Scheduling Sporadic Jobs with EDF

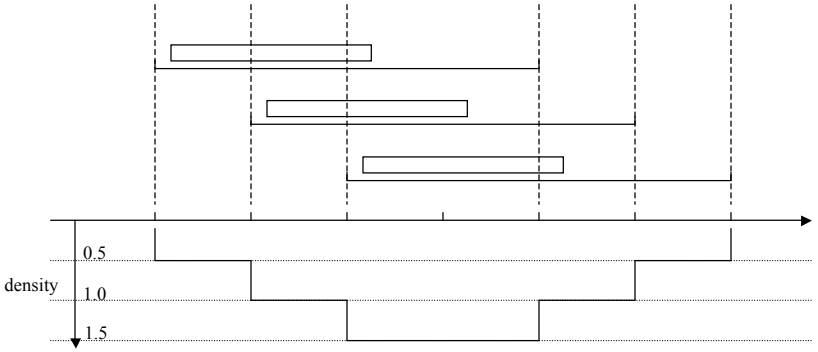
- **Definition:** *Density* of sporadic job J_i with release time r_i, maximum execution time e_i and deadline d_i:

$$density_i = e_i/(d_i - r_i).$$

- **Theorem:** A system of independent, preemptable sporadic jobs is schedulable according to EDF if the total density of all active jobs in the system is no greater than 1 at all times.

Scheduling Sporadic Jobs with EDF (cont)

- Theorem is not necessary!
- Example:



Scheduling Sporadic Jobs with EDF (cont)

- Sporadic task S_i as stream of sporadic jobs S_{i1}, S_{i2}, S_{i3}, ...
- Execution time of S_{ij} is e_{ij}.
- "Period" p_{ij} is time between invocation of S_{ij} and $S_{i(j+1)}$.

- Instantaneous utilization of sporadic job S_{ij} : e_{ij}/p_{ij}.
- Instantaneous utilization of sporadic task S_i: $u_i = max_j(e_{ij}/p_{ij})$.

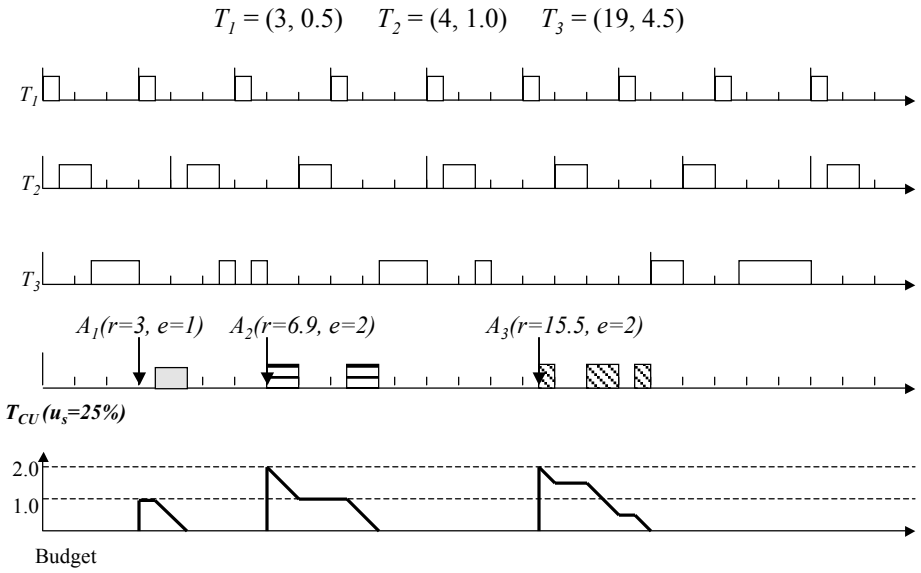
- **Corollary:** A system of n independent, preemptable sporadic tasks, which is such that the relative deadline of every job is equal to its period, is schedulable on a processor according to the EDF algorithm if the total instantaneous utilization is equal or less to 1.

Constant Utilization Server Algorithm

- A constant utilization server emulates a sporadic task with a constant instantaneous utilization.

- Consumption rule:
  - A server consumes its budget only when it executes.
- Replenishment rules (assume: server is allocated utilization u_s):
  - **R1** Initially, set e_s := 0 and d :=0.
  - **R2** When an aperiodic job with execution time e arrives at time t to an empty aperiodic job queue,
    (a) if $t < d$, do nothing.
    (b) if $t >= d$, set $e_s := e$, and $d := t + e_s/u_s$.
  - **R3** At the deadline d of the server,
    (a) if the server is backlogged, set $e_s := e$ and $d := d + e/u_s$
    (b) if the server is idle, do nothing.

Constant Utilization Server: Example

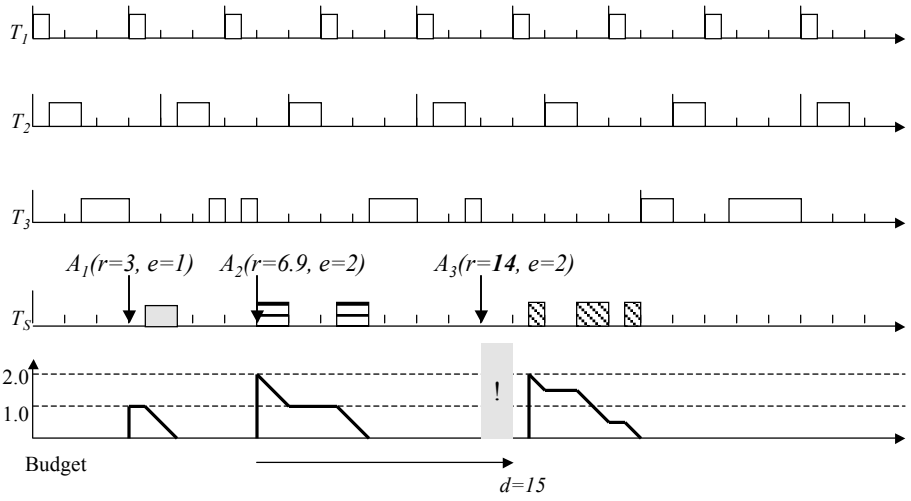


What about Unknown Execution Times?

- Assumption for constant utilization server: execution times of aperiodic jobs are known upon arrival.
  ⇒ Restrictive.
- Possible solution: Assign fixed bandwidth to server:
  - fixed budget e_s
  - fixed period e_s/u_s
- Upon job completion of job with execution time $e < e_s$, reduce current deadline of server by $(e_s-e)/u_s$ before replenishing again.
- For execution time $e > e_s$, use more than one server period.

Problems with CUS: Unused Capacity

$T_1 = (3, 0.5)$ $T_2 = (4, 1.0)$ $T_3 = (19, 4.5)$



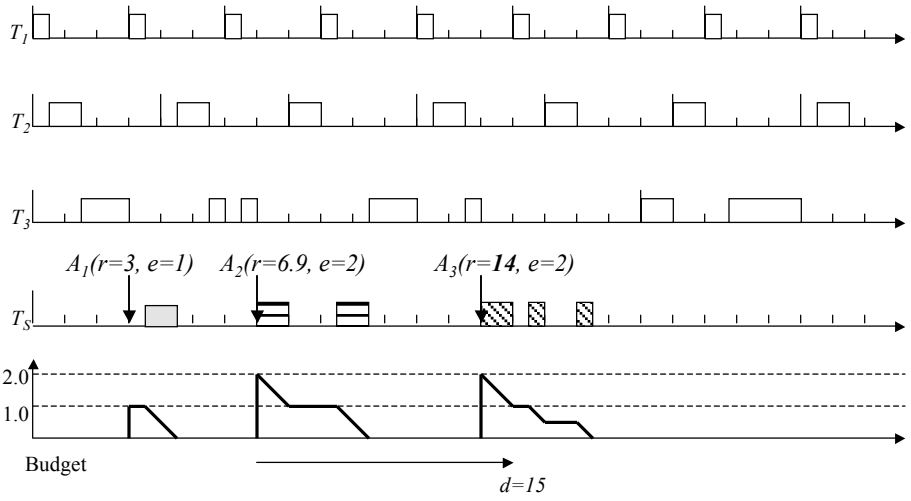
Total Bandwidth Server

- Allow server to use background time.

- Consumption rule:
  - A server consumes its budget only when it executes.
- Replenishment rules:
  - **R1** Initially, set $e_s := 0$ and $d := 0$.
  - **R2** When an aperiodic job with execution time e arrives at time t to an empty aperiodic job queue, set $d := max(d,t) + e_s/u_s$, and $e_s := e$.
  - **R3** Upon completion of the current aperiodic job, remove job from queue.
    (a) if the server is backlogged, set $d := d + e/u_s$ and $e_s := e$;
    (b) if the server is idle, do nothing.

Unused Capacity Eliminated with TBS

$T_1 = (3, 0.5)$ $T_2 = (4, 1.0)$ $T_3 = (19, 4.5)$



Correctness of Total Bandwidth Server

- Constant Utilization Server is correct.
- How does Total Bandwidth Server affect periodic tasks differently?
- Only interesting case:
  - Budget of Total Bandwidth Server replenished at time *t* before its deadline.
  - New deadline is $d' = d + e/u_s$.
- How does this affect the execution of periodic tasks?
  - Case 1: Current periodic job $J_{i,c}$ has deadline before $d' \Rightarrow$ execution of periodic job is not affected.
  - Case 2: Current periodic job $J_{i,c}$ has deadline after d'
    - Case 2.1: Current periodic job $J_{i,c}$ is ready before time $t \Rightarrow$ execution time demanded by Total Bandwidth Server from $r_{i,k}$ to d' is same as for Constant Utilization Server.
    - Case 2.2: Current periodic job $J_{i,c}$ is ready after time $t \Rightarrow$ execution time demanded by Total Bandwidth Server from $r_{i,k}$ to d' is less than that of Constant Utilization Server.