MANTECH Publications

Recent Trends in Computer Science and Software Technology
Volume 2, Issue 3, September-December 2017

Study of Scheduling Algorithm for Real Time Multiprocessing System

Pradnya Sumit Moon[1], Vrushali K. Moon[2], Karishma Chavhan[3]

Assistant Professor

Department of Computer Technology

KDK College of Engineering, Nagpur, Maharashtra, India

Corresponding Author's email: *pardhepradnya@gmail.com*

Abstract

Scheduling is a technique which makes an arrangement of performing certain tasks at specified period. The intervals between each function have been clearly defined by the algorithm to avoid any overlapping. The real time computing systems are those in which there are strict timing constraints that have to be met to get the correct output i.e. the output not only depend on the correctness of the outcome but also on the time at which results are produced. Real time systems are expected to change its state in real time even after the controlling processor has stopped its execution. The bound in which real time applications are needed to respond to the stimuli is known as deadline. In order to achieve optimized results in real rime operations the scheduling techniques has been used. In the paper we classify the various scheduling techniques based on different parameters. Also techniques used for scheduling in real time environment are analyzed and comparison between different techniques has been done.

Keywords: *Real Time Operating System, uniprocessor, Multi-Processor, Scheduling, periodic, soft real time, hard real time.*

I. INTRODUCTION

In real time operating system the output of the system is not only depending on logical correctness of the result but also the specific time at which the result will be produced. In real time system the excellent quality of

result is based on specific time constraint. Real-time applications extend over a large range of activities, like embedded systems, Industrial Applications, Automotive and Transportation, Peripheral Equipment, Telecommunication Applications, Aerospace, Internet and Multimedia Applications, Consumer Electronics, Defense Applications, Miscellaneous Applications etc.

To handle the multiple task scheduling is an important element in real-time systems to assure hard and soft timing constraints. Task scheduling allocates processor to the tasks which are executed under a real-time operating system with the aim of achieving optimal or near optimal quality of service.

In real time system scheduling is the aspect of resource management concerned with ensuring that a set of tasks meets its timing requirements. In a real-time application, real time tasks are the basic executable entities that are scheduled. In real time task occur over a period of time and they are classified into periodic, sporadic and a periodic and they have soft or hard real-time constraints. Periodic task occur after a fixed time instants. Sporadic task occur after a random time instants. A periodic task can occur in quick succession.

A. Real-Time Scheduling Strategies

The job release time are not priori, so the pre scheduling off-line is not possible with sporadic and some asynchronous periodic task systems. To overcome from off-line scheduling on-line scheduling are required. In On-line scheduling algorithm M jobs are scheduled according to assign highest priority at any instance. First we will discuss scheduling on uniprocessor and after that scheduling on multiprocessor.

B.Scheduling on Uniprocessors

Uniprocessor platforms include one processor on which jobs can get executed. In uniprocessor scheduling there is central state information of the entire task. In a single processor multi-programming system, multiple processes are contained within memory. Processes survive between Running, Ready, waiting, Blocked, and Suspend. A main goal is to keep the processor busy, by allocating task to the processor to execute, and always having at least one process able to execute.

The key to keeping the processor busy is the main purpose of process scheduling.

Uniprocessor scheduling is categorized into three types: Long term scheduling, medium term scheduling and short term scheduling.

Long-term scheduling: It will take the decisions to introduce new processes for execution or re-execution.

Medium-term scheduling: the decision to add to the processes that are fully or partially in memory.

Short-term scheduling: the decisions as to which process to execute next.

The goal of the scheduling in uniprocessor is to achieve high processor utilization, high throughput, number of processes completed per unit time and Low response time. But uniprocessor scheduling affects the performance of the system, because it determines which process will wait and which will progress.

C. Scheduling on Multiprocessors

Multiprocessors platforms include more than one processor on which jobs can get executed. Multiprocessor support for the multiprogramming as well as the parallel programming

The approaches to multiprocessor real-time scheduling can be categorized into two broad classes: partitioned and global. Under partitioning, the set of tasks is statically partitioned among processors, that is, each task is assigned to a unique processor upon which all its jobs execute. In contrast to partitioning, under global scheduling, a single, system wide, priority space is used, and a global ready queue is used for storing ready jobs.

The performance of each and every scheduling algorithm depends on performance parameters like deadline of a task, release time of a task, execution time of a task, laxity of a task, CPU utilization of a task, number of preemption, resource utilization, high throughput etc and all the tasks will be scheduled according to their unique assigned priority. In this paper we are going to show comparative study of the multiprocessor scheduling algorithm with some performance parameter.

II. MULTIPROCESSOR SCHEDULING ALGORITHM

A. Base-utilization Partitioning Algorithm for Multiprocessors

The main purpose of base-utilization partitioning algorithm [1] is according to the

utilization of each task allocate the tasks directly or indirectly onto right processors. It will find the closest full utilization and balance the workload for each processor. A base-utilization algorithm to allocate tasks on to appropriate processor of multi-core platform actualizes a bin-packing algorithm with the higher utilization and lower time complex. Suppose there are m+1 processor in the multi-core system, one processor is the center processor and other m processors are the boundary processors.

The main work of the center processor is distributing the period tasks to the boundary processor‟s scheduling queue. The base-utilization scheduling algorithm has described as.

First, partitioned the period task into two tasks set, one is the high-utilization set and the other is the low utilization set. After partitioning the two task sets, select all complementary tasks or similar complementary tasks according the utilization into different sets, then chose m different sets dispatch them into the m processors‟ scheduling queue. Thirdly, the m processors can use the deadline drive scheduling algorithm execute the tasks in their own scheduling queue.

In this proposed algorithm authors have concluded that this algorithm performs significantly better with respect to system schedulability and use the fewer processors complete the same work.

B. Energy Efficient Technique

In 2008 Euiseong Seo et. al. presented an energy efficient technique for scheduling real time tasks on multicore processors to lower the power consumption and increasing the throughput[2]. They presented two techniques which modify existing techniques of unicore processors for multicore processors. The two techniques suggested by them are:

(i) Dynamic Repartitioning algorithm, which dynamically balances the task loads on multiple cores to minimize the power consumption during execution.

(ii) Dynamic core scaling algorithm, which reduces leakage power consumption by adjusting the number of active cores. The simulation results show that 25% of energy consumed can be conserved by dynamic repartitioning and 40 % is conserved by dynamic core scaling.

C. Constant Complexity Scheduling

In 2008, S. Roman et. al. Presented Constant complexity scheduling for hardware multitasking in two dimensional reconfigurable field-programmable gate arrays to manage the FPGAs it described the need of extending operating system [3]. An algorithm has been presented which decides the scheduling and placing of arrival tasks with real time constraints, like deadline, in FPGA device.

The FPGA is divided into to four parts. Each part has an associated queue where the hardware manager places each arriving task depending on its size, shape and real time parameters as deadline requirements. The algorithm may change the queue selection policy, partition strategies and the sizes or the number of partitions at run-time in order to adapt itself in function of special characteristics of task profiles, taking into account task sizes and execution times for tasks in each queue. It was also possible to make changes in the size of partition, number of partitions, queue selection policy etc. at the run time.

The main purpose, in this paper, is to prove that „area-greedy "algorithms used in other approaches are not the most effective in environments where fast decision making is crucial and that this excessive focusing on area use does not really result in a significant improvement of total execution time of a certain set of tasks. The results proved that their four partition configuration is equally or more effective.

D. Response-Time Bound In Fixed-Priority Scheduling With Arbitrary Deadlines

In 2009, Enrico Bini et. al. Presented a response-time bound in fixed-priority scheduling with arbitrary deadlines[4] in which they indentified three desirable properties of estimates of the exact response time. Repeated computation of the worst case response times slows down the system which is undesirable for real time applications.

They proposed a technique which possessed the properties; those are continuity with respect to system parameters, efficient computability and approximability for the estimation of worst case response time of sporadic task systems which are fixed priority scheduled on a preemptive uniprocessor.

They have derived the continuous upper bound on the response times of task systems and the proved that the exact response time should be as large as this bound if the system is implemented on the processor which is at most only 50% as fast.

E. Fixed Priority Zero Laxity

In 2011, Robert I. Davis and Alan Burns presented ,"FPZL Schedulability Analysis". FPZL is a minimally dynamic global scheduling algorithm [5]. Under minimally dynamic also it follows preemptive scheduling. FPZL algorithm does not work on any priority techniques. The aim behind FPZL algorithm was to schedule as many tasks as far as possible. The key idea behind FPZL is that the jobs are scheduled according to the fixed priorities assigned until a zero laxity state is reached.

Zero Laxity state is the state where remaining execution time of a job is equal to the time to its deadline. This kind of Zero Laxity job will be missing its deadline unless it executes continually until completion. In FPZL such zero laxity jobs gets the higher priority. As the priority of a job changes at most once during its execution this algorithm is considered as a minimally dynamic scheduling algorithm and hence it does not follow any priority order.

F. Dynamic Priority Zero Laxity

There are many situations where some portions of tasks are very important which is to be scheduled on a priority basis than the other portions of the task. In this algorithm [6] the task is logically divided into two subtasks, mandatory and optional.

Using this concept the mandatory portion of every task will be executed first and later on the optional portions will be executed. It will take input of tasks containing computation time, mandatory portion, Optional portion, deadline, and period. As well as some input for processor parameters including Number of processors, period and capacity. After assign the input priorities are assigned to all the tasks including mandatory and optional tasks according to the Utilization.

The task which will utilize the system more is assigned highest priority and accordingly the priorities are assigned. Schedule tasks according to these assigned priorities. The mandatory tasks will be executed first and then the optional tasks will be executed. While executing mandatory tasks if any zero

laxity task arrives, then give that zero laxity task a higher priority than the task which was currently executing and add that pre-empted task to the ready queue and allocate the processor to the zero laxity tasks till it completely gets executed.

III. COMPARISON BETWEEN DIFFERENT MULTI-PROCESSOR SCHEDULING ALGORITHMS

It will show the comparison between the scheduling algorithms for multi-Processor real time system on the basis of performance parameters.

Table: 1

SR. No.	ALGORITHM	AUTHOR	PERFORMANCE PARAMETER
1	Base-utilization Partitioning Algorithm for Multiprocessors	Huang Shujuan and Zhu yi"an [1]	Total number of task, Utilization of processor, load balance
2	Energy Efficient Technique	Euiseong Seo Jinkyu Jeong, Seonyeong Park, and Joonwon Lee [2]	Dynamic power consumption
3	Constant Complexity Scheduling	S. Roman, H. Mecha, D. Mozos and J. Septien[3]	Workload Volume
4	Response-Time Bound In Fixed-	Enrico Bini, Thi Huyen Chau Nguyen,	Response-Time Analysis

	Priority Scheduling With Arbitrary Deadlines	Pascal Richard, and Sanjoy K. Baruah[4]	
5	FPZL	Robert I. Davis and Alan Burns [5]	Deadline Analysis, CPU Utilization and Response Time, Zero Laxity
6	DPZL	Komal Bhalotiya1, Dr. D. V. Padole and Prof. Radhakrishna Naik[6]	Deadline Analysis, CPU Utilization and Response Time, Zero Laxity

V. CONCLUSION

In this paper, some scheduling algorithm algorithms for multiprocessor are studied and a comparative study has been done on the basis of performance parameters for measuring effectiveness of algorithms. From the comparative study it can be concluded that real time systems have became an inevitable part of our lives and their accurate performance has been a challenge. From the comparison it is found that number of factors was found that affect the performance of the scheduling algorithms. Under workloads the Base-utilization Partitioning Algorithm, Energy Efficient Technique, Constant Complexity Scheduling algorithms perform comparably. Under Response-Time Analysis Response-Time Bound, FPZL, DPZL perform comparably.

REFERENCES

[1] Huang Shujuan and Zhu yi‟an , “Base-Utilization Partitioning Alogrithm for ultiprocessors” College of Computer Science &Engineering Northwestern Polytechnical University 2011 IEEE

[2] Euiseong Seo Jinkyu Jeong, Seonyeong Park, and Joonwon Lee., "Energy Efficient Scheduling of

Real-Time Tasks on Multicore Processors", IEEE transactions on parallel and distributed systems, vol. 19, no. 11, November 2008, pp 1540-1552.

[3] S. Roman, H. Mecha, D. Mozos and J. Septien, " Constant complexity scheduling for hardware multitasking in two dimensional reconfigurable field-programmable gate arrays", IET Computer Digit. Tech., 2008, Vol. 2, No. 6, pp. 401–412.

[4] Enrico Bini, Thi Huyen Chau Nguyen, Pascal Richard, and Sanjoy K. Baruah, "A Response-Time Bound in Fixed-Priority Scheduling with Arbitrary Deadlines", IEEE Transactions On Computers, Vol. 58, No. 2, February 2009, pp. 279-286.

[5] Robert I. Davis and Alan Burns, "FPZL Schedulability Analysis" 2011 17th IEEE Real-Time and Embedded Technology and Applications Symposium

[6] Komal Bhalotiya1, Dr. D. V. Padole and Prof. Radhakrishna Naik, "A Variant of FPZL Algorithm" IJCSI International Journal of Computer Science Issues, Vol. 9, Issue 4, No 1, July 2012

[7] Komal Bhalotiya1Customized Multiprocessor Scheduling Algorithms for Real time Systems, MPGI National Multi Conference 2012 (MPGINMC-2012), 7-8 April, 2012

[8] Li Jie; Guo Ruifeng; Shao Zhixiang, "The research of scheduling algorithms in real-time system" IEEE CONFERENCE PUBLICATIONS 2010 , Page 333 – 336

[9] Ishan Khera , Ajay Kakkar Comparative Study of SchedulingAlgorithms for Real Time Environment",International Journal of Computer Applications (0975 – 8887) Volume 44– No.2, April 2012.

[10] Apurva shah,ketan Kotecha,"Adaptive Scheduling Algorithm for real time multiprocessor System",IEEE Advance computing Conference,2009.