A Modified for Largest Processing Time Scheduling Algorithm in Multiprocessor

Isra Natheer Alkallak
Dep. Basic sciences , College of Nursing , University of Mosul , Mosul , Iraq
(**Received:** 26 / 12 / 2010 ---- **Accepted:** 28 / 6 / 2011)

Abstract
The research reviewed new priority allocated to the independent tasks in the graph in the modified algorithm for largest processing time scheduling algorithm in multiprocessor. Schedule length was taken criterion in determining the efficiency of the algorithm. The priority was calculated from the proportion of task's execution time to the total execution time for all tasks in the graph and then arranged in descending order. Practical side was simulated by computer program in Visual Basic 6 language. A modified algorithm is more efficient than shortest processing time scheduling algorithm. In practical side confirmed efficiency of modified algorithm by selecting free of first task from set of tasks in graph through scheduling.

Key word : independent tasks, scheduling algorithm, priority.

1. Introduction
Scheduling is a key concept in computer multitasking. Scheduling as set of independent tasks for parallel execution on a set of processors is an important problem. Parallel program can be decomposed into a set of small tasks that generally have dependencies. The goal of task scheduling is to assign tasks to available processors such that precedence requirements between tasks are satisfied and overall time required to execute all tasks, the makespan, is minimized [1,2]. Note that in the defined multiprocessor task scheduling problem, a schedule is feasible if each task *T* can be exactly processed by *j* processors simultaneously, and the performance of a feasible schedule is measured by its schedule length such that a feasible schedule is called an optimal schedule if it is of minimum schedule length [3].

The aims of this paper to proposed algorithm for independent tasks scheduling with non-preemptive in multiprocessor to minimize the schedule length through modified algorithm for largest processing time scheduling algorithm. The paper is organized as follows. In section 1 present introduction and related work. The considered task scheduling problem, list scheduling algorithm, the largest processing time scheduling algorithm and structure of graph discussed in detail section 2**.** Section 3 presents proposed algorithm, results and discussion. Conclusions are offered in section 4.

1-1 Related Work
Several algorithms are proposed in literature to solve task scheduling problem. Various studies have proven that finding an optimal schedule is an NP-complete problem. Lin and Chen (1994) consider the problem of scheduling non-preemptive multiprocessor tasks in a homogeneous system processors. Alkallak (2003) comparison study of list scheduling algorithm. Alkallak and Sha'ban (2008) proposed a genetic algorithm for independent tasks scheduling. Lee and Kim (2009) proposed an intelligent priority to minimize the number of control steps of overall operation when scheduling is performed with fuzzy logic. Selvan and Venkatesh (2009) studied parallel implementation of task scheduling using ant colony optimization. Kaur and Singh (2010) modified genetic algorithm for task scheduling in homogeneous parallel system using heuristics. Vijayalakshmi and Padmavathi (2010) proposed comparison study of genetic algorithm and list scheduling algorithm.

2. Task Scheduling Problem
The tasks should be non-preemptive i.e., task execution must be completely done before another task takes control of the processor and the processor environment is homogeneous. Homogeneous of processor means that the processors have same speeds or processing capabilities [1]. In the task scheduling problem, tasks require simultaneously arbitrary processors when they start to be processed. Tasks considered are assumed to be independent. The precedence relation does not exist among them. A schedule is feasible if each task *T* can be exactly processed by *j* processors simultaneously, and the performance of a feasible schedule is measured by its schedule length such that a feasible schedule is called an optimal schedule if it is of minimum schedule length [3].

2-1 List Scheduling Algorithm
List scheduling techniques assign a priority to each task to be scheduled then sort the list of tasks in decreasing priority. As processors become available, the highest priority task in the task list is assigned to be processed and removed from the list. If more than one task has the same priority, selection from among the candidate tasks is typically random [2,4]. Through multiprocessor tasks considered are assumed to be independent, i.e., the precedence relation does not exist among them, such a problem of scheduling non-preemptive multiprocessor task is NP-complete.

2-2 The Largest Processing Time Scheduling Algorithm
The largest processing time scheduling rule (LPT) was first proposed by Graham, R.L 1969. The algorithm was a well known rule for the conventional scheduling of independent tasks non-preemptive on identical processors. In the other word, the LPT is a rule for scheduling those independent non-preemptive tasks T^1 in a homogeneous system of processors. The principle of LPT is that whenever a processor becomes

free, select a task whose computation time is the largest of those not yet assigned tasks to assign to it [3], while the principle of SPT is that whenever a processor becomes free, select a task whose computation time is the smallest of those not yet assigned tasks to assign to it.

2-3 Structure of Graph

A task is typically a program or program-like set of instructions that is executed by a processor. These tasks can be characterized by a node [٥]. Independent tasks are set of tasks (nodes) without precedence relation. All of tasks have the same level. Each of task had computation time [6,7,8]. Each node generally has the form shown in Figure (1), for example the number 1 means first node (number of node) and the number 6 means execution time of task. The Gantt chart gives the schedule shows the start and finish times for all tasks. When one or more processors remained idle period that means no task allocated to them, which the processors had a symbol ϕ in Gantt chart [9]. The arbitrary selection task does not affect the progress of implementation of tasks unless the restrictions are controlled by an algorithm [6].

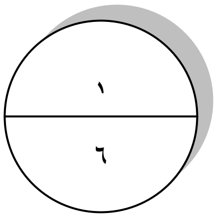

Figure (1) Task graph

3. Proposed Algorithm

The steps in modified algorithm for largest processing time scheduling algorithm with non-preemptive in Multiprocessor as follows :

Step 1 Input tasks set {initialize the number of tasks = n}
Step 2 Schedule length = 0 ; total time = 0 ; time ratio = 0
Step 3 assigned time for each task in graph as time [i] {arbitrary}
Step 4 For i = 1 to n
    total time = total time + time [i]
  End
Step 5 For i = 1 to n
    time ratio[i] = time [i] / total time
  End
Step 6 Decreasing sort of time ratio
Step 7 Initially assign any task from tasks set to free processor {arbitrary}
Step 8 Schedule length = Schedule length + time [i]
Step 9 Remove this task from tasks set
Step 10 While (there tasks unassigned) do
Begin
  IF there a free processor Then {processor not idle}
   Begin
    Assign task whose the time ratio is the largest among those
    tasks to processor
    Schedule length = Schedule length + time [i]
    Remove this task from tasks set
   End IF
End while
Step11 Print Schedule length

3-1 Results of Implementing for Proposed Algorithm and Discusion

Below Table (1) an input was illustrated each task in graph and execution time for each task to implement the proposed algorithm. Schedule were implemented in Visual Basic 6 language. The Table (2) show time ratio and descending order for tasks.

Table (1) Tasks and time

no. of task	execution time
1	4
2	1
3	2
4	3
5	4
6	5

Table (2) Descending order for tasks

no. of task	Execution time	time ratio	descending order for tasks
1	4	4/19 = 0.21	6
2	1	1/19 = 0.05	1
3	2	2/19 = 0.10	5
4	3	3/19 = 0.15	4
5	4	4/19 = 0.21	3
6	5	5/19 = 0.26	2
	total = 19		

To show the performance of proposed algorithm, a number of arbitrary generated test cases were used for simulation. Experimental results showed proposed algorithm to suit (correspond) traditional scheduling method as longest processing time algorithm (LPT). Schedule length in proposed algorithm is shorter than schedule length in algorithm as shortest processing time algorithm (SPT). Accordingly the proposed algorithm suited with longest processing time algorithm and therefore proposed algorithm is more efficient than shortest processing time algorithm (SPT). The Figure (2) an output was illustrated a schedule displayed as Gantt charts for two processors.

When initially from task 1

Processor 1	1	1	1	1	5	5	5	5	3	3
Processor 2	6	6	6	6	6	4	4	4	2	ϕ
	1	2	3	4	5	6	7	8	9	10

Schedule length is 10

When initially from task 2

Processor 1	2	1	1	1	1	5	5	5	5	ϕ
Processor 2	6	6	6	6	6	4	4	4	3	3
	1	2	3	4	5	6	7	8	9	10

Schedule length is 10

When initially from task 3

Processor 1	3	3	1	1	1	1	4	4	4	2
Processor 2	6	6	6	6	6	5	5	5	5	ϕ
	1	2	3	4	5	6	7	8	9	10

Schedule length is 10

When initially from task 4

Processor 1	4	4	4	1	1	1	1	3	3	2
Processor 2	6	6	6	6	6	5	5	5	5	ϕ
	1	2	3	4	5	6	7	8	9	10

Schedule length is 10

When initially from task 5

Processor 1	5	5	5	5	1	1	1	1	3	3
Processor 2	6	6	6	6	6	4	4	4	2	ϕ
	1	2	3	4	5	6	7	8	9	10

Schedule length is 10

When initially from task 6

Processor 1	6	6	6	6	6	4	4	4	3	3
Processor 2	1	1	1	1	5	5	5	5	2	ϕ
	1	2	3	4	5	6	7	8	9	10

Schedule length is 10

Figure (2) Gantt charts for proposed algorithm

While to show implementation for LPT algorithm and SPT algorithm for example in Table (1) were illustrated in Figure (3) and Figure (4).

Processor 1	6	6	6	6	6	4	4	4	3	3
Processor 2	1	1	1	1	5	5	5	5	2	ϕ
	1	2	3	4	5	6	7	8	9	10

Figure (3) Gantt chart for LPT algorithm
Schedule length is 10

Processor 1	2	4	4	4	5	5	5	5	ϕ	ϕ	ϕ
Processor 2	3	3	1	1	1	1	6	6	6	6	6
	1	2	3	4	5	6	7	8	9	10	11

Figure (4) Gantt chart for SPT algorithm
Schedule length is 11

The efficiency of the proposed algorithm through notic in this algorithm it is possible to start from any task in graph and gives the same solution (equal schedule length) in all cases, while in the case of the LPT algorithm which restrict the implemention time is the largest processing time of task assigned to processor. This supports the efficiency of the proposed algorithm. Alkallak (2003) tackled studied on the optimal scheduling for multiprocessors and concluded that the scheduling of independent tasks that the schedule length in the algorithm LPT is less than the schedule length in the algorithm SPT. Genetic algorithm developed by Alkallak and Sha'ban (2008) that suggested research addressed a heuristic genetic algorithm for independent task scheduling, and concluded the schedule length of the proposed genetic algorithm in their search corresponds to the length of the scheduling algorithm in the LPT, and in the case of genetic algorithm mentioned therein restrict genetic algorithm.

4. Conclusions

In this paper, a non-preemptive scheduling algorithm was presented that uses priority to schedule graph on multiprocessor with the objective of minimizing the schedule length. Experiment results of proposed algorithm is the best solution. Priority list is decided by a priority function, which affects the result of scheduling. Therefore, the choice of proper priority function is important for the performance of scheduling.

The proposed algorithm is more efficient than LPT algorithm. Also this algorithm is more efficient than SPT algorithm by noting in the proposed algorithm it is possible to start from any task and gives the same solution of schedule length in all cases. Proposed algorithm could be used in list scheduling applications with non-preemptive in multiprocessor.

References

1)K. Kaur, A. Ch, G. Singh, Modified Genetic Algorithm for Task Scheduling in Homogeneous Parallel System Using Heuristics, International Journal of Soft Computing, Medwell Journals: 5(2010)2, pp 42-51.

2)S.R.Vijayalakshmi, G. Padmavathi, A Performance Study of GA and LSH in Multiprocessor Job Scheduling, IJCSNS International Journal of Computer Science Issue: 7(2010)1.pp37-42.

3)J.F. Lin, S.J. Chen, Scheduling Algorithm for Nonpreemptive Multiprocessor Tasks, Computers Math. Applic. Elesevier Science Ltd, Printed in Great Britain: 28(1994)4, pp85-92.

4)J.Y. Lee, S. Kim, An Intelligent Priority Decision Making Algorithm for Competitive Operators in List-based Scheduling, IJCSNS International Journal of Computer Science and Network Security: 9(2009) 1.pp81-86.

5)T.V. Selvan, P. Ch, P. Venkatesh, Parallel Implementation of Task Scheduling Using Ant Colony Optimization, International Journal of recent trends in engineering: 1(2009)1.pp339-343.

6)I. N. H. Alkallak, An Optimal Scheduling of Multiprocessor, University of Mosul, (2003), Iraq, M.Sc. Research, College of Computer and Mathematics Sciences.

7)M. R. Garey, R. L. Graham, D. S. Johnson, Performance Guarantees for Scheduling Algorithms, Operations Research Society of America: 26(1978)1.pp3-19.
8)K. Hwang and F.A. Briggs, Computer Architecture and Parallel Processing. McGraw-Hill International Edition. (1984), Computer Science Series, New York, pp590-604.
9)I. N. Alkallak, R. Z. Sha'ban, A heuristic Genetic Algorithm for Independent Task Scheduling, Dirasat, Pure Sciences, University of Jordan: 35(2008) 2. pp180-193.

تطوير خوارزمية جدولة زمن التنفيذ الأكبر في المعالجات المتعددة

إسراء نذير الكلاك

فرع العلوم الأساسية ، كلية التمريض ، جامعة الموصل ، الموصل ، العراق

(**تاريخ الاستلام:** ٢٦ / ١٢ / ٢٠١٠ ---- **تاريخ القبول:** ٢٨ / ٦ / ٢٠١١)

الملخص

استعرض البحث أسبقية جديدة تُخصص للعمليات المستقلة في البيان لغرض تنفيذها على معالجين في خوارزمية مطورة لخوارزمية جدولة زمن التنفيذ الأكبر في المعالجات المتعددة. أُتخذ طول الجدولة معياراً في تحديد كفاءة الخوارزمية المطورة. تم حساب الأسبقية من نسبة زمن تنفيذ العملية إلى الزمن الكلي للعمليات جميعها في البيان ومن ثم ترتب تنازلياً. تم محاكاة الجانب العملي بشكل برنامج حاسوبي بلغة فيجول بيسك٦.
تبين إن نتائج تطوير الخوارزمية المذكورة أكفأ من خوارزمية زمن التنفيذ الأصغر، في حين أثبت الجانب العملي كفاءة الخوارزمية المطورة في حرية اختيار العملية الاولى من بين العمليات في البيان أثناء الجدولة.

الكلمات المفتاحية: العمليات المستقلة، خوارزمية الجدولة، اسبقية.