
Sistemas de Tempo Real:

Escalonamento Baseado em Prioridades Fixas

Rômulo Silva de Oliveira
Departamento de Automação e Sistemas - DAS – UFSC

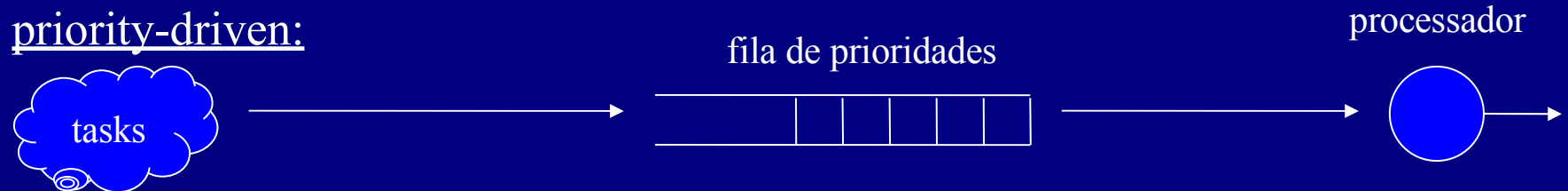
romulo@das.ufsc.br
<http://www.das.ufsc.br/~romulo>

Bettati

- Preemptivo vs não preemptivo
- Determinação da escala (executivo cíclico, hiperperíodo, $n!$)
- WCET Worst Case Execution Time
- Custo da preempção
- tarefas periódicas, aperiódicas e esporádicas
- tarefas periódicas e jobs
- instante crítico
- deadline relativo e absoluto

Escalonamento Priority-Driven de Tarefas Periódicas

- Escalonamento dirigido a prioridades vs. clock-driven:

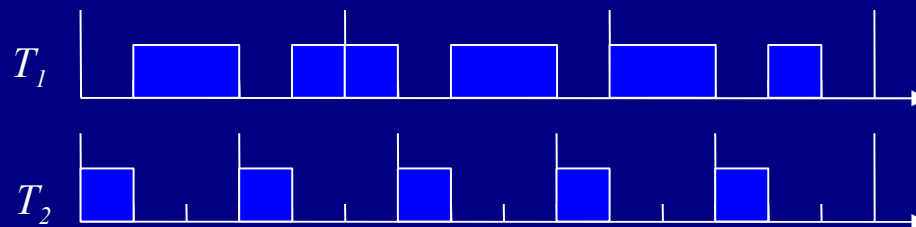


Static-Priority vs. Dynamic Priority

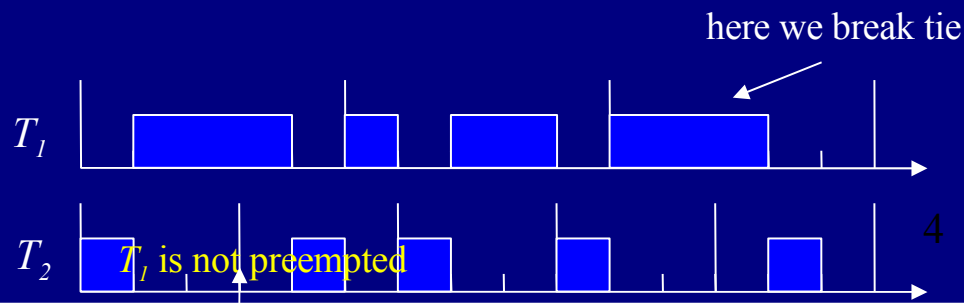
- **Static-Priority:** Todos jobs na tarefa têm mesma prioridade.
exemplo: **Rate-Monotonic:** "Quão menor o período, maior a prioridade."

$$T_1 = (5, 3, 5)$$

$$T_2 = (3, 1, 3)$$



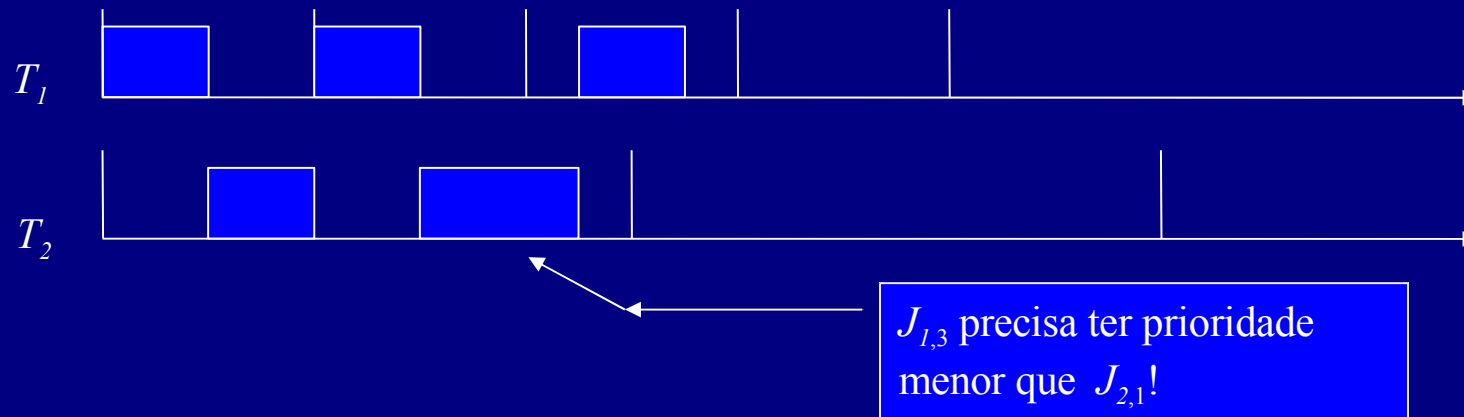
- **Dynamic-Priority:** Pode atribuir prioridades diferentes a jobs individuais.
exemplo: **Earliest-Deadline-First:** "Quão mais próximo o deadline absoluto, maior a prioridade."



E sobre prioridade estática?

- Prioridade estática não é ótima!

$$\left. \begin{array}{l} T_1 = (2, 1, 2) \\ T_2 = (5, 2.5, 5) \end{array} \right\} U = \frac{e_1}{p_1} + \frac{e_2}{p_2} = 1 \leq 100\%$$



- Então: Por que se incomodar com prioridade estática?
 - simplicidade
 - previsibilidade

Escalonamento Baseado em Prioridades Fixas

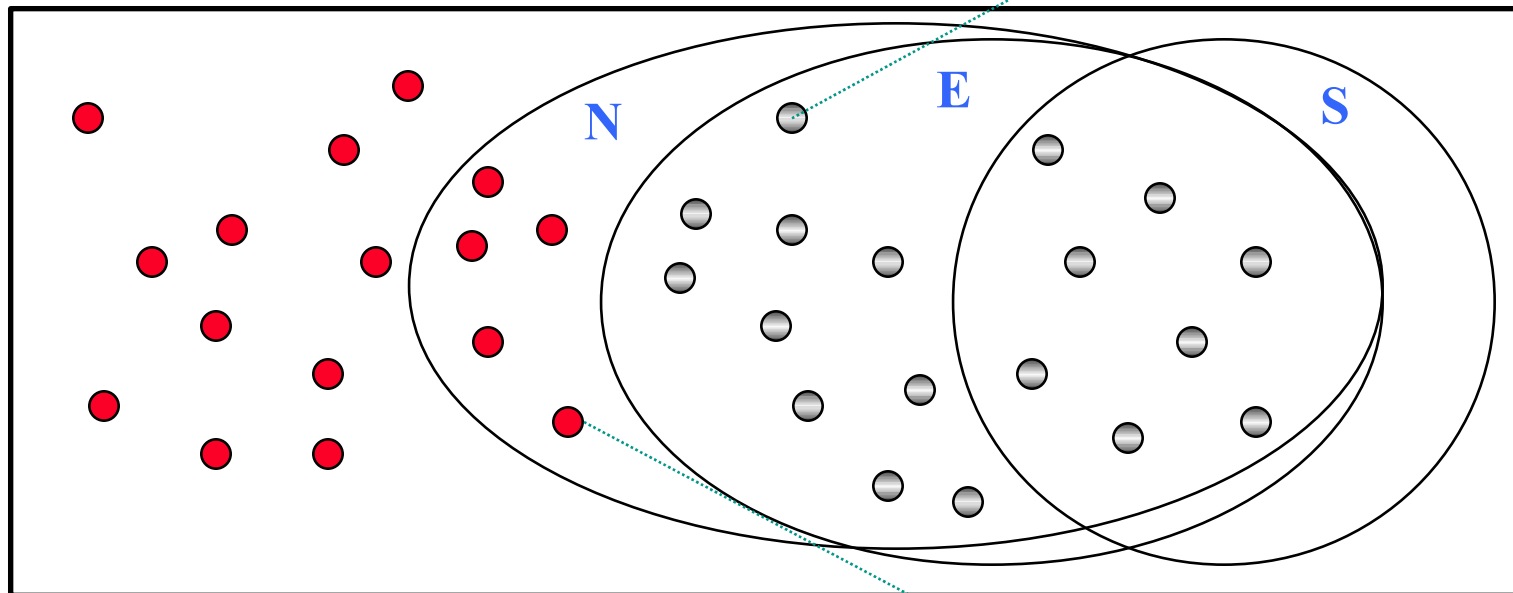
- Caracterização
- Rate Monotonic
- Análise da Utilização
- Análise do Tempo de Resposta
- Deadline Monotonic
- Release Jitter
- Bloqueios

Escalonamento Baseado em Prioridades Fixas

- Aplicação composta por tarefas (processos)
- Estados de uma tarefa:
 - Liberada (pronta para executar, apta, ready)
 - Suspensa esperando por evento ou passagem do tempo
 - Bloqueada
- Em geral escalonamento é preemptivo
- Tarefas possuem **prioridade fixa** definida em projeto
- Garantia exige
 - Tarefas periódicas ou esporádicas
 - Tempo máximo de computação conhecido
 - Teste de escalonabilidade apropriado

Teste de Escalonabilidade

- Teste de Escalonabilidade pode ser
 - Suficiente
 - Exato
 - Necessário



Conjunto de todos os sistemas

não escalonável

Rate Monotonic

- Quanto menor o período, maior a prioridade
- Ótimo quando
 - Tarefas são periódicas
 - Deadline é sempre igual ao período
- Exemplo:

– Tarefas	T1	T2	T3
– Períodos	P1=30	P2=40	P3=50
– Prioridades	p1=1	p2=2	p3=3
- Cuidado!
 - Número menor indica prioridade maior
 - Muitas vezes é o contrário

Análise da Utilização

- Utilização de uma tarefa:
 - Tempo máximo de computação dividido pelo período
 - T1 tem $C1=12$ e $P1=50$, então $U1 = 12 / 50 = 0.24$
- Teste para Rate Monotonic, sistema é escalonável se:

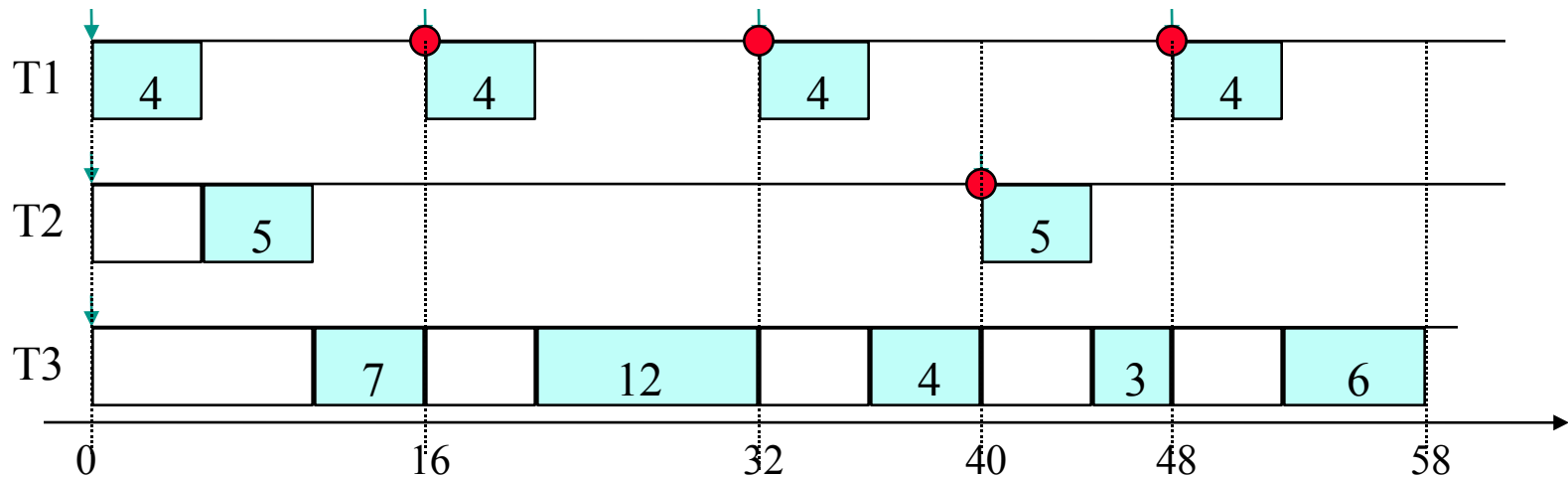
$$\sum_{i=1}^N \left(\frac{C_i}{P_i} \right) < N(2^{1/N} - 1)$$

- Para $N=1$ utilização máxima é 100%
- Para $N=10$ utilização máxima é 71.8%
- Para N grandes utilização máxima tende para 69.3%
- Baseado no conceito de Instante Crítico
- Teste é suficiente mas não necessário

Análise da Utilização

- Exemplo:

	T1	T2	T3
– Períodos	P1=16	P2=40	P3=80
– Computação	C1=4	C2=5	C3=32
– Utilização	U1=0.250	U2=0.125	U3=0.400
– Prioridades	p1=1	p2=2	p3=3
- Utilização total é 0.775, abaixo do limite 0.780



Rate monotonic e análise de utilização

- $U \leq 1$ se período das tarefas são múltiplos do período da tarefa mais prioritária
- mudar os períodos para facilitar a escalonabilidade do sistema?
- mudar os períodos para tolerância a faltas (Previsibilidade na falha: tarefas menos prioritárias falham primeiro)?

Análise do Tempo de Resposta

- Limitações da análise baseada em Utilização
 - Não é exata
 - Aplicável apenas a modelos de tarefas muito simples
- Análise baseada em **Tempo de Resposta**
 - Abordagem analítica calcula tempo de resposta no pior caso
 - Tempo de resposta de cada tarefa é comparado com o deadline da tarefa

Análise do Tempo de Resposta

- Como calcular o tempo de resposta de cada tarefa ?
- Para a tarefa mais prioritária temos $R1 = C1$
- Demais tarefas sofrem **Interferência** das tarefas com prioridade maior
- Neste caso, $R_i = C_i + I_i$
- Interferência é máxima a partir do **Instante Crítico**
 - Todas as tarefas são liberadas simultaneamente
 - Suposto instante zero na análise

Análise do Tempo de Resposta

- Seja T_j uma tarefa com prioridade maior que T_i
- Quantas vezes T_j pode acontecer durante a execução de T_i ?

$$\left\lceil \frac{R_i}{P_j} \right\rceil$$

- Qual a interferência total de T_j sobre T_i ?

$$\left\lceil \frac{R_i}{P_j} \right\rceil \times C_j$$

- Qual a interferência total sobre T_i ?

$$\sum_{j \in hp(i)} \left\lceil \frac{R_i}{P_j} \right\rceil \times C_j$$

Análise do Tempo de Resposta

- O tempo máximo de resposta de T_i é $R_i = C_i + I_i$

$$R_i = C_i + \sum_{j \in hp(i)} \left\lceil \frac{R_i}{P_j} \right\rceil \times C_j$$

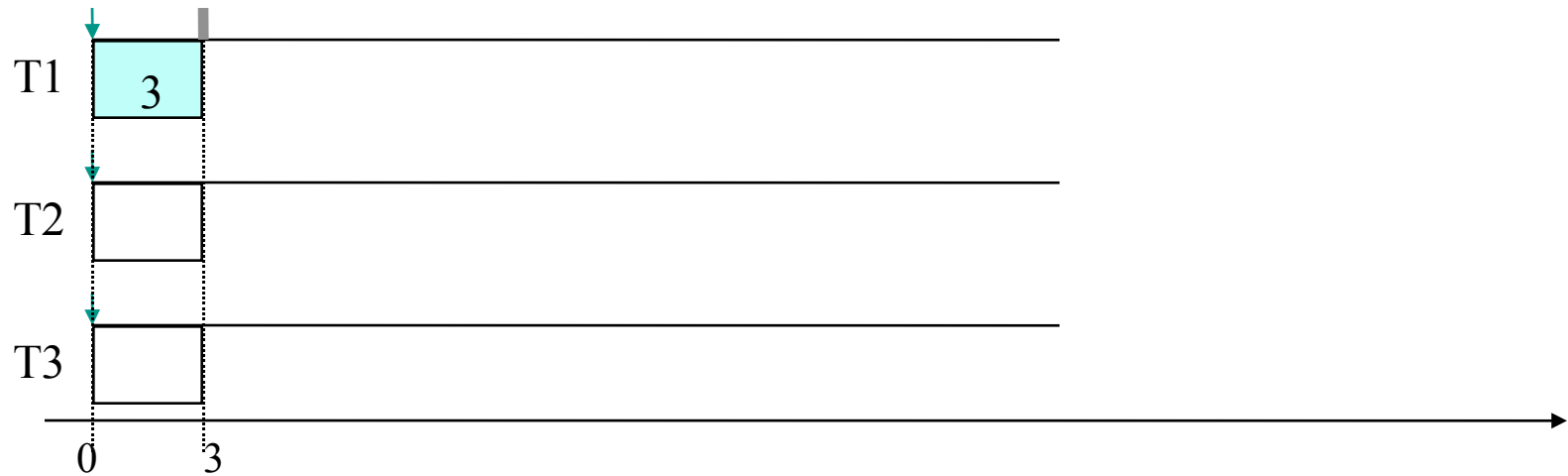
- Equação é recursiva
- Calculada através de iterações sucessivas, até:
 - Tempo de resposta passar do deadline
 - Resultado convergir, iteração $x+1$ igual a iteração x

$$w_i^{x+1} = C_i + \sum_{j \in hp(i)} \left\lceil \frac{w_i^x}{P_j} \right\rceil \times C_j \quad w_i^0 = C_i$$

Análise do Tempo de Resposta

- Exemplo:

	T1	T2	T3
– Períodos	$P1=7$	$P2=12$	$P3=20$
– Computação	$C1=3$	$C2=3$	$C3=5$
– Prioridades	$p1=1$	$p2=2$	$p3=3$
- $R1 = C1 = 3$

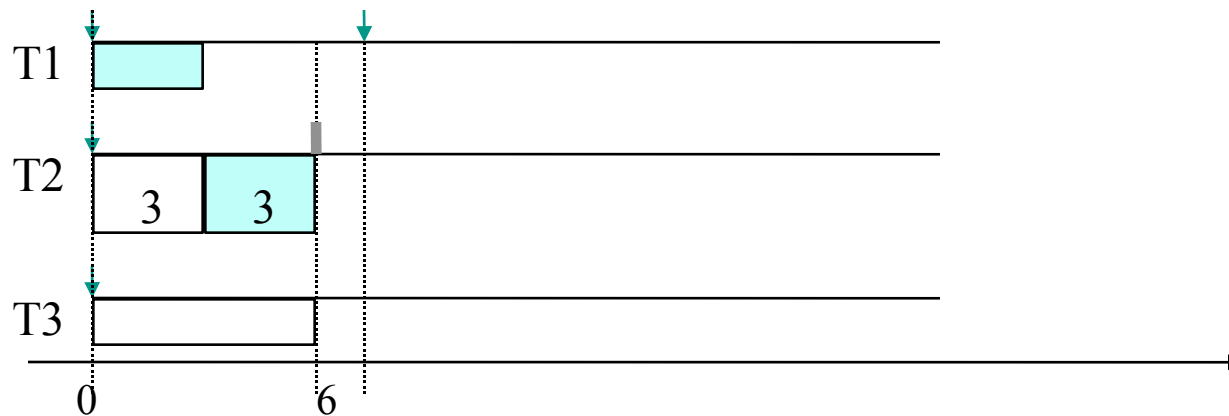


Análise do Tempo de Resposta

- Análise da tarefa T2: $w_2^0 = C_2 = 3$

$$w_2^1 = C_2 + \left\lceil \frac{w_2^0}{P_1} \right\rceil \times C_1 = 3 + \left\lceil \frac{3}{7} \right\rceil \times 3 = 6$$

$$w_2^2 = C_2 + \left\lceil \frac{w_2^1}{P_1} \right\rceil \times C_1 = 3 + \left\lceil \frac{6}{7} \right\rceil \times 3 = 6$$



Análise do Tempo de Resposta

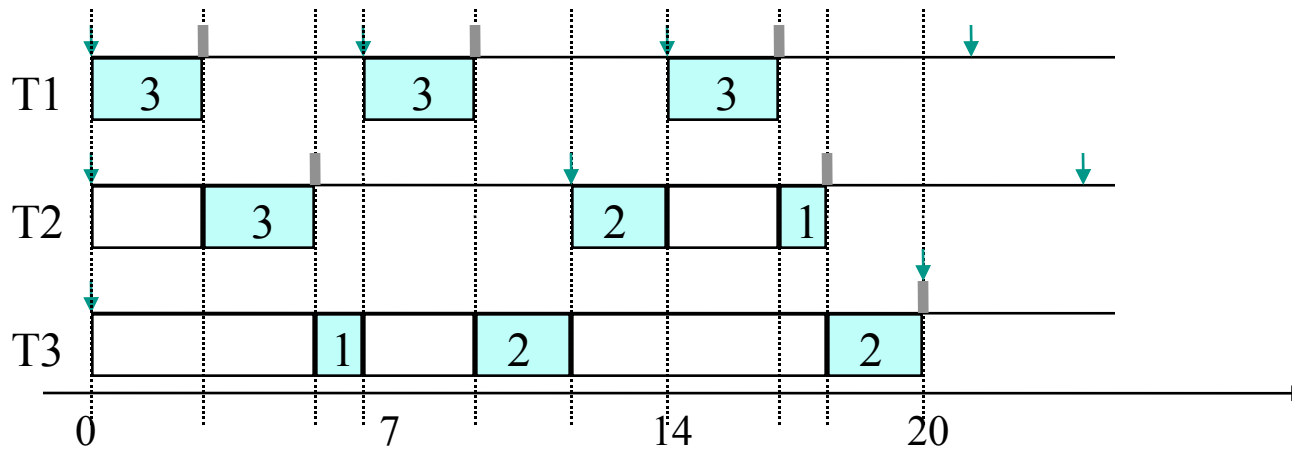
- Análise da Tarefa T3: $w_3^0 = C_3 = 5$

$$w_3^1 = C_3 + \sum_{j \in hp(3)} \left\lceil \frac{w_3^0}{P_j} \right\rceil \times C_j = 5 + \left\lceil \frac{5}{7} \right\rceil \times 3 + \left\lceil \frac{5}{12} \right\rceil \times 3 = 11$$

$$w_3^2 = 5 + \left\lceil \frac{11}{7} \right\rceil \times 3 + \left\lceil \frac{11}{12} \right\rceil \times 3 = 14 \quad w_3^3 = 5 + \left\lceil \frac{14}{7} \right\rceil \times 3 + \left\lceil \frac{14}{12} \right\rceil \times 3 = 17$$

$$w_3^4 = 5 + \left\lceil \frac{17}{7} \right\rceil \times 3 + \left\lceil \frac{17}{12} \right\rceil \times 3 = 20 \quad w_3^5 = 5 + \left\lceil \frac{20}{7} \right\rceil \times 3 + \left\lceil \frac{20}{12} \right\rceil \times 3 = 20$$

Análise do Tempo de Resposta



- Exemplo:

	T1	T2	T3
– Períodos	$P1=7$	$P2=12$	$P3=20$
– Computação	$C1=3$	$C2=3$	$C3=5$
– Prioridades	$p1=1$	$p2=2$	$p3=3$
– Tempo Máximo de Resposta	$R1=3$	$R2=6$	$R3=20$

Análise do Tempo de Resposta

- Teste de escalonabilidade **exato**
- Deadline pode ser menor que o período
 - Basta comparar o tempo de resposta com o deadline
- Deadline maior que o período exige análise mais complexa
 - Tarefa pode interferir com ela mesma
- Tarefas esporádicas podem ser tratadas como periódicas
 - Intervalo mínimo entre ativações é usado como período
- A forma como prioridades são atribuídas NÃO é importante
 - Funciona pois “hp(i)” sempre indica as tarefas mais prioritárias do que a tarefa “i”

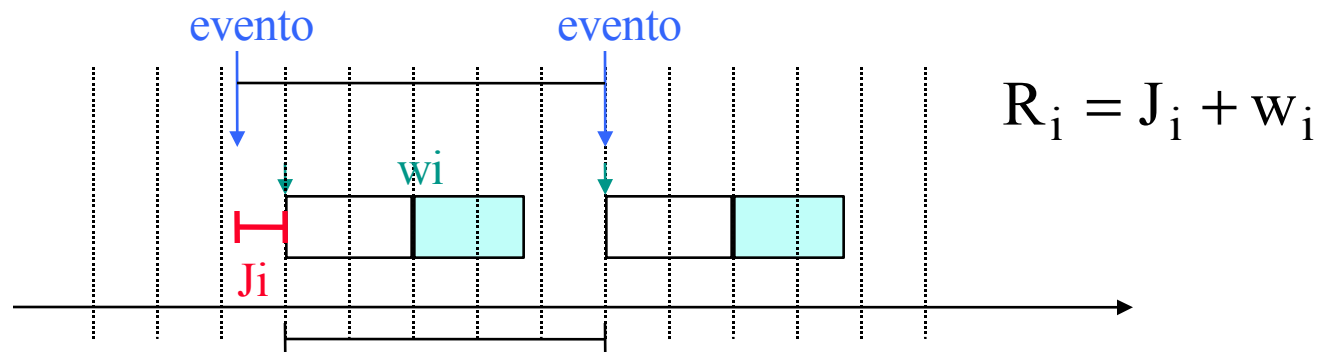
Deadline Monotonic

- Quanto menor o deadline, maior a prioridade
- Ótimo quando deadline é menor ou igual ao período
- Exemplo:

– Tarefas	T1	T2	T3	T4
– Períodos	P1=20	P2=15	P3=10	P4=20
– Tempo máximo de computação	C1=3	C2=3	C3=4	C4=3
– Deadline	D1=5	D2=7	D3=10	D4=20
– Prioridades	p1=1	p2=2	p3=3	p4=4
– Tempo máximo de resposta	R1=3	R2=6	R3=10	R4=20
– Caso fosse RM	R1=10	R2=7	R3=4	R4=20

Release Jitter

- Suponha uma tarefa esporádica liberada por evento externo
 - Eventos podem ser amostrados periodicamente
 - Sinalização do evento pode ter atraso variável



- **Release Jitter:** Atraso máximo na liberação da tarefa

$$w_i = C_i + \sum_{j \in hp(i)} \left\lceil \frac{w_i + J_j}{P_j} \right\rceil \times C_j$$

Bloqueios

- Podem ocorrer bloqueios devido a relações de exclusão mútua
 - Estruturas de dados compartilhadas
 - Dispositivos compartilhados
- Suponha T1 e T2, T1 com maior prioridade
- Se T2 fica bloqueada, esperando por T1
 - Ok, T1 tem mesmo prioridade superior
- Se T1 fica bloqueada, esperando por T2
 - Cálculo do tempo de resposta deve incluir a espera máxima B_i

$$w_i = C_i + B_i + \sum_{j \in hp(i)} \left\lceil \frac{w_i + J_j}{P_j} \right\rceil \times C_j$$

$$R_i = J_i + w_i$$

- Cada tarefa recebe uma prioridade fixa
- Teste é desenvolvido para examinar a escalonabilidade
- Dois tipos de análise
- **Análise da Utilização**
 - Utiliza o valor C/P
- **Análise do Tempo de Resposta**
 - Tenta calcular o tempo de resposta no pior caso

Time-Demand Analysis

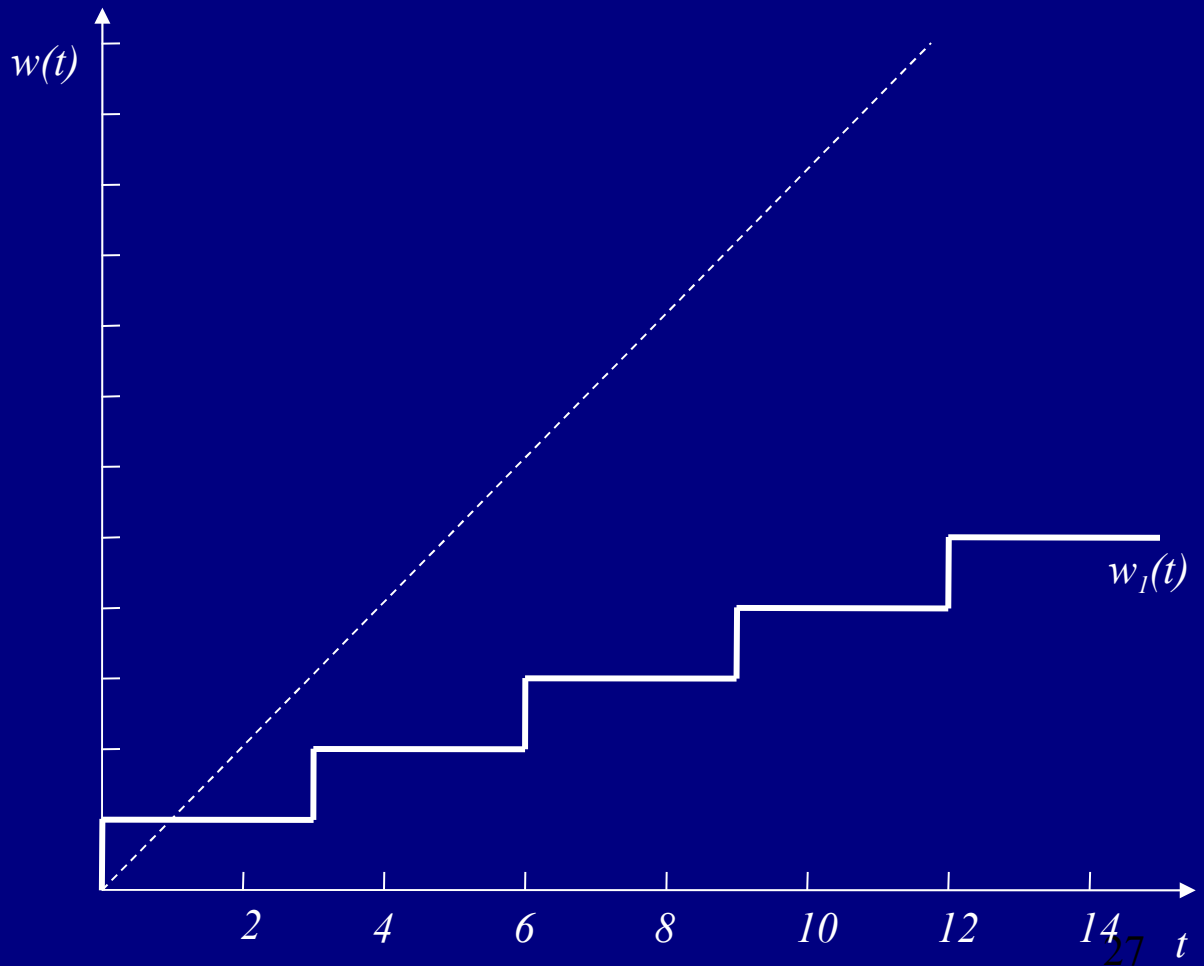
- Calcula a demanda total de tempo de processador do job liberado no instante crítico e pelas tarefas de maior prioridade
- Verifica se a demanda pode ser atendida antes do deadline.
- Determina se T_i é escalonável:
 - Um job em T_i liberado em um instante crítico de T_i :
 $w_i(t)$: Demanda de tempo de processador deste job e de todos os jobs liberados com prioridade mais alta em (t_o, t) :

$$w_i(t) = C_i + \sum_{k=1}^{i-1} \left\lceil \frac{t}{p_j} \right\rceil C_j$$

- Este job em T_i alcança seu deadline se, para algum $t_1 \leq D_i \leq p_i$: $w_i(t_1) \leq t_1$
- Caso contrário job não alcança seu deadline, e o sistema não é escalonável

Exemplo com RM

T_1	=	(3, 1)
T_2	=	(5, 1.5)
T_3	=	(7, 1.25)
T_4	=	(9, 0.5)



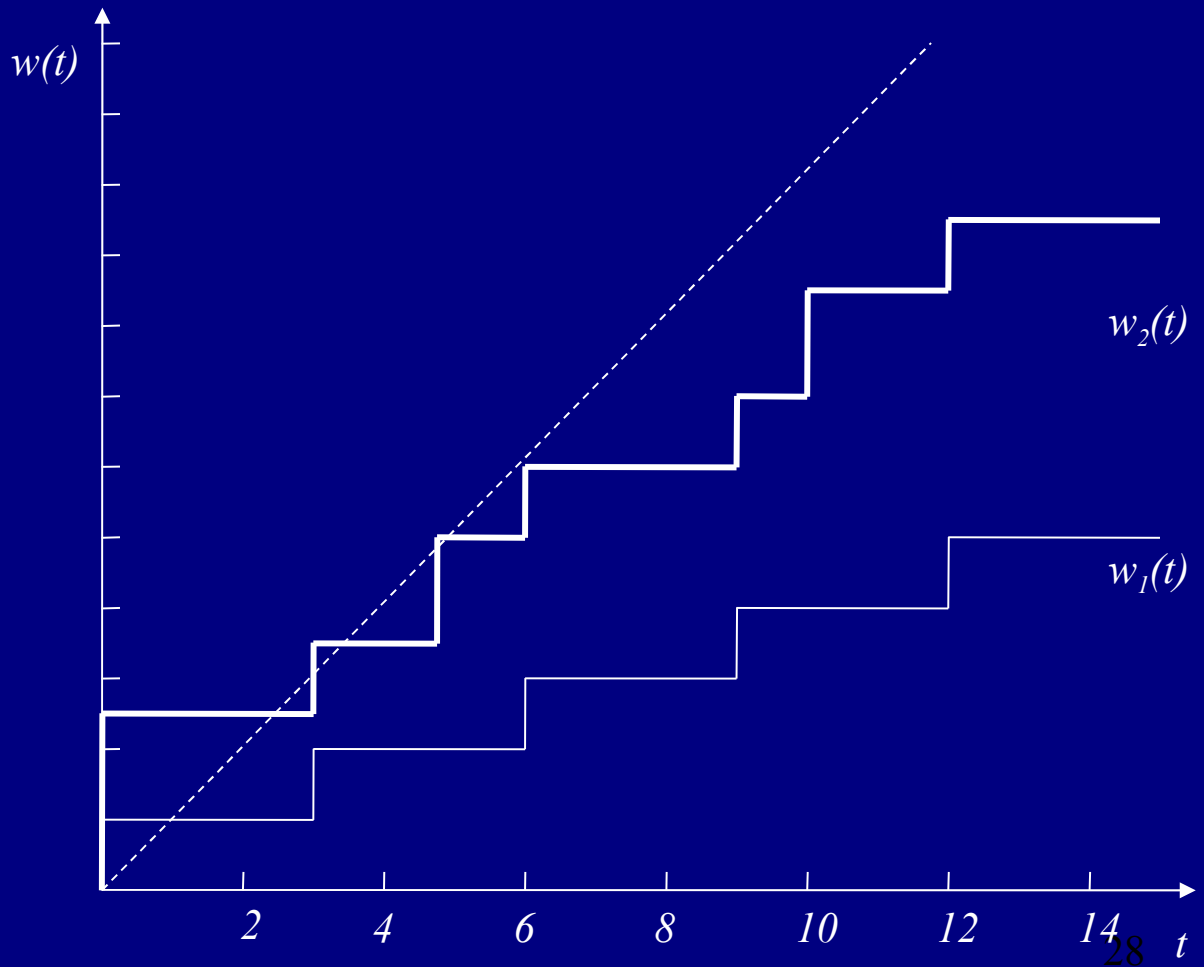
Example

$$T_1 = (3, 1)$$

$$T_2 = (5, 1.5)$$

$$T_3 = (7, 1.25)$$

$$T_4 = (9, 0.5)$$



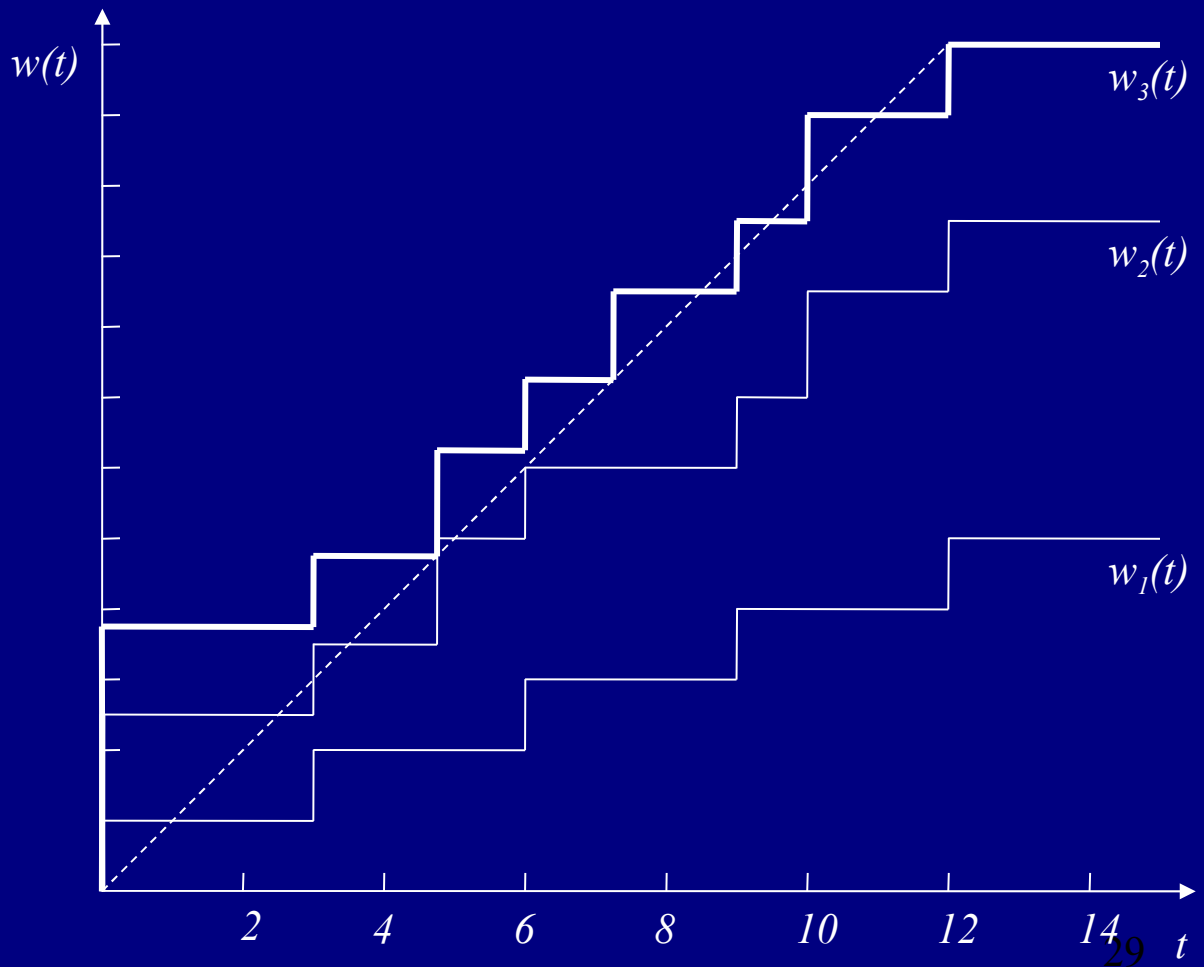
Example

$$T_1 = (3, 1)$$

$$T_2 = (5, 1.5)$$

$$T_3 = (7, 1.25)$$

$$T_4 = (9, 0.5)$$



Example

$$T_1 = (3, 1)$$

$$T_2 = (5, 1.5)$$

$$T_3 = (7, 1.25)$$

$$T_4 = (9, 0.5)$$

