Journal of Advanced Computing Research Vol. 1, Issue 2 (2016) 10-13

A New Efficient Algorithm based on Floyd's Algorithm for Scheduling Concurrent Tasks on Grid Computing

Nurul Sakinah Muhamad Mahadi [#1], Mohamed Faidz Mohamed Said [#2]
[#] *Universiti Teknologi MARA*
70300 Seremban, Negeri Sembilan, MALAYSIA
[1] sakinahmahadi92@gmail.com
[2] faidzms@ieee.org

***Abstract*—This article is about new efficient algorithm based on Floyd's algorithm for scheduling concurrent tasks on Grid computing. Floyd-Warshall (FW) algorithm is proficiently and concurrently used for the discovery of the shortest paths between every pair of vertices in a weighted and possible directed graph. Grid computing is a processor architecture that collaborates computer resources from different domains to accomplish a main goal.**

***Keywords:* Floyd's algorithm, scheduling concurrent task, Grid computing**

I. INTRODUCTION

A. Floyd-Warshall algorithm

FW algorithm is also commonly known as the Roy-Warshall algorithm, the Roy-Floyd algorithm, the WFI algorithm or Floyd's algorithm. The Floyd's algorithm is emphasized to be equivalent to the transitive closure algorithm found via Roy (1959) and Warshall (1962) (Pemmaraju and Skiena 2003), which provide the explanation that the algorithm is related to all the three founders.

B. Grid computing

With the collaboration of computers on the network, then it can be running as a supercomputer in Grid computing [1]. It is also able to work on specialized applications although a Grid works on different tasks inside a network, and it is invented to find an answer while retain the flexibility to process numerous smaller difficulties to the problems that are too large for a supercomputer. Computing grids give a multiuser infrastructure that satisfy the discontinuous requirement of large information processing.

II. ALGORITHM COMPARISON

The FW algorithm is one of the algorithms widely used for determining the least cost path between every pair of nodes. In paper [2], a new algorithm is initiated based on this problem that requires less computational effort than the FW algorithm.

Floyd–Warshall algorithm

Input: A digraph G with $V(G) = \{1, \ldots, n\}$ and weights $c : E(G) \to \mathbb{R}$

Output: An $n \times n$ matrix M such that $M[i, j]$ contains the length of a shortest path from vertex i to vertex j.

```
M[i, j] := ∞ ∀i ≠ j
M[i, i] := 0 ∀i
M[i, j] := c((i, j)) ∀(i, j) ∈ E(G)
for i := 1 to n do
    for j := 1 to n do
        for k := 1 to n do
            if M[j,k] > M[j,i] + M[i,k] then M[j,k] := M[j,i] + M[i,k]
for i := 1 to n do
    if M[i,i] < 0 then return ('graph contains a negative cycle')
```

Figure 1. The FW algorithm for computing the lengths of shortest paths between all pairs of vertices in a directed graph with possibly negative edge weights.

This paper shows that a number of alternatives of Floyd and Rivest's algorithm SELECT for finding the kth smallest of n elements require at most n+min{k, n-k}+o(n) comparison on average and with high probability. This improves the analysis of Floyd and Rivest, and expands it to the case of non-distinct elements. The computational results confirm that SELECT may be the most excellent algorithm in practice [3].

In this paper [4], the comparative study of FW algorithm and the Rectangular algorithm have been examined, and tested on random graphs produced by the Erdös – Renyi (ER) model. The FW algorithm also gives a little improved achievement for dense graphs while the Rectangular algorithm works better for sparse graphs when the interpretation of the algorithms for dissimilar probabilities is shown.

There is also a prospect of evaluating single-rule Datalog programs efficiently and with logarithmic work space by a natural expansion of the Floyd±Warshall algorithm for transitive closure [5].

In this work, the researchers parallelize the standard FW and two cache-friendly versions using three different parallel programming environments, namely OpenMP, Cilk and Threading Building Blocks.

The experiment is carried out with multiple alternative parallel versions, in order to gain insight on the execution behaviour of the parallelized algorithms on modern multicore platforms, and on the programmability of the environments. It is able to significantly accelerate FW performance utilizing the full capacity provided by the multicore architectures used [6].

To specify the general of heterogeneous, distributed and dynamic grid task requests, this paper proposed a directed hypergraph with parameters. The requests of grid task are uniformly encapsulated and recognized by heterogeneous computers with the design of Grid Heterogeneous task graph Definition Language (GHDL). Besides, distinct form of tasks can be conducted and maximized with self-adaptive scheduling strategy library [7].

Heterogeneity, dynamicity and scalability are namely attached to characteristics inherent to grid infrastructures. Therefore, it is more complicated to assign tasks with precedence constraints represented by a Directed Acyclic Graph and this task assignment problem is recognized as NP-complete. As noted from paper [8], a dependent task assignment strategy for Grids has been presented and evaluated. The objectives are two folds: firstly the average response time of tasks submitted to the grid will be reduced whenever possible, and secondly, the transfer cost inducing by the tasks transfer referring to the dependency constraints reduces.

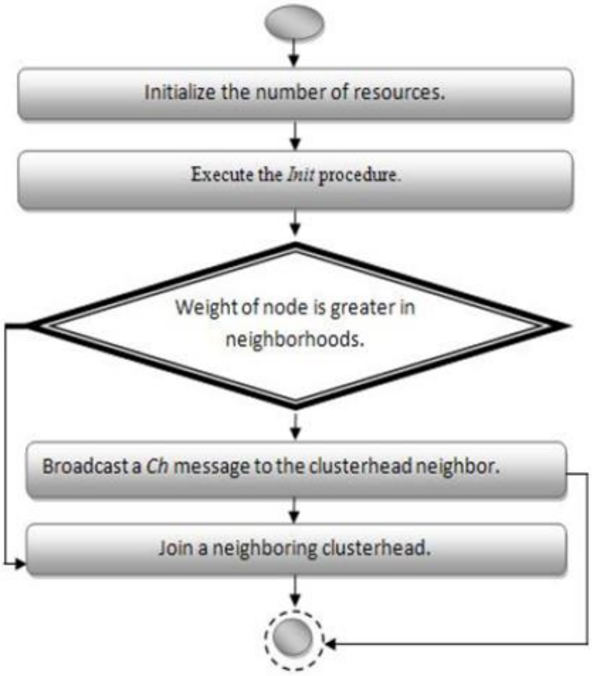

Figure 2. Distributed Clustering Algorithm

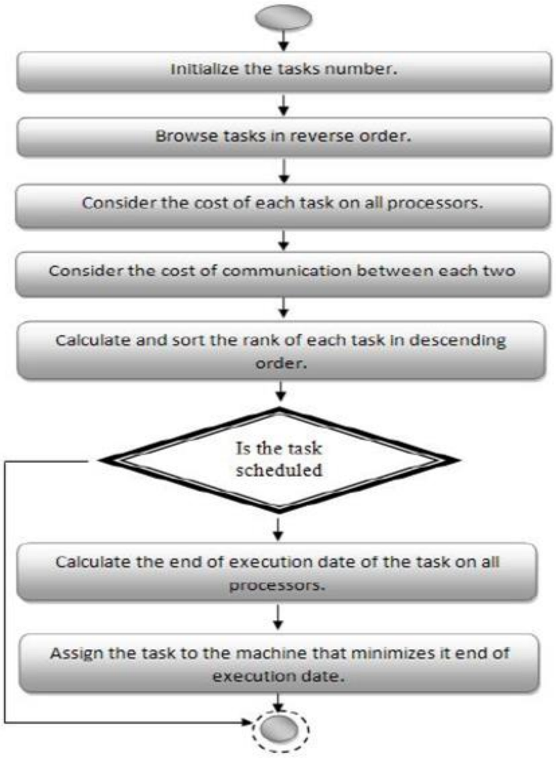

Figure 3. Heterogeneous Earliest Finish Time

Scheduling systems mostly are based on determined estimation of task completion time and as for current application-level, scheduling algorithms have several limitations and are too closely coupled with application internal structure. The application performance possibly will suffer when some resources stand for abnormal usage pattern during applications execution. Therefore, a prototype of Grid Harvest Service (GHS) has been developed to give dynamic and self-adaptive task scheduling. Testing result demonstrate GHS exceed existing system in scheduling huge application in a nondedicated heterogeneous environment [9].

Objective: Monitor the execution of an application in a set of machines and reallocate subtasks of the application if necessary.

Begin
Repeat
Measure the prediction error of the system utilization, PU_k, on machine M_k.
If $PU_k > Throttle$ **then** Calculate $E(T_{original})$;
List a set of machines that are current lightly loaded, $\{M_1, M_2, ..., M_q\}$;
If reallocation is one-to-one **then**
 For each machine, M_i, suppose it is the machine which the subtask will be assigned. Calculate $E(T_{reassign})^i$ with formula (2)
 End For
 Find the machine M_i, which has the maximum $E(T_{original}) - E(T_{reassign})^j$
 If $E(T_{original}) - (E(T_{reassign})^j > 0$ **then**
 Migrate the subtask on machine M_k to machine M_i.
 End If
End If
If reallocation is one-to-all **then**
 Sort the list of idle machines in a decreasing order with $(1-\rho_k)\tau_k$,
 Use bi-section search to find appropriate machine set *P*, which has the maximum $E(T_{original}) - E(T_{reassign})^P$
 If $E(T_{original}) - (E(T_{reassign})^P > 0$ **then**
 migrate the subtask on machine k to the machine set P.
 End If
End If
Until the application is completed
End

Figure 4. Self-adaptive task scheduling algorithm

By considering the implementation of environment as well as metrics used like soft computing, environment and metrics are used with the purpose to check the efficacy of each existing maximization approach as to show correlative study of different common grid computing based scheduling techniques. This reveals that the Ant colony maximization scheduling has slightly remarkable outcome exceeding the existing approach. The fitting future path increases the existing scheduling techniques because of decelerating growing rate, it turns into blockage of the optimistic scheduling [10].

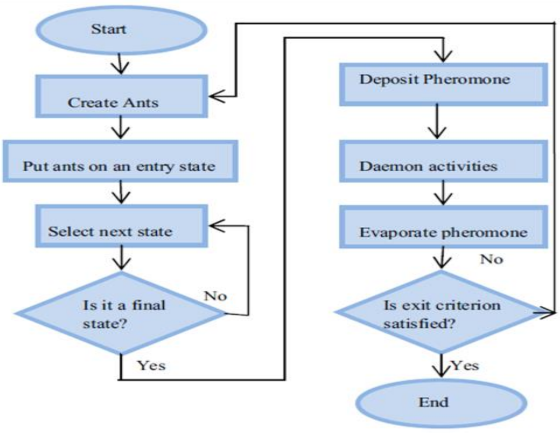

Figure 5. Flowchart 1 - Block Diagram of ACO

III. SCHEDULING

Scheduling tasks on heterogeneous resources distributed over a Grid computing system is an NP-complete problem. So, a new heuristic algorithm called Sort-Mid is proposed. Its purpose is to expand the implementation and minimizing the makespan. The advanced strategy of Sort-Mid algorithm is to find suitable resources. The basis initiative is to get the average value via sorting list of completion time of each task. Then, the maximum average is acquired [11].

Then different Grid scheduling algorithms are considered from different aspects like static vs. dynamic policies, objective functions, adaptation, applications models, strategies dealing with dynamic behavior of resources and QoS constraints, and others. A few general issues proposed worthy of further exploration are derived from a complete understanding of the challenge for Grid scheduling and the state of the art of current research [12].

Generally, scheduling acts as main part in Grid networks. So, choosing the type of scheduling algorithm involve two aspects in optimizing the reply and waiting time. Firstly, providing scheduling algorithms can reduce tasks runtime. Secondly, increase operational power that has remarkable influence in these categories.

As noted in paper [13], scheduling algorithms that include independent algorithms for example Minimum Completion Time, Minimum Execution Time , Min-min, Max-min and Xsuffrage are discussed.

MIN-MIN ALGORITHM

$$M = \left[\min_{0<j<\mu}(ct(t_i, m_i)), \text{for each } t_i \in U\right]$$

To a large learning curve of Grid computing, the requirement of brief comprehension of scheduling in Grid computing area is crucial. As for paper [14], comprehension of scheduling and Grid computing system are presented briefly. This paper also portrays complete concept of Grid computing and is considered essential subsystems that enable Grid computing achievable.

IV. CONCLUSION

With the collaboration of FW algorithm with other new algorithms, it can help the future research with it magnificent result that can maximize the quality of algorithm in solving the problem and minimize the obstacle that was faced to upgrade the existing algorithms.

REFERENCES

[1] "What is Grid Computing_ - Definition from Techopedia," https://www.techopedia.com/definition/87/grid-computing, 2016.

[2] A. Aini and A. Salehipour, "Speeding up the Floyd–Warshall algorithm for the cycled shortest path problem," *Applied Mathematics Letters*, vol. 25, pp. 1-5, 2012.

[3] K. C. Kiwiel, "On Floyd and Rivest's SELECT algorithm," *Theoretical computer science*, vol. 347, pp. 214-238, 2005.

[4] A. Singh and P. K. Mishra, "Performance Analysis of Floyd Warshall Algorithm vs Rectangular Algorithm," *International Journal of Computer Applications,* vol. 107, 2014.

[5] C. Papadimitriou and M. Sideri, "On the Floyd–Warshall algorithm for logic programs," *The journal of logic programming,* vol. 41, pp. 129-137, 1999.

[6] I.-M. Alifieraki, A. Andreou, C. Andrikos, D. Bliablias, Athanasios-Akanthos Chasapis, George Chatzikonstantis*, et al.*, "Parallelizing the Floyd-Warshall Algorithm on Modern Multicore Platforms: Lessons Learned," *5th Conference of Electrical and Computer Engineering Students, University of Thrace, Xanthi, Greece,* 2012.

[7] W. Qiao, G. Zeng, A. Hua, and F. Zhang, "Scheduling and executing heterogeneous task graph in grid computing environment," in *International Conference on Grid and Cooperative Computing*, 2005, pp. 474-479.

[8] M. Meddeber, B. Yagoubi, and W. Kadri, "A Static Tasks Assignment for Grid Computing," in *the Proceedings of the 3rd International Conference on Computer Science and its Applications (CIIA'11), Saida, Algeria*, 2011.

[9] M. Wu and X.-H. Sun, "A general self-adaptive task scheduling system for non-dedicated heterogeneous computing," in *Cluster Computing, 2003. Proceedings. 2003 IEEE International Conference on*, 2003, pp. 354-361.

[10] S. Sharma, A. Chhabra, and S. Sharma, "Comparative analysis of scheduling algorithms for grid computing," in *Advances in Computing, Communications and Informatics (ICACCI), 2015 International Conference on*, 2015, pp. 349-354.

[11] N. M. Reda, A. Tawfik, M. A. Marzok, and S. M. Khamis, "Sort-Mid tasks scheduling algorithm in grid computing," *J Adv Res,* vol. 6, pp. 987-93, Nov 2015.

[12] F. Dong and S. G. Akl, "Scheduling algorithms for grid computing: State of the art and open problems," 2006.

[13] F. S. Gharehchopogh, M. Ahadi, I. Maleki, R. Habibpour, and A. Kamalinia, "Analysis of Scheduling Algorithms in Grid Computing Environment," *International Journal of Innovation and Applied Studies,* vol. 4, pp. 560-567, 2013.

[14] H. B. Prajapati and V. A. Shah, "Scheduling in grid computing environment," in *2014 Fourth International Conference on Advanced Computing & Communication Technologies*, 2014, pp. 315-324.