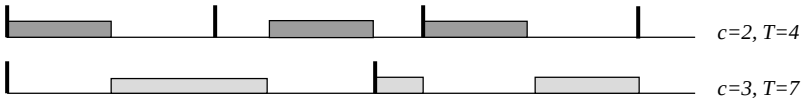
Scheduling Periodic tasks

Can apply EDF scheduling

- Example 2 tasks: (2,4) and (3,7)

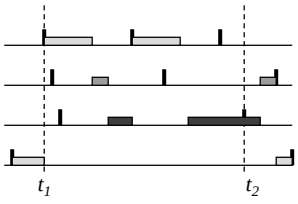


- Dynamic priority scheduling (high priority to task with earlier deadline)
- Note that we can look at each periodic task as a series of a-periodic tasks
- The Utilization of the task set $\{J_i , i=1,\ldots,n\}$ is defined by $U = \sum_{i=1}^{n} \frac{c_i}{T_i}$
- If deadlines are not equal to the periods, the necessary condition for the feasibility of EDF is an open problem.
- Which job misses its deadline in case $U > 1$ is unpredictable.

With EDF, all tasks will meet their deadlines if and only if $U \leq 1$

Proof:

- Assume that the overflow occurs at t_2 ,
- Let t_1 be the latest time before t_2 such that
  - the processor is fully utilized in $[t_1 , t_2]$
  - only instance with deadlines before t_2 executes in $[t_1 , t_2]$
- If cannot find such a t_1 , then set $t_1 = 0$.
- Let C_d be the computational demand in $[t_1 , t_2]$

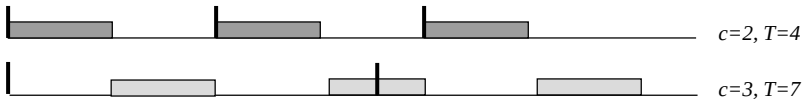


$$C_d = \sum_{r_i \geq t_1, d_i \leq t_2} \left\lfloor \frac{t_2 - t_1}{T_i} \right\rfloor c_i \leq \sum_{i=1,n} \frac{t_2 - t_1}{T_i} c_i = (t_2 - t_1)U$$

- But an overflow implies that $C_d > (t_2 - t_1)$ -- a contradiction if $U \leq 1$.

Rate monotonic Scheduling, RMS (fixed priority)

- Higher priority is given to tasks with shorter periods.



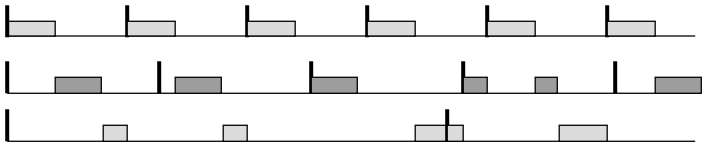
- If $c_2 = 3.1$, then c_2 will miss its deadline although the utilization is $\frac{2}{4} + \frac{3.1}{7}$, which is less than *1*.

Liu and Layland proved that RMS leads to a feasible schedule if
$U \leq n\,(2^{1/n} - 1)$

- The above bound is sufficient but not necessary
- $n\,(2^{1/n} - 1) = 0.828$, when $n = 2$, and $= ln\ 2 = 0.69$, when n is very large.
- RMS is an optimum fixed priority assignment algorithm (if RMS fails, all fixed priority scheduling fail).

Derivation of the Liu and Layland "RMS" bound.

(I) If a task set is feasible when the phases are aligned, then the same task set is feasible if the phases are not aligned (the critical instant is when all tasks are aligned). Hence, a task set is feasible if the first instant of each task finishes by its deadline.



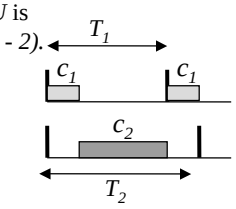
(II) For n tasks, find U_{lub} such that if a task set has a utilization less than or equal to U_{lub} , then it can be feasibly scheduled by RMS. Assume that the ratio of the largest to the smallest period (T_n / T_1) is no more than 2.

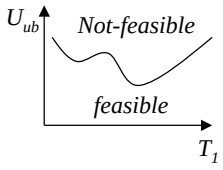
(III) generalize the result for a arbitrary T_n / T_1 ratio.

Proof of (II) for sets of two tasks ($n=2$).

Among all the feasibly scheduled task sets $\{J_1, J_2\}$, we want to find for any given combination of periods, the maximum feasible utilization U_{ub}.
Find, U_{lub}, the minimum value of U_{ub} for all possible combinations of T_i.

- Fix T_2 and consider all possible values of T_1, c_1, and c_2 (relative to T_2).
- For any T_1, assuming a fully utilized processor, the minimum U is when $c_1 = T_2 - T_1$ (thus $c_2 = T_1 - cr_1$ and $U = T_2 / T_1 + 2\,T_1 / T_2 - 2$).
  - If $c_1 = T_2 - T_1 - \varepsilon$, then for feasibility, $c_2 = T_2 - 2c_1$ and U increases.
  - If $c_1 = T_2 - T_1 + \varepsilon$, then for feasibility, $c_2 = T_1 - c_1$ and U also increases.
  - Note that we kept the processor fully utilized to obtain an upper bound on the feasible utilization.





- Next, differentiate U w.r.t. T_1 and equate the result to zero to obtain $T_2 / T_1 = \sqrt{2}$
- Hence, among all possible values of T_1, the minimum utilization is $2\sqrt{2} - 2$

Proof of (II) for sets of n tasks.

- For given values of $T_1, \ldots, T_n$, the upper bound, U_{ub}, on the utilization of feasible sets is obtained when
  $c_1 = T_2 - T_1$
  $c_2 = T_3 - T_2$
  ….
  $c_{n-1} = T_n - T_{n-1}$
  $c_n = T_1 - (c_1 + \ldots + c_{n-1}) = 2\,T_1 - T_n$
- For the above values, $U = \sum_{i=1}^{n-1} R_i + \frac{2}{R_1 R_2 \cdots R_{n-1}} - n$
  where $R_i = T_{i+1} / T_i$ and $R_1 \ldots R_{n-1} = T_n / T_1$
- Differentiate U w.r.t. $R_1, \ldots, R_{n-1}$ and equate the result to zero to obtain $R_1 = \ldots = R_{n-1} = 2^{1/n}$
- Hence, among all the T's and c's, the minimum utilization for feasibility is $U = n\,(2^{1/n} - 1)$

Time demand analysis

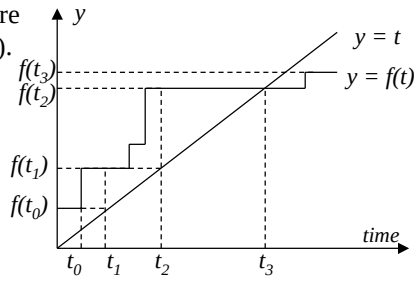
A necessary and sufficient test for RMS feasibility

- Sort the tasks $J_1, \ldots, J_n$ by order of priorities ($T_1 \leq \ldots. \leq T_n$)
- Task J_i , will meet its deadline T_i , if there is a time t earlier than T_i , such that

$$c_i + \sum_{k=1}^{i-1} \left\lceil \frac{t}{T_k} \right\rceil c_k \leq t$$

May iteratively find the first *t*, such that $f(t) = t$

- t is the response time of task J_i .
- Need to only consider the times t where some event occurs (periods boundaries).
- Repeat the above test for each task J_i .
- What is the test complexity?



Deadline monotonic scheduling

Used when the deadline of task J_i , is D_i which is less than T_i .

- Use priority proportional to the deadlines rather than periods
- Deadline monotonic is optimal among fixed priority schedules
- Necessary and sufficient test is that, for every *i*, the value of t that satisfies

$$c_i + \sum_{k=1}^{i-1} \left\lceil \frac{t}{T_k} \right\rceil c_k = t$$

also satisfies $t \leq D_i$. That is the response time is less than the deadline.

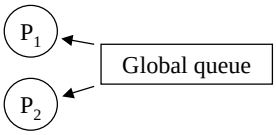
- Can show that a task set is feasible with DMS if $\sum_{i=1}^{n} \frac{c_i}{D_i} \leq n(2^{1/n} - 1)$

Note that when $D_i < T_i$, then EDF cannot guarantee feasibility if $U \leq 1$.

Some other conditions are needed. (for all $t > 0$, $t \geq \sum_{i=1}^{n} \left\lfloor \frac{t}{T_i} \right\rfloor c_i$).

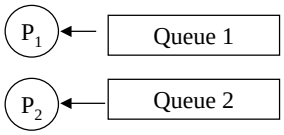
Periodic scheduling on multiprocessors

Global scheduling:



- One global ready queue sorted by priority
- The priority of executing tasks should be larger than the task at the head of the queue
- Both global RMS and EDF are very inefficient.
- Example: 2 processors, and 3 tasks; (0, ε, 1- ε), (0, ε, 1– ε), (0, 1 , 1).
- Utilization may be zero for large number of processors and very small ε.

Partitioned scheduling:



- Partition the tasks into separate queues (using any bin packing scheme (first fit, best fit ,…)
- Observe the feasibility of each processor