Real-Time Systems Lecture 3
These were some of the more important things written out during the class.

\[ n = 3: \quad e_1 = 1, \quad e_2 = 3, \quad e_3 = 5 \]
\[ p_1 = 4, \quad p_2 = 8, \quad p_3 = 10 \]

\[ T_1 \text{ schedulability}: \quad e_1 \leq p_1 \]
\[ T_2 \text{ schedulability}: \quad e_1 + e_2 \leq p_1 \quad 1 + 3 \leq 4 \]
\[ T_3 \text{ schedulability}: \quad e_1 + e_2 + e_3 \leq p_1 \quad \times \]

\[ 2e_1 + e_2 + e_3 \leq 8 \times \frac{1}{2} \]
\[ 3e_1 + 2e_2 + e_3 \leq 10 \quad \frac{1}{10} \]
\[ 3 + 6 + 5 \leq 10 \times \text{NOT SCHEDULABLE} \]

\[ e_3 \leq p_1 - (e_1 + e_2) \rightarrow 4 - (3 + 1) = 0 \]
\[ \text{OR} \quad e_3 \leq p_2 - (2e_1 + e_2) \rightarrow 8 - (2 + 3) = 3 \]
\[ \text{OR} \quad e_3 \leq p_3 - (3e_1 + 2e_2) \rightarrow 10 - (3 + 6) = 1 \]
\[ \text{Max value for } e_3 \text{ for schedulability} = 3 \]
Claim: If an iteration of $T_i$ is released at the same time as an iteration of every higher-priority task $T_j$, then $T_i$ will have the longest response time possible under any phasing.

\[
\sum_{i=1}^{n} \left\lceil \frac{t}{p_i} \right\rceil e_i = t \quad \text{for some } t \in (0, p_j],
\]

\[
\sum_{i=1}^{j} \left\lceil \frac{t}{p_i} \right\rceil e_i + e_j = t
\]
Check if
\[ \sum_{i=1}^{j} \left\lfloor \frac{t}{p_i} \right\rfloor e_i \leq t \]
for multiples of \( p_1, p_2, \ldots, p_{j-1} \)
and \( p_j \)
(Note: need to check only up to \( t = p_j \)).

This is the condition for schedulability when task phasing are all 0; write the equivalent condition for when the phasing of \( T_i = \phi_i \).
RM is an optimal static priority algorithm

Assume RM is suboptimal

⇒ ∃ some task set J and some other static priority algorithm, A, such that
  - A can successfully schedule J, and
  - RM cannot """

Suppose \( J = \{ T_1, T_2 \} \).

\( P_1 < P_2 \Rightarrow T_1 \not\preceq_{RM} T_2 \)

- If A assigns higher priority to \( T_2 \) than to \( T_1 \) (\( T_2 \succ_A T_1 \))
- By assumption, A meets all task deadlines:
  \[ \begin{align*}
  T_2 \text{ sched } & \Rightarrow e_2 \leq P_2 \\
  T_1 \text{ sched } & \Rightarrow \left\lfloor \frac{t}{P_1} \right\rfloor e_2 + e_1 = t \quad \text{for some } 0 < t \leq P_1.
\end{align*} \]
By assumption, RM fails

$$e_1 > p_1 \quad \text{(1)}$$

or

$$\left\lfloor \frac{t}{p_1} \right\rfloor e_1 + e_2 > t$$

for all $0 < t \leq p_2$.

Condition (1) would result in $T_i$ always missing, irrespective of the scheduling alg. used.
\[ \left[ \frac{t_1}{P_2} \right] e_2 + e_1 - \left[ \frac{t_1}{P_1} \right] e_1 - e_2 < 0 \]

where \( t_1 \) is a value of \( t \) that satisfies (3). \( 0 < t_1 \leq P_1 < P_2 \)

\[ e_2 + e_1 - e_1 - e_2 < 0 \]

\[ 0 < 0 \]

which is a contradiction.
RM is an optimal static priority alg. 
\[ P_1 \prec P_2 \prec \ldots \prec P_n \] (general n)

\[ T_1 \succ_T T_2 \succ_T \ldots \succ_T T_n \]

Proof: By Contradiction. Assume alg. \( A \) which can schedu

If \( A \) is different from \( RM \),
there must be \( T_i, T_{i+1} \)

such that \( T_i \succ_{RM} T_{i+1} \)

and \( T_{i+1} \nprecedes T_i \).

Construct \( A_i \) which has exactly the
same priority assignment as \( A \) except

that \( T_i \nprecedes T_{i+1} \).

\( A \)-schedulable \( \Rightarrow A_i \)-schedulable