A Hybrid Multiprocessor Task Scheduling Method Based on Immune Genetic Algorithm

Mostafa Rahimi Azghadi, Mohammad Reza Bonyadi
Sara Hashemi, Mohsen Ebrahimi Moghadam
Department of Electrical and Computer Engineering, Shahid Beheshti University; MC, Tehran, Iran
{m_rahimi, m_bonyadi}@std.sbu.ac.ir, sa.hashemi@mail.sbu.ac.ir, m_moghadam@sbu.ac.ir

Abstract

Multiprocessor task scheduling plays a fundamental role in parallel applications and distributed networks. All of the methods for this kind of scheduling are concerned with achieving optimal running time. In this way parallel execution of tasks on several processors based on precedence graph should be considered. In this study, first a new heuristic method has been introduced which improved the execution time of some precedence graphs. Furthermore, we presented a novel immune genetic approach for multiprocessor task scheduling problem. Finally, combination of the proposed heuristic and the genetic approach makes a new hybrid scheme which is better than other well known and recent methods.

1. Introduction

Scheduling problem in multiprocessor, parallel and distributed systems are placed in NP-hard problem arena. These scheduling problems are employed in different important applications such as information processing, whether forecasting, image processing, database systems, process control, economics, operation research, and other areas. The data for these applications should be disseminated on different processors. Consequently efficient communication and well-organized assignments of jobs to processors are our concerns in solving multiprocessor task scheduling problems [1].

Genetic Algorithm has long been inspected as a method to find an optimal solution for NP-hard problems. Therefore genetic algorithms have been developed to solve this problem recently. However, most of the presented methods suffer from the local optimum problem in solving the scheduling problem. To overcome this problem, in this paper we present an immune genetic algorithm. Also a heuristic method is presented to increase convergence speed of the genetic algorithm. It is worth mentioning that this heuristic method can be useful in scheduling algorithm separately. Simulation results show that the hybrid of heuristic method and immune genetic method works better than well known scheduling methods.

2. Multiprocessor task scheduling problem

As already mentioned, the multiprocessor task scheduling problem focuses on achieving minimum execution time to perform all of the determined tasks in a right manner depend on the predefined or non predefined number of processors. Based on several criteria different kinds of multiprocessor task scheduling problem exist. These criteria can be stated as the following [2]:

1- The tasks are primitive or non primitive.
2- The processors are homogeneous or heterogeneous.
3- The number of processors is predefined or non predefined.

Commonly the multiprocessor task scheduling problem is described by a Directed Acyclic Graph (DAG). This graph represents the dependency among tasks, execution time of each one and the communication cost between them if they execute on different processors. An example of such DAG is shown in Fig. 1. In this figure four tasks are available, e.g. the task T1 is the predecessor of T2 and the communication cost between them is 1. And the execution time of task T1 is 5. Therefore we suppose that algorithm input is a DAG with described specification.

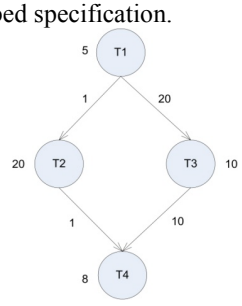

Figure 1. An example of a DAG

3. Related Works

Different approaches have been applied for multiprocessor task scheduling such as heuristic algorithm [3], evolutionary approaches [4-6] and hybrid methods.

QShine 2008, July 28-31, 2008, Hong Kong, Hong Kong.
Copyright 2008 ICST ISBN 978-963-9799-26-4
DOI 10.4108/ICST.QSHINE2008.4263

There are various heuristic works for task scheduling in multiprocessors, however, the most studied heuristic methods are list heuristics that can be employed in static scheduling [7]. Some related works for solving multiprocessor task scheduling problem using heuristic methods are DSC (Dominant Sequence Clustering) by Yang and Gerasoulis [1], MCP (Modified Critical Path) by Wu and Gajski [8], MD (Mobility Directed) by Wu and Gajski [8].

Also, solving the multiprocessor task scheduling problem using genetic algorithm has attracted many attentions and various studies have been reported in the literature [4-7]. One of the most recent and important works on the multiprocessor task scheduling using GA is [6] where the authors proposed the extension of the priority-based coding method as the priority-based multi-chromosome (PMC). Moreover, a new crossover method which is compatible with this new encoding has been proposed, that is called weight mapping crossover (WMX).

Another one classic work on the scheduling tasks in a multiprocessor system using GA is the HAR algorithm [4]. In this work, the heights of tasks in input DAG are main feature of them. The chromosomes in this study have a simple structure and each chromosome is composed of several strings. Using the height concept of the subtasks the precedence dependency among tasks is considered. Namely, the tasks are ordered according to their heights in each string. While this algorithm is very simple in terms of computational complexity, but it cannot ensure that the search space is a global one, so some feasible schedules are not reachable at all [7].

Correa et al. [9] is the other important work in the literature that tries to overcome the three drawbacks of the HAR. These drawbacks are: (1) naught of searches in the global solution space, (2) in the case of initializing population, load balancing between processors do not observed and (3) the absence of knowledge, since in creating individuals only the validity of individuals is observed and the quality of them is not monitored. In [9], a new combined approach where a GA is improved with the introduction of some knowledge about the scheduling problem represented by the use of list heuristic in the genetic operators. These improvements are useful in removing the drawbacks of the HAR algorithm. However, the presented method removes the HARs problems and it is a suitable algorithm in terms of solutions quality, but it causes a heavy computational load in crossover and mutation operators.

4. Proposed Method

The proposed method is composed of two parts. First a heuristic method, which can be employed for solving the desired problem separately or become a producer of a suitable initial population for contributing with genetic part, and second an immune genetic approach for solving the multiprocessor task scheduling problem.

4.1. Proposed Heuristic Method

The heuristic method can be described as follows:

1- Create a queue which will consist of potential executable tasks and make it empty. A potential executable task is a task that all of its predecessors have been executed. All the tasks in the first level are potentially executable.
2- The processor is dedicated to processes based on the Earliest Start Time (EST) algorithm. EST means that while a task is ready to be executed on a processor, we should select a processor which can carry out the task earlier than the other processors. However, the EST is computed for all the tasks in the queue, after entering or exiting a task. The task which has the smallest EST time has the highest priority.
3- After completing each task in every time, we check the next level and if there is a new task that all of its predecessors have been performed, this task is added to the list of potential executable tasks.
4- If the EST for a task is equal in all of the processors, it should be assigned to one processor randomly or be executed on the processor that has performed fewer tasks since the scheduling time.
5- If the EST of two or several tasks is equal on a processor, the task which has more offspring should take the processor before the others.

The mentioned algorithm has been applied on two graphs (Figures 2 and 4) which are presented in [6] and the simulation results show improvements in decreasing execution time. The main importance of this heuristic algorithm is that it was used to generate the initial population for the genetic algorithms to increase convergence speed. When the genetic algorithm has a suitable initial population, usually, it can reach the optimal or suboptimal solutions faster than when it uses a random initial population. Once again, we point out that the proposed heuristic algorithm can be useful for solving the problem disjointedly.

4.2. Proposed Immune Genetic Algorithm

The proposed Immune Genetic Algorithm (IGA) method has a similar encoding as is employed in [6]. Here, we use the coding of [6] and combine it to immune method and heuristic approach to reach the better answer in comparison with the PMC method. When a population is formed in genetic algorithm through the PMC method, some similar solutions maybe

created which cause the search process traps in a local optimum. In other words, the chromosomes become similar very fast. In this case, the search process stops in a local optimum point or in other words, evolution process advances slowly. To overcome this problem, we can check the solutions similarity in the evolution process and draw out solutions which are similar to the previous ones from the population and substitute a new solution instead of the exited one. The similarity checking between two tasks can be performed by calculating Euclidean distance between them. If the Euclidean distance is bigger than a threshold value, one of the similar tasks must be replaced with another one from the initial population or a new one from a random process. The proposed method can be stated in a sequential order:

1- The initial population is constituted through the proposed heuristic approach for realizing one of the schedules. The others are produced via a random process. This method for creating of initial population is useful in increasing the convergence speed of the genetic search process toward the optimal or suboptimal solution, because a good solution will be there at the first of the search process.

2- After that, the conventional genetic process is performed. The main points of this process are outlined in the following:

- Selection method is roulette wheel.
- Various cross over methods are checked on the algorithm and the method that has been used in our algorithm is WMX as it is presented in [6].
- Several different approaches for mutation phase have been carried out such as scrambling, swap and inversion. The best results are achieved using the Scrambling.
- To check the similarity of the produced solutions via an affinity mechanism and remove the similar tasks from our solution space (population) the Euclidean distance is used.
- Making schedules based on the recent population is the next part. We have some solutions in a list representation which a field of this list is a number that shows the precedence of each task, as shown in Fig. 3. Therefore, when some eligible tasks that all of their predecessors have been performed, are ready to be scheduled, one of them which has the most precedence should take the processor.
- To assign tasks to processors based on the scheduled list, each task assigns to the processor that its number is equal to the remainder of the task precedence number divided by the processors count [6].
- The last point is computing fitness of all schedules. The fitness here is the execution time value for each task schedule. Hence the smaller number of the fitness is desired.

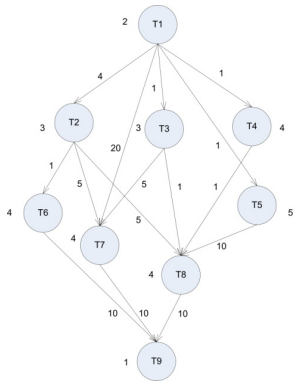

Figure 2. Example DAG with nine tasks

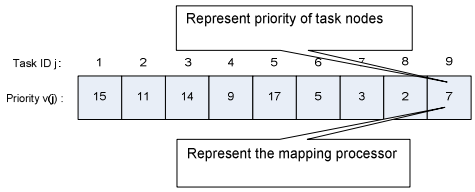

Figure 3. An Example of PMC and proposed chromosome

5. Simulation Results

To evaluate method functionality we implemented the presented approach and some previous works in MATLB and compared the results. The comparison part is composed of two phases. First, the proposed heuristic and hybrid IGA method are applied on two shown example graphs (Fig. 2 and 4) and results were compared with five previous methods such as DSC [1], DCP [3], MCP and MD [8] as heuristic based methods and PMC as a genetic based technique [6]. The simulation results for these evaluations are shown in Table 1. Moreover, for checking the proposed method and comparing it with the other previous ones on a larger DAG, the simulations are performed in different conditions and based on some standard task graph database [11]. Note that we add some communication cost to the database graphs and make some graph with communication cost to test our approach in a real condition and compare it with the PMC method. The related simulation results are shown in Table three and four.

In Table 2 we simulate two algorithms by different genetic operators. But the best results of both of them are reported in tables. As it is shown in these tables the

results of proposed method is better than the other methods.

The applied genetic algorithm used the following parameters throughout the simulations: Population size = 50, Maximum generation= 100, Crossover probability = 0.8, Mutation probability= 0.4, Terminating condition =100 generations, Affinity threshold= 0.8.

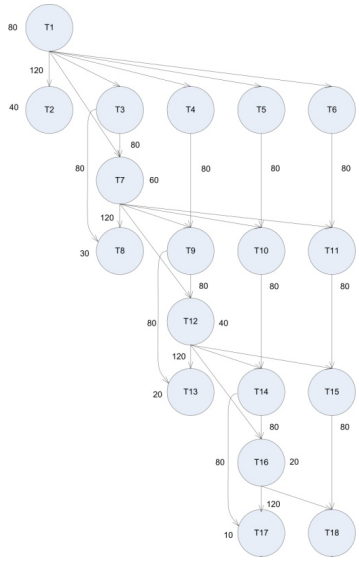

Figure 4. Example DAG with 18 tasks

Table 1. Comparative results with nine tasks, here with 1000 generation and population size equal to 100 (Fig. 2, Fig. 4)

Algorithms	MCP	DSC	MD	DCP	PMC	Proposed
No. Processors	3	4	2	2	2	2
Best Solution for DAG in Fig. 2	29	27	32	32	23	21
No. Processors	4	6	3	3	2	2
Best Solution for Fig. 4	520	460	460	440	440	440

Table 2. Comparative results with 50 tasks graphs (rnc50.tgz, rand0019, rand0160) [11]

Algorithms	PMC [6]	Proposed	PMC [6]	Proposed
No. Processors	2	2	8	8
Best Solution for rnc50.tgz, rand0019	252	243	176	167
Best Solution for rnc50.tgz, rand0160	380	361	365	361

6. Conclusions

A new heuristic and an innovative immune genetic method have been introduced which resulted in remarkable improvements in decreasing the execution time of parallel tasks in a multiprocessor system. The proposed method tries to overcome the previous methods shortcomings and cover them efficiently. In the IGA method, we only employ the affinity mechanism of the immune method for preventing of trapping genetic search process in local optimum. Moreover, the proposed method uses an initial population based on the heuristic approach which this caused in faster convergence and in some cases best solutions. The simulation results show that the presented hybrid IGA can be very effective in multiprocessor systems where the parallel and associated tasks are scheduled.

7. References

[1] Yang T, Gerasoulis A. DSC: scheduling parallel tasks on an unbounded number of processors. IEEE Transactions on Parallel and Distributed Systems 1994; 5(9).

[2] F. Montazeri, M. Salmani-Jelodar, S. N. Fakhraie and S. M. Fakhraie, “Evolutionary Multiprocessor Task Scheduling,” Proceedings of the International Symposium on Parallel Computing in Electrical Engineering (PARELEC'06), 2006.

[3] Yo-Kwong Kwok, Ishfaq Ahmad. Dynamic Critical Path scheduling: An Effective Technique for Allocating Task Graphs to Multiprocessors. IEEE Trans on Parallel and Distributed Systems, Vol. 7, No. 5, pp. 506-521, 1996.

[4] Hou ESH, Ansari N, Hong R. A genetic algorithm for multiprocessor scheduling. IEEE Transactions on Parallel and Distributed Systems 1994;5(2):113–20.

[5] Wu AS, Yu H, Jin S, Lin K-C, Schiavone G. An incremental genetic algorithm approach to multiprocessor scheduling. IEEE Transactions on Parallel and Distributed Systems 2004; 15(9):824–34.

[6] R. Hwang, M. Gen, H. Katayama, “A comparison of multiprocessor task scheduling algorithms with communication costs,” Computers & Operations Research 35, pp. 976 – 993, 2008.

[7] Y. W. Zhongiz, J. G. Yang, A genetic algorithm for tasks scheduling in parallel multiprocessor systems, In Proceedings of the Second International Conference on Machine Learning and Cybernetics, pp. 1785-1790, 2003.

[8] Wu MY, Gajski DD. Hypertool: a programming aid for message-passing systems. IEEE Transactions on Parallel and Distributed Systems, 1990; 1(3):330–43.

[9] Ricardo C Corrga, Afonso Ferreira and Pascal Rebreyend. Scheduling Multiprocessor tasks with Genetic Algorithm. IEEE transactions on parallel and distributed systems. 10(8):825-837, 1999.

[10] T. Tsuchiya, T. Osada and T. Kikuno, Genetics-based multiprocessor scheduling using task duplication, journal of Microprocessors and Microsystems, Vol. 22, Issues 3-4, pp. 197-207, 1998.

[11] Standard Task Graph Set is available online at: http://www.kasahara.elec.waseda.ac.jp/schedule.