
Variabilidade dos Tempos de Resposta



Fundamentos dos Sistemas de Tempo Real

2ª Edição

Rômulo Silva de Oliveira

Edição do Autor, 2020

www.romulosilvadeoliveira.eng.br/livrotemporeal

**Por que o tempo de resposta
de uma tarefa varia ?**

- Uma fonte de variabilidade para o tempo de resposta de uma tarefa é o seu próprio tempo de execução
 - O qual tipicamente varia bastante
- Mesmo que seu tempo de execução fosse constante, seu tempo de resposta variaria

pois varia o quanto a tarefa em questão é atrapalhada pelas demais atividades do sistema cada vez que ela executa

- Comportamento temporal do SO
afeta o comportamento temporal da aplicação
 - Exemplo: tratador de interrupções do timer

- Uma tarefa de tempo real pode ser atrapalhada pelas demais atividades do sistema

Variabilidade Causada por Outras Tarefas 1/1

- Fonte de variabilidade mais óbvia:
execução de outras tarefas da aplicação
- Uma aplicação de tempo real complexa pode incluir
 - diversas malhas de controle com períodos diferentes
 - interface humano-máquina gráfica ou textual
 - emprego de arquivos de configuração e de histórico
 - detecção de alarmes
 - comunicação via rede com outros computadores ou equipamentos
- Tempo de espera na fila de aptos irá somar-se ao próprio tempo de execução da tarefa para compor o seu tempo de resposta

Variabilidade Causada por Outros Processos 1/1

- Em sistemas operacionais maiores podem existir processos do sistema
- São processos normais
 - executam fora do kernel
 - fazem chamadas de sistema
- Porém, eles não executam código de aplicação mas sim realizam tarefas de manutenção e administração do sistema
 - fazem parte do sistema operacional
- São aqueles processos que já existem antes da aplicação iniciar
 - Lista de tarefas no Windows
 - ps -a no Linux

Variabilidade Causada por Threads do Kernel 1/1

- São empregadas threads dentro do kernel na implementação de funções
 - Por exemplo *device-drivers* associados com periféricos
 - Manutenção da gerência de memória
 - Gerência de sistema de arquivos
 - Etc
- Estas threads executam apenas código do kernel
 - possuem privilégios no acesso aos controladores de periféricos
 - são acionadas para atender as chamadas de sistema
- Na perspectiva do escalonador, são threads que competem pelo processador com as demais threads que existem no sistema
- Tipicamente executam com prioridade sobre as threads da aplicação

Variabilidade Causada por Tratadores de Interrupção 1/1

- A qualquer momento a thread de aplicação pode ser suspensa para a execução de um tratador de interrupção
 - Teclado, ethernet, etc
- Talvez após o tratador de interrupção, a thread interrompida retome sua execução
- Ou seja substituída por uma thread liberada pelo tratador de interrupções
- Interrupções são uma importante fonte de variabilidade no tempo de resposta
- Mesmo tarefa implementada como tratador de interrupções pode ser atrapalhada por outros tratadores de interrupções
 - Muitos tratadores de interrupções desabilitam as interrupções no computador durante pelo menos parte de sua execução

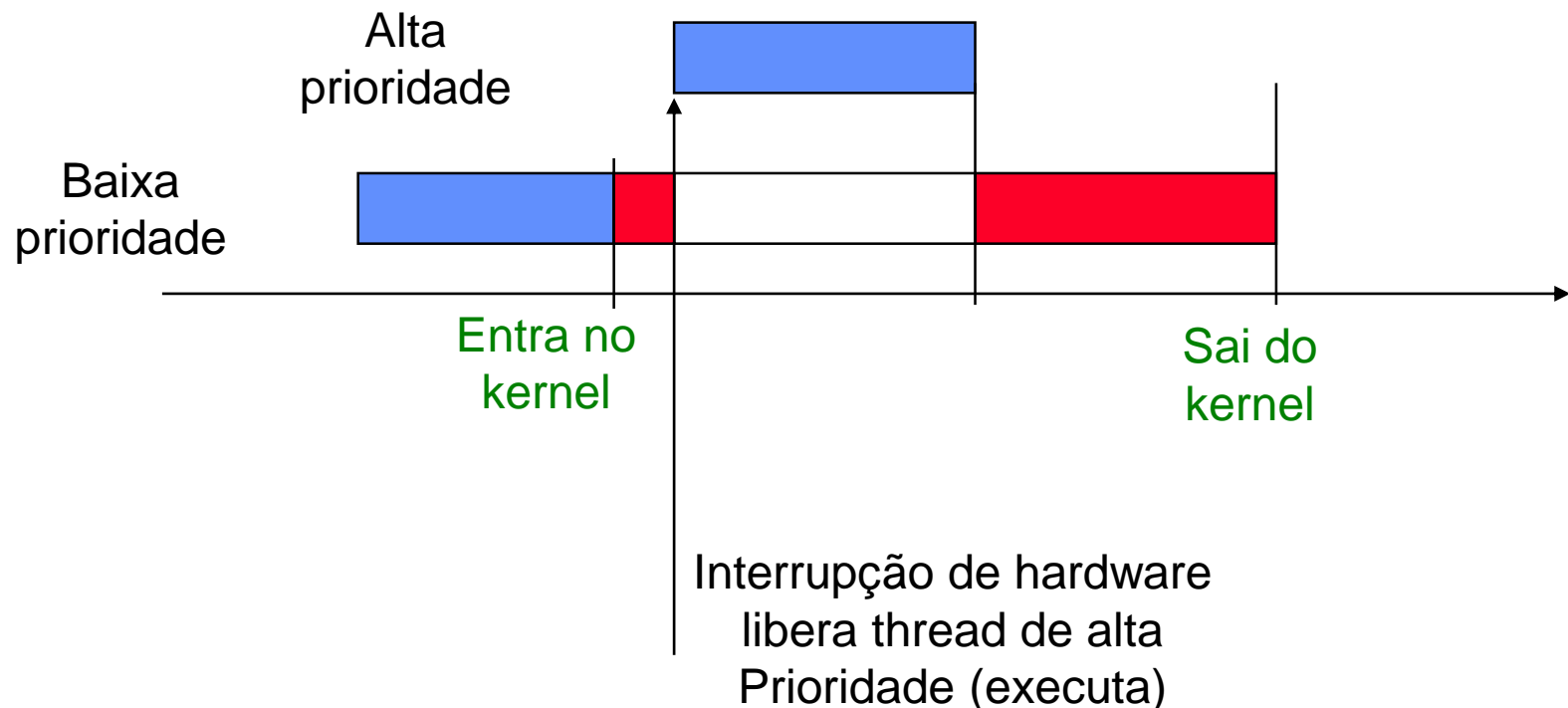
Variabilidade Causada p/Manipulação de Prioridade 1/1

- Muitos sistemas operacionais manipulam por conta própria as prioridades das tarefas
- Por exemplo, o mecanismo de envelhecimento (*aging*)
- Ou reduzem automaticamente a prioridade de uma thread na medida que ela consome tempo de processador

- Esses mecanismos fazem sentido em um SOPG
 - quando o objetivo é estabelecer um certo grau de justiça
- No contexto de tempo real:
preocupação é com o atendimento dos deadlines
- Aumentam a variabilidade dos tempos de resposta

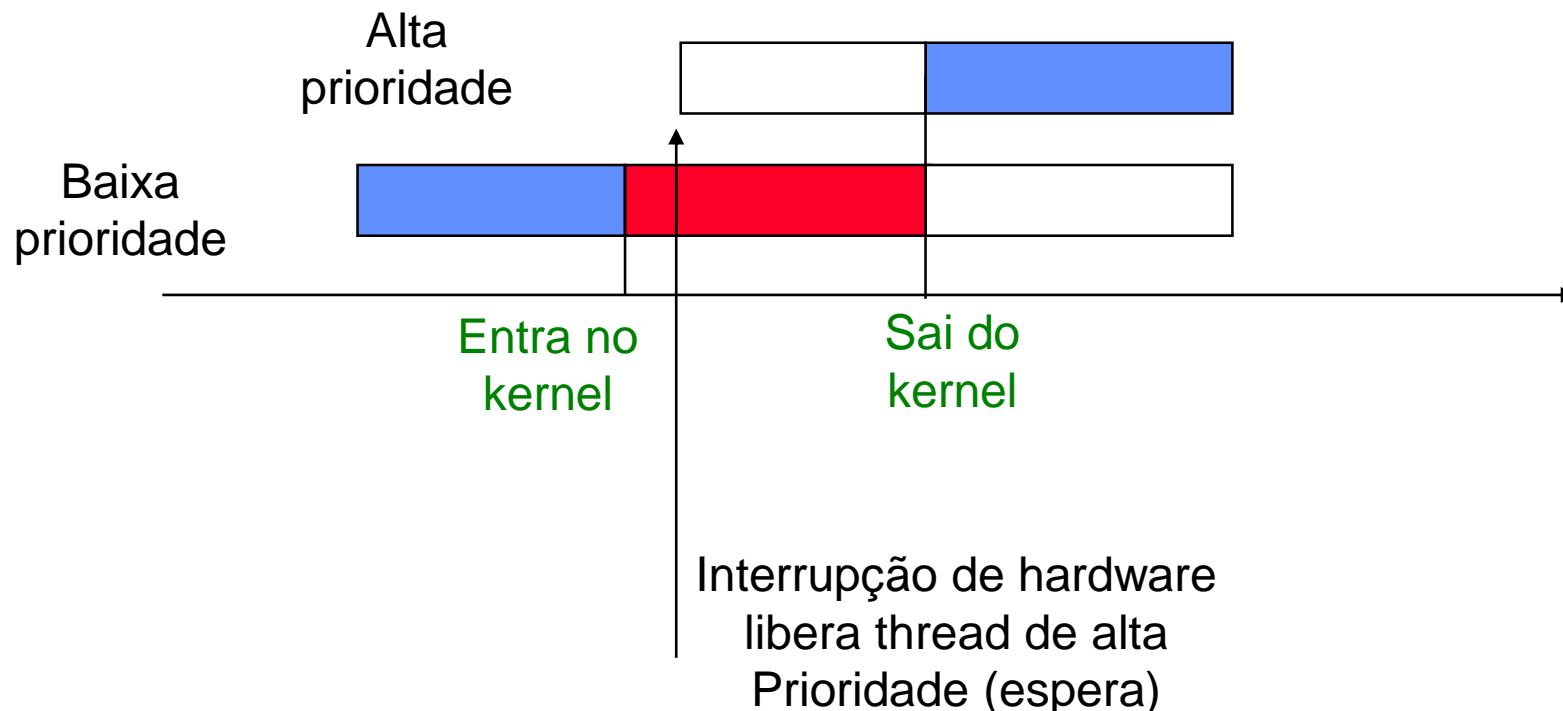
Variabilidade Causada por Kernel Não-Preemptivo 1/2

- **Kernel preemptivo** (*preemptive kernel*)
- Escalonador irá respeitar a ordem das prioridades
- A chamada de sistema da thread de baixa prioridade será suspensa e a thread de alta prioridade passará a executar



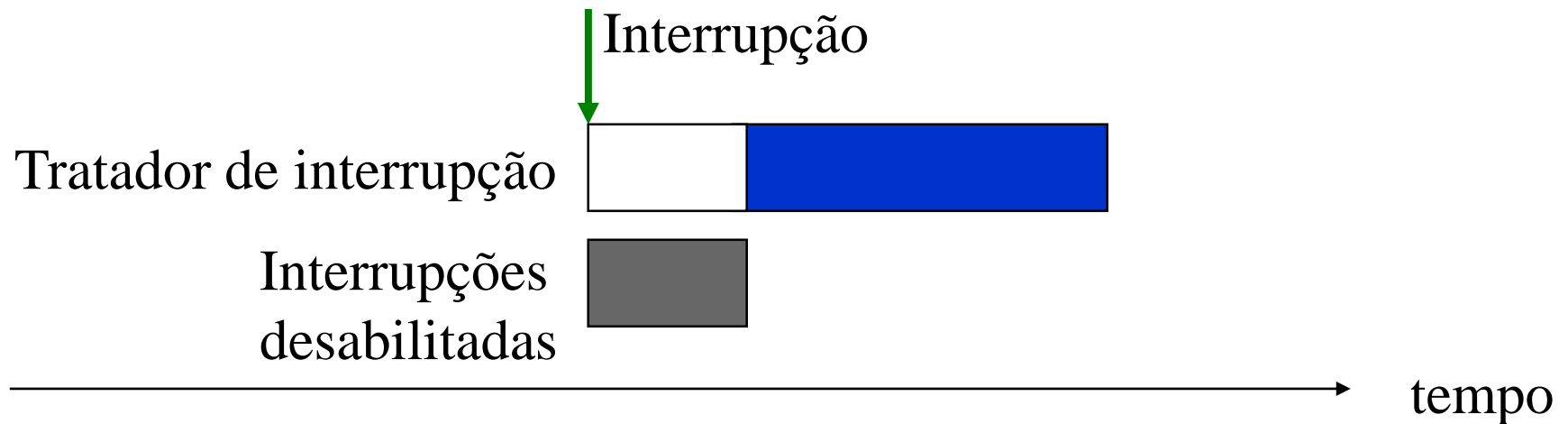
Variabilidade Causada por Kernel Não-Preemptivo 2/2

- **Kernel não preemptivo** (*non-preemptive kernel*), a execução de uma chamada de sistema jamais é suspensa para a execução de outra thread
- Thread de alta prioridade terá que esperar até que a chamada de sistema da thread de baixa prioridade termine, para poder executar
 - Inversão de prioridades, tempo de resposta da tarefa de alta prioridade irá variar



Variabilidade Causada por Desabilitar Interrupções 1/1

- Tempo entre a sinalização de uma interrupção no hardware e o início da execução de seu tratador varia
- Interrupções podem estar desabilitadas
- Tempo para executar o tratador e possivelmente liberar uma thread de alta prioridade varia

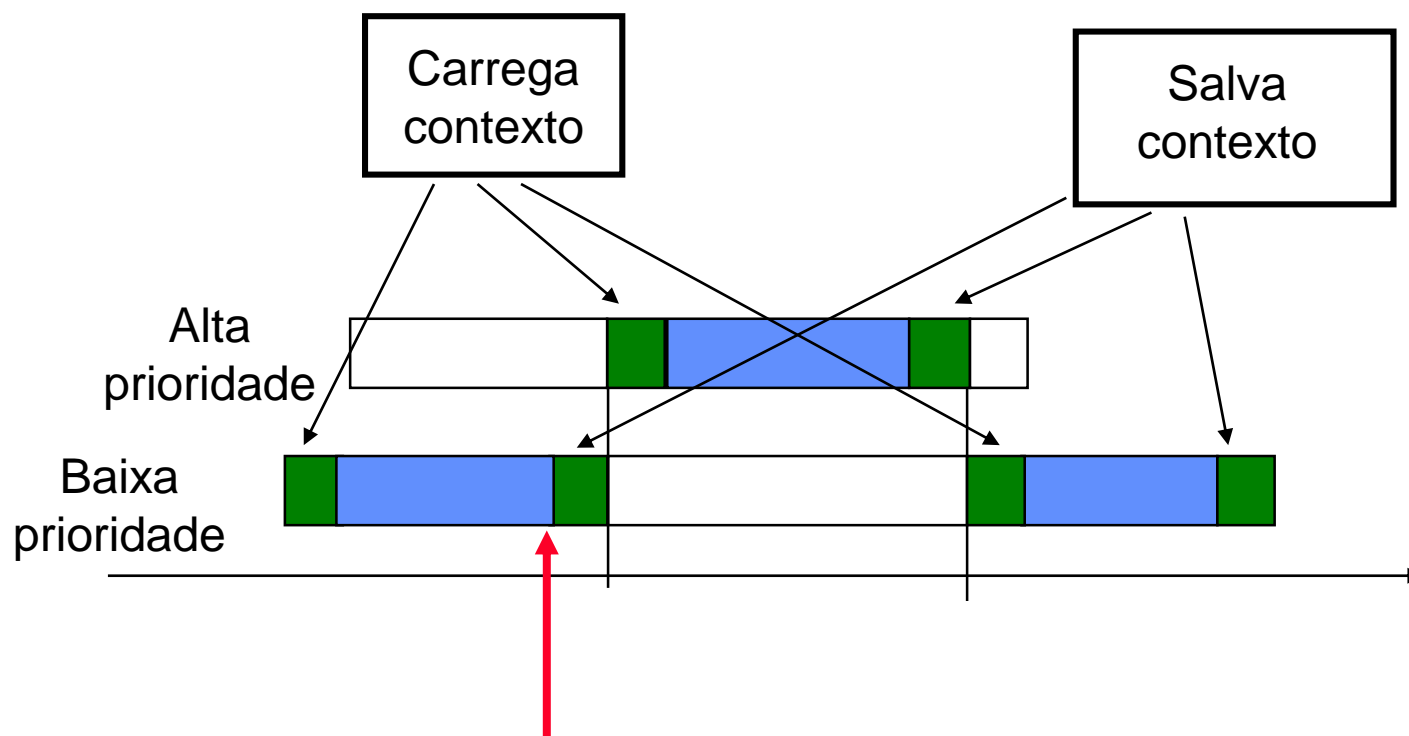


Variabilidade Causada por Overhead do Kernel 1/3

- Diversas atividades do sistema operacional demandam tempo de processamento
- Tempo de processador gasto com atividades de gerência do sistema e não com código de aplicação é normalmente chamado de *overhead*
- Tradicionalmente são utilizadas listas encadeadas na implementação de filas dentro do kernel
 - Introduz overhead variado
 - Minimizado através da substituição da tradicional implementação da fila por estruturas de dados mais eficientes

Variabilidade Causada por Overhead do Kernel 2/3

- Chaveamento de contexto deve ser considerado
- Tempo de chaveamento é somado ao tempo de resposta de cada tarefa
- O número de chaveamento de contexto é variado

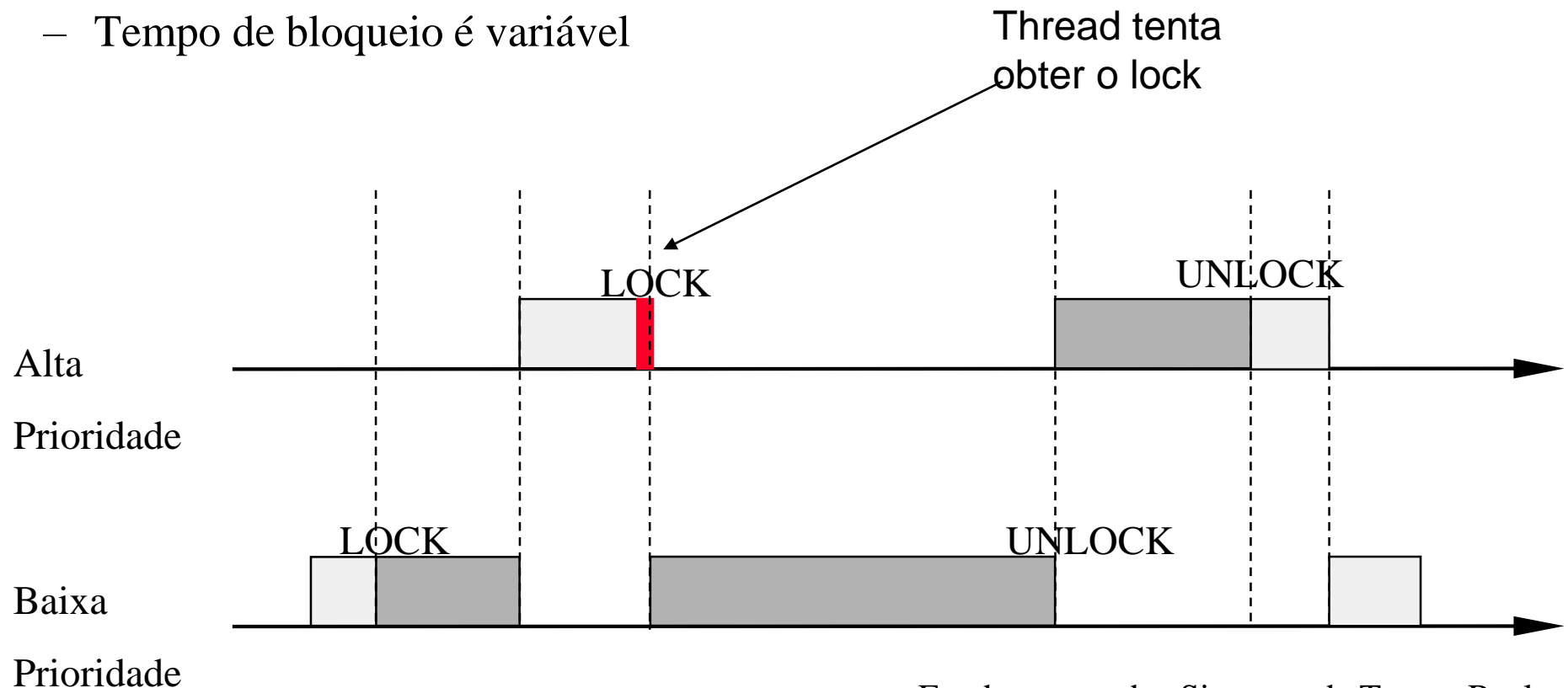


Variabilidade Causada por Overhead do Kernel 3/3

- Processador é apenas um recurso do sistema
- Memória, periféricos, controladores, servidores também são escalonados e geram filas de tarefas
- **Todas as filas** do sistema geram atrasos
 - Atraso depende da disputa em torno do recurso
 - Atraso que uma tarefa específica sofre é variável
 - Principalmente recursos acessados por ordem de chegada

Variabilidade Causada por Sincronização na Aplicação 1/1

- Quando duas ou mais threads acessam concorrentemente a mesma variável compartilhada existe a possibilidade de erros
 - Deve-se empregar métodos para proteger as seções críticas
- Mecanismo **mutex** oferece primitivas **LOCK** e **UNLOCK**
 - Tempo de bloqueio é variável

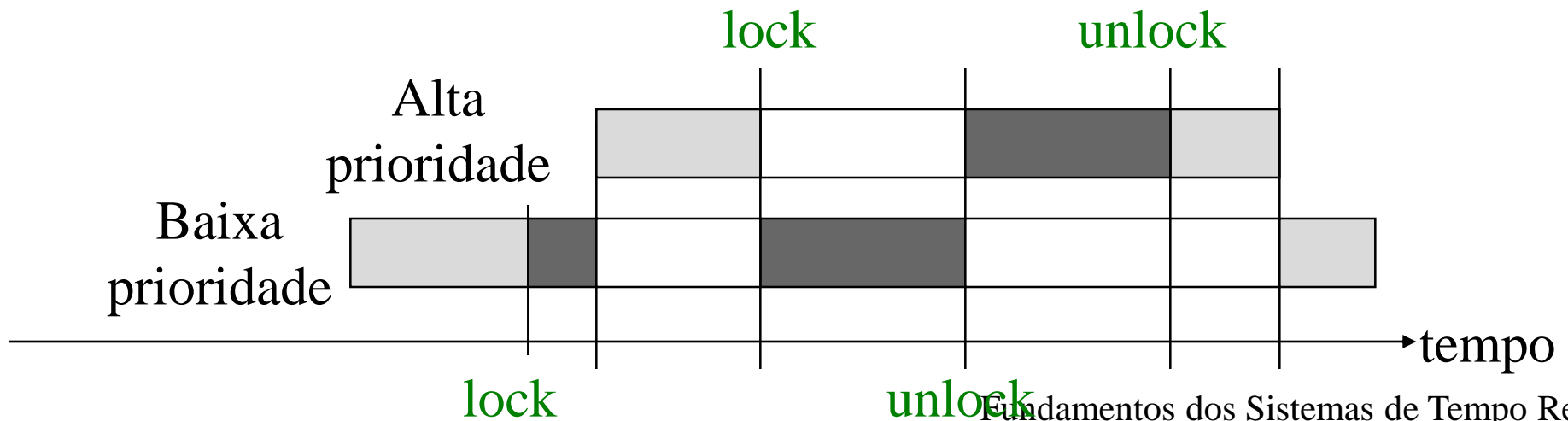
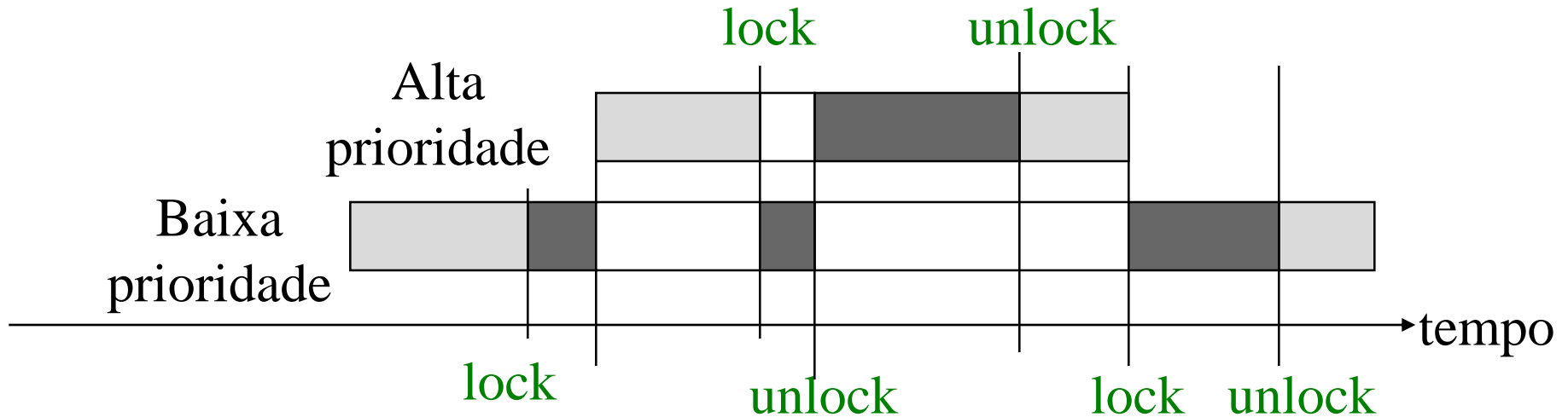


Variabilidade Causada por Sincronização no Kernel 1/2

- Código do kernel também tem sincronização
 - Mutex, etc
- Tempo de bloqueio é variável
 - Afeta tarefas do sistema
- Uma granularidade fina para as seções críticas dentro do kernel embora aumente a complexidade do código reduz o tempo de bloqueio em potencial das threads

Variabilidade Causada por Sincronização no Kernel 2/2

- Granularidade é importante



Determinação do Tempo de Resposta no Pior Caso 1/4

- Caso a especificação determine um deadline relativo
- O importante é conhecer qual o **tempo de resposta no pior caso** (**WCRT – *Worst-Case Response Time***) da tarefa
- A determinação do tempo de resposta no pior caso deve obviamente considerar como o sistema e a tarefa foram implementados

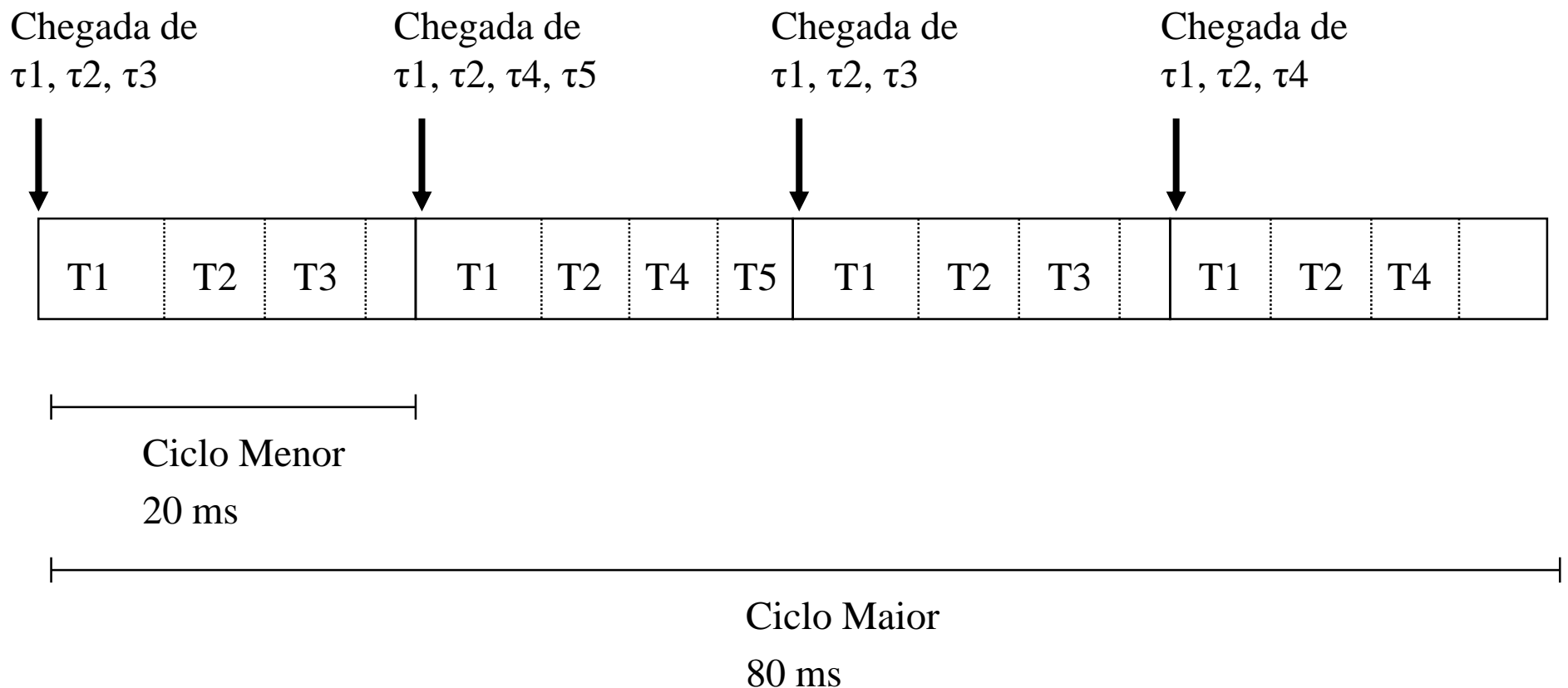
Determinação do Tempo de Resposta no Pior Caso 2/4

- No caso do sistema ser implementado como um executivo cíclico o trabalho é facilitado
- A escala de execução do sistema se repete a cada ciclo maior
- O tempo de resposta é definido como o intervalo de tempo entre a chegada da tarefa e sua conclusão

- Tempo de resposta no pior caso obtido da inspeção da tabela

Determinação do Tempo de Resposta no Pior Caso 3/4

- Tempo de resposta no pior caso obtido da inspeção da tabela



Determinação do Tempo de Resposta no Pior Caso 4/4

- Caso a tarefa de tempo real seja implementada como uma thread em um sistema operacional moderno
- Precisa considerar as fontes de variação descritas neste capítulo
- Podem ser usados
 - Métodos analíticos (modelos algébricos)
 - Medição
- Assunto para outro capítulo ...

- Introdução
- Fontes da Variabilidade do Tempo de Resposta
- Variabilidade Causada por Outras Tarefas da Aplicação
- Variabilidade Causada por Outros Processos do Sistema
- Variabilidade Causada por Threads do Kernel
- Variabilidade Causada por Tratadores de Interrupção
- Variabilidade Causada pela Manipulação de Prioridade dos Processos e Threads
- Variabilidade Causada por Kernel Não-Preemptivo
- Variabilidade Causada por Desabilitar Interrupções
- Variabilidade Causada por *Overhead* do Kernel
- Variabilidade Causada por Sincronização na Aplicação
- Variabilidade Causada por Sincronização no Kernel
- Determinação do Tempo de Resposta no Pior Caso

