International Journal of Pure and Applied Mathematics
Volume 118 No. 6 2018, 425-434
ISSN: 1311-8080 (printed version); ISSN: 1314-3395 (on-line version)
url: http://www.ijpam.eu
Special Issue

Dynamic Ant Colony Optimization (DyACO) in Heterogeneous Environments

D.I. George Amalarathinam[1] and A.Maria Josphin[2]
[1]Jamal Mohamed College (Autonomous), Tiruchirappalli–620 012.
di_george@ymail.com
[2]Jamal Mohamed College, Tiruchirappalli,
sr.mariajose25@gmail.com

Abstract

Efficient scheduling of tasks for an application is critical for achieving high performance in heterogeneous environment. The task scheduling has been shown to be NP complete in general case. Because of its key importance on performance, the task scheduling problem has been studied and proposed. This paper proposed a new method called Dynamic Ant Colony Optimization (DyACO), whose main objective is to get minimum makespan and maximize processors utilization in the heterogeneous systems. In this proposed algorithm the application program can be represented by a Directed acyclic graph (DAG), and the tasks are scheduled and assigned to the processors.

AMS Subject Classification: 03E72, 05C40, 05C72

Key Words and Phrases: Parallel processors, Directed Acyclic graph(DAG), Ant Colony Optimization, makespan, Processors Utilization, Heterogeneity.

1 Introduction

Parallel programming systems offer a promising and effective alternative choice for high performance computing. A parallel program is a collection of separate cooperating and communicating modules called tasks and processes. Tasks can execute in sequence or at the same time on two or more processors. Task mapping distributes the load among its processors so that the overall objective processing utilization is maximized and schedule length is minimized. The task scheduling activity determines the execution order [1]. To meet the computational requirements of a larger number of current and emerging applications, a satisfactory algorithm for task matching and scheduling is to be strengthen for the parallelization functions. One of the key challenges of such heterogeneous systems is the scheduling problem. The scheduling problem deals with mapping each task of the application onto the

available processors in order to minimize makespan. The task scheduling problem has been solved and is known to be NP complete [2].

Scheduling is the process which allocates the tasks to the available processors at particular time. The task scheduling is divided into static and dynamic. In static scheduling, execution process of each task is to be known in advance, takes place during compile time. Tasks are assigned to the processors upon their arrival in dynamic scheduling while the scheduling decisions are made at run time based on dynamic parameters that change during run time. Reallocation of scheduling tasks is possible during run time. Dynamic scheduling is flexible and faster than the static scheduling [2].

Ant colony algorithm is used to solve the problem of task scheduling in heterogeneous computing environments. In the Heterogeneous system, the processors speed varies, so there will be difference when they execute the same task. In addition there are obvious differences among tasks, so while allocating tasks to the processors; most tasks will be assigned to the processors which has better performance than others. In that case some processors will be overloaded and others will be in idle state. As a result, it will cause resource wastes. The load balance of processors will increase efficiency in heterogeneous systems, so this paper considers one more objective, the load balance of processor based on the ant colony algorithm to increase the utilization of processors [3].

The rest of the paper is organized as follows. Section 2 introduces the related work about ACO algorithm. Section 3 details of proposed ACO optimization algorithm, Section 4 contains the comparison of proposed algorithm with other existing methods in terms of makespan and processor utilization. Section 5 concludes paper.

2 Related work

Ant colony optimization (ACO) algorithm was proposed by Marco Dorigo [4]. The ACO algorithm is a meta heuristic inspired by the behaviour of real ants in their search for the shortest path to food sources. Ants tend to choose the paths marked by the strongest pheromone concentration. The ACO algorithm is an essential system based on agents that simulates the natural behavior of ants, including the mechanisms of cooperation and adaptation. The ACO algorithm simulates the techniques employed by real ants to rapidly establish the shortest route from a food source to their nest and vice versa without the use of visual information. The ACO algorithm consists of a number of cycles (iterations) of solution construction.

Smitha jha [5] proposed a method called Balanced Ant Colony Algorithm for Scheduling DAG to Grid Heterogeneous System. In this method Hybrid algorithm can be applied for dependent task scheduling, where tasks in Directed Acyclic Graph (DAG) are upward ranked and sorted decreasingly. This algorithm analysis gives better performance when compared with other algorithms.

Vahid Modiri et al [6] proposed a method called Fault Tolerance in Grid using Ant Colony Optimization and Directed Acyclic Graph. Tasks are entered to the system by Directed Acyclic Graph (DAG) and allocate the tasks to processors in short time and tolerability of system may be considered.

Arash Ghorbannia Delavar et al [7] proposed an algorithm called Task Scheduling

in Grid Environment with Ant Colony Method for Cost and Time. This proposed method used two parameters namely cost and time. Implementation of these parameters and more efficient scheduling of tasks increase the performance.

Ewa Figielska et al [8] proposed an algorithm called An Ant Colony Optimization for Scheduling Parallel Machines with sequence dependent set up costs. The objective of this paper is to minimize the weighted sum of makespan and total setup costs in the schedule by using column generation techniques and produced good results.

Pankaj d.Khambre et al [9] proposed an algorithm called Ant colony optimization with different variation methods to solve grid workflow scheduling problem. In this proposed method different QoS parameters are considered, including reliability, time and cost. Three variants of ACO for Network routing is proposed and implemented by standard network models. This algorithm is to overcome the problem of stagnation and congestion by using Multiple Ant Colony Optimization.

Kamolov et al [10] proposed an algorithm called Dynamic Task Scheduling Algorithm based on Ant Colony Scheme. In this proposal probability are used for ants to decide target machine and also to minimize makespan.

Umarani Srikanth et al [11] proposed an algorithm called Task Scheduling using probabilistic Ant Colony Heuristics. This algorithm is a novel algorithm feasible schedule for a task set on heterogeneous processors ensuring fair load balancing across the processors within a reasonable amount of time. Three parameters are used namely average waiting time of tasks, utilization of individual processors and the scheduling time of tasks.

Lei Wang et al [12] proposed an algorithm called Flexible job shop scheduling problem using an improved Ant Colony Optimization. The main objective of this algorithm is to minimize makespan. Select machine rule problems, initialize uniform distributed mechanism for ants, change pheromone's guiding mechanism, select node method and update pheromone's mechanism.

Liyun Zuo et al [13] proposed an algorithm called A multi-objective optimization scheduling method. This method considers the makespan and the user's budget costs as constraints of the optimization problem, achieving multi-objective optimization of both performance and cost. This method's performance was tested using four metrics: makespan, cost, deadline violation rate and resource utilization.

Li Zhou et al [14] proposed a method called Mapping DAG to Coarse-grained reconfigurable array (CGRA) using min–max Ant Colony System. Further optimization of MMAS is studied to reduce mapping time while maintaining the quality of solutions. This algorithm is a temporal mapping approach that reduces the execution latency of an application running on a given CGRA thus, optimized solutions can be found within a reasonable time.

3 Proposed algorithm

A Parallel program is represented by directed acyclic graph (DAG). The DAG model consists of two tuples $G = (V, E)$ where $V = \{T_1, T_2, \ldots, T_n\}$ finite set of tasks and $E = \{e_{ij}\}$ edges that connects two tasks T_i and T_j. The communication cost between two tasks is zero if they are scheduling on same processor. Precedence constraints are maintained. The task Tj executes if and only if its entire predecessor are executed.

A source node is called parent node and the sink node is called child node. A task with no parent is called entry node and with no child is called exit node [15].

Task scheduling is classified into deterministic scheduling also known as static scheduling and non-deterministic scheduling which is also known as dynamic scheduling. In deterministic scheduling, all the information about tasks that is communication time, execution time and their precedence constraint are well known in advance. In case of non deterministic scheduling, all the information are changed during run time. Scheduling decision are based on the dynamic parameters that may change during run time.

An Ant in the Ant system is a task in the Heterogeneous multiprocessor system. Pheromone value on the path in the Ant system is a load of the processor in the heterogeneous system. A scheduler assigns the task with high weight to high speed processor first, and then low weight task to low speed processor.

Initially, the parameters are created where we set the number of nodes, number of iterations and evaporation rate. In DyACO, the initial pheromone value is set to 0 because first the ant will fix a point and succeeding ants will follow the same point in search and of the best processor. In DyACO if the high speed processor completed task before its execution time, then the scheduler assign the next waited task to execute in the same processor and its pheromone value is updated.

Selection of processors is based on the Earliest starting Time (EST) and Earliest Finishing Time (EFT). Scheduling is done based on the EST (ni, pj) and EFT (ni, pj), the earliest execution start time and the earliest execution finish time of task ni on processor pj respectively.

In DyACO algorithm assignment of task is done using two steps. In the first step priorities of task in a DAG is calculated using ranking method and in the second step tasks are assigned to the processors using ACO method. The ranking method calculates the rank (R) from downwards to upwards, that is from the exit task to the task itself.

The Rank (R) is defined as:

$$R(v_i) = w(v_i) + \max_{v_j \in \text{succ}(v_i)} \{w(e_{ij}) + R(v_j)\} \tag{1}$$

where $\text{succ}(v_i)$ is the set of immediate successors of task v_i, $w(v_i)$ is the average computation cost of task v_i, and $w(e_{ij})$ is the average communication cost of edge e_{ij}. In the ACO method pheromone value of each processor is applied to the probability rule to identify the best processor.

$$P_{ij}^k = \frac{(\tau_{ij}^{\alpha})(\eta_{ij}^{\beta})}{\sum_{z \in \text{allowed } j}(\tau_{ij}^{\alpha})(\eta_{ij}^{\beta})} \tag{2}$$

where τ is the pheromone trail, while α and β controls the weight of the local heuristic and global heuristic. In DYACO after conducting experiments, α is set to 0.1 and β is set to 0.5 which reduces the makespan. Here η is the local heuristic and it is calculated every time a probability is generated. τ is the global heuristic which is the pheromone of the ants and it is updated in each iteration of the algorithm. The proposed DyACO algorithm is given in Fig. 1.

Step 1 Input the task priorities using DAG.
Step 2 Initialize ACO parameters α and β.
Step 3 Calculate the rank (R) for each task using the ranking method.
$R(v_i) = w(v_i) + \max_{v_j \text{succ}(v_i)}\{w(e_{ij}) + R(v_j)\}$
Step 4 Place all ants at the starting processors randomly.
Step 5 Every ant chooses the processors for the next task based on the probability rule.
$P_{ij}^k = \frac{(\tau_{ij}^{\alpha})(\eta_{ij}^{\beta})}{\sum_{z \in \text{allowed}\, j}(\tau_{ij}^{\alpha})(\eta_{ij}^{\beta})}$
Step 6 When a task is scheduled on a processor its pheromone value is updated.
Step 7 Repeat Step 5 and Step 6 until all tasks are scheduled.

Figure 1: The Proposed DyACO Algorithm.

Table 1: Computation cost

Tasks	Execution Time	P_0	P_1	P_2
T_0	4	2.67	3.20	2.29
T_1	2	1.33	1.60	1.14
T_2	6	4.00	4.80	3.43
T_3	4	2.67	3.20	2.29
T_4	12	8.00	9.60	6.86
T_5	2	1.33	1.60	1.14
T_6	14	9.33	11.20	8.00
T_7	2	1.33	1.60	1.14
T_8	10	6.67	8.00	5.71
T_9	10	6.67	8.00	5.71

4 Experiments and results

The proposed work is implemented in Java Net Beans and a DAG is randomly generated. Tests are done for different cases. Sample run of the given task set is shown in figure 2.and the computation cost table is shown in the Table 1 Processor speed is one of the CPU characteristics considered for this result. The speed is varied as 1, 1.25, 1.5 and 1.75. Figure 3 shows the comparative study of the makespan and processor utilization with various algorithms using arbitrary graph. DyACO scheduling algorithm has two metrics such as makespan and processor utilization. They are the performance parameters, of parallel programs considered in this paper. Table 1 tabulates the execution time of the tasks in various processors. The makespan is better in DyACO algorithm when compared to ACO algorithm. The resource utilization is also much far efficient in DyACO algorithm.

4.1 Makespan

The main performance of a scheduling algorithm is the total execution time of exit task called as makespan. Table 2 shows the comparison study of makespan with various algorithms. In all cases makespan is minimized. The graphical representation of Table 2 is shown in Fig. 3.

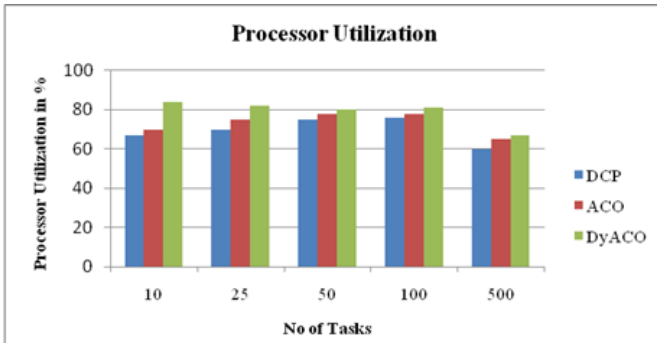

Figure 2: Sample DAG with 10 Tasks.

Table 2: Makespan for Arbitrary Graph.

Tasks	Processors	DCP	ACO	DyACO
10	4	46	26	20
25	8	139	191	150
50	10	159	228	225
100	12	197	550	450
500	16	585	701	625

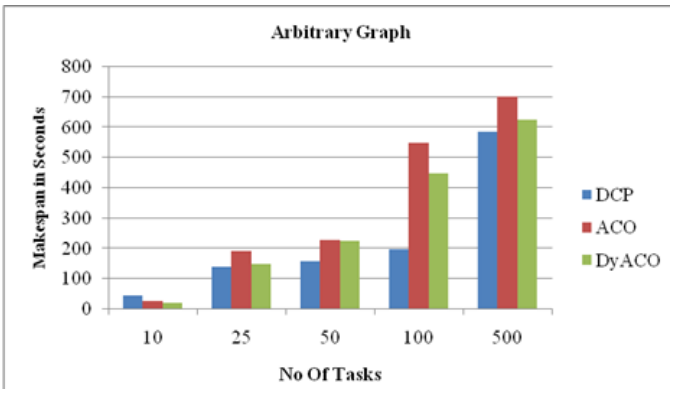

Figure 3: Comparison study of makespan with various algorithms.

4.2 Processor Utilization

The processors which have been reserved in advance are available till the completion of exit task. The load balance of processors will increase efficiency in heterogeneous systems, load balancing maximizes the processor utilization based on the ant colony algorithm. Table 3 shows the processors utilization percentage of DCP, ACO and DyACO algorithms. Fig. 4 shows the comparison study with various algorithms and in all cases processor utilization is maximized.

Table 3: Processor Utilization for Arbitrary Graph.

Tasks	Processors	DCP	ACO	DyACO
10	4	67	70	84
25	8	70	75	82
50	10	75	78	80
100	12	76	78	81
500	16	60	65	67

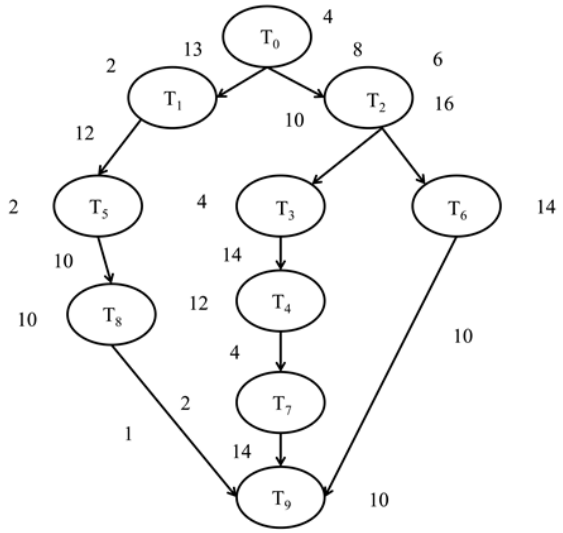

Figure 4: Comparison study of processor utilization with various algorithms.

5 Conclusion

The ACO algorithm is stochastic, population-based, evolutionary search algorithm. It is an efficient and powerful optimization algorithm, which widely applied in scientific research and engineering field. This paper proposes an efficient and intelligent ant colony optimization (DyACO) algorithm. The DyACO algorithm minimizes the makespan and maximizes the processor utilization. Comparison study with various algorithms in all cases shows that makespan is minimized and balance the load among the processors that maximize the processor utilization. This algorithm has been tested by varying the number of tasks and it is found that DyACO produces consistent results.

References

[1] T. Vetri Selvan, Mrs. P. Chitra, Dr. P. Venkatesh, "Parallel Implementation of Task Scheduling using Ant Colony Optimization", Academy Publisher, ACEEE, International Journal of Recent Trends in Engineering, Vol. 1, No. 1, pp. 339-343, 2009.

[2] N. Srinivasu, G. Krishna Chaitanya, "Advanced Task Scheduling in Heterogeneous Multiprocessor Systems Using Evolutionary Algorithms", International Journal of Innovative Research in Computer and Communication Engineering, Vol. 3, No. 11, 2015.

[3] Shengjun Xue, Mengying Li, Xiaolong Xu, and Jingyi Chen, "An ACO-LB Algorithm

for Task Scheduling in the Cloud Environment", Journal of Software, Vol. 9, No. 2, pp. 466–473, 2014.

[4] Ping Duan, Yong AI, "Research on an Improved Ant Colony Optimization Algorithm and its Application", International Journal of Hybrid Information Technology, Vol. 9, No. 4, pp. 223–234, 2016.

[5] Smitha Jha, "Balanced Ant Colony Algorithm for Scheduling DAG to Grid Heterogeneous System", International Journal of Scientific & Engineering Research Vol. 2, No. 6, 2011.

[6] Vahid Modiri, Morteza Analoui And Sam Jabbehdari, "Fault Tolerance In Grid Using Ant Colony Optimization And Directed Acyclic Graph", International Journal of Grid Computing & Applications (IJGCA) Vol. 2, No. 1, 2011.

[7] Arash Ghorbannia Delavara, Javad Bayrampoor, "Task Scheduling In Grid Environment With Ant Colony Method For Cost And Time", International Journal of Computer Science, Engineering And Applications (IJCSEA) Vol. 2, No. 5, 2012.

[8] Ewa Figielska, "An Ant Colony Optimization Algorithm for Scheduling Parallel Machineswith Sequence-Dependent Setup Costs", Zeszyty Naukowe Warszawskiej Wyż szej Szkoły Informatyki Vol. 9, No. 7, pp. 15–26, 2013.

[9] Pankaj D. Khambre, Aarti Deshpande, Deeksha Singh, Sajju Gautam, "Ant Colony Optimisation Algorithm with Different Variation Methods to Solve Grid Workflow Scheduling Problem", International Journal of Advanced Research in Computer Science & Technology (IJARCST), Vol. 2, No. 2, 2014.

[10] Kamolov Nizomiddin Baxodirjonovich, Tae-Young Choe, "Dynamic Task Scheduling Algorithm based on Ant Colony Scheme", International Journal of Engineering and Technology (IJET), Vol. 7 No. 4, pp. 1163–1172, 2015.

[11] Umarani Srikanth, Uma Meheswari, Shanthi Palaniswami and Arul Siromoney, "Task Scheduling using probabilistic Ant Colony Heuristics", The International Arab Journal of Information Technology, Vol. 13, No. 4, 2016.

[12] LeiWang, Jingcao Cai, Ming Li, and Zhihu Liu, "Flexible Job Shop Scheduling Problem Using an Improved Ant Colony Optimization", Hindawi, Scientific Programming, Vol. 2017, pp. 1–11, 2017.

[13] Liyun Zuo, Lei Shu, Shoubin Dong, Chunsheng Zhu, and Takahiro Hara, "A Multi-Objective Optimization Scheduling Method Based on the Ant Colony Algorithm in Cloud Computing", IEEE Access, The Journal for rapid open access publishing, Vol. 3, pp. 2687–2699, 2015.

[14] Li Zhou, Hengzhu Liu, "Mapping DAG to CGRA using min–max Ant Colony System", International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery, IEEE, 2013.

[15] Ms. Garima Garg, Mrs. Sheetal Kundra, "Task Scheduling in Grid Environment with Ant Colony method for Reliability and Time, International Journal of Advances in Computer Science and Communication Engineering (IJACSCE), Vol. 1, No. 1, pp. 14–21, 2013.