
Escalonamento em Sistema de Tempo Real

Fundamentos dos Sistemas de Tempo Real
2ª Edição

Rômulo Silva de Oliveira
Edição do Autor, 2020

www.romulosilvadeoliveira.eng.br/livrotemporeal



Como escalonar as tarefas em um sistema de tempo real ?

Escalonamento de Tarefas

- Sistemas de tempo real são organizados em torno do conceito de tarefas
- Tarefas podem ser implementadas como:
 - Funções em um executivo cíclico
 - Tratadores de interrupções
 - Threads
- Forma mais comum: tarefa implementada como thread
 - Pode ser com microkernel ou com kernel complexo
- Em geral existem muito mais tarefas do que processadores
- É preciso definir que algoritmo será usado para escolher qual tarefa será executada a seguir
 - No caso de multicore, quais serão executadas

Escalonamento em Sistemas de Tempo Real 1/2

- Ênfase do escalonamento está no atendimento dos requisitos temporais: deadlines
- Executivo cíclico:
 - Ordem de execução das tarefas expressa no próprio código do executivo
 - Solução não preemptiva
- Laço principal com tratadores de interrupção:
 - Tratadores de interrupção possuem prioridade preemptiva mais alta que o laço principal
 - Entre interrupções o hardware define uma ordem de prioridade
 - Ordem pode ser definida pelo software durante a inicialização do sistema
 - Interrupção de mais alta prioridade ocorre durante tratador de mais baixa prioridade ?
 - Em alguns sistemas todos os tratadores executam com interrupções desabilitadas
 - Em outros sistemas é permitido que uma interrupção interrompa o tratador em execução
 - Tratadores de interrupção podem ou não ser preemptados, depende do sistema

Escalonamento em Sistemas de Tempo Real 2/2

- Restante do capítulo:
- Escalonamento das tarefas quando um microkernel ou kernel é usado
- Tarefas de tempo real são implementadas como threads
- Algoritmo preferido: prioridades preemptivas
 - desenvolvedor gerencia o uso do processador
- Como definir as prioridades ?

Prioridades Fixas 1/2

- Aplicação composta por tarefas
- Estados de uma tarefa:
 - Liberada (pronta para executar, apta, ready)
 - Executando (running)
 - Suspensa esperando pela próxima ativação
 - Outros estados serão acrescentados mais adiante
- Em geral escalonamento é preemptivo
- Tarefas possuem **prioridade fixa** definida em projeto
- Garantia exigirá ainda
 - Tarefas periódicas ou esporádicas
 - Tempo máximo de computação conhecido
 - Teste de escalonabilidade apropriado

- Rate Monotonic – RM (Taxa Monotônica)
 - Prioridade mais alta para a tarefa com período menor
 - Prioridade fixa
- Deadline Monotonic – DM (Deadline Monotônico)
 - Prioridade mais alta para a tarefa com deadline relativo menor
 - Prioridade fixa
 - Igual ao RM quando $D = P$
- Importância
 - Prioridade mais alta para a tarefa mais importante da aplicação
 - Prioridade fixa
- Outras

Prioridades Variáveis 1/3

- Cada tarefa recebe uma prioridade que varia ao longo do tempo
- Prioridade leva em conta informações relativas à execução
- Diferentes jobs da mesma tarefa podem receber prioridades diferentes
- Cada Job em particular pode receber prioridade fixa ou variável

- A escala de execução só é conhecida durante a execução

- Necessário Teste de Escalonabilidade
 - Para saber antes se todos os deadlines estão garantidos ou não

- ***EDF – Earliest Deadline First***
 - Inversamente proporcional ao deadline absoluto
 - Ótimo em relação aos critérios de prioridades variáveis
- ***LSF (LST ou LLF) – Least Slack First***
 - Inversamente proporcional ao tempo livre (*laxity* ou *slack*)
 - Ótimo em relação aos critérios de prioridades variáveis
 - Overhead maior que EDF
- ***FCFS – First Come First Served***
 - Inversamente proporcional ao tempo de espera por serviço
 - Não é ótimo com respeito ao cumprimento de deadlines
- **EDF é o mais usado**

Prioridades Variáveis 3/3

- **Least Slack First - LSF**
- Quanto menos tempo livre (*slack*), maior a prioridade
- LSF é ótimo quando EDF for ótimo
- Prioridade das tarefas na fila de aptos aumenta com passar do tempo
- Prioridade da tarefa em execução mantém-se constante
 - Gera número maior de chaveamento de contextos que EDF
 - Maior overhead
- **Não apresenta vantagens face a EDF**

Prioridades + Teste de Escalonabilidade

- Cada tarefa recebe uma prioridade
- Escalonamento em geral é preemptivo
- Teste realizado antes da execução determina escalonabilidade
 - Teste considera como são as tarefas (modelo de tarefas)
 - Periódica, esporádica, $D \leq P$, bloqueios, etc
 - Teste considera forma como prioridades são atribuídas
 - Validade do teste é demonstrada como teorema
 - Complexidade do teste depende do modelo de tarefas
- Na execução:
 - Escalonador dispara as tarefas conforme as prioridades

Teste baseado na Utilização

- Utilização de uma tarefa:
 - Tempo máximo de computação dividido pelo período
 - Exemplo: T1 tem $C1=12$ e $P1=50$, então $U1 = 12 / 50 = 0.24$
- Utilização do sistema
 - Somatório da utilização de todas as tarefas
- Dado
 - Um modelo de tarefas
 - Uma política de atribuição de prioridades
- Existe um limiar de utilização para o processador, de tal sorte que:
 - Se a utilização do processador for menor que o limiar
 - Então jamais um deadline será perdido
- Limiar demonstrado como teorema

Prioridades Variáveis – Teste para EDF

- Supondo um conjunto de n tarefas
 - independentes e periódicas
- EDF como política de atribuição de prioridades
- Liu & Layland, 1973

- Se $D=P$, sistema é escalonável quando:

- Permite usar 100% do processador mantendo os deadlines
- Teste exato

$$\sum_{i=1}^N \left(\frac{C_i}{P_i} \right) \leq 1$$

- Se $D < P$, sistema é escalonável quando:

- Teste suficiente

$$\sum_{i=1}^N \left(\frac{C_i}{D_i} \right) \leq 1$$

- Para D arbitrário, sistema é escalonável quando:

- Teste suficiente

$$\sum_{i=1}^N \left(\frac{C_i}{\min(D_i, P_i)} \right) \leq 1$$

Prioridades Variáveis – Teste para EDF

- Existe grande variedade de testes de escalonabilidade para EDF
- Testes mais complexos
 - São menos pessimistas
 - Porém requerem esforço computacional maior
- EDF
 - As prioridades de todas as tarefas aptas e em execução aumentam
 - de igual modo com o passar do tempo
- Tarefa possui prioridade variável
- Mas um job em particular possui prioridade fixa

Prioridade Fixa – Teste para RM

- RM – Rate Monotonic (Taxa Monotonica)
- Quanto menor o período, mais alta 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

Prioridade Fixa – Teste para RM

- Utilização de uma tarefa:
 - Tempo máximo de computação dividido pelo período $U_i = C_i / P_i$
 - T1 tem $C_1=12$ e $P_1=50$, então $U_1 = C_1 / P_1 = 12 / 50 = 0.24$
- Liu & Layland, 1973
- Teste para Rate Monotonic, sistema é escalonável se:

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

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

Prioridade Fixa – Teste para RM

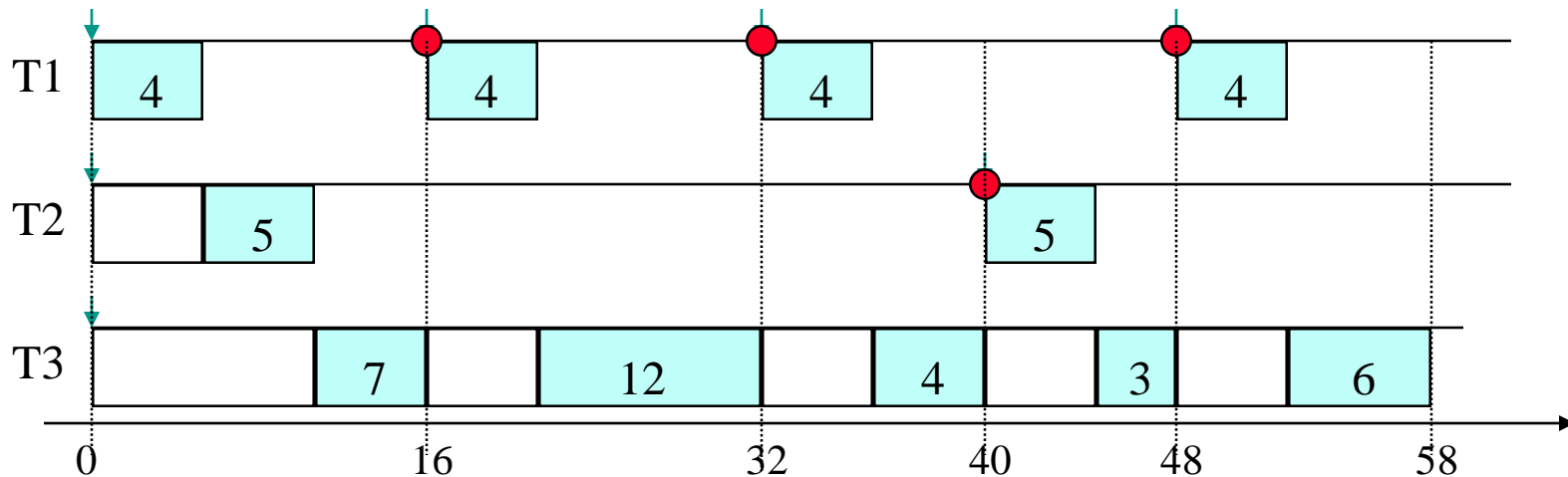
- N Limiar de Utilização
- 1 100.0%
- 2 82.8%
- 3 78.0%
- 4 75.7%
- 5 74.3%
- 10 71.8%

- infinito 69.3%

Prioridade Fixa – Teste para RM

- Exemplo:

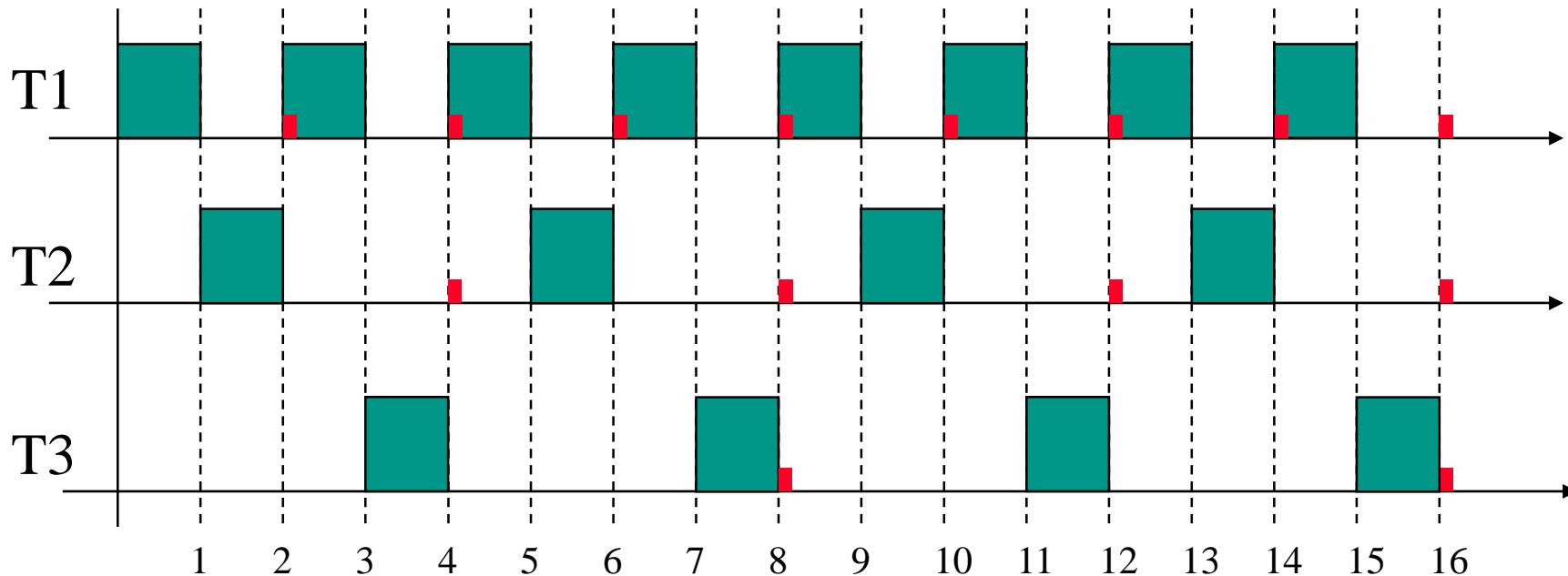
	T1	T2	T3
– Períodos	$P1=16$	$P2=40$	$P3=80$
– Computação	$C1=4$	$C2=5$	$C3=32$
– Utilização	$U1=0.250$	$U2=0.125$	$U3=0.400$
– Prioridades	$p1=1$	$p2=2$	$p3=3$
- Utilização total é 0.775, abaixo do limite 0.780



Prioridade Fixa – Teste para RM

- Exemplo:

	T1	T2	T3
– Períodos	$P1=2$	$P2=4$	$P3=8$
– Computação	$C1=1$	$C2=1$	$C3=2$
– Utilização	$U1=0.500$	$U2=0.250$	$U3=0.250$
– Prioridades	$p1=1$	$p2=2$	$p3=3$
- Utilização total é 1, acima do limiar 0.780, mas conjunto é escalonável



Prioridade Fixa – Teste para RM

- Limiar Hiperbólico (Hyperbolic Bound)
- Bini & Buttazzo & Buttazzo, 2001
- Tarefas independentes, P=D, Rate Monotonic
- Se

$$\prod_{i=1}^N (C_i / P_i + 1) \leq 2$$

- Então uma instância de cada tarefa está garantida a cada período
- Suficiente mas não necessário
- Menos pessimista que o teste de Liu & Layland, 1973

Teste baseado na Utilização – Resumo

- Testes baseados em utilização
 - Não são gerais
 - Não são exatos
 - Mas são rápidos, $O(N)$
- Exemplo de teste necessário mas não suficiente para este modelo de tarefas

$$\sum_{i=1}^N \left(\frac{C_i}{P_i} \right) \leq 1$$

- Escalonamento em Sistemas de Tempo Real
- Prioridades Fixas
- Prioridades Variáveis
- Testes Baseados em Utilização
- Questões Práticas



Escalonamento em Sistema de Tempo Real

Fundamentos dos Sistemas de Tempo Real
2ª Edição

Rômulo Silva de Oliveira
Edição do Autor, 2020

www.romulosilvadeoliveira.eng.br/livrotemporeal



Escalonamento em Sistema de Tempo Real

Parte II: Algumas Questões Práticas

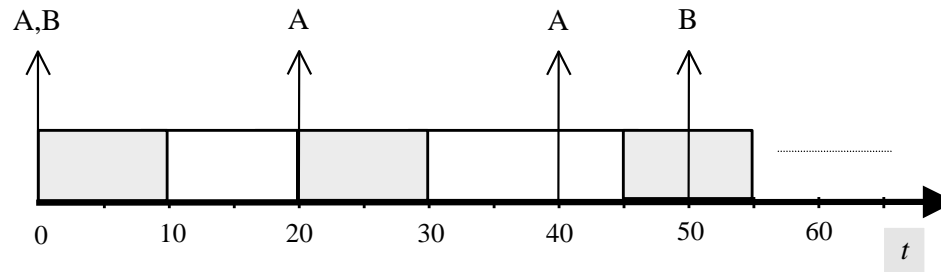
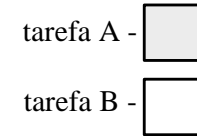


Prioridade Fixa – Deadline Monotonic

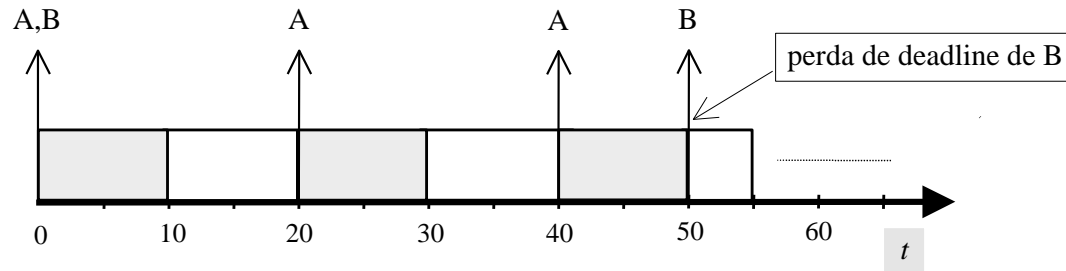
- Deadline monotonic (DM) será ótimo se qualquer conjunto de tarefas Q , o qual é escalonável por uma política de atribuição de prioridades W , também for escalonável por DM
- Pode-se provar a optimalidade de DM através da transformação das prioridades de Q (atribuídas por W) até que a ordenação seja aquela do DM
 - Desde que cada passo da transformação preserve a escalonabilidade
- Leung & Whitehead, 1982

Questão Prática: Escalonabilidade EDF versus RM 1/1

<i>tarefas periódicas</i>	C_i	P_i	D_i
tarefa A	10	20	20
tarefa B	25	50	50



(a) Escalonamento EDF

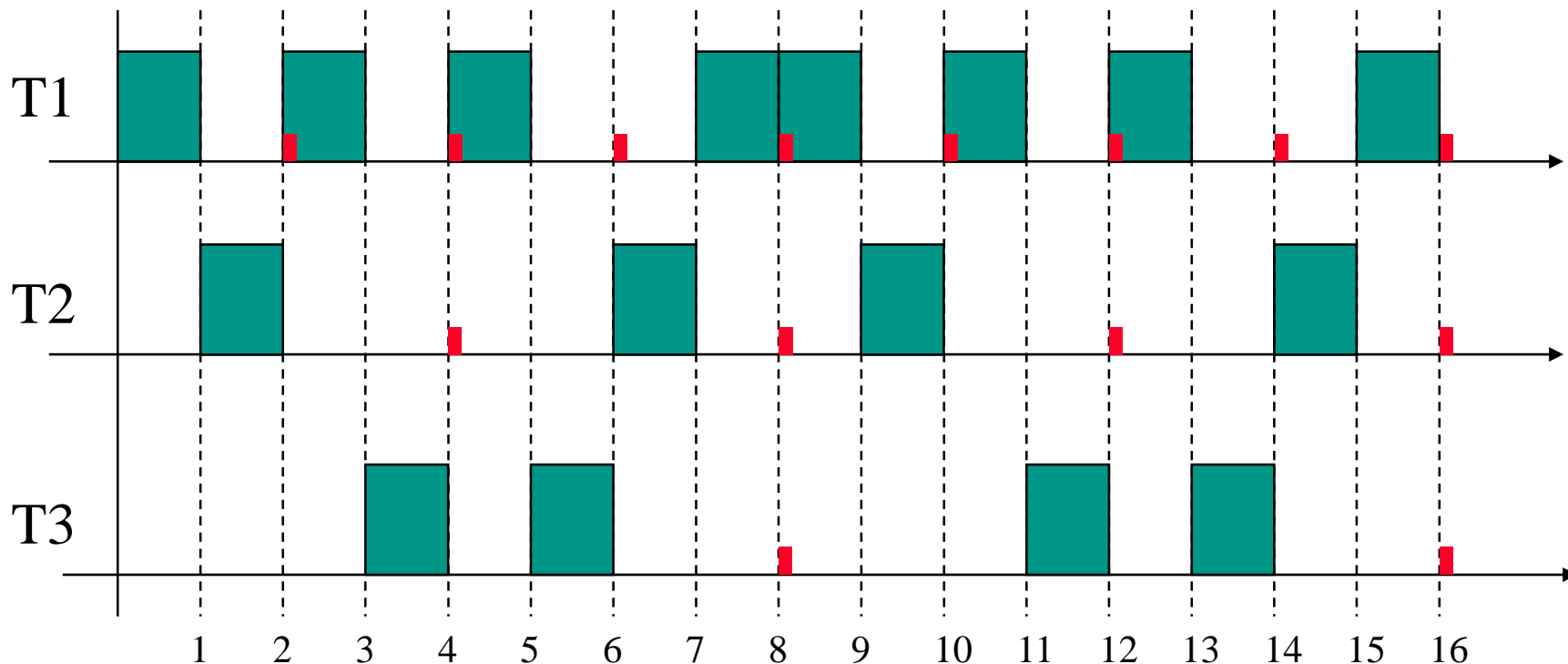


(b) Escalonamento RM

Figura 2.6: Escalas produzidas pelo (a) EDF e (b) RM

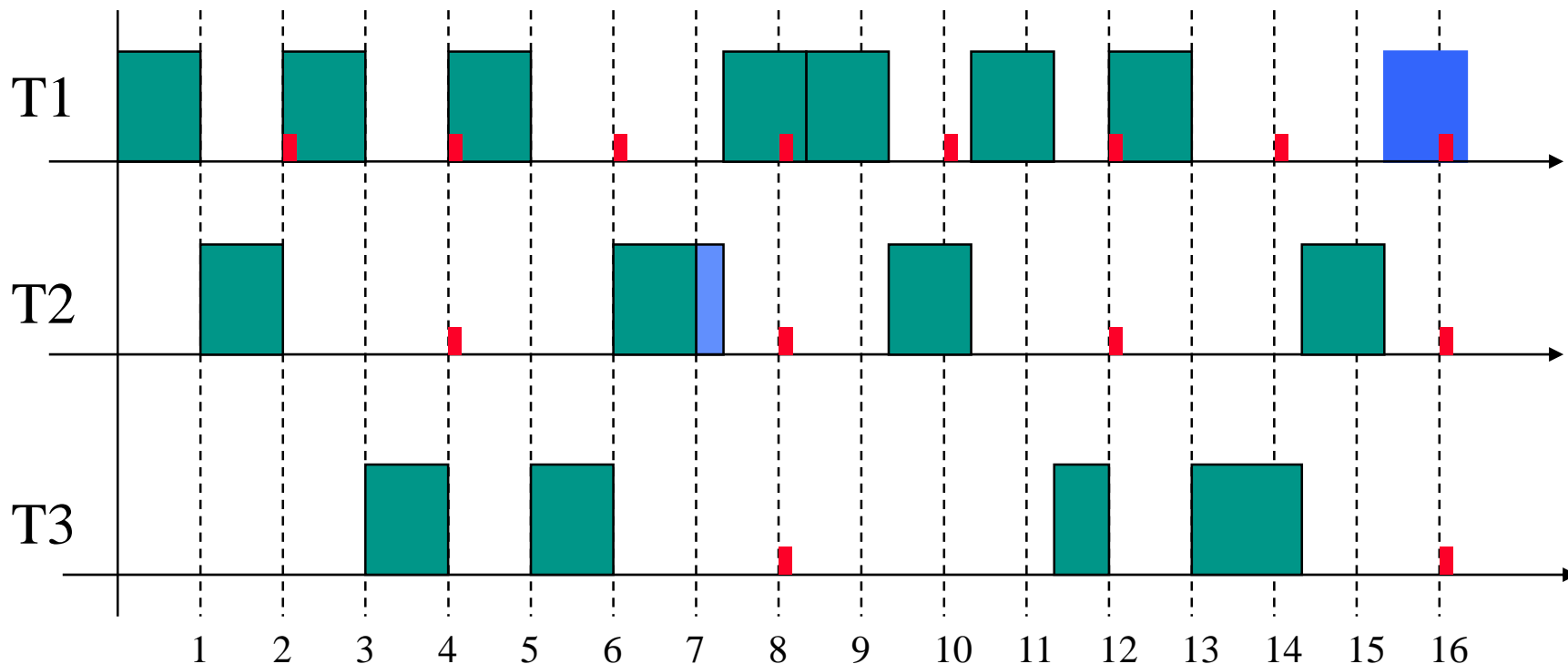
Questão Prática: Sobrecarga EDF versus RM 1/6

- T1: C1=1 P1=D1=2
- T2: C2=1 P2=D2=4
- T3: C3=2 P3=D3=8
- Execução normal com EDF



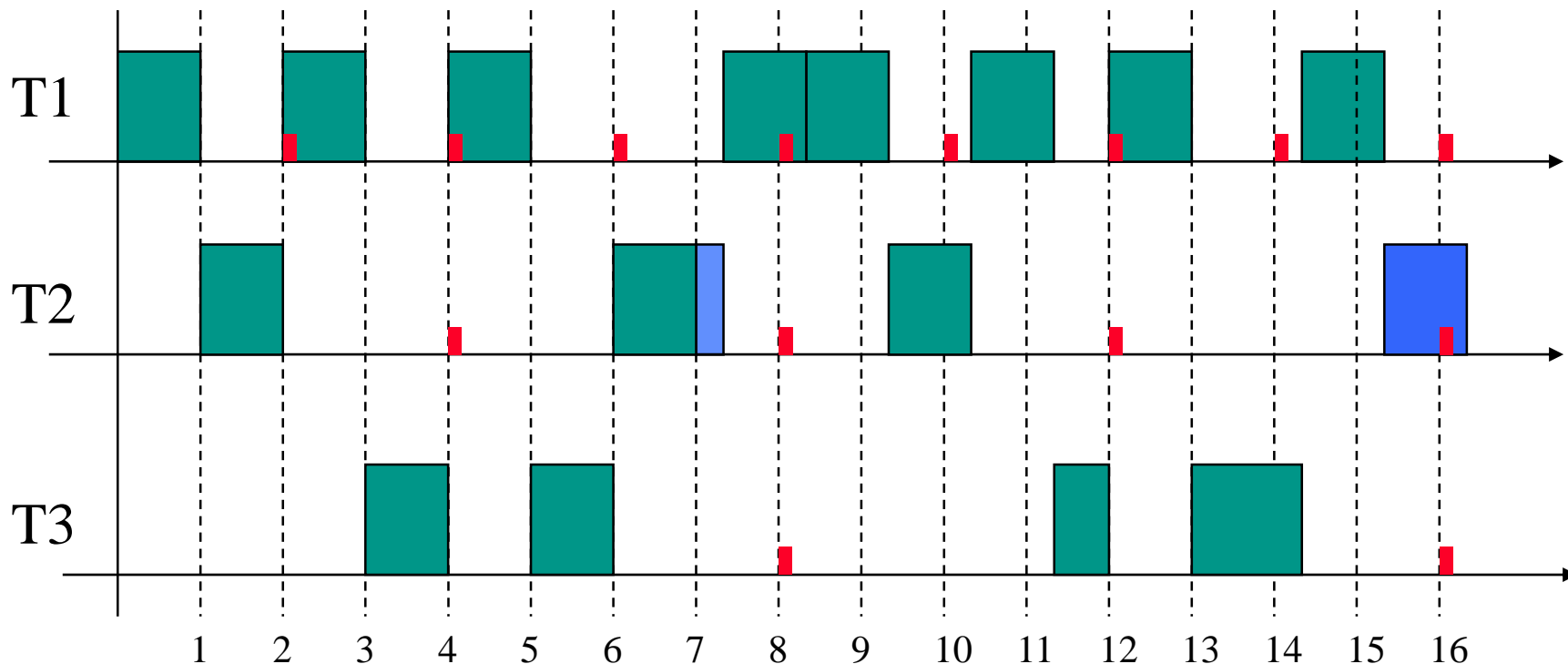
Questão Prática: Sobrecarga EDF versus RM 2/6

- T1: C1=1 P1=D1=2
- T2: C2=1 P2=D2=4
- T3: C3=2 P3=D3=8
- Sobrecarga devido a falha do projeto com EDF



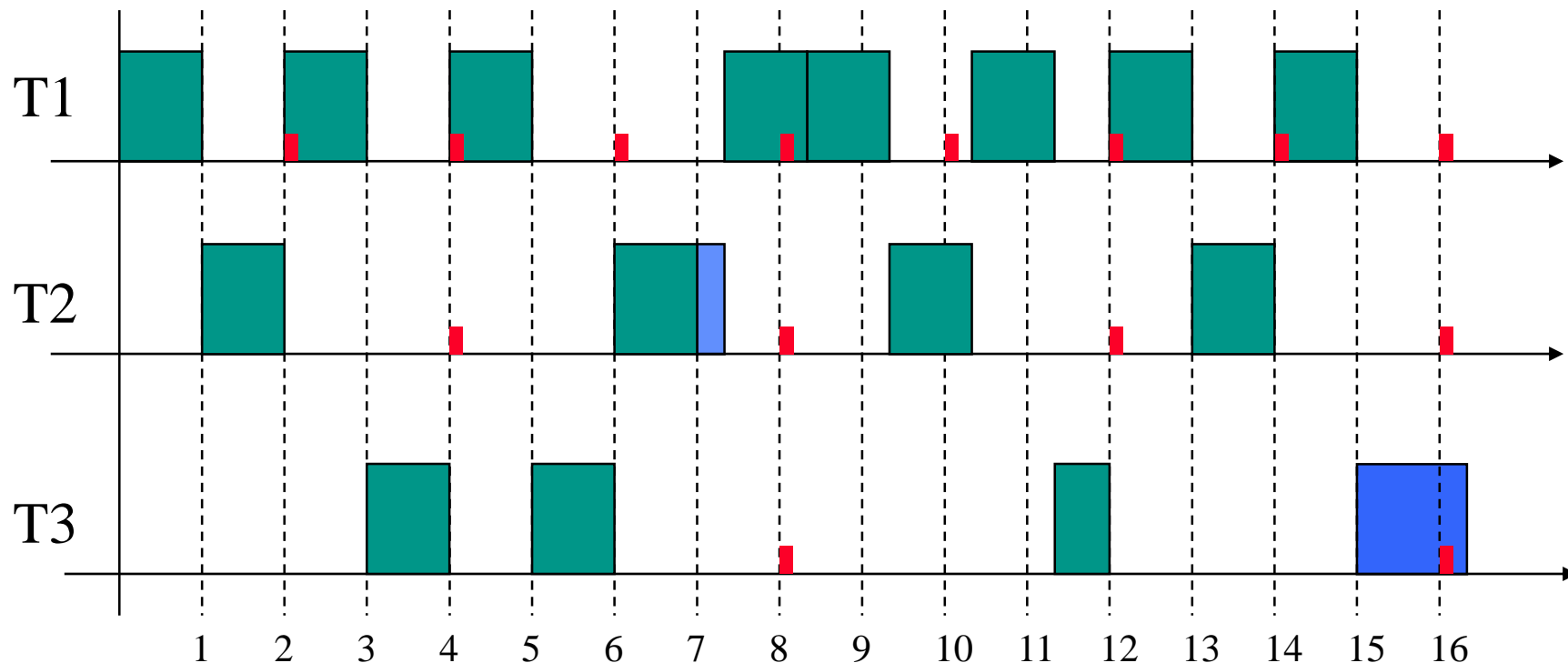
Questão Prática: Sobrecarga EDF versus RM 3/6

- T1: C1=1 P1=D1=2
- T2: C2=1 P2=D2=4
- T3: C3=2 P3=D3=8
- Sobrecarga devido a falha do projeto com EDF



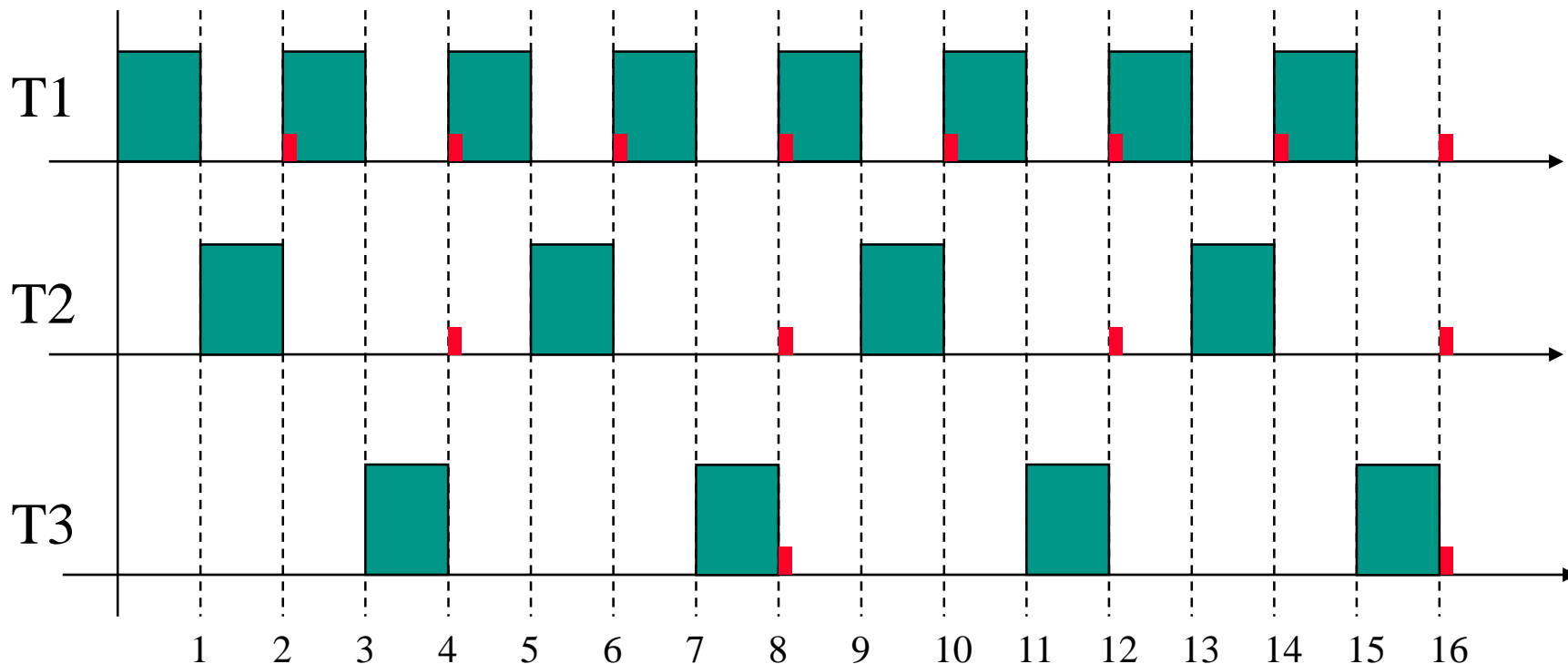
Questão Prática: Sobrecarga EDF versus RM 4/6

- T1: C1=1 P1=D1=2
- T2: C2=1 P2=D2=4
- T3: C3=2 P3=D3=8
- Sobrecarga devido a falha do projeto com EDF



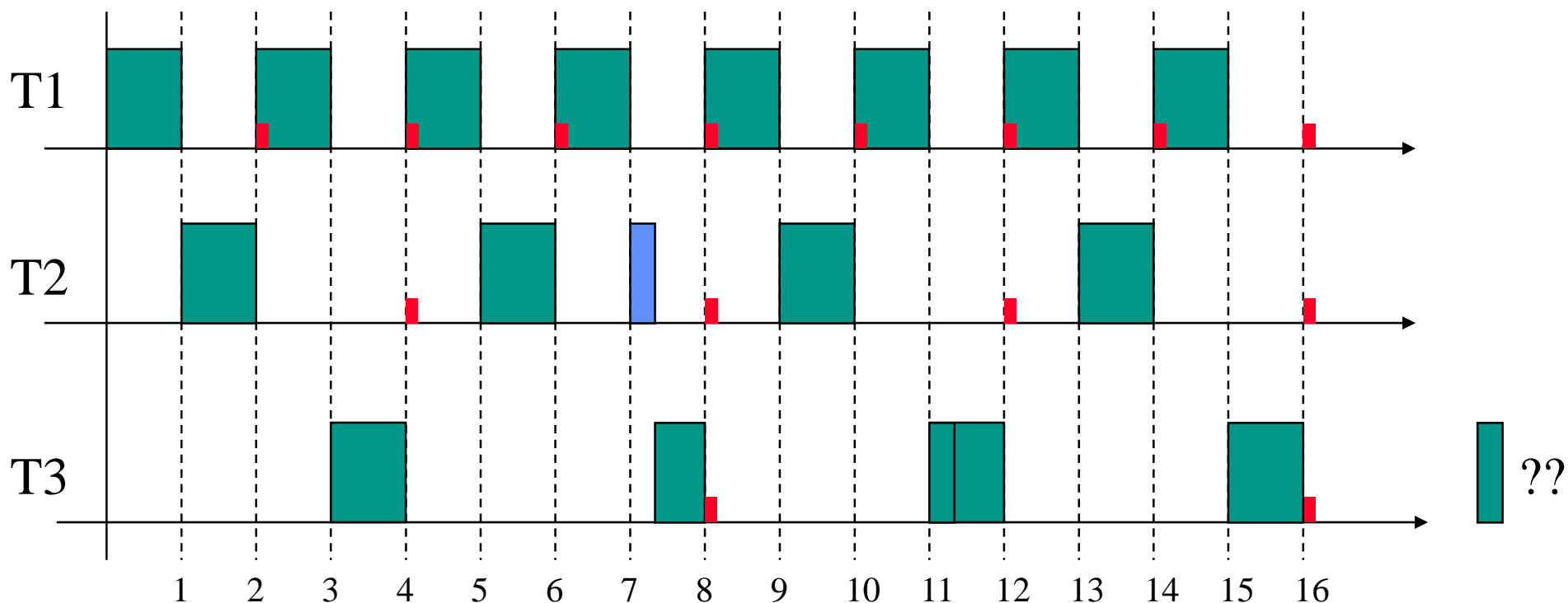
Questão Prática: Sobrecarga EDF versus RM 5/6

- T1: C1=1 P1=D1=2
- T2: C2=1 P2=D2=4
- T3: C3=2 P3=D3=8
- Execução normal com RM



Questão Prática: Sobrecarga EDF versus RM 6/6

- T1: C1=1 P1=D1=2
- T2: C2=1 P2=D2=4
- T3: C3=2 P3=D3=8
- Sobrecarga devido a falha do projeto com RM



Questões Práticas Prioridades Variáveis versus Fixas – Comentários

- Implementação é mais complexa do que com prioridade fixa
 - Requer um kernel que aceita prioridades variáveis
 - Precisa recalcular o deadline absoluto para cada job
- *Overhead* de execução pode ser elevado caso seja necessária reordenação dinâmica da fila de aptos (depende do algoritmo)
- Instabilidade face a sobrecargas
 - Não é possível saber antecipadamente quais tarefas vão perder deadline
- Escalonabilidade é superior em EDF do que em prioridade fixa
 - Qualquer sistema escalonável com prioridade fixa também será escalonável com EDF
 - O contrário não é verdadeiro
- Entretanto, prioridade fixa é mais usado na prática

- Escalonamento em Sistemas de Tempo Real
- Prioridades Fixas
- Prioridades Variáveis
- Testes Baseados em Utilização
- Questões Práticas

