Constant Utilization Server

- Consumption rule:
  - A constant utilization server only consumes budget when it executes
- Replenishment rules:
  - Initially, budget $e_s = 0$ and deadline $d = 0$
  - When an aperiodic job with execution time e arrives at time t to an empty aperiodic job queue
    - If $t < d$, do nothing ($\Rightarrow$ server is busy; wait for it to become idle)
    - If $t \geq d$ then set $d = t + e/\tilde{u}_s$ and $e_s = e$
  - At the deadline d of the server
    - If the server is backlogged, set $d = d + e/\tilde{u}_s$ and $e_s = e$
      $\Rightarrow$ was busy when job arrived
    - If the server is idle, do nothing

i.e. the server is always given enough budget to complete the job at the head of its queue, with known utilization, when the budget is replenished

Total Bandwidth Server

- A constant utilization server gives a known fraction of processor capacity to a task; but cannot claim unused capacity to complete the task earlier
- A *total bandwidth* server improves responsiveness by allowing a server to claim background time not used by the periodic tasks
  - Change the replenishment rules slightly, leave all else the same:
    - Initially, $e_s = 0$ and $d = 0$
    - When an aperiodic job with execution time e arrives at time t to an empty aperiodic job queue
      - Set $d = \max(d, t) + e/\tilde{u}_s$ and $e_s = e$
    - When the server completes the current aperiodic job, the job is removed from the queue and
      - If the server is backlogged, set $d = d + e/\tilde{u}_s$ and $e_s = e$
      - If the server is idle, do nothing
  - Always ready for execution when backlogged
  - Assigns at least fraction $\tilde{u}_s$ of the processor to a task

Weighted Fair Queuing Server

- Aim of the constant utilization and total bandwidth servers is to assign some fraction of processor capacity to a task
- When assigning capacity there is the issue of *fairness*:
  - A scheduling algorithm is *fair* within any particular time interval if the fraction of processor time in the interval attained by each backlogged server is proportional to the server size
    - Not only do all tasks meet their deadline, but they all make continual progress according to their share of the processor, no *starvation*
  - Constant utilization and total bandwidth servers are fair on the long term, but can diverge significantly from fair shares in the short term
    - Total bandwidth server partly by design, since it uses background time, but also has fairness issues when there is no spare background time

- As we discuss in lecture 16, the *weighted fair queuing* algorithm can also be used to share processor time between servers, and is designed to ensure fairness in allocations