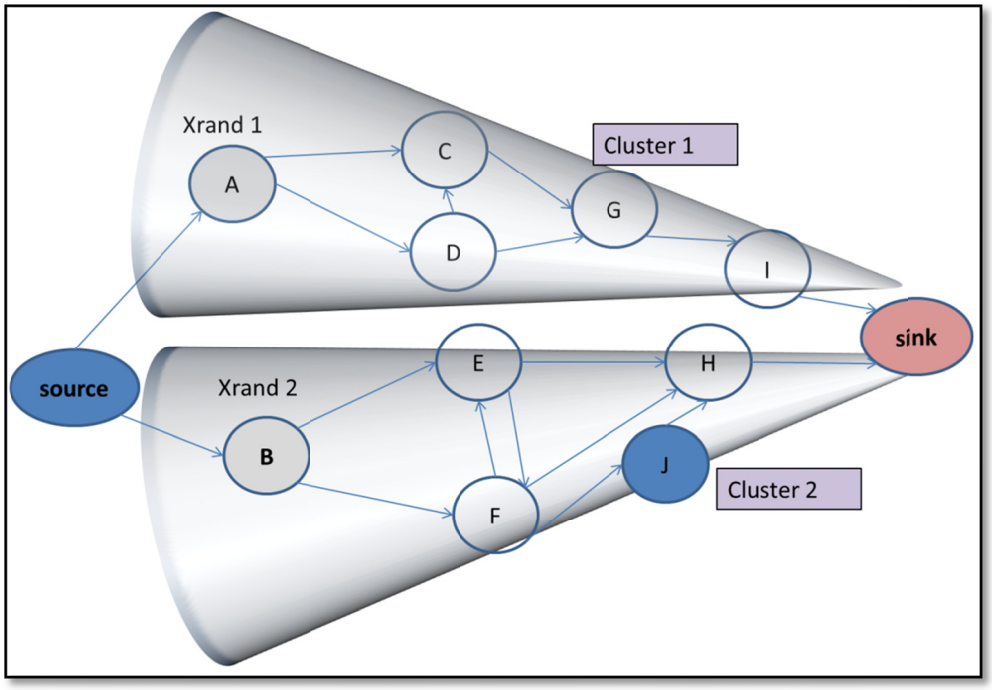

Figure 7. Updating position and joining the nearest cluster

6.3 Parallel-MF-WOA

Each node joins a cluster after updating its position and clustering phase. Therefore, each cluster will compute the MF from the source to the sink simultaneously by using Equation. (1) as shown in Figure 7. Thus, each cluster gets its local MF then sends it to the sink in order to calculate the global MF for the given network. In other words, the sink obtains set of local MFs as a number of clusters and computes the global MF using Equation. (10). Figure 8 shows the parallel process of MF.

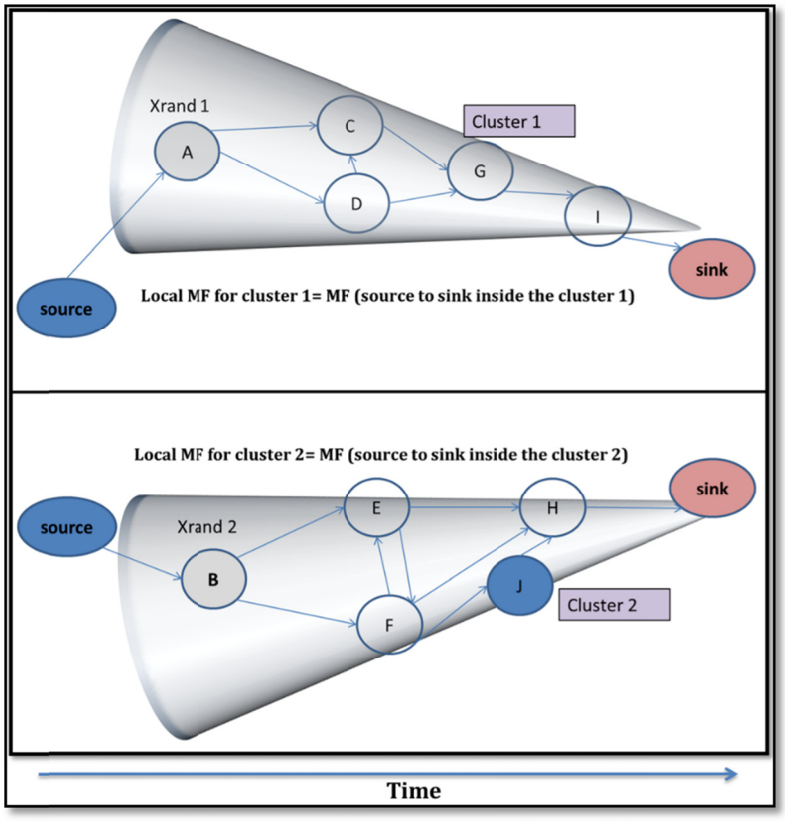

Figure 8. Parallel MF_WOA

7. Simulation Results of Parallel-WOA

Matlab R2016a software with Parallel Toolbox was employed in order to evaluate the performance of Parallel-WOA for solving MF problem. For sequential mode, only one worker (process) is used to do the computation. On the other hand, four workers (processes) were used in order to do computation in parallel. In addition, the sizes of the used datasets were between 1000 and 10000. Every scenario was generated 5 experiments to have their average run time.

The simulation platform is a portable computer with following specifications: Intel (R) core (TM) i7-4510U CPU with 2.40 GHz, 16 GB RAM and Windows 8.1, 64-bit operating system.

7.1 Data Sets

The data sets that used to assess the proposed technique were randomly generated graphs with different sizes and large number of edges in these graphs; for that each node (*i)* in the graph is connected to every other nodes that those distances from the source node is greater than the current one (*i)*; Table 1 shows the graphs that had been used in the experiments.

The proposed technique is based on clustering the nodes in the graph's space. The data partitioning is an important issue and should be kept into account to accurately compare sequential with parallel mode, for that the random graph generator was oriented to make the random generation smart in somehow to fairly distributing the data.

Table 1. Randomly generated Data sets

Graph number	*Number of Nodes*	*Number of Edges*
Graph1	1000	502,498
Graph2	2000	2,004,998
Graph3	3000	4,507,498
Graph4	4000	8,009,998
Graph5	5000	12,512,498
Graph6	6000	18,014,998
Graph7	7000	24,517,498
Graph8	8000	32,019,998
Graph9	9000	40,522,498
Graph10	10000	50,024,998

7.2 Computation Time

In this subsection, the computation time was examined which means the time that each processor spends in computing the MF values. The computation time depends on the number of processors and the input size. When the input size is large, it is better to have many processors to do the computation, especially in the proposed work different clusters used to represent the partition of the graphs, for that the number of clusters was determined by the number of processors that are available in simulation platform. For sequential mode only one processor runs to compute the MF value for each cluster in a sequential manner; on the other hand, four processors are used to make each processor works on individual cluster. Figure 9 shows the computation time that was taken by only one worker (processor) and four workers.

Figure 9. The computation time of running sequential MF-WOA algorithm and Parallel MF-WOA

As clearly shown in Figure 9, the computation time was minimized by using parallel mode with four processors because each processor works on partition of the data simultaneously. For instance, in the graph size of 10000, it takes approximately 844.8 seconds for one processor, where it takes approximately 222.7 seconds for four processors which show 3.79 speedup.

7.3 Communication Cost

Communication cost determines the number of communication steps needed to accomplish the computation process. Figure 10 shows the communication steps required for computing the MF values on varying dataset of

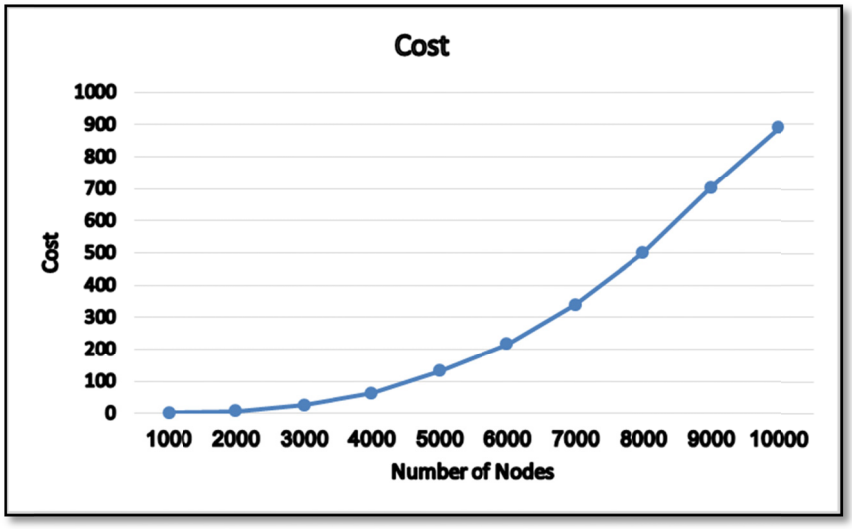

Figure 10. Communication cost for parallel WOA

7.4 Relative Speedup

In the proposed work, the speedup value is the ratio of computation time that required finishing the work on one processor over the computation time on four processors. Figure 11 shows the speedup values that gained from using parallel mode against sequential mode.

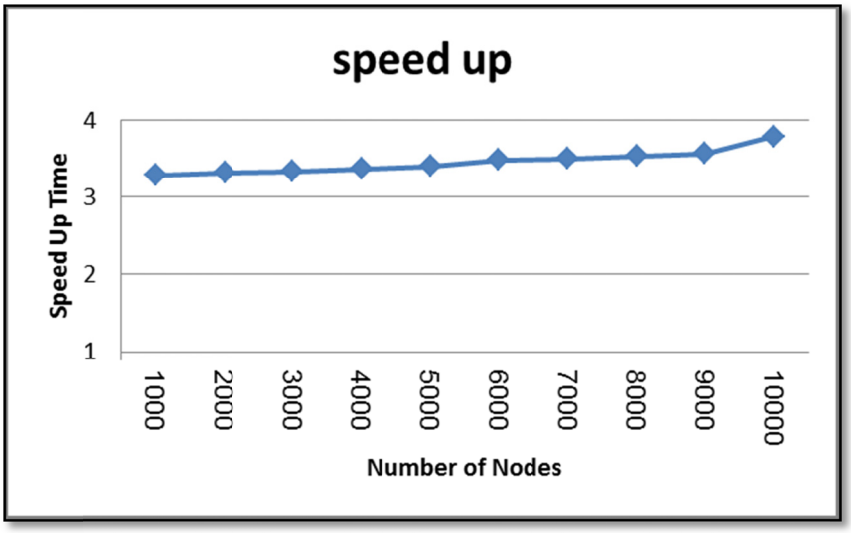

Figure 11. Speedup Values for sequential WOA against parallel WOA

As shown in Figure 11, the speedup values vary between 3 and 4 since using four processors. Thus, the computation time will be decreased by using parallel mode between 3 and 4 approximately and it is not exactly 4 because there are times taken into communications. The minimum value of speedup is approximately 3.3 on graph size 1000 nodes and the maximum one is approximately 3.8 on graph size 10000 nodes, because when the size of graph increased the computation time also increased too.

7.5 Relative Efficiency

Relative efficiency is a performance metric closely related to relative speed up. In the proposed work, it is a ratio of relative speed up that is computed in the previous subsection and the number four that represent the number of used processors. Figure 11 shows the speedup values that gained from using parallel mode against sequential mode. From Figure 12 the efficiency values vary between 0.82 and 0.95. This is a promise outcome as the efficiency measures amount of time that processors are usefully utilized, which means that the processors are approximately usefully used, it ranges between 82% and %95.

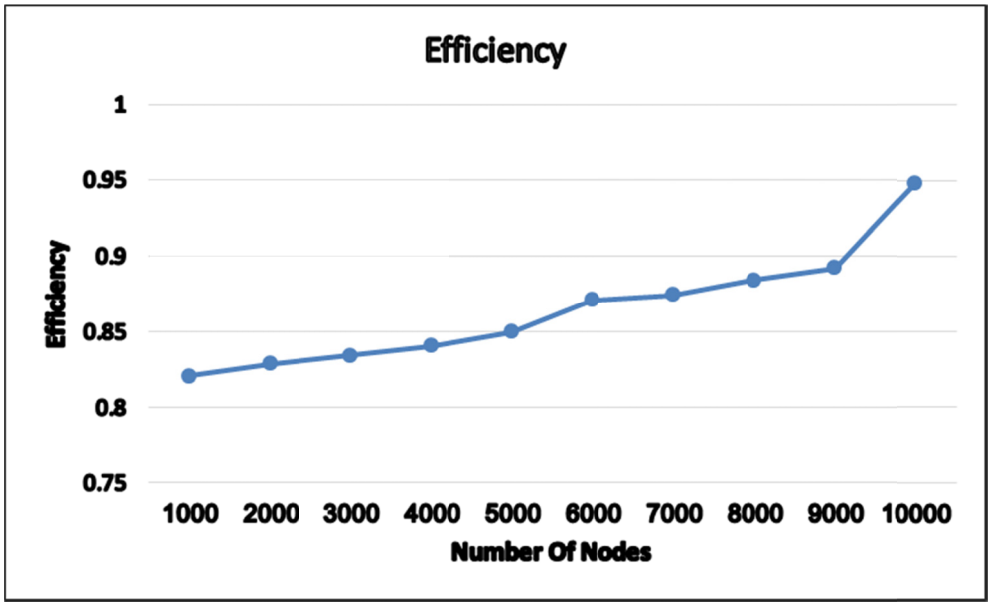

Figure 12. Efficiency for parallel WOA

8. Conclusion and Future Work

The maximum flow problem is an essential problem in network flow theory and it has been studied deeply from different aspects. In this study, we presented a parallel whale optimization algorithm for solving the maximum flow problem, which is simple to implement. Different from the intuitive idea, the basic idea of the proposed algorithm is clustering the graph and finds the maximum flow for each cluster simultaneously as a step to find the overall maximum flow of the graph. The simulation result showed that the given parallel algorithm outperforms the sequential one and has a great enhancement of the computing time, since it achieves 3.79 of speedup.

As a future work, the proposed algorithm was tested on a single machine with multi-processors so for future apply the proposed algorithm on distributed machines to take the overall advantages of hardware.

References

Alatas, B. (2011). ACROA: artificial chemical reaction optimization algorithm for global optimization. *Expert Systems with Applications, 38*(10), 13170-13180. https://doi.org/10.1016/j.eswa.2011.04.126.

Alzaqebah, A., Masadeh, R. & Hudaib, A. (2018, April). *Whale optimization algorithm for requirements prioritization.* In 2018 9th International Conference on Information and Communication Systems (ICICS) (pp. 84-89). IEEE. https://doi.org/10.1109/iacs.2018.8355446.

Barham, R., Sharieh, A. & Sliet, A. (2016). Chemical reaction optimization for max flow problem. *IJACSA) International Journal of Advanced Computer Science and Applications, 7*(8), 189-196. https://doi.org/10.14569/ijacsa.2016.070826.

Cormen, T. H. (2009). *Introduction to algorithms.* MIT press.

Dinic, E. A. (1970). Algorithm for solution of a problem of maximum flow in a network with power estimation. *soviet math. doll. 11*(5), 1277-1280.

Dorigo, M., Birattari, M., Blum, C., Clerc, M., Stützle, T. & Winfield, A. (Eds.). (2008). *Ant Colony Optimization and Swarm Intelligence: 6th International Conference*, ANTS 2008, Brussels, Belgium, September 22-24, 2008, Proceedings (Vol. 5217). Springer. https://doi.org/10.1007/978-3-540-87527-7

Edmonds, J. & Karp, R. M. (1972). Theoretical improvements in algorithmic efficiency for network flow problems. *Journal of the ACM (JACM), 19*(2), 248-264. https://doi.org/10.1007/3-540-36478-1_4.

Eppstein, D. (1998). Finding the k shortest paths. *SIAM Journal on computing, 28*(2), 652-673. https://doi.org/10.1137/s0097539795290477.

Ford, L. R. & Fulkerson, D. R. (1956). Maximal flow through a network. *Canadian journal of Mathematics, 8*(3), 399-404. https://doi.org/10.4153/cjm-1956-045-5.

Goldberg, A. V. &Tarjan, R. E. (1988). A new approach to the maximum-flow problem. *Journal of the ACM (JACM), 35*(4), 921-940. https://doi.org/10.1145/48014.61051.

Holland, J. H. (1992). Genetic algorithms. *Scientific american, 267*(1), 66-73. https://doi.org/10.1038/scientificamerican0792-66.

Khanafseh, M. Y., Surakhi, O. M., Sharieh, A. &Sleit, A. (2017). A Comparison between Chemical Reaction Optimization and Genetic Algorithms for Max Flow Problem. *International Journal of Advanced Computer Science And Applications, 8*(8), 8-15. https://doi.org/10.14569/ijacsa.2017.080802

King, V., Rao, S. & Tarjan, R. (1994). A faster deterministic maximum flow algorithm. *Journal of Algorithms, 17*(3), 447-474. https://doi.org/10.1006/jagm.1994.1044

Kirkpatrick, S., Gelatt, C. D. &Vecchi, M. P. (1983). Optimization by simulated annealing. *science, 220*(4598), 671-680. https://doi.org/10.1126/science.220.4598.671

Lam, A. Y. & Li, V. O. (2010). Chemical-reaction-inspired metaheuristic for optimization. *IEEE Transactions on Evolutionary Computation, 14*(3), 381-399. https://doi.org/10.1109/tevc.2009.2033580

Leiserson, C. E., Rivest, R. L., Cormen, T. H. & Stein, C. (2001). *Introduction to algorithms* (Vol. 6). Cambridge, MA: MIT press.

Masadeh, R., Hudaib, A. & Alzaqebah, A. (2018). WGW: A hybrid approach based on whale and grey wolf optimization algorithms for requirements prioritization. *Advances in Systems Science and Applications, 18*(2), 63-83.

Masadeh, R., Mahafzah, B. A. & Sharieh, A. (2019). Sea Lion Optimization Algorithm. *(IJACSA) International Journal of Advanced Computer Science and Applications, 10*(5), 388-395. https://doi.org/10.14569/ijacsa.2019.0100548

Masadeh, R., Alzaqebah, A. & Sharieh, A. (2018). Whale optimization algorithm for solving the maximum flow problem. *Journal of Theoretical & Applied Information Technology, 96*(8), 2208-2220. https://doi.org/10.1109/iacs.2018.8355446

Masadeh, R., Sharieh, A. &Sliet, A. (2017). Grey wolf optimization applied to the maximum flow problem. *International Journal of Advanced and Applied Sciences, 4*(7), 95-100. https://doi.org/10.21833/ijaas.2017.07.014

McHugh, J. A. (1990). *Algorithmic graph theory (Vol. 68056)*. New Jersey: Prentice Hall, 1-85. https://doi.org/10.5860/choice.27-4552

Mirjalili, S. & Lewis, A. (2016). The whale optimization algorithm. *Advances in Engineering Software, 95*, 51-67. https://doi.org/10.1016/j.advengsoft.2016.01.008

Mirjalili, S., Mirjalili, S. M. & Lewis, A. (2014). Grey wolf optimizer. *Advances in engineering software, 69*, 46-61. https://doi.org/10.1016/j.advengsoft.2013.12.007

Munakata, T. &Hashier, D. J. (1993, July). *A Genetic Algorithm Applied to the Maximum Flow Problem.* In ICGA (pp. 488-493).

Nemhauser, G. L. & Wolsey, L. A. (1989). Chapter vi integer programming. *Handbooks in Operations Research and Management Science, 1*, 447-527. https://doi.org/10.1016/s0927-0507(89)01007-8

Orlin, J. B. (2013, June). Max flows in O (nm) time, or better. *In Proceedings of the forty-fifth annual ACM symposium on Theory of computing*, 765-774. ACM. https://doi.org/10.1145/2488608.2488705

Surakhi, O. M., Qatawneh, M. & Hussein, A. (2017). A Parallel Genetic Algorithm for Maximum Flow Problem. *International Journal of Advanced Computer Science and Applications.* https://doi.org/10.14569/ijacsa.2017.080620

Tarjan, R. E. (1986). *Data Structures and Network Algorithms* (Society for Industrial and Applied Mathematics, Philadelphia, 1983) and G. Chartrand, Graphs and Digraphs.

Watkins, W. A. &Schevill, W. E. (1979). Aerial observation of feeding behavior in four baleen whales: Eubalaenaglacialis, Balaenoptera borealis, Megapteranovaeangliae, and Balaenopteraphysalus. *Journal of Mammalogy, 60*(1), 155-163. https://doi.org/10.2307/1379766

Topcoder, Community - Maximum Flow (Section 1). https://www.topcoder.com/community/data-science/data-science-tutorials/maximum-flow-section-1/ Last visited (14/01/2020)

Topcoder, Community - Maximum Flow (Section 2). https://www.topcoder.com/community/data-science/data-science-tutorials/maximum-flow-section-2/ Last visited (14/01/2020)