Resource Reservation Servers

Prof. Dr. Jian-Jia Chen

LS 12, TU Dortmund

17 May 2016

Sporadic Tasks and Aperiodic Tasks

CPU is a shared resource for hard real-time tasks and soft real-time tasks

- For hard real-time tasks:
  - The schedulability has to be assured.
- For soft real-time tasks:
  - The performance (e.g., average response time) should be optimized.

Approaches

- Resource reservation servers
  - Guarantee the maximal interference to the periodic/sporadic tasks.
  - Ensure the minimal progress inside the servers.

Well-Known Protocols

- Servers for fixed-priority systems
  - Polling Server (PS): provide a fixed execution budget that is only available at pre-defined times.
  - Deferrable Server (DS): provide a fixed budget, in which the budget replenishment is done periodically.
  - Sporadic Server (SS): provide a fixed budget, in which the budget replenishment is performed only if it was consumed.
  - Others (not included): priority exchange (PE) server, slack stealer, etc.
- Servers for dynamic-priority systems
  - Total bandwidth server (TBS): provide a fixed utilization for executing jobs, in which the deadline for execution is *dependent* on the execution time of jobs.
  - Constant bandwidth server (CBS): provide a fixed utilization for executing jobs, in which the deadline for execution is *independent* on the execution time of jobs.
  - Others (not included): dynamic priority exchange (DPE) server, dynamic slack stealer, dynamic sporadic server, etc.

Outline

Servers for fixed-priority systems

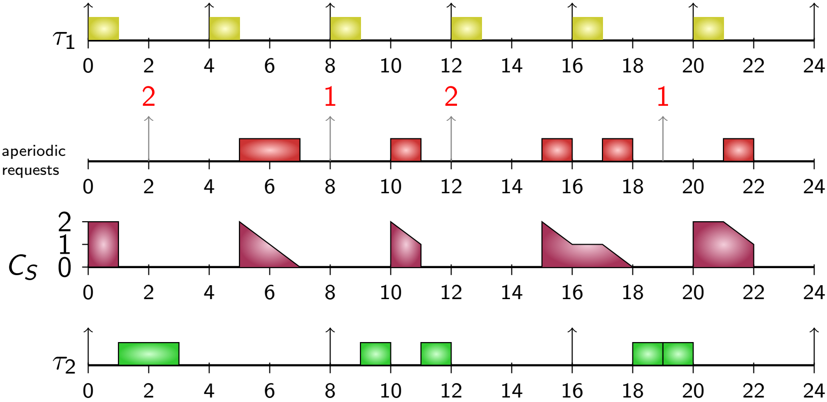
Servers for dynamic-priority systems

Polling Server (PS)

- Behavior: periodic task with specified priority
  - period: T_{S_i}
  - capacity (computation time): C_{S_i}
- Consumption rule:
  - upon activation of the task, the server starts executing events until either no request is in the ready queue of the server or the capacity C_{S_i} is exhausted.

An Example of PS

$\tau_1 = (1, 4, 4)$, $\tau_2 = (2, 8, 8)$.
Polling server: $C_S = 2$ and $T_S = 5$.
Priority: $\tau_1 > PS > \tau_2$.



Properties of PS

- Schedulability Guarantee:
  - Suppose that there are n periodic tasks, a polling server with utilization $U_S = \frac{C_S}{T_S}$ and that the RM scheduling algorithm is adopted.
  - The schedulability of the periodic task set is guaranteed if

$$U_S + \sum_{i=1}^{n} \frac{C_i}{T_i} \leq U_{lub}(RM, n+1) = (n+1)(2^{\frac{1}{n+1}} - 1).$$

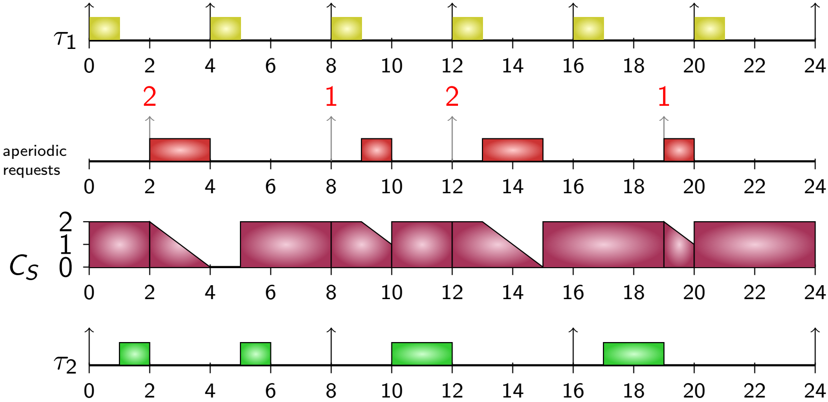
  - The proof can be done by imaging that a periodic task represents the polling server, which is executed for at most C_i time units after the right to execute is granted.
  - Since the capacity is greedily set to 0 if there is no request for the polling server to be executed, the periodic task that represents the polling server can be imaged as early completion of the task instead of task self-suspension.

Deferrable Server (DS)

- Behavior: periodic task with a specified priority
  - period: T_{S_i}
  - capacity (computation time): C_{S_i}
- Replenishment rule:
  - upon activation of the task (periodic replenishment at the multiple of T_{S_i})
- Consumption rule:
  - aperiodic requests are served when the server still has capacity
  - capacity is lost at the end of the period

An Example of DS

$\tau_1 = (1, 4, 4)$, $\tau_2 = (2, 8, 8)$.
Deferrable server: $C_S = 2$ and $T_S = 5$.
Priority: $\tau_1 > DS > \tau_2$.



Schedulability Guarantee of DS

- Suppose that there are n periodic tasks, a deferrable server with utilization $U_S = \frac{C_S}{T_S}$ and that the RM scheduling algorithm is adopted.
- RM analysis is incorrect for such a case, since the periodic task that represents the deferrable server can be imaged as using self-suspension of the task (e.g., time 1 in the example).

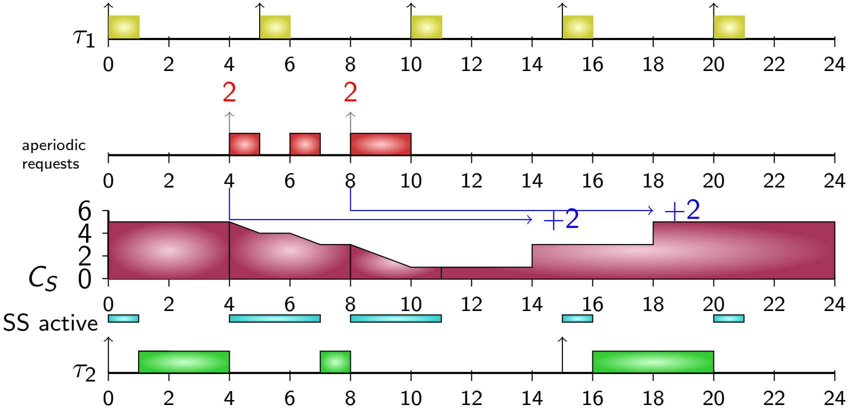
- The analysis in the textbook by Buttazzo is not complete.
- We will discuss this next week.

Sporadic Server (SS)

- Behavior: sporadic task with a specified priority
  - period: T_{S_i}
  - capacity (computation time): C_{S_i}
- Rules:
  - Let $\pi_{exe}(t)$ be the priority level that is executing at time t
  - The server is Active when the $\pi_{exe}(t)$ has no lower priority than SS.
  - The server is Idle when the $\pi_{exe}(t)$ has lower priority than SS.
  - Initially, the server is Idle and its budget is C_{S_i}. When the server becomes Active at time t_1, the replenishment time is set to $t_1 + T_{S_i}$
  - When the server becomes Idle at time t_2, the (next) replenishment amount is set to the amount of capacity consumed in the time interval between the last replenishment time and t_2

An Example of SS

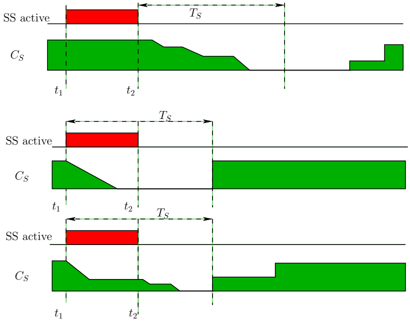
$\tau_1 = (1, 5, 5)$, $\tau_2 = (4, 15, 15)$.
Sporadic server: $C_S = 5$ and $T_S = 10$.
Priority: $\tau_1 > SS > \tau_2$.



Schedulability Guarantees of SS

Theorem

If a periodic task set is schedulable, replacing a task τ_i by a sporadic server SS_i with the same period and execution time is still schedulable.



Case 1: Imagine that the arrival of a periodic task is delayed to arrive at time t_2.

Case 2: Imagine the behavior like a periodic real-time task

Case 3: Imagine that there are multiple tasks with the same period but with different arrival times, and their total execution time is the same as that of τ_i.

Overview of PS, DS, and SS

	Performance	computation	memory	implementation complexity
PS	poor	excellent	excellent	excellent
DS	good	excellent	excellent	excellent
SS	excellent	good	good	good

Outline

Servers for fixed-priority systems

Servers for dynamic-priority systems

Total Bandwidth Server (TBS)

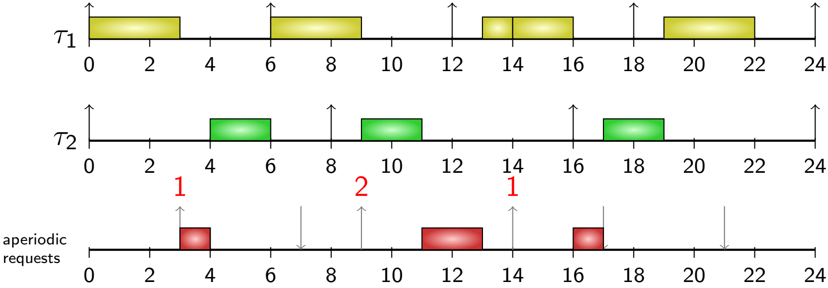
- Behavior: assign the absolute deadline of an incoming job such that the utilization for this server is at most U_{S_i}, which is the only parameter required for a TBS server S_i.
- Initialization: assign server deadline D_{S_i} to $-\infty$.
- Deadline assignment rule: when a job arrives at time t
  - The absolute deadline of this job is set to

    $$\max\{t, D_{S_i}\} + \frac{C_j}{U_{S_i}},$$

    where C_j is the required (maximum) computation time of the job and D_{S_i} is the server deadline.
  - The server deadline D_{S_i} is also updated to the above absolute deadline.

An Example of TBS

$\tau_1 = (3, 6, 6)$, $\tau_2 = (2, 8, 8)$.
TBS: $U_S = 0.25$



At time $t = 0$, $D_{S_i} = -\infty$
At time $t = 3$, $D_{S_i} = \max\{t, D_{S_i}\} + \frac{1}{U_S} = 3 + 4 = 7$
At time $t = 9$, $D_{S_i} = \max\{t, D_{S_i}\} + \frac{2}{U_S} = 9 + 8 = 17$
At time $t = 14$, $D_{S_i} = \max\{t, D_{S_i}\} + \frac{1}{U_S} = 17 + 4 = 21$

Properties of TBS

- Schedulability Guarantee:
  - Suppose that there are n periodic tasks and a total bandwidth server with utilization U_S.
  - The schedulability of the whole task set is guaranteed if

$$U_S + \sum_{i=1}^{n} \frac{C_i}{T_i} \leq 1.$$

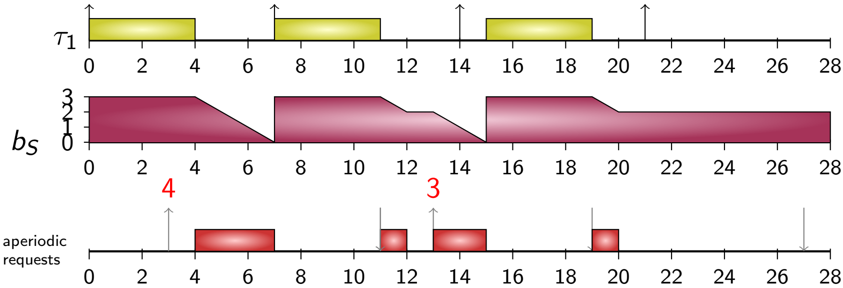
Again, proven by contradiction.
The key point is that the total execution time demanded by an aperiodic requests that arrived at time t_1 or later and served with a deadline less than or equal to t_2 is no more than $(t_2 - t_1)U_S$.

Constant Bandwidth Server (CBS)

- Behavior: assign the absolute deadline of an incoming job such that the utilization for CBS server S_i is at most $U_{S_i} = C_{S_i}/T_{S_i}$.
  - Capacity: C_{S_i}
  - Period: T_{S_i}
- Initialization: assign server deadline D_{S_i} to $-\infty$ and budget to 0.
- Definition: The server is active at time t if there are pending jobs; otherwise it is idle.
- Deadline assignment rule:
  - When the server is idle at time t and a job arrives, if $t < D_{S_i}$ and $\frac{b_{S_i}}{D_{S_i}-t} < \frac{C_{S_i}}{T_{S_i}}$, the server becomes active with the same budget and server deadline; otherwise, D_{S_i} is set to $t + T_{S_i}$ and b_{S_i} is set to C_{S_i}.
  - The first job, if exists, in the ready queue of server S_i is assigned to the current server deadline D_{S_i}.
  - The budget b_{S_i} is decreased by the served execution (computation) time while server is executing jobs
  - When b_{S_i} reaches 0, the new server deadline D_{S_i} becomes $D_{S_i} + T_{S_i}$ and b_{S_i} is replenished to C_{S_i} immediately.

An Example of CBS

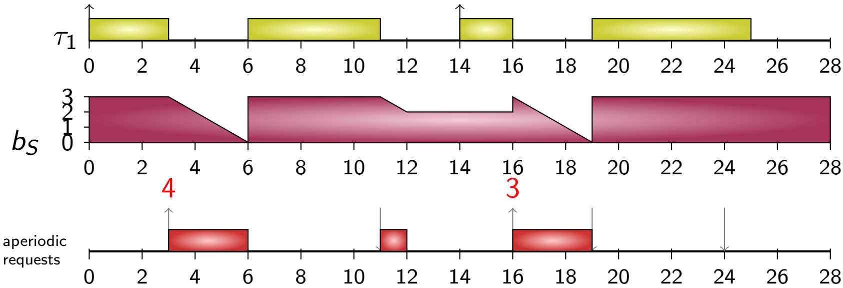
$\tau_1 = (4, 7, 7)$
CBS: $C_S = 3,\ T_S = 8$



At time $t = 0$, $D_{S_i} = -\infty$. At time $t = 3$, $D_{S_i} = 11$ and $b_{S_i} = 3$
At time $t = 7$, b_{S_i} becomes 0 and immediately is replenished to 3 by setting $D_{S_i} = 19$.
At time $t = 13$, since $\frac{b_{S_i}}{19-13} < \frac{3}{8}$, server becomes active with the same deadline/budget.
At time $t = 15$, b_{S_i} becomes 0 and immediately is replenished to 3 by setting $D_{S_i} = 27$.

Another Example of CBS

$\tau_1 = (8, 14, 14)$
CBS: $C_S = 3,\ T_S = 8$



At time $t = 0$, $D_{S_i} = -\infty$. At time $t = 3$, $D_{S_i} = 11$ and $b_{S_i} = 3$
At time $t = 6$, b_{S_i} becomes 0 and immediately is replenished to 3 by setting $D_{S_i} = 19$.

At time $t = 16$, since $\frac{b_{S_i}}{19-16} > \frac{3}{8}$, server becomes active by setting $b_{S_i} = 3$ and $D_{S_i} = 16 + 8 = 24$.

Properties of CBS

- Schedulability Guarantee:
  - Suppose that there are n periodic tasks and a constant bandwidth server with utilization U_S.
  - The schedulability of whole task set is guaranteed if

$$U_S + \sum_{i=1}^{n} \frac{C_i}{T_i} \leq 1.$$

The key point is the isolation property, in which

- the CPU utilization of a CBS server is U_S, independently from the computation times and the arrival pattern of the served jobs.