ENSC 351: Real-time and Embedded Systems

Craig Scratchley, Spring 2005

Earliest-Deadline First (EDF) scheduling

Earliest-Deadline First (EDF) is a simple and effective scheduling algorithm. As its name implies, the algorithm simply schedules at any point in time the task with the earliest deadline first. The task with the next deadline is scheduled second, etc. EDF is optimal on a uniprocessor in the sense that if it is possible to schedule a task set without missing any deadlines, EDF can find such a schedule.

Figure 2.8 (from the book Scheduling in Real-Time Systems by Cottet et al.) gives an example of an EDF schedule for a set of three periodic tasks. Consider that a timer is associated with each periodic task. Each time that the timer for a periodic task triggers, consider that a thread is created to handle the task invocation, after which the thread will terminate. Each timer will first trigger at time 0 (r_0=0). Task 1 has a period of 20 units (say twenty seconds), so the timer will have a reload interval of 20 seconds (T = 20). The thread created when the task 1 timer triggers will require 3 seconds of computation time on the processor (C = 3), and the thread should complete before 7 seconds elapses from the time the timer triggered (D = 7). That is, the *deadline* of the thread is 7 seconds after the time the timer triggered. If the timer triggered at time t = 20 seconds, then the deadline is at time t = 27 seconds.

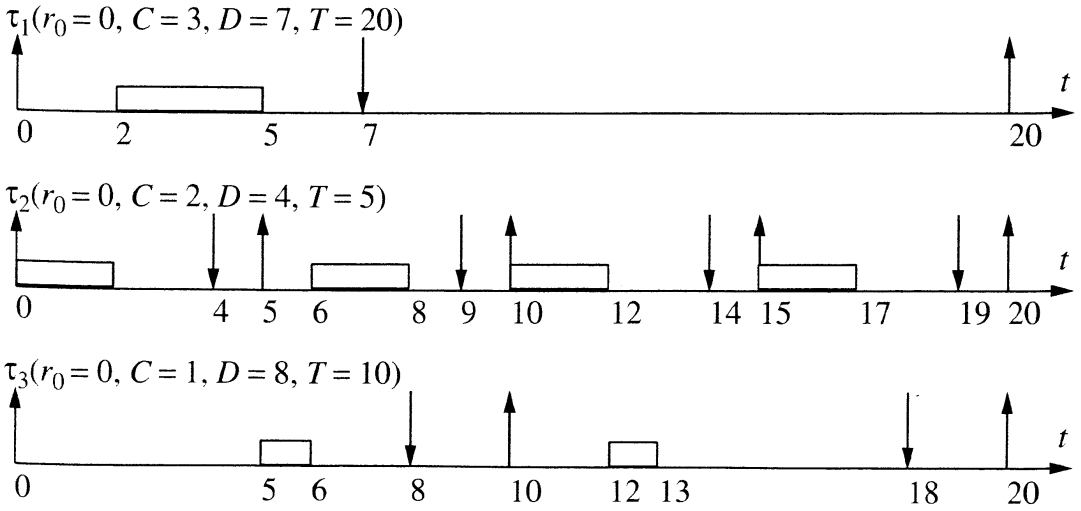


Figure 2.8 EDF schedule

EDF will first schedule an instance of task 2, then an instance of task 1, then an instance of task 3, then an instance of task 2, then after some idle CPU time, another instance of task 2, etc.

EDF is generally considered a better scheduling algorithm than Rate Monotonic Scheduling, but the 63 or so priority levels provided by QNX make it difficult to use EDF in many cases.

Figure 4.9 (from the book Hard Real-Time Computing Systems by Giorgio C. Buttazzo) shows two periodic tasks scheduled with both Rate Monotonic Scheduling and Earliest-Deadline First Scheduling. Task 1 has C = 2 and D = T = 5. Task 2 has C = 4 and D = T = 7. Both tasks have r_0=0. Note that the first invocation of task 2 misses its deadline (D = 7) with Rate Monotonic Scheduling, but no deadlines are missed with Earliest-Deadline First Scheduling.

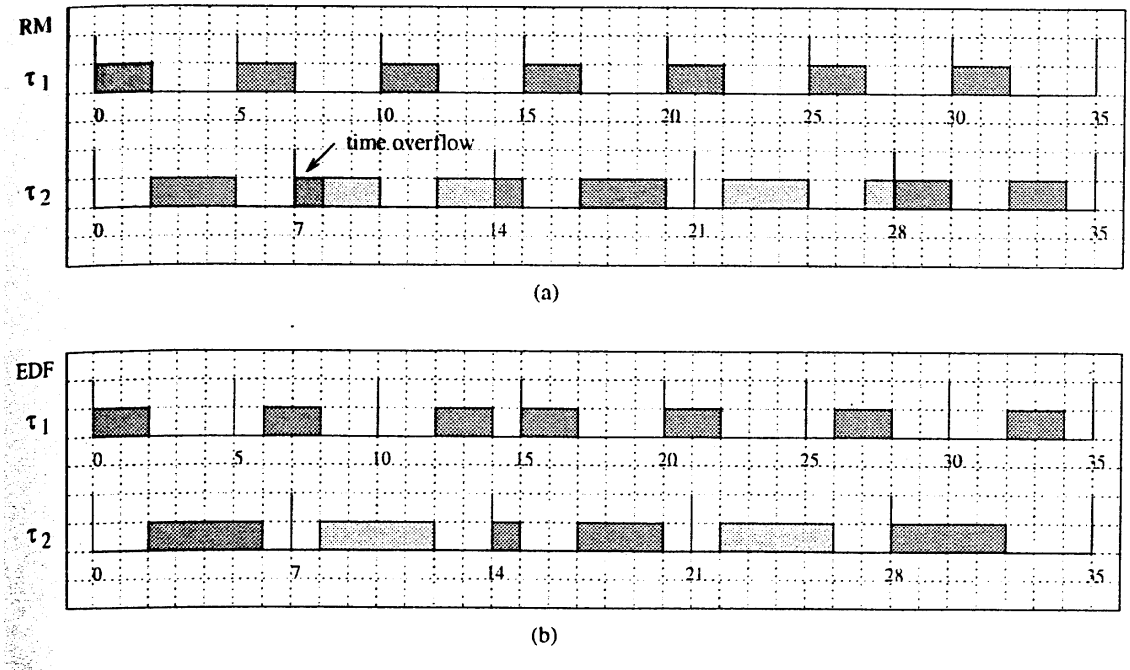
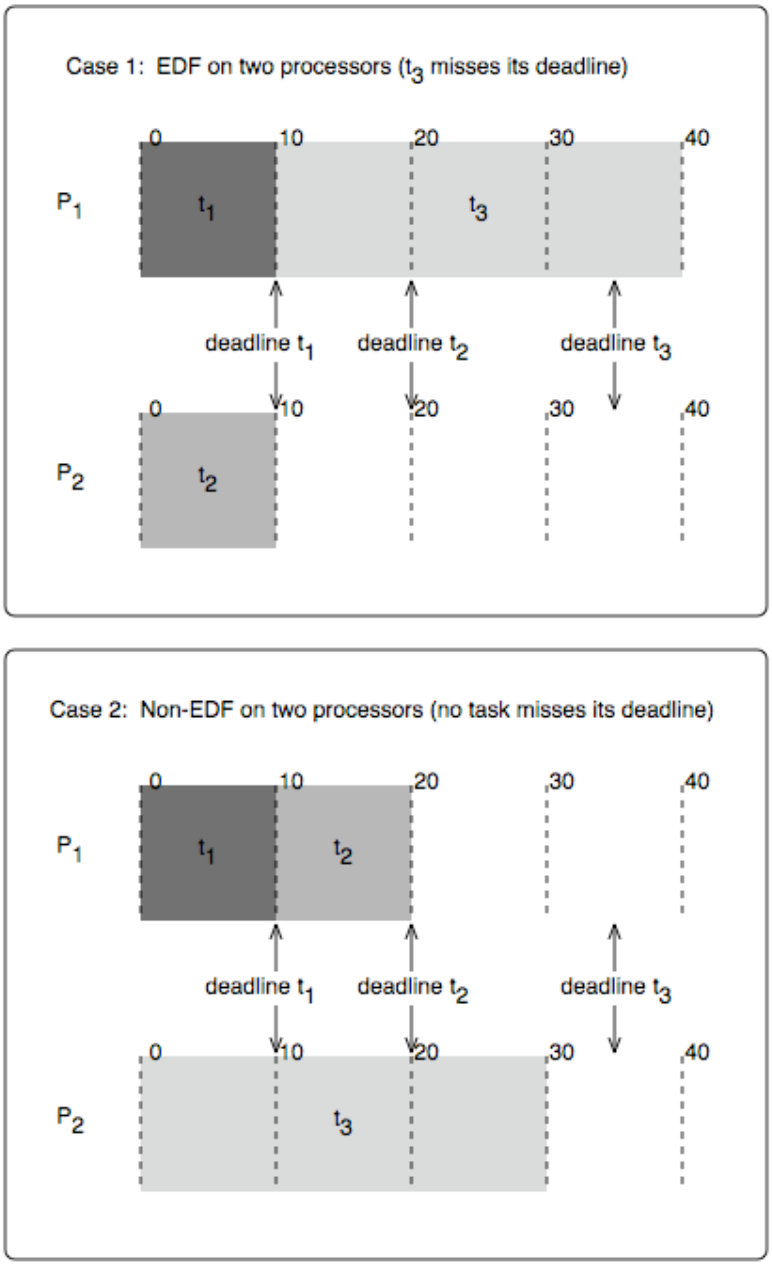


Figure 4.9 Schedule produced by RM (a) and EDF (b) on the same set of periodic tasks.

Although EDF for a set of cpu-consuming tasks can be considered optimal on a uniprocessor, it is not optimal on a multiprocessor or if the tasks use multiple devices. Short of traversing the space of possible schedules to find one that meets all deadlines (an inefficient process), there is no simple algorithm which is optimal in these cases.

The diagram below demonstrates that EDF is not optimal on a multiprocessor. The diagram shows the first invocation of each of three tasks. All tasks have r_0=0. Task 1 has C = D = 10. Task 2 has C = 10 and D = 20. Task 3 has C = 30 and D = 35.

Demonstration that EDF is not optimal on a multiprocessor



The diagram below demonstrates that EDF is not optimal when a task uses multiple devices. The diagram shows the first invocation of each of two tasks. Both tasks have r_0=0. Task 1 first uses the CPU for 10 units and then uses a disk for 20 units and has D = 35. Task 2 has C = 10 and D = 20.

Demonstration that EDF is not optimal when a task uses multiple devices