International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 2 Issue 1, January- 2013

Feasibility Analysis of RM Scheduling Under Fluid Traffic Conditions

Kedilaya Shreeganesh. B[1], Dr. G. M. Patil[2]

[1]Professor, Srinivas School of Engg, Mukka, Mangalore, India.
[2]Principal, Basava Kalyan Engg College, Bidar, India.

ABSTRACT

Scheduling refers to the process of allocating resources to tasks or assigning tasks to resources. This is done by generating a table with explicit start times for each task such that only one task at a time requests the resource. This is called static scheduling. Another solution for scheduling is to assign priorities to tasks and choose this task for execution with highest priority. Scheduling decision arises whenever the number of tasks to be executed is more than the available resources. Under fluid traffic flow Pro-active scheduling needs a re look with respect to traffic pattern analysis and scheduling accordingly[1]. Here we enumerate the whys and how's of disruptions in scheduling. Despite the plethora of studies and commercial solutions proposed, scheduling is still considered as one of the scientific areas where substantial improvements can be gained by the development and application of new research approaches[2]. Transferring this knowledge in to practice is really difficult because poor evaluation of benefits that can be got by adopting the methods in real time processes and under*standing which solution can give better results according to a pre defined set of tasks.*

Key words: VLSI, Traffic, Multiprocessor, Robustness.

1.Introduction

In the emerging VLSI Technology paradigm the consumer plays an important role. The tasks initiated by the consumer drive the systems traffic. In the historical evolution of scheduling technique first approach is based on a determinate view with implicit assumption that the proposed schedule will be carried as predicted before hand. In reality changeability and exceptions have demanded the stochastic based scheduling approach. The ever growing external and internal changing conditions which heavily affect the quality of of schedule are still under investigation in the research community. A scheduling scheme for tasks which guarantees the quality of services for video and audio is arrived at by means of dynamic priority scheduling .Feasibility of offline schedulers are affected by a large variety of dynamic events.

2. RELATED WORKS

It has been shown that the Liu and Layland results break down on multiprocessor systems [3]. Dhall and Liu [4] gave examples of task sets for which global RM scheduling can fail at very low processor utilizations, essentially leaving all but one processor idle nearly all of the time. Reasoning from such examples it may be perhaps easy to conjecture that Rate Monotonic is not a good or efficient scheduling policy for multiprocessor systems. However, at the moment these conclusions have not yet been formerly justified [5].

The partitioning scheme has received greater attention than the global scheme because the scheduling problem can be reduced to the scheduling on a single processor, where at the moment a great variety of scheduling algorithms exist. It has been proved by Leung and Whitehead [6] that the partitioned and global approaches to static-priority scheduling on identical multiprocessors are incomparable in the sense that

1) There are task sets that are feasible on *m* identical processors under the partitioned approach but for which no priority assignment exists which would call all jobs of all tasks to meet their deadlines under global scheduling in the same *m* processors, and

2)There are task sets that are feasible on *m* identical processors under the global approach, which cannot be partitioned into *m* distinct subsets such that each individual partition is feasible on a single static-priority uni processor.

In this work, therefore only the partitioning scheme has been dealt with and not the global scheme.

3. MODEL DESCRIPTION AND PROBLEM FORMULATION

We consider a queuing architecture of a homogeneous system in which 'n' processors are connected to 'm' classes of tasks through a Scheduler. 'm' and 'n' are integers greater than or equal to 1.All processors functions in same speed and are identical. The system architectural model consists of a task scheduler queue, a scheduling strategy for multiple classes of tasks and 'n' local task queues. The aim of the scheduler is to make good task allocation decision for each class of tasks and maintain an ideal performance in response time. A schedule queue stores incoming tasks and maintain an ideal performance in response time. The scheduler processes the tasks in FCFS manner and are dispatched to one of the designated processors. The processor maintains a local queue. We formulate the scheduling problem as a problem between number of incoming tasks and the number of processors required to execute this task when the scheduling policy is RMFF and RMBF.

A. SCHEDULABILITY CONDITION IP (INCREASING PERIOD):

In condition IP [7], it is required that the periods of the tasks are ordered increasingly, thus the timing constraints of the task set must be known a priori. Condition IP, introduced by Dhall and Liu [7] is used in the implementation of the multiprocessor algorithms Rate Monotonic Next Fit and Rate Monotonic First Fit.

Theorem 1: **(Condition IP)** [6]: Let $\tau = \{ \tau_1, \tau_2, \ldots, \tau_n \}$

be a set of tasks with $T_1 \le T_2 \le \ldots \le Tn$ and let

$$u = \sum_{i=1}^{n-1} C_i / T_i \le (n-1)\left(2^{1/(n-1)} - 1\right) \quad (3.1)$$

If the following condition is met,

$$C_n / T_n \le 2\left(1 + \frac{u}{(n-1)}\right)^{-(n-1)} - 1 \quad (3.2)$$

then the set of tasks will have a feasible schedule under the RM algorithm. When $n \to \infty$, the minimum utilization of task T_n approaches to $(2e^{-u} - 1)$.

4. RATE MONOTONIC SCHEDULING ON MULTIPROCESSOR SYSTEMS

When can a set of tasks be scheduled on a multiprocessor system? The result has to be found for the minimal number of processors that is needed to find a feasible assignment of a task set to a multiprocessor system. Obviously, in the worst case, only one task can be assigned to each processor. Therefore, the following question should be taken into concern: When can a set of tasks be scheduled on less than one processor for each task? The answer to this question is given in Theorem 2. Roughly, Theorem 2 says, that if K processors are needed to schedule a set of K tasks, then the load on each processor cannot be much less than 1/2.

Theorem 2: If the total load of a set of K real-time tasks satisfies

$$U \le \frac{K}{2^{1/K} + 1} \quad (4.1)$$

then the task set can be scheduled with the RM algorithm on less than K processors. The condition is tight.

RATE MONOTONIC ALGORITHMS UNDER PARTITIONING SCHEME

In the following sections, the heuristic static scheduling algorithms executed under RM on multiple processors are described. It is assumed that in the notation used the allocation of task $_{Ti}$ to processor P_j consists on increasing the utilization of the processor by, $U_j = U_j + u_i$, where U_j denotes the utilization of processor P_j and u_i denotes the utilization of task $_{Ti}$.

A.RATE MONOTONIC FIRST FIT (RMFF):

In the allocation process, some algorithms verifies the current processor to see if a feasible schedule can be found for a new task (along with the previously allocated tasks). If there is no feasible schedule in that processor, it allocates the task on an idle processor, even though the task could be feasibly scheduled in another non-idle processor previously used. To overcome this waste of processor utilization, RMFF always checks the schedulability of the new task from the first processor to the current one on which the task can be scheduled.

The Rate Monotonic First Fit (RMFF) algorithm [7], illustrated in Figure 4.1, orders tasks increasingly according to their periods, before the allocation process. To allocate task $_{Ti}$, it is required to select the processor with smallest index, such that task $_{Ti}$ along with previously allocated tasks, finds a feasible schedule using the schedulability condition IP.

In the algorithm, if task $_{Ti}$ satisfies condition IP (step 4 from the algorithm) then it is schedulable, so the task is allocated to processor P_m (step 7). k_m denotes the number of tasks allocated to processor P_m, and U_m denotes the total utilization of the k_m tasks. If ($i > n$), its means that all tasks have already been allocated (step 3), then the algorithm finishes, otherwise, the index is increased (step 10), and the algorithm continues with steps 4 to 10. When the algorithms finishes, the total number of processors required for the feasible scheduling of the task set is given by variable j.

The performance of the RMFF algorithm is $2 \leq \Re_{RMFF} \leq (4 \times 2^{1/3}) / (1 + 2^{1/3}) \approx 2.23$ [6] and its computational complexity is O(n log n).

Input: Task set $_T = \{T_1, T_2, \ldots, T_n\}$

and a set of processors $\{P_1, \ldots, P_m\}$.

Output: j; number of processors required.

1. Sort the tasks by increasing periods
2. i:= 1; j:= 1; / * i = i^{th} task, j = j^{th} processor * /
3. **while** ($i \leq n$) **do**
4. q : = 1; ,
5. **while** ($u_i > 2(1 + U_q/k_q)^{-k}{}_q - 1$) **do**
6. q : = q + 1; / * q denotes the processor index * /
7. $U_q : = U_q + u_i$; $k_q := k_q + 1$
8. **if** ($q > j$) **then**
9. j: = q;
10. i : = i + l;
11. **return** (j);
12. **end**

Fig. 4.1.Rate Monotonic First Fit (RMFF) Algorithm

B.RATE MONOTONIC BEST FIT (RMBF):

Algorithm RMBF[7], illustrated in Figure 4.2, allocates the task to the processor with smallest available utilization, on which it can be feasibly scheduled.RMBF orders tasks increasingly according to their periods, before the allocation process. In this algorithm, the allocation of task $_{Ti}$ to processor P_j (step 5 from the algorithm) requires finding the smallest j such that task $_{Ti}$ together with all the task previously allocated to processor P_j, can be feasibly scheduled according to the schedulability condition IP, and that the value of $2(1 + U_j / k_j)^{-kj} - 1$ be as small as possible (step 4).

Input: Task set $\tau = \{\tau_1, \tau_2, \ldots, \tau_n\}$

and a set of processors $\{P_1, \ldots, P_m\}$.

Output: j; number of processors required.

1. Order tasks increasingly by their periods
2. i:= 1; j:= 1; / * i = i^{th} task, j = j^{th} processor * /
3. **while** ($i \leq n$) **do**
4. **if** ($u_i \leq$ Condition IP) **and**
   ($2(1 + U_j / k_j)^{-k}{}_j - 1$) is the smallest possible
5. $U_j := U_j + u_i$;
6. else

7. task τ_i is allocated to an idle processor

8. i := i + 1;

9. **return** (j);

10.**end**

Fig. 4.2 Rate Monotonic Best Fit (RMBF) Algorithm

The processor with the smallest available utilization, on which the new task can be feasibly scheduled, will be the one where the task will be allocated.

The performance of RMBF is $\Re_{RM\ BF}\ 2 + (3 - 2^{3/2}) / a \approx 2.33$, where $a = 2\ (2^{1-3} - 1)$ [7], and its computational complexity is O(n log n).

5. EXPERIMENTAL RESULTS AND PERFORMANCE EVALUATION

To obtain the average case performance of the algorithms one can analyze the scheme with probabilistic assumptions, or conduct simulation experiments to empirically study the average performance. Since a probabilistic analysis of our algorithms is beyond the scope of this study simulation is used to gain insight in to the average case behavior.

The simulation experiments are presented with task sets ranging from $5 \leq K \leq 20$ In each experiment the value of α is varied $\alpha = \text{Max}_{i=1,\ldots,k} V$; the maximal load factor of any task in the set. The task periods are assumed to be uniformly distributed with values $5 \leq T_i \leq 100$. The execution times of the tasks are also taken from a uniform distribution with range $1 \leq C_i \leq \alpha T_i$ The performance metric in all experiments is the number of processors required to assign a given task set. All the assignment schemes are executed on identical task sets.

The outcome of the simulation experiments is shown for $\alpha = 0.2$, 0.5 and 0.8. The graphs show that in all schemes the number of processors required for the respective schemes increases proportionally to the total load.

For α = 0.2

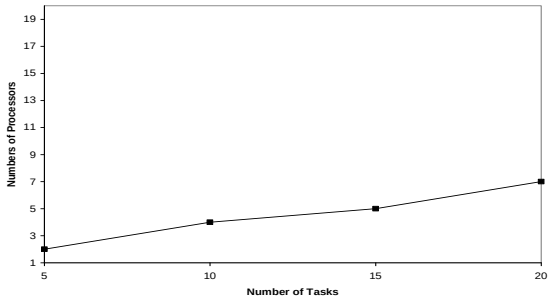

Fig. 5.1 Performance of RMBF

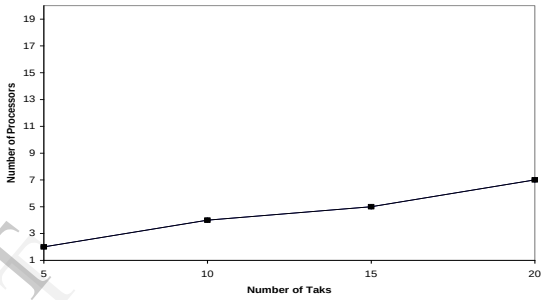

Fig. 5.2 Performance of RMFF

For α = 0.5

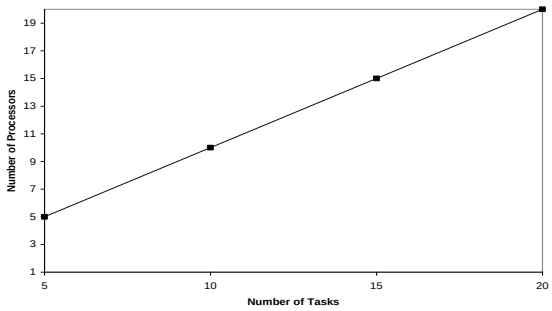

Fig. 5.3 Performance of RMBF

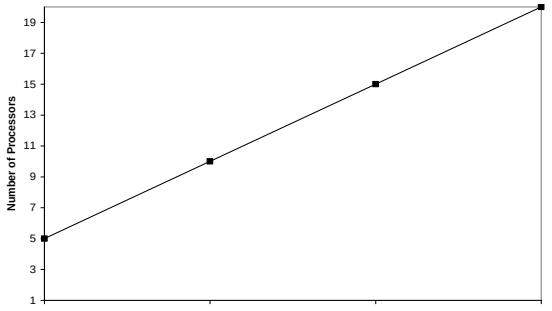

Fig. 5.4 Performance of RMFF

For α = 0.8

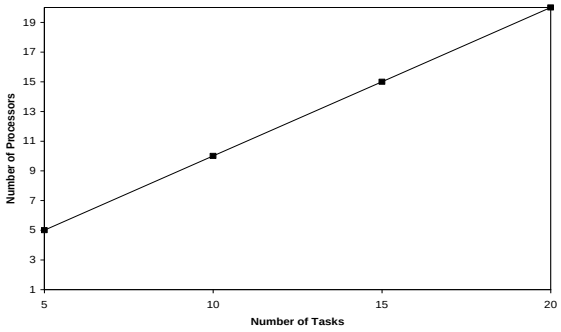

Fig. 5.5 Performance of RMBF

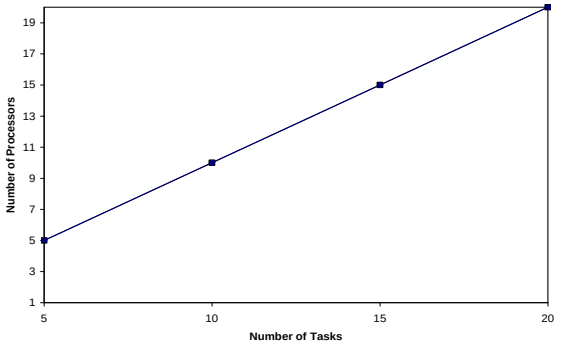

Fig. 5.6 Performance of RMFF

6. SUMMARY AND FUTURE WORK

An increasing number of applications with traffic constraints are running on homogeneous processors. In both scheduling schemes, it is shown that for particular number of tasks, if the suggested number of processors are not available ,scheduling fails. The traffic/task pattern which causes this is called disruptive traffic. Present schedulers focus on bin-packing algorithms and not capable of handling disruptive traffic. Schedulers to work with disruptive conditions are to be formulated. Also scheduling algorithms must look into handling different types of traffic cases from Poisson to general arrivals.

References:

[1] Kai Nagel," Traffic Networks" ETH Zurich,CH-8092 Zurich ,Switzerland 2002

[2] Sergio Cavalientand Sergio Terzi," Proposal of a performance Measurement system for the evaluation of scheduling solutions" Int.J. Manufacturing Technology and Management, Vol 8,Nos1/2/3,2006,Department of Industrial Engineering,University of Bergmo,Dalmine,BG,Italy.

[3] J. W. S. Liu, "Real-Time Systems", Prentice-Hall, 2000.

[4] C. L. Liu and W. Layland, ""Scheduling Algorithms for Multiprogramming in a Hard Real-Time Environment", Journal of ACM, vol. 20, number 1, pp.46-61, January 1973.

[5] T. P. Baker, "Multiprocessor EDF and Deadline Monotonic Schedulability Analysis", IEEE Real-Time Systems Symposium, Dec. 2003.

[6] J. Y. T. Leung and J. Whitehead. On the Complexity of Fixed-Priority Sceduling of Periodic, Real-Time tasks. Performance Evaluation, 2:237-250,1982.

[7] S. K. Dhall and C. L. Liu. On a real-time scheduling problem. Operations Research, 26(1):127-140, January/February 1978.