Providing an algorithm for optimal prioritization of workloads on distributed systems using ALO meta-heuristic method

Raana Moallaeyan[1*], Ali Haroun Abadi, Seyed Javad Mirabedini

[1*]Master of Science, Department of Computer Engineering, Bushehr Branch, Islamic Azad University Bushehr, Iran
Department of Computer, Central Tehran Branch, Islamic Azad University, Tehran, Iran

Abstract
A new model of heavy parallel computing, distributed systems development aims at creating an abstract cloud computer using the free resources on WANs for uncertain contexts such as the Internet. As in distributed systems, there is a dynamic communication platform and resources, resource management including load balancing is an important issue. The purpose of load balancing is to find an efficient mapping of things on the processors in the system so that each processor carries out an almost equal amount of works to minimize the run time and its total cost. This paper presents a novel method to schedule and prioritize tasks in a distributed computing environment using the ant milk optimization algorithm which can cause a good balance between various factors. The simulation results indicated a good quality of the proposed method compared to other methods.

Key-words:
distributed computing, workflow, scheduling, prioritizing, ant milk algorithm.

1. INTRODUCTION

Due to the rapid growth of distributed computing, we need to use tools such as virtual machine for better management of computing resources and reducing energy consumption. This is why virtualization plays a major role in all strategies and algorithms proposed by researchers [1]. Finding a near-optimal solution to the problem of timing of tasks by more than one goal in a flexible and efficient manner in terms of time, is a difficult end evor. The nature of network is dynamic and reallocation of tasks is quite revealing in this situation [2]. Obtaining an optimal schedule for carrying out the tasks and activities in a distributed computing environment based on all available resources in the system, has become a very important responsibility in research [3]. In this paper, a new algorithm derived from nature called ant loin optimizer (ALO) was used to solve the problem of optimization. Network viewing was used for planning and scheduling activities and the mathematical model was used to minimize the overall time and cost of production used [4]. We used a new mechanism for prioritizing to reduce the execution time of the tasks set.

In this article, the literature is presented in section 2. In Section 3 we reviewed the available tasks. In Section 4, we evaluated the proposed method. Section 5 presented the results and section 6 was devoted to conclusion.

2. LITERATURE

An application is modeled as a directed graph without loop G (T, E) in which the T id set of n task and E is a set of arcs. Each arc represents a sequence that defines limits to first end the task t_i to initiate the task t_j [5]. In a duty graph [6], a task that does not have any parent is called input task, and a task that has no children is call output task. Because we need an algorithm with a graph which only has an input task, and one individual output task, we always add two t_{entery} and t_{exit} fabricated duties to the beginning and end of our tasks' graph. The implementation time of these two tasks is zero and are connected to the actual input and output tasks with arcs with zero weight [7]. The following figure shows an example of a task graph.

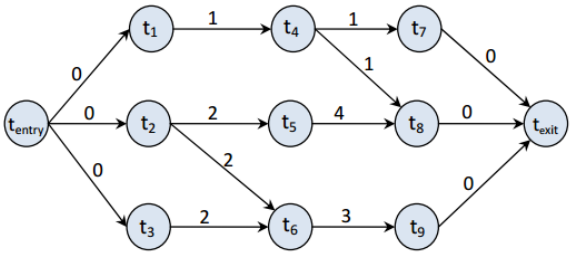

Fig. 1 An example of a workflow Application

General mosaics workflow system model is formed of a workflow management system and a number of mosaics service providers, each of which provides services to users [8,9]. The users send their workflow programs to workflow management system which acts as an intermediary between users and service providers [10]. In this sense, it receives the required information, schedule the workflow tasks on the appropriate services, books the services in advance and finally sends the tasks to the service providers [11,12].

Manuscript received May 5, 2017
Manuscript revised May 20, 2017

3. Review of literature

Some related research is examined here. The authors in [13] used a genetic algorithm for scheduling tasks in a distributed network environment. A directed acyclic graph (DAG) was shown for each task that has limitations and the processing time. The results showed that the scheduler reduced overall execution time and CPU time. The authors in presented a novel genetic algorithm based on the combination of Genetic Algorithm and simulated annealing for timing distribution network. The procedure can perform effectively and do the general optimization very well.

In [14] a tasks scheduling algorithm for grid computing is presented based on ant colony optimization algorithm that is the Monte Carlo method. This algorithm ensures appropriate load balancing of the machines. The authors [15] showed genetic algorithm for scheduling work on computational grids and the results showed the overall time optimization of tasks. In [16,17] a hybrid method based on particle swarm optimization algorithm has been used to solve the problem of scheduling tasks in distributed environments. Tests showed that this algorithm is able to have a better timing than genetic algorithm [18].

In [19,20], a multi-level decision engine and a two-level decision-making algorithm were proposed. In the first level decisions, the preliminary scheduling was created using an algorithm and then, in the later stage, scheduling is done using genetic algorithms. This approach can effectively use the algorithms in both stages. In [21] the Min-Min optimized algorithm which selects a set of couples of work / processor at each stage with the closest completion time. Then, from this collection of work / processor, it selects the one with the shortest amount of time to be completed and devote that task to that processor. Min-Min is an innovative method based on minimum completion time [22].

4. Proposed model

In this paper, optimization of the overall production time and cost of production have been considered. Operation time is one of the criteria which must be minimized which is widely used in optimal workflow scheduling. In this study, total operation time, is composed of processing time and communication time.

The processing time (TW) is the time required for machining a request by a number of operations that can be calculated as follows:

$$TW = \sum_{i=1}^{n} TWI(i, j) \qquad (1)$$

Here, n is the total number of operations in the workflow and i processing time on j machinery. Time Communications (TT) is the time taken to transfer data and the computational results between machines which can be calculated as follows:

$$TT = \sum_{i=1}^{n-1} TTI((i, j_1),(i+1, j_2)) \qquad (2)$$

Here, J1 and J2 are communication time for two consecutive operations. The total processing time (TPT) is defined as follows:

$$TPT = TW + TT \qquad (3)$$

Implementation cost is the other measure which is widely used in the distributed environment to increase the flexibility of workflow scheduling process increasing the quality and usability of timing mechanisms. The total cost of implementations includes the cost of executive machinery which can be described and calculated as follow.

machine cost (MC) is the total cost of the machines in the flexible application process calculated as follows:

$$MC = \sum_{i=1}^{n} MCI_i \qquad (4)$$

Here n is the number of overall operations and MCIi is the predetermined cost index for using i machinery which is fixed for a particular machine. Machine change cost (MCC) should be considered when two successive operations are carried out on different machines and it can be calculated as follows:

$$MCC = MCCI \times \sum_{i=1}^{n} \Omega(M_{i+1} - M_i) \qquad (5)$$

Here, MCCI is the machine change cost index and Mi is the identity of machine used for the operation i. In this section, two different performance goals were introduced for scheduling workflow. The proposed approach would establish an appropriate balance between these objectives. Also, on the other hand, these goals may have different priorities depending on different applications. For example, in a particular application, machinery costs may be more important than the cost of machinery change and communication tools, or vice versa. So, to achieve this goal, in this paper, we changed the objective function in a way that the user has the authority to continue to maintain an appropriate balance between these factors. And, for this, the user gives a series of weight to each of these goals. So, the merit function of the proposed method is as follows.

$$Fitness = \alpha \times MC + \beta \times MCC \quad (6)$$

Where A and B are weight coefficients which are given to each performance goals and α + β = 1 in it.

A. Prioritizing the tasks

The tasks in our proposed priority algorithm are determined by scheduling which is based on upward and downward ranking. In upward ranking, ni task is defined recursively and as follows:

$$rank_u(n_i) = \overline{\omega_i} + \max_{n_j \in succ(n_i)} (\overline{C_{i,j}} + rank_u(n_j)), \quad (7)$$

Where succ (ni) is set of past direct ni tasks and the average of communication cost is the edge and the average of calculation cost is ni. As these rankings is returning upward and is calculated by the start from output work, it is called upward ranking. For output function, the upward ranking value is:

$$rank_u(n_{exit}) = \overline{\omega_{exit}}. \quad (8)$$

In fact, rank u (ni) is the length of critical path from ni task to the output task, which includes the cost of calculating ni task. In various articles, there are algorithms which only use computational costs to determine ranking value. The upward ranking is called static. Similarly, downward rating of ni task is defined recursively.

$$rank_u(n_i) = \max_{n_j \in pred(n_i)} \{rank_d(n_j) + \overline{\omega_j} + \overline{C_{i,j}}\}, \quad (9)$$

Where pred (ni) is a set of ni direct former tasks. Downside rows recursively traverse the task downward graph calculated starting from the entry task. For nentry task, the value of downward rank is zero. In fact, is the longest distance from entry task to the work excluding the cost of automatic calculation. The proposed algorithm is a scheduling algorithm applied to a limited number of heterogeneous processor with two primary phases. One phase is prioritization which calculates the priorities of all tasks and the other phase is processor selection to choose their priorities and timelines and allocation of work in the "best" processor that minimizes the end time. This phase needs to prioritize each task by allocating an upward value to any task that is based on calculating the average and the average cost of communication. Tasks list is created by arranging the tasks in descending order based on $rank_u$. Multiple alternative policies can be used to break the knot, such as selecting a task that has a greater than the former direct and immediate task; and has higher upward rank'. Descending sorting based on expresser is a topological order of the tasks that provides a linear arrangement for the implementation and preserves the initial restrictions.

B. Ant lion optimization

Ant lions belong to a group of insects in the family of a. Two main stages of the life cycle of the ant lion are larva and adult stage . Ant lion insect larvae is often called "holes digger insect" because when searching for a good position in order to create a trap, it creates holes on the sand. Ant lion makes some holes in the soft and sandy ground during the hunt and then waits patiently at the bottom of the hole (Figure 2). When the bait slips into the hole, it is quickly haunted by ant lion. Or, if the bait is trying to escape, ant lion began to throw the sands at the edge of the hole to make the bait slip to the bottom of the hole. When the ant lion throws the sand to the side edges, they crash and the bait move downfall to the hunter. Mathematical modeling behavior of ant lions and ants lion are provided in the next section.

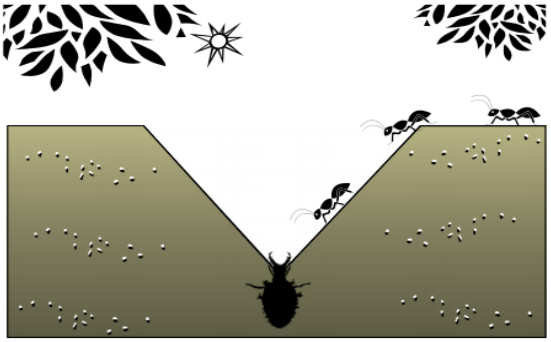

Fig. 2 Haunting behavior of ant lions

C. Mapping the workflow scheduling on ant lion algorithm

In the previous sections, in the workflow modeling of distributed systems, prioritizing tasks and ant lion optimization algorithm was explained in details. In this section, we show how to map the concepts to express ant lion algorithm. The following algorithm illustrates the mapping details.

Entry: a directed acyclic graph (DAG) for the implementation of activities
Calculating the priority of each task based on priority mechanism proposed
1. Descending sorting of tasks assigned on a priority basis
2. For n to k = 1 (n) is the number of nodes of DAG
3. Select n-th task from the ordered list
4. This task was assigned to the nearest available machinery.
5. The first ant in the lion ant is mapped to each of the mappings performed in the stage 3.

6. Mapping the other ants in the ant lion algorithm, with random assignment of all existing tasks to different machineries available.
7. Determine the ant lions position randomly.
8. Perform the ant lion algorithm
9. Return as the solution, the solution attributable to the most expert ant
10. End

Fig. 3 Mapping the workflow scheduling on ant lion algorithm

5. Evaluation Results

To evaluate the proposed method in this paper several criteria were considered. The proposed method based on these criteria is initially measured in terms of the various inputs. Then, the performance of the proposed method is compared with other methods based on the criteria of an average execution time, productivity and the average total cost.

A. Simulation model

Real systems of heterogeneous computing, such as grid computing and cloud computing, are a complex mix of hardware, software, and network components and it is often difficult to compare between the different techniques that are used on different systems. To solve this problem, Brown and his colleagues proposed a simulation model for comparing static scheduling algorithm in a heterogeneous computing environments. Their defined duty as a collection of independent works without any dependencies. The purpose of a scheduling problem here is to minimize the total execution time of the tasks. As static scheduling is used, we assumed that all the necessary information about available tasks and machines in the system are already available. As the expected running time of each user on each machine should be known, this information will be stored in ETC matrix. However, in an actual scheduling, some differences may be between the estimated run time and real time. But, in this article, we assume that the value in the ETC matrix is the runtime. Brown and colleagues considered the scheduling issues in heterogeneous computing environments as realistic as possible and defined different types of ETC matrix according to three criteria: job heterogeneity, machinery heterogeneity and compliance. Work heterogeneity is defined as the possible variance (changes) among tasks running time with two possible values: strong and weak. Machinery heterogeneity also reflects possible changes in the implementation of a specific work on all machines, again with weak and strong values. In order to attempt to consider some other features of real scheduling problems, three different ETC compatibility were used: Compatible, incompatible and semi-compatible. An ETC matrix is consistent when the mj machinery performs the ji task faster than mk machinery and the mj machinery performs all other tasks faster than mk. So a compliant ETC Matrix can be used as a model of heterogeneous systems where the machines are different only in the processing speed. In an incompatible ETC matrix, Pj machine may so some tasks faster and some slower than pk. So, incompatible ETC matrix can simulate a network in which various types of machines are available. For example, a Linux machine may perform tasks that involve a lot of symbolic computation faster than a Windows machine, but perform tasks involving a large number of floating point operations slower than it.

B. Results

In this section, the proposed method based on ant lion algorithms are compared with some existing methods in terms of scheduling in distributed environment. We compare this algorithm with those presented in [13] and [14]. The reason for using the method in [13] is that it also uses an evolutionary algorithm like this article. This method uses genetic algorithms and the method presented in [14] also uses the ant colony optimization method which is one of the best method of scheduling. For comparison of proposed method with other methods, some conditions of all states of Brown benchmark are used. In this, compatibility, incompatible or semi-compatible modes are considered and the proposed method was compared with the previous two methods. Charts presented in Figure 5 shows the results of the comparison very well.

Method proposed for [14]	Method proposed for [13]	Proposed method	Operational sample
1874	1987	956	u_c_hihi. 0
2031	2219	1187	u_c_hilo. 0
2145	2350	1298	u_c_lohi. 0
1708	1830	1056	u_c_lolo. 0
1924	2076	1021	u_i_hihi. 0
2217	2492	1349	u_i_hilo. 0
2189	2461	1621	u_i_lohi. 0
1920	2204	1106	u_i_lolo. 0
1876	2089	1150	u_s_hihi. 0
2108	2320	1239	u_s_hilo. 0
1569	1690	1331	u_s_lohi. 0
1894	2149	1198	u_s_lolo. 0

Fig. 4 The comparison of different methods' runtime (seconds)

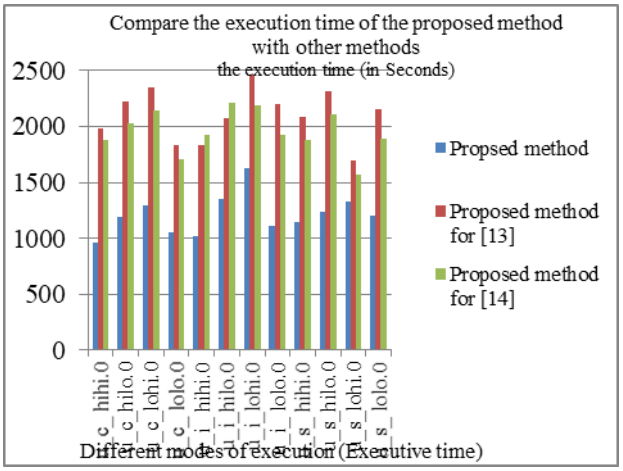

Fig. 5 Graph comparing the runtime of different methods (in seconds)

As is clear from the results, the proposed method in this article had good performance in all running modes and had a run time less than other methods. This is due to the use of ant lion algorithm with appropriate objective function in this article, which is well capable of searching the problem space and adequate enforcement of scheduling action. The purpose of scheduling tasks of distributed environment is reaching the top power of the system and matching application requirements with available computing resources. This matching of resources is done in a heterogeneous, community and uncertain environment and therefore the complexity of scheduling increases with the increase in the size of distributed environment. In this section, the efficiency of the proposed method using ant lion algorithm was compared with other methods presented in [13] and [14]. Provided graph in Figure 7 shows the comparison well.

Proposed method for [14]	Proposed method for [13]	Proposed method	Operational sample
0.96	0.92	0.97.5	u_c_hihi. 0
0.99	0.84	0.94.3	u_c_hilo. 0
0.85	0.81	0.91.7	u_c_lohi. 0
0.83	0.82	0.91.3	u_c_lolo. 0
0.88	0.83	0.92.4	u_i_hihi. 0
0.83	0.82	0.90.6	u_i_hilo. 0
0.82	0.79	0.88.2	u_i_lohi. 0
0.80	0.78	0.78.9	u_i_lolo. 0
0.93	0.91	0.93.8	u_s_hihi. 0
0.84	0.80	0.78.2	u_s_hilo. 0
0.82	0.83	0.91.4	u_s_lohi. 0
0.84	0.85	0.98.3	u_s_lolo. 0

Fig. 6 Comparison of productivity (in seconds)

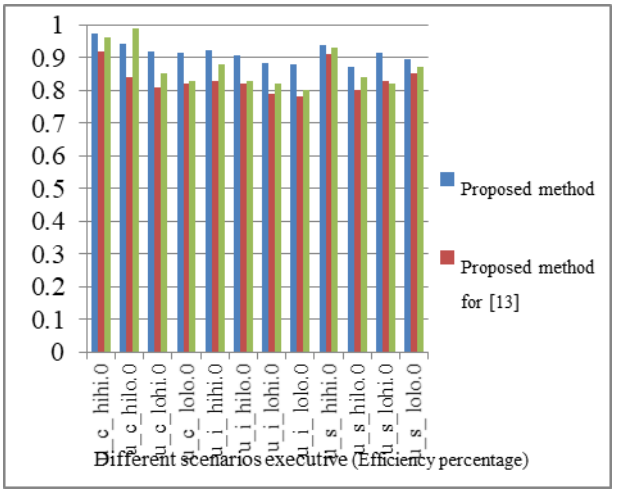

Fig. 7 Graph comparing the efficiency of different methods (in seconds)

6. Conclusion

The method presented in this paper was clearly mentioned in the previous sections based on ant lion algorithms. The proposed method was simulated to assess and demonstrate the quality of it. The results were compared over different metrics such as average time performance and efficiency of computing resources. For further studies, new algorithms and methods for scheduling can be announced that consider various parameters for running. For example, the dynamic changes in the cost and running work or even the possibility of opting out of the service of some tasks before logging in can be among the other parameters which make scheduling more dynamic and balancing load in the distributed environment. Also, the parallelism of ant lion algorithm to run in parallel the selection functions and compatibility function, can also be introduced as an idea of future research.

References

[1] Naghibzadeh M., (2016) “Modeling and scheduling hybrid workflows of tasks and task interaction graphs on the cloud”, Future Gen Comput Syst 65.

[2] Deldari A., Naghibzadeh M., Abrishami S., (2016) “CCA: a deadline-constrained workflow scheduling algorithm for multicore resources on the cloud”, The Journal of Supercomputing, pp 1–26.

[3] Kardani-Moghaddam. S, Khodadadi. F, Entezari-Maleki. R, and Movaghar. A, (2012) "A hybrid genetic algorithm and variable neighborhood search for task scheduling problem in grid environment". Procedia engineering, 29, PP. 3808-3814.

[4] Seyedali Mirjalili, (2015), “ The Ant Lion Optimizer” , Advances in Egineering Software 83 (2015) 80-89,ELSEVIRE.A

[5] Shojafar. M, Pooranian. Z, Abwajy. J. H, Singhal. M, (2013)”An efficient scheduling method for grid systems based on a hierarchical stochastic petri net”, Journal of computing science and engineering,Vol. 7, No. 1, March.

[6] Fidanova S., Durchova M., (2012) “Ant Algorithm for Grid Scheduling Problem”, Springer-Verlag Berlin Heidelberg, pp. 405–412.
[7] Wang, J., Duan, Q., (2011) “A New Algorithm for Grid Independent Task Schedule: Genetic Simulated Annealing”, World Automation Congress (WAC), IEEE.
[8] Sharma. Raksha, Kant Soni. Vishnu, Kumar Mishra. Manoj, Bhuyan. Prachet, (2012) “A survey of job scheduling and resource management in grid computing” World academy of science, engineering and technology.
[9] Ruhana, Mahamud and Husna Jamal Abdul Nasir. (2010) “Ant colony algorithm for job scheduling in grid computing”, In fourth asia international conference on mathematical/analytical modelling and computer simulation, IEEE computer society.
[10] Abdulal. W, Jabas. A, Ramachandram. S, Al Jadaan. O, (2012) "Task scheduling in grid environment using simulated annealing and genetic algorithm". Book: Grid computing-technology and applications, Widespread coverage and new horizons, PP. 89-110.
[11] Ahmadizar. F., (2012) "A new ant colony algorithm for makespan minimization in permutation flow shope", Computers & industrial engineering,Vol. 63, PP.355-361.
[12] Gupta S., Gabrani G., (2016) “ A dynamic two-level priority based authentication system for job scheduling in a heterogeneous grid environment”, SAI Computing Conference (SAI).
[13] Kołodziej J., Khan S. U., Wang L., Byrski A., Madani S. A., (2013) “Hierarchical genetic-based grid scheduling with energy optimization”, Cluster Computing, Volume 16, Issue 3, pp 591–609.
[14] Li K., Xu G., Zhao G., Dong T., Wang D., (2011) “Cloud Task scheduling based on Load Balancing Ant Colony Optimization”, 2011 Sixth Annual ChinaGrid Conference.
[15] Carretero. J, Xhafa. F, (2015) ”GENETIC ALGORITHM BASED SCHEDULERS FOR GRID COMPUTING SYSTEMS”, International Journal of Innovative Computing, Information and Control,vol. 3, No.6.
[16] Zhang. L, Chen. H, R. Sun, S. Jing, B. Yang, (2012) ”A Task Scheduling Algorithm Based on PSO for distributed Computing”, International Journal of Computational Intelligence Research, Vol. 4, No. 1, pp. 37–43.
[17] Karimi. M, Motameni. H, (2013) “Tasks scheduling in computational grid using a hybrid discrete particle swarm optimization," International journal of grid and distributed computing, Vol. 6, No. 2, PP.117-125.
[18] Mathiyalagan. M. P, Suriya. S, Dr. Sivanandam. S. N,(2010) ” Modified ant colony algorithm for grid scheduling, International journal on computer science and engineering(IJCSE), Vol. 02, No. 02.
[19] Ghaedrahmati. V, Enayatallah Alavi. S, Attarzadeh. I, (2014) "A reliable and hybrid scheduling algorithm based on cost and time balancing for computational grid "ACSIJ advances in computer science: An international journal, Vol. 3, Issue 3, No. 9, May, ISSN: 2322-5157.
[20] Mathiyalagan, P., Suriya, S., Sivanandam, N., (2013) “Modified Ant Colony Algorithm for Scheduling”, (IJCSE) International Journal on Computer Science and Engineering, Vol. 02, No. 02, pp. 132-139.
[21] Kaur D., Singh S., (2014), “An Improved min - min Algorithm for Job Scheduling using Ant Colony Optimization”, A Monthly Journal of Computer Science and Information Technology.
[22] Vivekanandan. K, Ramyachitra. D, (2013) “A study on scheduling in grid environment”, International journal on computer science and engineering, Vol. 3, No. 2.

Raana Moallaeyan received her B.S. degree in computer engineering Software from Islamic Azad University, Sepidan, Iran in 2009, then continued her study for M.S. degree in computer engineering Software Islamic Azad University, Science and Research Branch, Bushehr, Iran. Her research interests include distributed systems, computer network, grid computing, cloud computing, ACO, ALO and load balancing.

Ali Haroun Abadi is a faculty member of Azad University, central Tehran branch. He was born in 1976 and got PHD degree from Islamic Azad University, research and science branch (Tehran), in 2008. His work scope is software engineering and web mining.

Seyed Javad Mirabedini received his B.S in Shahid Beheshti University in computer engineering branch software, his M.S. In computer engineering branch machine intelligence and then continued his study for Ph.D. in computer science branch software in Islamic Azad University, Science and Research Branch, Tehran, Iran. Since 2007, he is assistant professor in Islamic Azad University, Central Tehran Branch and his research interests includes computer networks as well as intelligent systems.