Priority Exchange (PE) servers

Alberto Tessari & Sebastiano Scodro

June 10

Recap: Polling servers (PS)

- PS recharged at its full value C_s at the beginning of each period
- if there are no aperiodic tasks pending PS suspends itself until the next period (C_s goes to 0)
- if there are aperiodic tasks pending, PS serves them until $C_s > 0$
- In the worst case $U_s = C_s / T_s$
- Schedulability is guaranteed if $U_p \leq n \left[\left(\frac{2}{U_s + 1} \right)^{\frac{1}{n}} - 1 \right]$ where periodic tasks are scheduled with Rate Monotonic (RM) algorithm

Recap: Deferrable servers (DS)

- Server capacity is preserved during the period and replenished at its full value at the beginning of the next period
- Aperiodic requests can be served at any time as long as C_s has not been exhausted
- Schedulability is guaranteed if $U_p \leq n \left[\left(\frac{U_s+2}{2U_s+1} \right)^{\frac{1}{n}} - 1 \right]$ where periodic tasks are scheduled with Rate Monotonic (RM) algorithm
- Responsiveness is improved (compared to PS) but utilization bound is decreased (as it is more "invasive" for the periodic tasks)

Priority exchange (PE)

- Like DS, the only difference is the way the capacity is preserved
- C_s is replenished at its full value at the beginning of each period
- if there are aperiodic tasks pending and the server is the task with the highest priority then the requests are served at that highest priority until C_s is exhausted, otherwise C_s is exchanged for the execution time of the active periodic task with the highest priority

Exchanging priority

- When a priority exchange occurs between a periodic task and the server, the server accumulates C_s at the priority of the periodic task (so it is preserved, not lost)
- priority exchange continues with lower priority periodic tasks until either the capacity is used for serving aperiodic requests or it is degraded to the priority level of background processing.
- the objective is to provide high responsiveness to aperiodic requests: all priority ties are broken in favor of aperiodic tasks

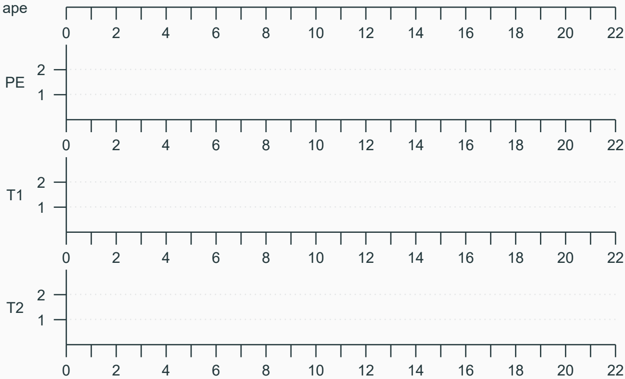
PE server example 1

	C_i	T_i
$T1$	4	10
$T2$	8	20

Ap. req.	C_i	a_i
$ape1$	1	5
$ape2$	1	12

Server
$C_s = 1$
$T_s = 5$

$P(S) > P(T1) > P(T2)$



PE server example 2

	C_i	T_i
$T1$	2	10
$T2$	12	20

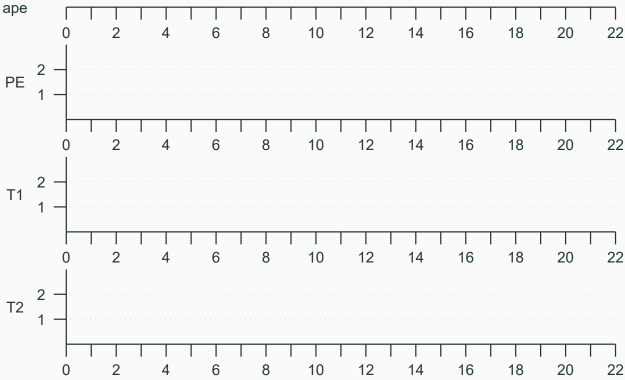
Ap. req.	C_i	a_i
$ape1$	2	11
$ape2$	1	18

Server

$C_s = 1$

$T_s = 5$

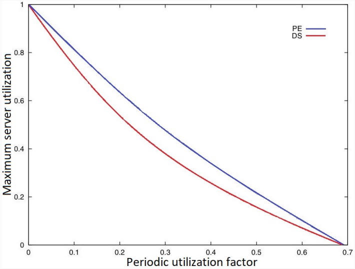
$P(S) > P(T1) > P(T2)$



Priority Exchange server analysis

- Like in PS, schedulability is guaranteed if $U_p \leq n \left[\left(\frac{2}{U_s+1} \right)^{\frac{1}{n}} - 1 \right]$ where periodic tasks are scheduled with Rate Monotonic (RM) algorithm
- Compared to PS, PE is more responsive
- Compared to DS, PE is slightly less responsive (to aperiodic tasks) but provides a better schedulability bound for the periodic tasks

Strenghts and weaknesses of PE



- DS is much easier to implement as there are no exchanges of priorities
- PE increases the overhead (due to the additional work required to manage priorities)
- DS pays with a lower utilization bound its simplicity (meaning that, for a given periodic load U_p, the maximum size of a DS server that can guarantee the periodic tasks is smaller than the maximum size of a PE server).

Dynamic Priority Exchange servers (DPE)

- The scheduling algorithm for the periodic tasks in now EDF: this means that the priority of the server capacity exchanged when periodic tasks are being executed follows the same rules
- as for the static variant, ties of priority are broken in favor of the aperiodic tasks
- periodic task set guaranteed for $U_s + U_p \leq 1$

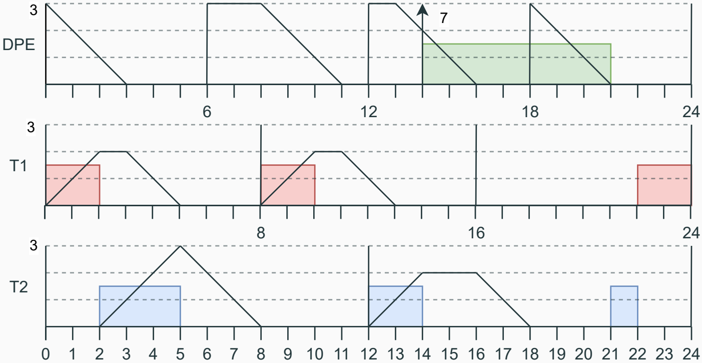
DPE example 1

	C_i	T_i
$T1$	2	8
$T2$	3	12

Server

$C_s = 3$

$T_s = 6$


DPE
T1
T2