Summary: Rate Monotonic vs EDF, Judgment day

In this paper the author attempts to compare the Rate Monotonic theory and EDF theory under various conditions and prove that the claims such as the RM is easier to analyze than EDF, EDF increases runtime overhead, is less predictable in overload situations, and causes jitter in task execution are either wrong or are confined to specific situations.

In the next section the author talks about the two cases which we have to consider while looking at the implementation complexity of a scheduling algorithm. The first one being implementing the algorithm on a priority based OS and the second one being that implementing the algorithm from scratch. The author states that the development of EDF algorithm on top of a kernel with a set of fixed priorities is not easy or efficient. If the algorithm is developed from scratch using the list from the ready queue, then both RM and EDF have the same complexity. But, an advantage of RM over EDF is that for less number of priorities, the RM can be implemented more efficiently by splitting the ready queue, typically it can be implemented wit O(1) complexity. This cannot be adopted for EDF policy since the number of queues would be too large. Another disadvantage of EDF is that the absolute deadlines for the jobs change from one to another and have to be calculated at the activation of each job. This causes overhead which is absent in the EM policy.

In the third section, the author discusses the complexity of determining the runtime overhead introduced by the two algorithms. Inspite of the fact that EDF needs updated deadlines computed by the kernel before every job activation, it introduces lees overhead a compared to RM during a context switch. Also in RM the higher priority task has to preempt every instance of the lower priority task. Whereas in EDF the preemption occurs only once. As the number of task sets increases, the preemptions in EDF increase but as the task sets keep on increasing, the execution time for each task decreases hence the preemption count decreases. Also, the number of preemptions constantly increase with the load for RM policy. Under EDF however increasing the load does not necessarily mean higher preemptions because a task with longer periods can have a shorter absolute deadline.

In the fourth section, the author analysis the schedulability test for RM and EDF. The author presents data regarding the complexity of the feasibility tests for RM and EDF. Through these results the author points out that for RM as the number of tasks increase, there is an increase in the number of steps involved in calculating the feasibility increase exponentially.

In the fifth section the author compares the behavior of RM and EDF in overload conditions. The algorithms are compared under permanent overload and then under transient overload conditions. In a permanent overload situation, the EDF policy automatically performs a period rescaling whereas in RM policy, it might cause permanent blocking of the lower priority tasks. Although above said, in permanent overload situation, the decision of implementing RM or EDF algorithm is highly application specific.

In sixth section, the author analysis the jitter during task execution. He states that the belief that fixed priority in RM causes reduces the jitter as compared to EDF is a misconception. This might be the case for highest priority task but it’s not a general case.

In the last section, the author considers resource sharing in both the algorithms. In this section the author claims that the classical shared resource protection protocol like the Priority Inheritance protocol, Priority Ceiling Protocol are not only implementable on RM policy but also equally applicable on the EDF policy. The author also claims that the EDF performs more efficiently while handling aperiodic requests which might be attributed to higher CPU utilization.

In conclusion, the author conveys that the only advantage of RM over EDF is its easy implementation. The other apparent advantages of RM are only apply to highest priority tasks. On the other hand EDF allows more processor utilization and hence efficient exploitation of computational power. These properties become extremely important in time critical and resource limited systems.