Dynamic Priority Driven Scheduling

Sérgio Campos

Priority Driven Scheduling

Restrições:

- Release time correto;
- Preempção perfeita — a qualquer instante;
- Processos não se suspendem;
- Troca de contexto a custo zero;
- Número fixo de processos;

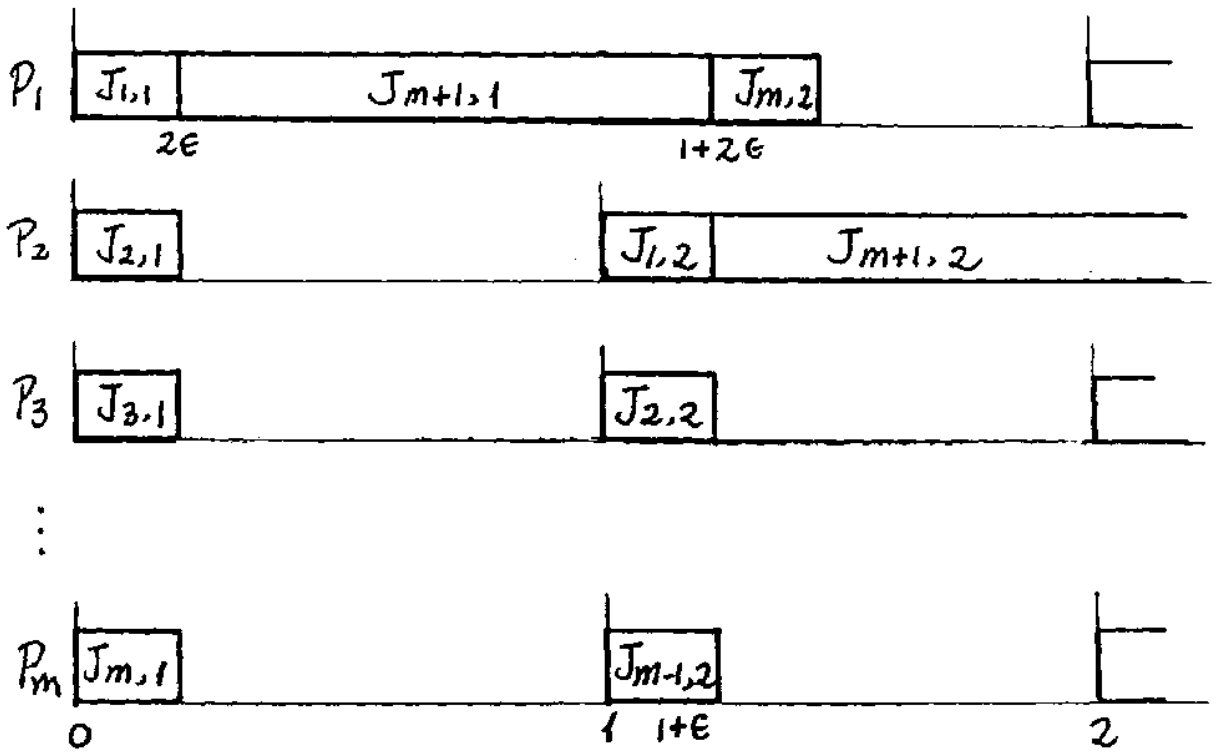
Particionamento Estático X Dinâmico

Estático: Jobs atribuidos aos vários processadores

- Uma fila por processador

Dinâmico: jobs são colocados em uma única fila de prioridades.

- Sistemas dinâmicos são mais difíceis de analisar;
- Por isto sistemas estáticos são mais usados.



Algoritmos de Priorização

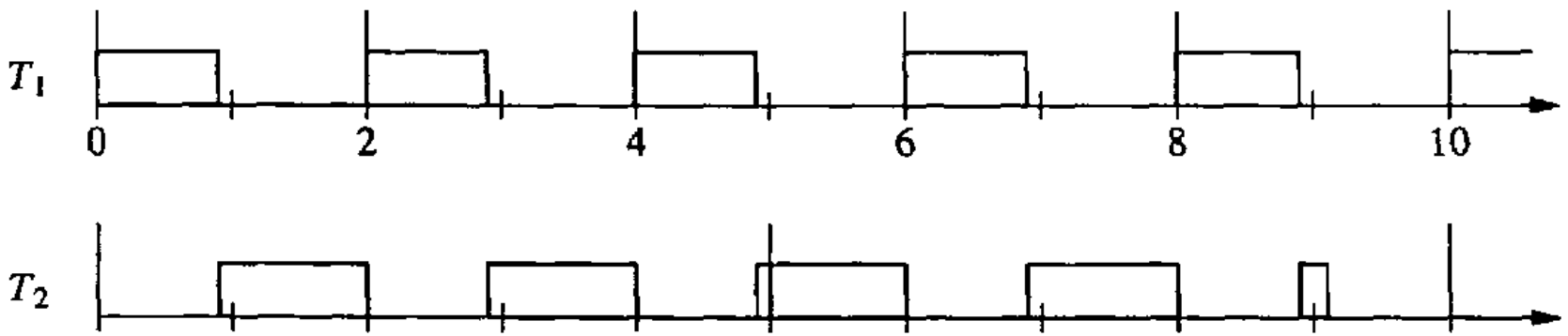
Prioridade Estática: Rate Monotonic — RM

Atribui prioridades maiores a processos com frequência de execução maior

- período menor $\rightarrow$ prioridade maior
- Não leva em conta: tempo de execução, importância, ...

Variação: Deadline Monotonic — DM

- RM = DM se período = deadline

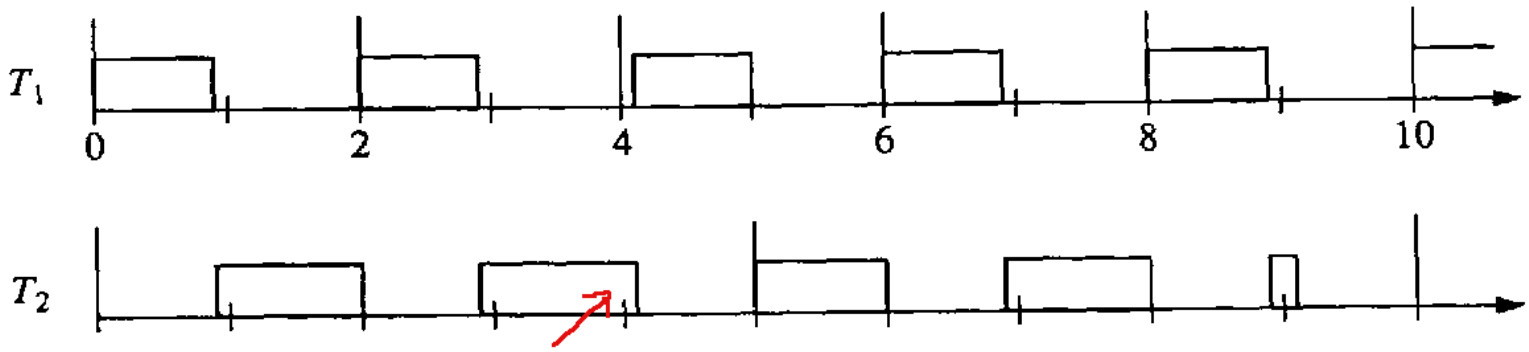


$$T_1 = (2, 0.9), T_2 = (5, 2.3)$$

Algoritmos de Priorização

Prioridades dinâmicas:

- Earliest Deadline First — EDF



$$T_1 = (2, 0.9),\ T_2 = (5, 2.3)$$

- Least Slack-time First — LST
  - slack: quanto falta para o deadline
  - quanto menor o slack, maior a prioridade
  - LST estrito é difícil de implementar

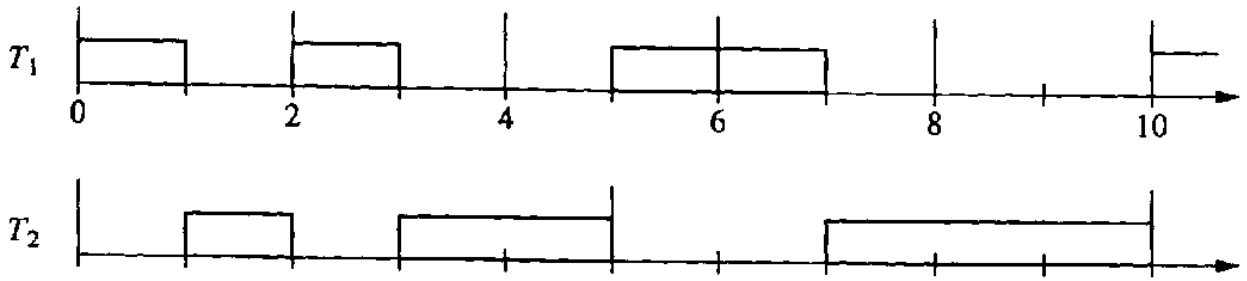
Utilização de Escalonabilidade

Soma-se o percentual de utilização da CPU por cada processo

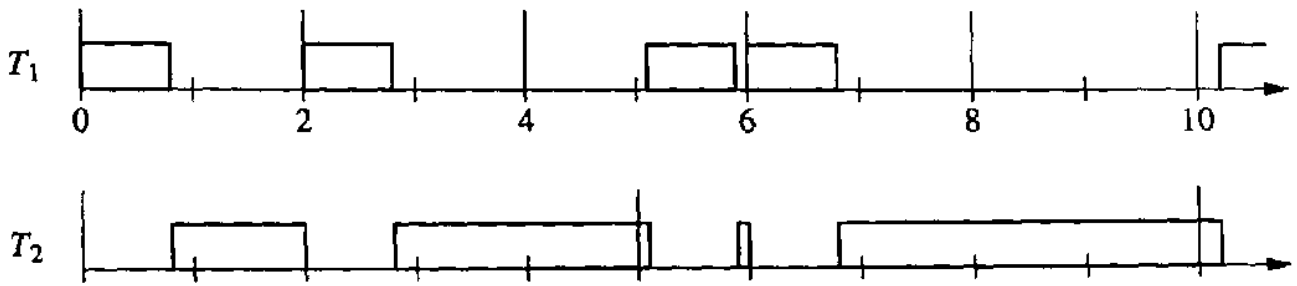
- Um algoritmo de escalonamento garante que tarefas podem ser escalonadas se a utilização total for menor que o percentual de utilização do algoritmo
- Quanto maior melhor; sempre ≤ 1, portanto $= 1$ é ótimo
- EDF é ótimo; RM e DM não
  - Contudo, prioridades estáticas são mais previsíveis
  - Overruns não afetam prioridades maiores
  - Com isto consegue-se saber quem vai dançar em caso de problemas.

Exemplo de Imprevisibilidade de EDF

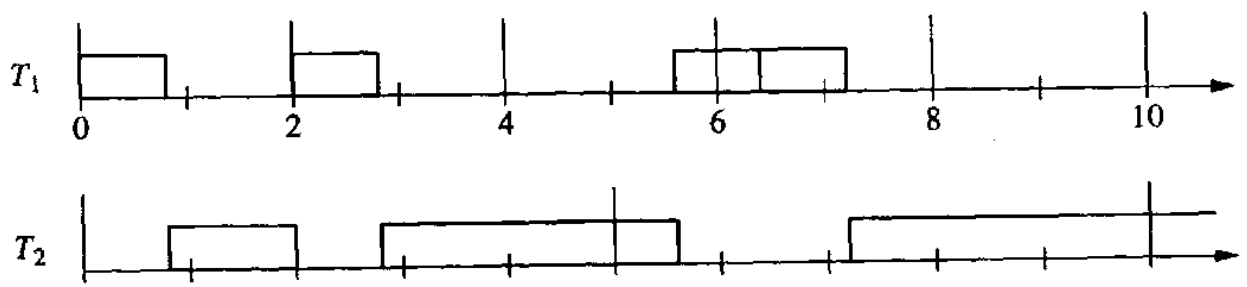
- $T_1 = (2, 1)$; $T_2 = (5, 3.0)$; $U = 1.1$



- $T_1 = (2, 0.8)$; $T_2 = (5, 3.5)$; $U = 1.1$



- $T_1 = (2, 0.6)$; $T_2 = (5, 4.0)$; $U = 1.1$



Escalonabilidade de EDF

Teorema: Um sistema com T processos independentes e preemptáveis com deadline = período pode ser escalonado em um processador sse U ≤ 1

Corolários:

- Se deadline $>$ período, ok.
- A utilização escalonável de EDF $= 1$

Contudo:

- Se deadline $<$ período, nada se garante.

Teste de Escalonabilidade

Dados:

- Um conjunto de tarefas independentes definidas por (p_i, e_i, D_i)
- Determinar se todas as tarefas podem ser escalonadas ou não

Teste de escalonabilidade para EDF

- sim se $\sum_{(k=1)}^{n} \frac{e_k}{D_k} \leq 1$

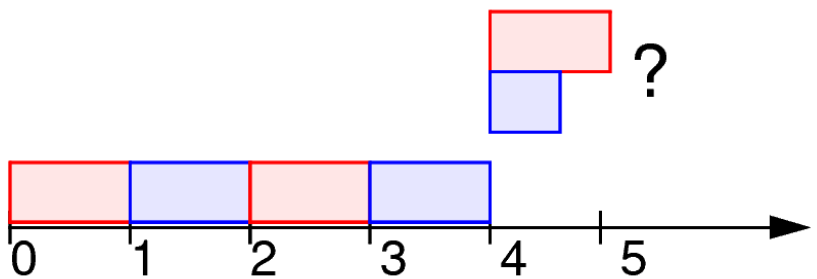
Exemplo: um controlador com

- Função de controle: $e_1 = 8ms, p_1 = 10ms$
- Built in self test: $e_2 = 50ms, p_2 = ?? \leq 1s$
- Comunicação: $e_3 = 15ms, p3 = ??$

Limitações de RM

Rate monotonic é garantidamente não ótimo!
Suponha:

- $T_1 = (2, 1)$, $T_2 = (5, 2.5)$
- U $= 1 \rightarrow$ escalonável com EDF
- Em $(0, 4]$ prioridade de T_1 tem que ser maior
- Mas em $t = 4$, prioridade de T_2 tem que ser maior



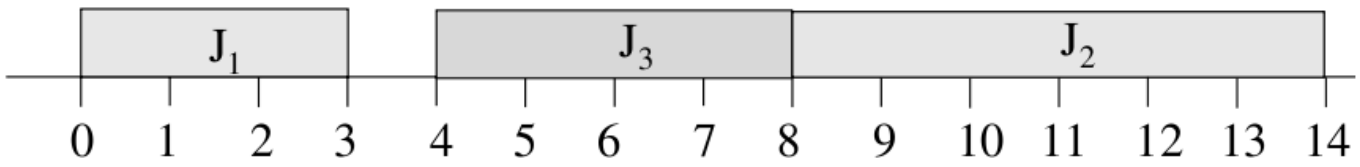
Contudo RM é ótimo se todos os períodos são harmônicos (simply periodic)

- Neste caso $U_{RM} = 1$.

Limitações de EDF

Suponha: $J_1 = (r_1, e_1, d_1) = (0, 3, 10); J_2 = (2, 6, 14); J_3 = (4, 4, 12)$

- Preemptivo:



- Não preemptivo: