1. **EDF and Queue/Stack disciplines**

“Queue discipline” is another term for FIFO, while “Stack discipline” refers to LIFO. An EDF scheduler does not always obey the FIFO (queue) discipline, because tasks with shorter deadlines that are added later are allowed to preempt (or at least cut in front of) tasks with longer deadlines that are added earlier. EDF schedulers are also not required to obey the LIFO (stack) discipline, because a task that is added later but has a long deadline will have to wait for all previously added tasks with shorter deadlines to finish. EDF schedules are not directly related to arrival order, but instead use deadlines as the parameter for deciding execution order.

Example of how EDF does not obey FIFO (queue) discipline:

|  |  |  |  |
| --- | --- | --- | --- |
| Task | Arrival | Exec Time | Deadline |
| J­1 | 0 | 5 | 10 |
| J2 | 0 | 5 | 15 |
| J3 | 2 | 5 | 10 |

The order of these tasks will be J1, J3, J2. Even though J2 arrived before J3, it will not be scheduled until after J3 because J3’s deadline is sooner than J2’s.

Example of how EDF does not obey LIFO (stack) discipline:

|  |  |  |  |
| --- | --- | --- | --- |
| Task | Arrival | Exec Time | Deadline |
| J­1 | 0 | 5 | 11 |
| J2 | 0 | 5 | 10 |
| J3 | 2 | 5 | 15 |

The order of these tasks will be J2, J1, J3. Even though J3 arrived last, it will not be chosen as the next task to run after J2 is finished because J1’s deadline is sooner than J3’s.

1. **Subtask scheduling**

Each subtask may have to wait until there is just enough time before the deadline to run itself and all the other remaining subtasks. This moment would correspond to laxity = 0 for the subtask relative to the set of subtasks. A formula to represent the latest allowable start time could be:

for each subtask J­i.

1. **LLF scheduling**

Yes, the task set is schedulable using LLF. The diagram below shows a schedule.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **t** | 0 |  |  | 3 | 4 |  |  |  | 8 |  |  |  | 12 |  | 14 |  | 16 | 17 | 18 |  |  |  | 22 |  | 24 | 25 | 26 | 27 |  |  |  | 31 |  | 33 |  | 35 | 36 | 37 |  |  |  | 41 |  |  | 44 | 45 |  |  | 48 |  | 50 | 51 |  | 53 | 54 |  | 56 | 57 | 58 | 59 | 60 |
| **J** | E | | | B | C | | | | A | | | | E | | A | | E | B | C | | E | | C | | E | B | D | A | | | | E | | A | | E | B | C | | | | E | | | B | A | | | C | | A | E | | A | C | | A | E | B | D |  |
| Arrival | A |  |  |  |  |  |  |  |  |  | B |  |  |  |  | C |  |  |  |  | A |  |  |  |  |  |  |  |  |  | B |  |  |  |  |  |  |  |  |  | A |  |  |  |  | C |  |  |  |  | B |  |  |  |  |  |  |  |  |  |  |
| B |  |  |  |  |  |  |  |  |  | E |  |  |  |  |  |  |  |  |  | B |  |  |  |  |  |  |  |  |  | C |  |  |  |  |  |  |  |  |  | B |  |  |  |  |  |  |  |  |  | E |  |  |  |  |  |  |  |  |  |  |
| C |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | E |  |  |  |  |  |  |  |  |  | D |  |  |  |  |  |  |  |  |  | E |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| D |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | E |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| E |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

1. **Scheduling a given task set with multiple schedulers**

FCFS schedule:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **t** | | 0 | | 1 | 2 | |  |  | 5 | |  |  |  | 9 | |  |  |  |  |  | 15 | | 16 | |  |  | 19 | |  |  |  | 23 | | 24 | |  |  | 27 | |  |  |  |  |  | 33 | | 34 | | 35 | |  |  | 38 | |  |  |  | 42 | | 43 | |  |  | 46 | |  |  |  |  |  | 52 | |  |  |  | 56 | | 57 | |  |  | 60 | |
| **J** | B | | D | | | E | | | | C | | | | | A | | | | | | | B | | E | | | | C | | | | | B | | E | | | | A | | | | | | | B | | D | | E | | | | C | | | | | B | | E | | | | A | | | | | | | C | | | | | B | | E | | | |  |
| Arrival | | A | |  |  | |  |  |  | |  |  |  |  | | B |  |  |  |  | C | |  | |  |  |  | | A |  |  |  | |  | |  |  |  | |  |  | B |  |  |  | |  | |  | |  |  |  | |  | A |  |  | |  | |  | C |  | |  |  |  | B |  |  | |  |  |  |  | |  | |  |  |  | |
| B | |  |  | |  |  |  | |  |  |  |  | | E |  |  |  |  |  | |  | |  |  |  | | B |  |  |  | |  | |  |  |  | |  |  | C |  |  |  | |  | |  | |  |  |  | |  | B |  |  | |  | |  |  |  | |  |  |  | E |  |  | |  |  |  |  | |  | |  |  |  | |
| C | |  |  | |  |  |  | |  |  |  |  | |  |  |  |  |  |  | |  | |  |  |  | | E |  |  |  | |  | |  |  |  | |  |  | D |  |  |  | |  | |  | |  |  |  | |  | E |  |  | |  | |  |  |  | |  |  |  |  |  |  | |  |  |  |  | |  | |  |  |  | |
| D | |  |  | |  |  |  | |  |  |  |  | |  |  |  |  |  |  | |  | |  |  |  | |  |  |  |  | |  | |  |  |  | |  |  | E |  |  |  | |  | |  | |  |  |  | |  |  |  |  | |  | |  |  |  | |  |  |  |  |  |  | |  |  |  |  | |  | |  |  |  | |
| E | |  |  | |  |  |  | |  |  |  |  | |  |  |  |  |  |  | |  | |  |  |  | |  |  |  |  | |  | |  |  |  | |  |  |  |  |  |  | |  | |  | |  |  |  | |  |  |  |  | |  | |  |  |  | |  |  |  |  |  |  | |  |  |  |  | |  | |  |  |  | |

\*To break ties, a SJF policy was used.

EDF schedule:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **t** | | 0 | |  |  |  |  | |  |  |  |  | |  |  |  |  |  |  | |  | |  |  |  | |  | |  |  |  |  | |  | |  |  |  | |  |  |  |  |  |  | |  | |  |  |  | |  |  |  |  | |  | |  |  |  | |  | |  |  |  |  | |  | |  |  |  | |  |  |  |  |  | 60 | |
| **J** | B | | E | | | | | C | | | | | A | | | | | | | B | | E | | | | D | | C | | | | | B | | E | | | | A | | | | | | | B | | E | | | | C | | | | | B | | E | | | | D | | C | | | | | B | | E | | | | A | | | | | | |  |
| Arrival | | A | |  |  |  |  | |  |  |  |  | |  | B |  |  |  |  | | C | |  |  |  | |  | | A |  |  |  | |  | |  |  |  | |  |  | B |  |  |  | |  | |  |  |  | |  |  | A |  | |  | |  |  | C | |  | |  |  |  | B | |  | |  |  |  | |  |  |  |  |  |  | |
| B | |  |  |  |  | |  |  |  |  | |  | E |  |  |  |  | |  | |  |  |  | |  | | B |  |  |  | |  | |  |  |  | |  |  | C |  |  |  | |  | |  |  |  | |  |  | B |  | |  | |  |  |  | |  | |  |  |  | E | |  | |  |  |  | |  |  |  |  |  |  | |
| C | |  |  |  |  | |  |  |  |  | |  |  |  |  |  |  | |  | |  |  |  | |  | | E |  |  |  | |  | |  |  |  | |  |  | D |  |  |  | |  | |  |  |  | |  |  | E |  | |  | |  |  |  | |  | |  |  |  |  | |  | |  |  |  | |  |  |  |  |  |  | |
| D | |  |  |  |  | |  |  |  |  | |  |  |  |  |  |  | |  | |  |  |  | |  | |  |  |  |  | |  | |  |  |  | |  |  | E |  |  |  | |  | |  |  |  | |  |  |  |  | |  | |  |  |  | |  | |  |  |  |  | |  | |  |  |  | |  |  |  |  |  |  | |
| E | |  |  |  |  | |  |  |  |  | |  |  |  |  |  |  | |  | |  |  |  | |  | |  |  |  |  | |  | |  |  |  | |  |  |  |  |  |  | |  | |  |  |  | |  |  |  |  | |  | |  |  |  | |  | |  |  |  |  | |  | |  |  |  | |  |  |  |  |  |  | |

\*To break ties, a SJF policy was used. If deadlines for newly arrived tasks tied with the running task, the running task was not preempted.

RM schedule:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **t** | 0 |  |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | | 60 | |
| **J** | B | E | |  | |  | | C | |  | |  | |  | | A | |  | | B | | E | |  | |  | | A | | C | |  | |  | |  | | A | | \* | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  |
| Arrival | A |  |  | |  | |  | |  | |  | |  | |  | |  | | B | |  | |  | |  | |  | | C | |  | |  | |  | |  | | A | |  | |  | |  | |  | |  | |  | |  | |  | |  | | B | |  | |  | |  | |  | |  | |  | |  | |  | |  | | A | |  | |  | |  | |  | | C | |  | |  | |  | |  | | B | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |
| B |  |  | |  | |  | |  | |  | |  | |  | |  | | E | |  | |  | |  | |  | |  | |  | |  | |  | |  | | B | |  | |  | |  | |  | |  | |  | |  | |  | |  | | C | |  | |  | |  | |  | |  | |  | |  | |  | |  | | B | |  | |  | |  | |  | |  | |  | |  | |  | |  | | E | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |
| C |  |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | | E | |  | |  | |  | |  | |  | |  | |  | |  | |  | | D | |  | |  | |  | |  | |  | |  | |  | |  | |  | | E | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |
| D |  |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | | E | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |
| E |  |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |  | |

This task set is not RM-schedulable. Task A misses its deadline at time = 20. Schedulability test 3 shows this because:

* cB + cE + cC + cA + cD = 1 + 3 + 4 + 6 + 1 = 15 > 10 = pB = pE.
* 2cB + 2cE + cC + cA + cD = 2 + 6 + 4 + 6 + 1 = 19 > 15 = pC.
* 2cB + 2cE + 2cC + cA + cD = 2 + 6 + 8 + 6 + 1 = 23 > 20 = pA.
* 3cB + 3cE + 2cC + 2cA + cD = 3 + 9 + 8 + 12 + 1 = 33 > 30 = pD.

1. **Scheduling with precedence constraints**

Revise the deadlines to integrate information about precedence.

* Map out scheduling blocks over a timespan = LCM of periods of all tasks.

LCM(42, 42, 14) = 42

* Assign an absolute deadline to each block.

d1,1 = 42; d1,2 = 42; d1,3 = 42; d2,1 = 12; d2,2 = 12; d2,3 = 26; d2,4 = 26; d2,5 = 40; d2,6 = 40; d3,1 = 42

* Topologically order the blocks and then reverse it.
  + Draw a precedence graph.

1

2

3

1

2

3

6

4

5

1

T1

T2

T3

* + A forward topological could be: (T3,1 T2,­1 T1,­1 T2,­2 T1,­2 T2,­3 T1,­3 T2,­4 T2,­5 T2,­6)
  + Reversing that = (T2,­6 T2,­5 T2,­4 T1,­3 T2,­3 T1,­2 T2,­2 T1,­1 T2,­1 T3,1)
* Work through the reverse topological order and adjust deadlines for blocks with outgoing edges in the precedence graph.
  + T2,­6 has no outgoing edges. d’2,6 = d2,6 = **40**.
  + T2,­5 has an edge to T2,­6. d’2,5 = min(d2,5, d’2,6 – c2,6) = min(40, 40 – 4) = min(40, 36) = **36**.
  + T2,­4 has an edge to T2,­5. d’2,4 = min(d2,4, d’2,5 – c2,5) = min(26, 36 – 2) = min(26, 34) = **26**.
  + T1,­3 has an edge to T2,­6. d’1,3 = min(d1,3, d’2,6 – c2,6) = min(42, 36 – 4) = min(42, 32) = **32**.
  + T2,­3 has edges to T1,­3 and T2,4. d’2,3 = min(d2,3, d’2,4 – c2,4, d’1,3 – c1,3)   
    = min(26, 26 – 4, 42 – 6) = min(26, 22, 36) = **22**.
  + T1,­2 has edges to T1,­3 and T2,4. d’1,2 = min(d1,2, d’1,3 – c1,3, d’2,4 – c2,4)   
    = min(42, 32 – 6, 26 – 4) = min(42, 26, 22) = **22**.
  + T2,­2 has an edge to T2,­3. d’2,2 = min(d2,2, d’2,3 – c2,3) = min(12, 26 – 2) = min(12, 24) = **12**.
  + T1,­1 has edges to T1,2 and T2,2. d’1,1 = min(d1,1, d’1,2 – c1,2, d’2,2 – c2,2)   
    = min(42, 22 – 6, 12 – 4) = min(42, 16, 8) = **8**.
  + T2,­1 has edges to T1,2 and T2,2. d’2,1 = min(d2,1, d’1,2 – c1,2, d’2,2 – c2,2)   
    = min(12, 22 – 6, 12 – 4) = min(12, 16, 8) = **8**.
  + T3,­1 has no outgoing edges. d’3,1 = d3,1 = **42**.
* We can schedule the adjusted task blocks with EDF. Original arrival times are respected.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **t** | 0 |  |  |  | 4 |  | 6 |  |  |  | 10 |  |  |  |  |  | | 16 |  | | 18 |  |  |  | | 22 |  |  |  |  |  | | 28 |  | | 30 |  |  |  | | 34 |  |  |  |  |  |  |  | 42 | |  | |
| **J** | T1,1 | | | | T2,1 | | T2,2 | | | | T1,2 | | | | | | T2,3 | | | T2,4 | | | | | T1,3 | | | | | | | T2,5 | | | T2,6 | | | | | T3,1 | | | | | | | | |  |  | |  |

1. **Task set density**

The density of the task system is defined as the sum of the densities of all the tasks in the system. If there is only one task, the system is trivially schedulable, since the single task may start as soon as it arrives, and it is an absurdity to have a task whose deadline cannot be met even if execution is immediate and uninterrupted. Say we have a system S­­ of n – 1 many tasks with density D <= 1 which is schedulable. Consider a task Jn with density Dn such that D + Dn is still <= 1. This implies D <= 1 - Dn. This means that there is an average of >= cn units of time not used by tasks in S over each window of width dn within the hyperperiod of S. Since the tasks are independent and preemptable, move their execution blocks so that at least cn units of idle time are placed between each arrival time of an instance of Jn and the instance’s deadline, even if the idle time is not contiguous. Then this idle time can be filled by Jn, and it will meet its deadline. So, the new system S’ = S + Jn is still schedulable.

1. **RM and EDF schedulability tests**

Since all 3 systems have 3 tasks, the expression from Schedulability Test 2 = 3 \* (21/3 – 1) = 0.78

System (a):   
U = (4 / 10) + (4 / 16) + (4 / 20) = 0.4 + 0.25 + 0.2 = 0.85  
Yes, system (a) is EDF schedulable by Schedulability Test 4.

U > 0.78, so we need to try Schedulability Test 3 to check if the system is RM schedulable.

* J1 is RM schedulable because c1 = 4 <= 10 = p1.
* J2 is RM schedulable because c1 + c2 = 8 <= 10 = p1.
* J3 is RM schedulable because 2\*c1 + c2 + c3 = 16 <= 16 = p2.

So yes, system (a) is RM schedulable.

System (b):  
U = (2 / 5) + (3 / 10) + (3 / 12) = 0.4 + 0.3 + 0.25 = 0.95  
Yes, system (b) is EDF schedulable by Schedulability Test 4.

* J1 is RM schedulable because c1 = 2 <= 5 = p1.
* J2 is RM schedulable because c1 + c2 = 5 <= 5 = p1.
* J3 is RM schedulable because 2\*c1 + c2 + c3 = 10 <= 10 = p2.

So yes, system (b) is RM schedulable.

System (c):  
U = (3 / 7) + (5 / 9) + (3 / 15) = 0.43 + 0.45 + 0.2 = 1.08  
No, system (c) is not EDF schedulable, by Schedulability Test 4.

System (c) is not RM schedulable either, since EDF is an optimal scheduler.

1. **Constructing task set**

This problem is equivalent to creating a task set that is RM schedulable, since EDF and LLF are optimal and can schedule any task set that is schedulable by another method.

J1: c = 2; p = 4  
J2: c = 2; p = 6  
J3: c = 2; p = 12

3\*c1 + 2\*c2 + c3 = 6 + 4 + 2 = 12 <= 12 = p3 so the task set is RM schedulable.

1. **Non-simple schedulable utilization (RM schedulability test 3)**

The task set shown in problem (8) does not pass schedulability test 2 because its utilization is (2/4) + (2/6) + (2/12) = 0.50 + 0.33 + 0.17 = 1.00, which is greater than the utilization for a set of 3 tasks = 0.78 (already calculated in problem (7)). But it is RM schedulable, as shown in problem (8).

1. **Multiprocessor scheduling**

The problem of finding a set of n4 tasks grows quickly as n increases. Any set of this many tasks needs to have very widely spaced periods in order to be schedulable. According to the utilization formula in schedulability test 10, the period of each task Ji­ should be at least ci \* n3 in order for the sum of all ratios ci/pi to be <= n. For example, with n = only 3, the minimum period of any task with c = 1 would have to be 33 = 27, while with n = 4, this minimum would be 43 = 64, etc. The minimum period for tasks with longer computation times would be even greater. Any tasks with periods narrower than this minimum for a given n would make the task set non-schedulable. So it is possible, but not very likely when n is not trivial.

1. **RM and EDF equivalence**

The RM and EDF algorithms are equivalent when the tasks with the narrowest periods also have the earliest deadlines every cycle. This can occur when all tasks have deadlines = periods and each period is an integer multiple of all lesser periods of tasks in the set. However, since the EDF algorithm breaks ties for earliest deadline using an arbitrary strategy, a task with a wider period could be chosen in favor of the task with the narrowest period to break the tie during the last cycle before a new instance of the task with the wider period arrives, or the latest instance of the task with a narrow period could fail to preempt the running instance of a task with a wider period. So the tie-breaking strategy of the EDF algorithm would also have to be based on least period first in order for the algorithms to be equivalent all the time.

1. **Is EDF ever not optimal?**

EDF is always optimal on a uniprocessor system, but there are cases where EDF is not optimal on a multiprocessor system. This frequently happens when there is a task with a late deadline but also a very high computation time, resulting in a low value of slack/laxity.

For example, say a system has 3 processors. There are 4 tasks ready, including J1 = {d1 = 8, c1 = 8}, J2 = {4, 1}, J3 = {4, 3} and J4 = {5, 4}. If we use EDF to schedule, then the tasks J2 through J4 will be scheduled first across the 3 processors. Task J2 will finish first, at time = 1, leaving room on processor 1 for task J1 to begin executing. But J1 will miss its deadline if it waits for even one time unit before it executes, because it has 8 computation units to complete before time = 8. This means that the task set is not schedulable with EDF on this system.

In contrast, using LLF will recognize that task J1 has laxity = 0, so that task will be scheduled at time = 0 along with tasks J3 (L = 1) and J4 (L = 1). When J3 completes early at time = 3 on processor 2, there is just enough time to execute task J2 before its deadline. Meanwhile, the other processors complete their tasks right before their deadlines. So the task set is schedulable with LLF. This means that EDF is not optimal, since another scheduler could schedule a task set that EDF failed to schedule.

1. **RM schedulability test 3**

Let’s test the set of tasks using schedulability test 2 first. The desired utilization is n \* (21/n - 1) = 6 \* (21/6 - 1) = 0.735; the utilization of the system is (5/50) + (11/70) + (14/100) + (15/120) + (26/140) + (60/300) = 0.908 so test 2 does not work. We need to try test 3 to check if the set is RM schedulable.

S1 = T1 is trivially schedulable.

S2 = S1 + T2 is schedulable because c1 + c2 = 5 + 11 = 16 <= 50 = p1.

S3 = S2 + T3 is schedulable because c1 + c2 + c3 = 5 + 11 + 14 = 30 <= 50 = p1.

S4 = S3 + T4 is schedulable because c1 + c2 + c3 + c4 = 5 + 11 + 14 + 15 = 45 <= 50 = p1.

S5 = S4 + T5 is schedulable because 2c1 + 2c2 + c3 + c4 + c5 = 10 + 22 + 14 + 15 + 26 = 87 <= 100 = p3.

S6 = S5 + T6 is **not** schedulable because:

* c1 + c2 + c3 + c4 + c5 + c6 = 5 + 11 + 14 + 15 + 26 + 60 = 131 > 50 = p1.
* 2c1 + c2 + c3 + c4 + c5 + c6 = 10 + 11 + 14 + 15 + 26 + 60 = 136 > 70 = p2.
* 2c1 + 2c2 + c3 + c4 + c5 + c6 = 10 + 22 + 14 + 15 + 26 + 60 = 147 > 100 = p3.
* 3c1 + 2c2 + 2c3 + c4 + c5 + c6 = 15 + 22 + 28 + 15 + 26 + 60 = 166 > 120 = p4.
* 3c1 + 2c2 + 2c3 + 2c4 + 2c5 + c6 = 15 + 22 + 28 + 30 + 52 + 60 = 207 > 140 = p5.
* 6c1 + 5c2 + 3c3 + 3c4 + 3c5 + c6 = 30 + 55 + 42 + 45 + 78 + 60 = 310 > 300 = p6.

So no, the system with all 6 tasks is not RM schedulable.