Available Online at www.ijcsmc.com

International Journal of Computer Science and Mobile Computing

A Monthly Journal of Computer Science and Information Technology

ISSN 2320–088X
IMPACT FACTOR: 6.017

IJCSMC, Vol. 6, Issue. 3, March 2017, pg.67 – 71

Comparative Study of RM and EDF Scheduling Algorithm in Real Time Multiprocessor Environment

SUMIT KUMAR NAGER, NASIB SINGH GILL

Department of Computer Science & Applications, Maharshi Dayanand University, Rohtak, Haryana, India

Email: sumitnager18@gmail.com, nasibsgill@gmail.com

Abstract: This paper discusses an analysis between scheduling algorithms in real-time multiprocessor environment. Many studies are already present regarding real-time scheduling algorithms. These constitute observations of real-time scheduling algorithms from the viewpoint of different traits particular to the system characteristics. In this paper algorithms especially EDF (Earliest deadline First) and RM (rate-monotonic scheduling) are studied. There are numerous misguided judgments exist about the properties of these two scheduling algorithm. This paper looks at RM against EDF under a few perspectives, utilizing existing hypothetical results or straightforward counterexamples to demonstrate that numerous regular convictions are either false or just limited to particular circumstances. Also, the parameters that can be used to evaluate algorithms also are defined. The paper is concluded by way of discussing outcomes of the study and futuristic research inside the subject of RM and EDF in real-time multiprocessor environment.
Keywords: CPU Scheduling, Rate-Monotonic, Earliest Deadline First, Data-Monotonic, DCP

1. INTRODUCTION

A real-time machine is one in which the correctness of the computations now not only depends on their logical correctness, however also at the time at which the result is produced. Real-time systems have timing necessities that ought to be assured. It leads to the proposal of the deadline which is a fashioned thread among all real-time approach items and the core of the change between real-time programs and time-sharing techniques. The period at which an outcome is anticipated to attain called a deadline. The deadline of an activity is the point in time before which the project need to complete its execution. There are 3 varieties of deadlines: **Soft Deadline:** If the after-effects produced afterward the deadline has anesthetized and is still advantageous again this blazon of the deadline is accepted as a soft deadline. **Firm deadline:** This deadline is one in which the after-effects produced afterward the deadline is absent is of no utility. Infrequent deadline misses are tolerable. **Hard deadline:** If accident after-effects on missing the deadline again this blazon of the deadline is accepted as a hard deadline.

There are varieties of real-time activity, depending on their arrival sample: periodic activity (the activity has a normal inter-arrival time referred to as the period, a deadline and a computation time) and aperiodic activity (the activity can arrive at any time; this sort of undertaking is characterized with the aid of a computation time and a deadline; the latter is most likely regarded as soft). Scheduling mechanism is the maximum critical factor of a computer system. Scheduling is the approach with the aid of which the device decides which task should be finished at any given time. There may be the difference among real-time scheduling and multiprogramming timesharing scheduling.

1.1 WRITINGS EXAMINATION

This paper presents a comparison of different scheduling algorithms for a real-time multiprocessor environment. In the year 2011, Devendra Thakor achieved a work," D-EDF [1]: A green Scheduling algorithm for real-Time Multiprocessor device". The proposed set of rules D-EDF [1] performs properly at some point of overloaded conditions. In underneath loaded conditions it makes use of EDF coverage and in overloaded situations, it uses deadline Monotonic (DM) [18] policy. It takes permissions of both Earliest deadline First (EDF)[7] and DM[18] algorithm .It's miles simulated and examined for preemptive independent periodic responsibilities on tightly coupled real-time multiprocessor machine under international scheduling[1]. In the year 2008, Murthy Durbhakula accomplished a work," Sharing-conscious OS Scheduling Algorithms for Multi-socket Multi-center Servers". This paper affords that to reduce the impact of inter-socket cache to cache transfers this paper gives a brand new OS scheduling optimization. For every pair of thread for each scheduling quantum, it examines the sample of cache to cache transfers and applies for an exclusive algorithm. Originator computes a new timetable of threads for every quantum. The brand new agenda reduce down inter-socket cache to cache transfer for subsequent timetable quantum [2]. In the year 2010, Andreas Merkel executed a work," resource-aware scheduling for energy performance on Multicore Processors". Writer rents the concept of task interest vectors for characterizing applications with the aid of resource usage. author combine the guidelines into an operating device scheduler and into a virtualization machine, allowing placement selections to be made both inside and across bodily nodes, and lowering contention both for individual responsibilities and complete packages.[3] In the year 2009, Jin Cui achieved a work," Dynamic Thermal-aware Scheduling on Chip Multiprocessor for soft real-time gadget". creator recommend two special metrics for figuring out wherein to place the challenge: one primarily based on the maximum middle temperature in the publish-thermal map; and some other based on the sum of the made from the publish-thermal map temperature and the remaining challenge runtime for every center. [4] Liu and Layland [5] came up with Rate Monotonic scheduling (precedence pushed scheduling) of real- time running systems that are greatest scheduling set of rules carried out in almost all the real-time working machine. From the paintings performed via the various researchers in the discipline of real time scheduling; to date, it's been observed that: a) Scheduling will have to be accomplished as a way to guarantee the agenda of the strategies relatively and throughput ought to be highest. b) Real-time scheduling algorithms are consistently pre-emptive which can accomplish bigger if the pre-emption is limited. c) Static precedence scheduling algorithms are used for scheduling actual time obligations for maximum CPU utilization but it could be extended extra the use of dynamic priorities. d) The schedulability of scheduling algorithm ought to be checked the use of schedulability checks. e) Starvation should not be there which agency an accurate action should not be captivated indefinitely. Allocation of resources should be such that all the processes get able CPU time in adjustment to anticipate starvation. f) In the case of precedence based algorithms, there should be openness in the pre-emption policy. Low antecedence tasks should not delay indefinitely because of college antecedence tasks.

2 TAXONOMY OF SCHEDULING ALGORITHMS

A scheduler affairs that arranges jobs or a computer's operations into an adapted sequence, according to some criteria. A job scheduler can admit and administer jobs automatically by processing able job ascendancy accent statements or through agnate alternation with the human operator. There are altered types of scheduling. Here are the lot of important: **Optimal or non-optimal:** A most beneficial scheduling can schedule an undertaking set even supposing the mission set is schedulable by way of some scheduler. **Preemptive or non-preemptive:** In preemptive scheduling a task may be suspended in between if there comes a activity with higher assets and reschedule for latter time. Non-preemptive scheduling does no longer suspend the activities in such way, it waits for the mission to complete first after which the mission with higher priority is assigned. **Static or dynamic:** Static scheduling calculates the execution order of tasks before run-time. It requires knowledge of task characteristics but produces little run-time overhead. Dynamic scheduling makes decisions during the run-time of the system. This allows to design a more flexible system, but it means some overhead.

Precedence-driven scheduling is dynamic which makes a decision what task to execute relying on the priority of the undertaking. This style of scheduling also is also fixed or dynamic, depending on whether the activity priority varies in the course of run-time. Precedence-driven schedulers had been largely utilized in real-time methods.

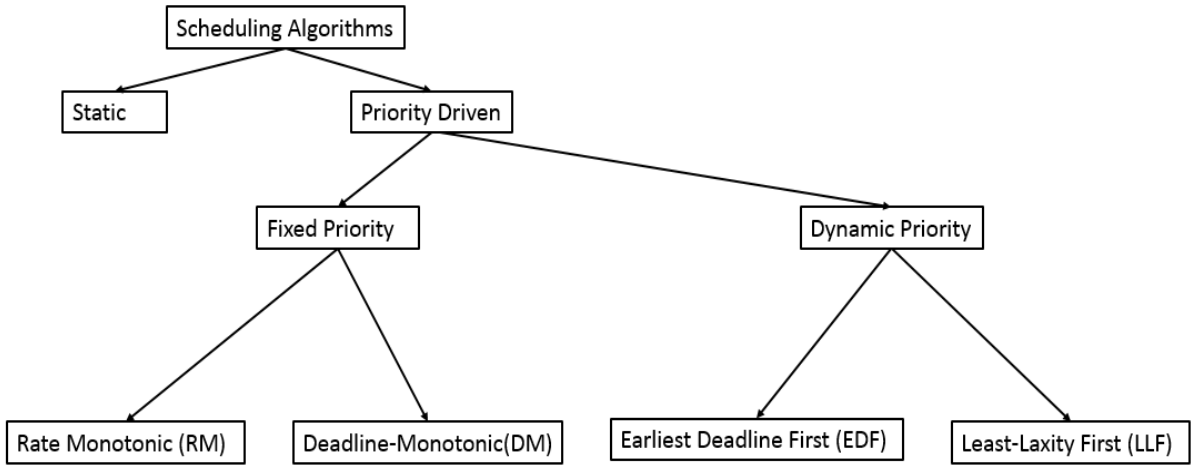

Figure -1. Scheduling Algorithms classification

3 PRIORITY-DRIVEN ALGORITHMS

A priority driven algorithm uses a priority driven scheduler that doesn't pre-empt the scheduled task mean the task which are going to be executed are put in simple queue and await for their turn to be executed based on a certain algorithm.

3.1. Fixed-priority scheduling: The maximum essential scheduling algorithms on this class are Rate Monotonic (RM) [6] and deadline Monotonic (DM) [7]. The previous assigns the better priority to the activity with the shortest length, assuming that durations are identical to time limits. The latter assigns the highest priority to the challenge with the shortest deadline. Each algorithm is the premiere.

3.2. Dynamic-priority scheduling: In this class, Earliest deadline First (EDF) [6] and Least Laxity First (LLF) [8] are the most essential. Each is top of the line, if any algorithm can find a program in which all obligations meet their deadline then EDF can meet the deadline. In EDF, the assignment with the closest absolute deadline has the very best priority. The Absolute deadline is the instant in time at which the response have to be completed. The Least-Laxity-First (LLF) [8] scheduling set of rules assigns better priority to an activity with the least laxity. The time period laxity is outlined as the temporal difference between the deadline, the ready time and the run time. The term laxity is defined because of the temporal difference between the deadline, the equipped time and the run time. Dynamic-priority algorithms achieve excessive processor utilizations they could adapt to dynamic environments, wherein mission parameters are not acknowledged.

4. ELEMENTARY FACTORS THAT HAVE TO IMPACT THE PERFORMANCE IN REAL-TIME MULTIPROCESSOR ENVIRONMENT

Following are the components that assume fundamental part in the performance in real-time multiprocessor environment:

4.1 Scheduler: A scheduling algorithm determines how Real-time working process assigns CPU time for activity execution. The scheduler is the high element of the kernel because it affects the way in which wherein the approach will execute duties and computer cycles. The scheduler executes periodically and for the duration of the state transition of the thread. Essentially the most original scheduling used in actual time operating approach is Preemptive scheduling.

4.2 Event Latency and Jitter: An occasion could be an interrupt request by means of a few hardware or an activity being scheduled by the working machine. For a hardware interrupt, the latency is the time elapsed from the interrupt request generation to the execution of the primary line of ISR code. For a system occasion, latency is the time elapsed from the era of the sign project to the execution of the primary instruction of the mission. Jitter is any variant of the constant term for a challenge execution, servicing of an interrupt and many others.

4.3 Priority Inversion: Priority inversion is a major drawback of the most real-time operating process. On this, the scale lower priority activity pre-empts the higher priority activity and this reverses the priority structure. The most real-time running system has a provision to avoid priority inversion and the long-established ones are priority ceiling and priority inheritance. In priority ceiling, each system that shares a resource is assigned a priority equal to the best possible priority system which may lock the resource. In priority inheritance, if a high priority activity is ready for a currently being executed lower priority activity, then the low precedence activity is quickly (until its execution) assigned the priority of the waiting highest priority activity. This avoids the pre-emption of lower priority activity by added average priority activity.

4.4 Memory Management: It's the allocation of memory space accessible in the system, with any activity that requests it. Afterward, the activity achievement appliance using allocated the memory area and does not require it now any in addition, then the memory needs to be deallocated and made to the other threads. Distinct kernels use exclusive techniques to make sure this. The two common models are the Flat Memory Model and Segmented Address Model.

5 COMPARATIVE ANALYSIS

In 1973, Liu and Layland [9] analyzed the houses of two common priority task rules: the Rate Monotonic (RM) Fixed priority scheduling algorithm and the Earliest deadline First (EDF) dynamic scheduling algorithm. In keeping with RM, duties are assigned fixed priorities which might be proportional to their rate, so the task with the smallest interval receives the best priority. In step with EDF, priorities are assigned dynamically and are inversely proportional to the deadline of the active task.

5.1 Implementation complexity: If the range of priority ranges isn't excessive, the RM algorithm may be applied greater efficiently by splitting the ready queue into several FIFO queues, one for every precedence degree. In the case of EDF the absolute deadline exchange from a task to the other and want to be computed at each process activation. Which isn't always known at the start, this it makes the EDF extra complex to implement.

5.2 Runtime overhead: It is not present beneath RM, back periods are about fixed. However, the drawback of evaluating the runtime overhead offered EDF introduces a greater runtime overhead than RM, considering the fact that in EDF absolute deadline ought to be up to date from a job to the opposite, so reasonably increasing the time needed to execute the job activation primitive.

5.3 Number of pre-emption: Beneath RM, the number of preemptions always increases with the burden, on the grounds that activities with longer execution instances have extra probabilities to be preempted via activities with larger priorities. Below EDF, however, increasing activity execution instances does now not continually imply a better quantity of preemptions, due to the fact that an undertaking with an extended period would have an absolute deadline shorter than that of an activity with a smaller interval. In distinctive instances, an expanded execution time may intend a curb number of preemptions.

5.4 Schedulability Analysis: if comparative deadlines are according to periods, exact schedulability study can be performed in O(n) beneath EDF, however, is pseudo-polynomial beneath RM. If comparative deadlines are beneath than periods, the study is pseudo-polynomial for both scheduling algorithms.

5.5 Robustness during Overloads: If the system turns into overloaded, the most effective difference among RM and EDF is that, beneath RM, an overrun in mission τi cannot purpose activities with higher priority to missing their deadline, whereas under EDF another challenge should pass over its deadline.

5.6 Jitter: The maximum time variation (relative to the release time) in the occurrence of a particular event in two consecutive instances of a task defines the jitter for that event. In real-time applications, the jitter can be tolerated when it does not degrade the performance of the system. In many control applications, however, a high jitter can cause instability or a jerky behaviour of the controlled system [10], hence it must be kept as low as possible. EDF always introduces fewer jitter than RM.

5.7 Resource Sharing: Altered protocols are acclimated in the aggregate with both the RM ((Priority Inheritance Protocol (PIP) or the Priority Ceiling Protocol (PCP) [11]) and Dynamic Priority Ceiling (DPC) [12], the Stack Resource Policy (SRP) [13],)to accord with the botheration of priority inversion.[14, 15]

5.8 Resource Reservation: The trouble brought about with the aid of execution overruns may also be solved via implementing temporal isolation amongst activities through a resource reservation mechanism in the kernel. Below RM, the situation has been investigated via Mercer, Savage and Tokuda [16], who proposed a capability reserve mechanism that assigns each and every activity a given funds in each period and downgrades the assignment to a background level when the reserved capability is exhausted. Beneath EDF, a similar mechanism has been proposed by Abeni and Buttazzo, via the regular Bandwidth Server [17]. This system additionally uses a budget to reserve a preferred processor bandwidth for each and every project, however, it is more effective in that the task is not completed in the background when the funds are exhausted. As a substitute, deadline postponement mechanism ensures that the used bandwidth by no means exceeds the reserved value.

6. CONCLUSION AND FUTURE SCOPE

From the comparative study, we may establish that the idea of "time" is vital in real-time systems. These system for the most part include various procedures that challenge each other. The need of scheduling is particularly imperative for constant programming design and examination. From the examination of the real-time scheduling algorithm, we can derive that the earliest deadline first is the proficient scheduling algorithm if the CPU usage is not more than 100% but rather scales well when the system is over-burden. The genuine advantage of RM concerning EDF is its less difficult implementation in commercial kernels that don't give unequivocal support to timing requirements such as periods and deadline. Different properties ordinarily asserted for RM, such as consistency amid over-loaded conditions, or better jitter control, only apply for the highest priority task, and do

not hold in general. Then again, EDF permits a full processor use, which suggests a more proficient exploitation of computational assets and a greatly improved responsiveness of aperiodic activities. In future, new algorithm ought to be created which is a blend of fixed and dynamic priority or a scheduling algorithm switch consequently between EDF algorithm and fixed based scheduling algorithm to deal over-loaded and under loaded conditions. The new algorithm will be exceptionally valuable when the future workload of the system is alterable.

REFERENCES

[1] DevendraThakor," D-EDF: An efficient Scheduling Algorithm for Real-Time Multiprocessor System", 2011 World Congress on Information and Communication Technologies 978-1-4673-0125-1@2011 IEEE (pp 1048-1053).

[2] Murthy Durbhakula," Sharing-Aware OS Scheduling Algorithms for Multi-socket Multi-core Servers", ACM 2008 ISBN: 978-1-60558-407-2/08/11

[3] Andreas Merkel," Resource-conscious Scheduling for Energy Efficiency on Multicore Processors", EuroSys'10, April 13–16, 2010, Paris, France. ACM 978-1-60558-577-2/10/04 (pp 153-166)

[4] Jin Cui," Dynamic Thermal-Aware Scheduling on Chip Multiprocessor for Soft Real-time System", GLSVLSI'09, May 10–12, 2009, Boston, Massachusetts, USA. ACM 978-1-60558-522-2/09/05 (pp 393-396)

[5] C. Liu and James Leyland, January 1973. *Scheduling algorithm for multiprogramming in a hard real-time environment.* Journal of the Association for Computing Machinery, 20(1): 46-61.

[6] C. Yaashuwanth and R. Ramesh, 2010. *Design of real time scheduler simulator and development of modified round robin architecture.* International Journal of Computer Applications.

[7] Fengxiang Zhang and Alan Burns, September, 2009. *Schedulability analysis for real-time systems with EDF scheduling.* IEEE Transactions on computers, vol. 58, no. 9.

[8] S. Baskiyar and N. Meghanathan, 2005. *A Survey On Real Time Systems.* Informatica (29), 233-240.

[9] C.L. Liu and J.W. Layland, "Scheduling Algorithms for Multiprogramming in a Hard real-Time Environment", Journal of the ACM 20(1), 1973, pp. 40–61.

[10] P. Marti, G. Fohler, K. Ramamritham, and J.M. Fuertes, "Control performance of flexible timing constraints for Quality-of-Control Scheduling," Proc. of the 23rd IEEE Real-Time System Symposium, Austin, TX, USA, December 2002.

[11] L. Sha, R. Rajkumar, and J. P. Lehoczky, "Priority Inheritance Protocols: An Approachto Real-Time Synchronization," IEEE Transactions on Computers, 39(9),pp. 1175–1185, 1990.

[12] M.I. Chen, and J.K. Lin, Dynamic Priority Ceilings: A Concurrency Control Protocolfor Real-Time Systems, Journal of Real-Time Systems 2, (1990).

[13] T.P. Baker, Stack-Based Scheduling of Real-Time Processes, The Journal of Real-Time Systems 3(1), pp. 76–100, (1991).

[14] K. Jeffay, "Scheduling Sporadic Tasks with Shared Resources in Hard-Real-TimeSystems," Proceedings of IEEE Real-Time System Symposium, pp. 89–99, December1992.

[15] J. Stankovic, K. Ramamritham, M. Spuri, and G. Buttazzo, Deadline Schedulingfor Real-Time Systems, Kluwer Academic Publishers, Boston-Dordrecht-London,1998.

[16] C.W. Mercer, S. Savage, and H. Tokuda, "Processor Capacity Reserves for MultimediaOperating Systems," Technical Report, Carnegie Mellon University, Pittsburg(PA), CMU-CS-93-157, May 1993.

[17] L. Abeni and G. Buttazzo, "Integrating Multimedia Applications in Hard Real-Time Systems", Proc. of the IEEE Real-Time Systems Symposium, Madrid, Spain,December 1998.

[18] J.Leung and J. Whitehead, 1982. Performance Evaluation, on the complexity of fixed-priority scheduling of periodic, real-time tasks.

