Fatec
Araras

Servidores e seus sistemas operacionais

Prof. Orlando Saraiva Júnior
orlando.saraiva@unesp.br

Processos

Prof. Me. Orlando Saraiva Júnior

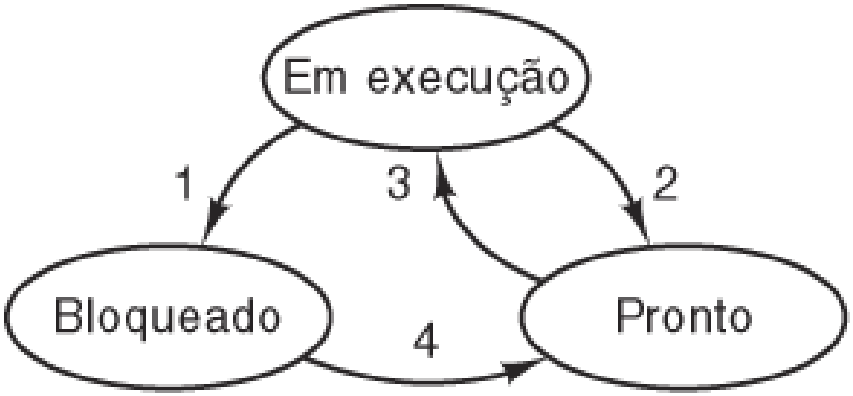
Processos
Estados de um processo

Em execução: realmente usando a CPU naquele instante

Pronto: Executável, temporariamente parado para dar lugar a outro processo

Bloqueado: incapaz de executar enquanto um evento externo não acontecer

Processos
Estados de um processo



1. O processo bloqueia aguardando uma entrada
2. O escalonador seleciona outro processo
3. O escalonador seleciona esse processo
4. A entrada torna-se disponível

O sistema operacional mantem uma tabela (vetor de estruturas) chamada tabela de processos.

Deadlock

Prof. Me. Orlando Saraiva Júnior

Deadlock

Fatec Araras

Deadlock

Fatec Araras

Deadlock
Definição

Fatec Araras

Deadlock: podem ocorrer quando vários processos recebem direitos de acesso exclusivo a recursos da máquina.

Recursos podem ser dispositivos de hardware (acesso ao pendrive ou disco rígido, por exemplo) ou informação (tabela ou registro do banco de dados ou arquivo aberto, por exemplo)

Recursos podem ser **preemptíveis** ou **não preemptíveis**.

Deadlock
Recursos

Recursos **preemptíveis** são aqueles que podem ser tirados do processo proprietário sem nenhum prejuízo.
Exemplo: Processo em memória ser armazenado em disco.

Recursos **não preemptíveis** são aqueles que não podem ser tirados do processo proprietários. Caso isto ocorra, acontecerá um erro.

Exemplo: Processo gravando um CD, caso o recurso gravador de CD seja passado a outro processo, gerará um CD com erros.

Deadlock
Recursos

A sequência de eventos necessários ao uso de um determinado recurso é:

1) Requisitar o recurso;
2) Usar o recurso;
3) Liberar o recurso;

Caso o recurso não estiver disponível quando requisitado, o processo solicitante será forçado a esperar.

Deadlock
Definição

Fatec Araras

“Um conjunto de processos estará em situação de deadlock se todo processo pertencente ao conjunto que estiver esperando por um evento que somente um outro processo desse mesmo conjunto poderá fazer acontecer.”

Para este modelo, consideramos que os processos possuem apenas um thread e que não existem interrupções possíveis para acordar o processo.

Deadlock
Definição

Segundo Coffman(*), há quatro condições para que ocorra o deadlock:

1) Condição de exclusão mútua: Em um determinado instante, cada recurso está em uma de duas condições: ou associado a um processo, ou disponível;

2) Condição de posse e espera. Processos que, em determinado instante, retêm recursos concedidos anteriormente podem requisitar novos recursos.

(*) Coffman, E.G., Elphick, M.J. E Shoshani,A. “System Deadlocks”, Computing Surveys, jun 1971, vol 3. p. 67-68

Deadlock
Definição

3) Condição de não preempção: Recursos concedidos previamente a um processo não podem ser forçosamente tomados deste processo. Eles devem ser ser explicitamente liberados pelo processo que os retém;

4) Condição de Espera circular. Deve existir um encadeamento circular de dois ou mais processos; cada um deles encontra-se à espera de um recurso que está sendo usado pelo membro seguinte dessa cadeia.

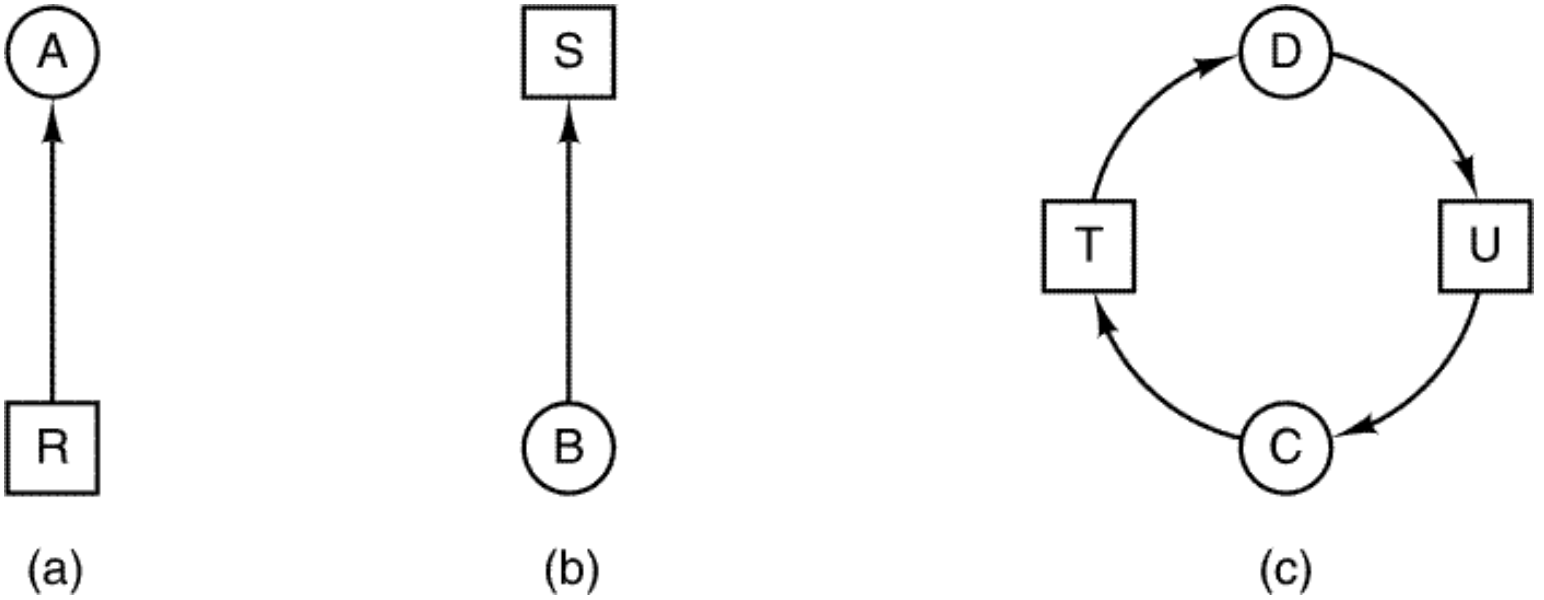
Todas estas quatro condições devem estar presentes para que o deadlock ocorra.

Deadlock
Modelagem

Fatec Araras

Essas quatro condições podem ser modeladas a partir do uso de grafos dirigidos. Esses grafos possuem dois tipos de nodos: processos (simbolizados com círculos) e recursos (simbolizados com quadrados).

Deadlock
Modelagem



(a) Processo A de posse do recurso R
(b) Processo B requisitando o recurso S
(c) *Deadlock*: Processo C de posse do recurso U e solicitando o recurso T, enquanto o processo D de posse do recurso T solicitando o recurso U.

Fatec Araras

Modelagem de *deadlocks:*

Três processos, A, B e C;
Três recursos, R, S e T;

Processo A:	**Processo B:**	**Processo C:**
Requisita R	Requisita S	Requisita T
Requisita S	Requisita T	Requisita R
Libera R	Libera S	Libera T
Libera S	Libera T	Libera R

Deadlock
Modelagem

Fatec Araras

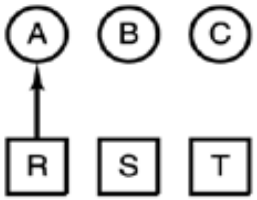
A
Requisita R
Requisita S
Libera R
Libera S
(a)

B
Requisita S
Requisita T
Libera S
Libera T
(b)

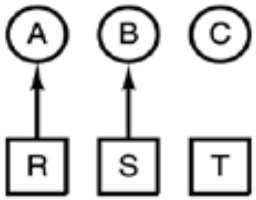
C
Requisita T
Requisita R
Libera T
Libera R
(c)

1. A requisita R
2. B requisita S
3. C requisita T
4. A requisita S
5. B requisita T
6. C requisita R
   deadlock

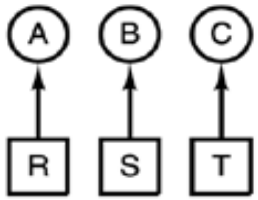
(d)



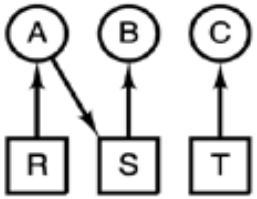
(e)



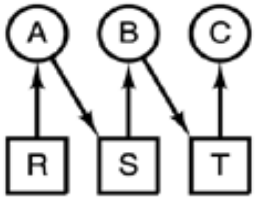
(f)



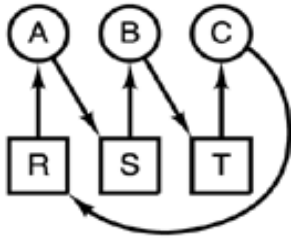
(g)



(h)



(i)



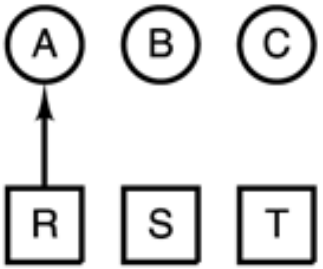
(j)

Deadlock

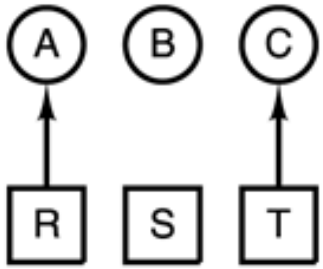
1. A requisita R
2. C requisita T
3. A requisita S
4. C requisita R
5. A libera R
6. A libera S

nenhum deadlock

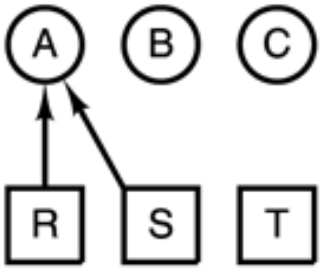
(k)



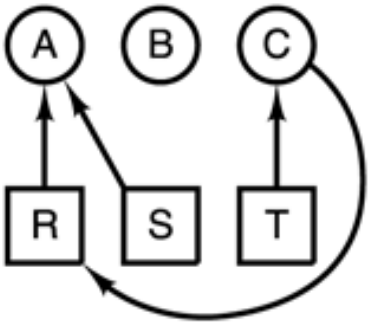
(l)



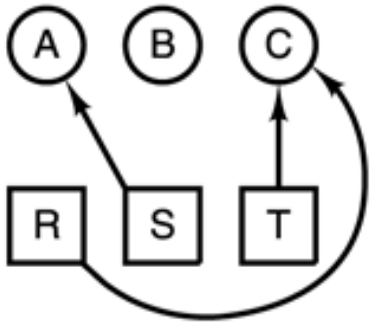
(m)



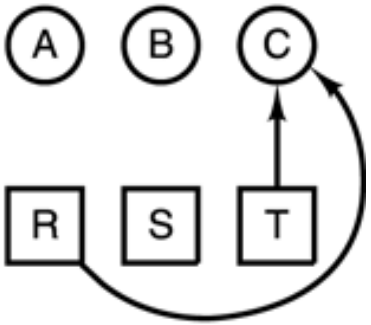
(n)



(o)



(p)



(q)

Deadlock

O sistema operacional não é obrigado a colocar os processos em ordem de execução especial.

Se o sistema operacional soubesse do deadlock iminente, poderia bloquear o processo B, até que a solicitação possa ser atendida com segurança.

Deadlock

O sistema operacional não é obrigado a colocar os processos em ordem de execução especial.

Se o sistema operacional soubesse do deadlock iminente, poderia bloquear o processo B, até que a solicitação possa ser atendida com segurança.

Comunicação Inter Processos
IPCs

Prof. Me. Orlando Saraiva Júnior

IPCs

Fatec Araras

Normalmente, processos precisam se comunicar com outros processos. Esta comunicação precisa ser feita de forma estruturada.

Mas lembre-se !

Com a alternância da CPU entre os processos, a taxa que o processo realiza sua computação não é uniforme e provavelmente não reproduzível.
Processos não podem ser programados com hipóteses predefinidas de temporização.

Race Conditions

Fatec Araras

Uma condição de disputa ocorre quando duas ou mais threads (ou processos) necessitam efetuar operações em uma mesma área de memória, mas os resultados destas operações dependem da ordem em que essas operações são executadas.

Processos não podem ser programados com hipóteses predefinidas de temporização.

Race Conditions

Fatec Araras

Enquanto o código-fonte pode aparecer para nós na tela do computador na ordem em que devem ser executadas, as *threads* podem ser escalonadas e executadas em ordem aleatória, além de seus códigos terem tamanhos diferentes. Além disso, as *threads* manipulam as mesmas variáveis porque compartilham uma mesma área do processo. Com isso, podem gerar resultados inesperados.

Essa situação, na qual vários *threads* acessam e manipulam os mesmos dados concorrentemente e na qual o resultado da execução depende da ordem específica em que o acesso ocorre, é chamada de condição de disputa.

Exclusão mútua

Exclusão mútua é um dos meios de garantir o controle de acesso a um recurso compartilhado.
A região crítica (que gera condição de disputa) é acessada por uma única thread em um mesmo instante, ou seja, quando uma thread está executando seu código na região crítica, nenhuma outra thread pode acessar a mesma região.

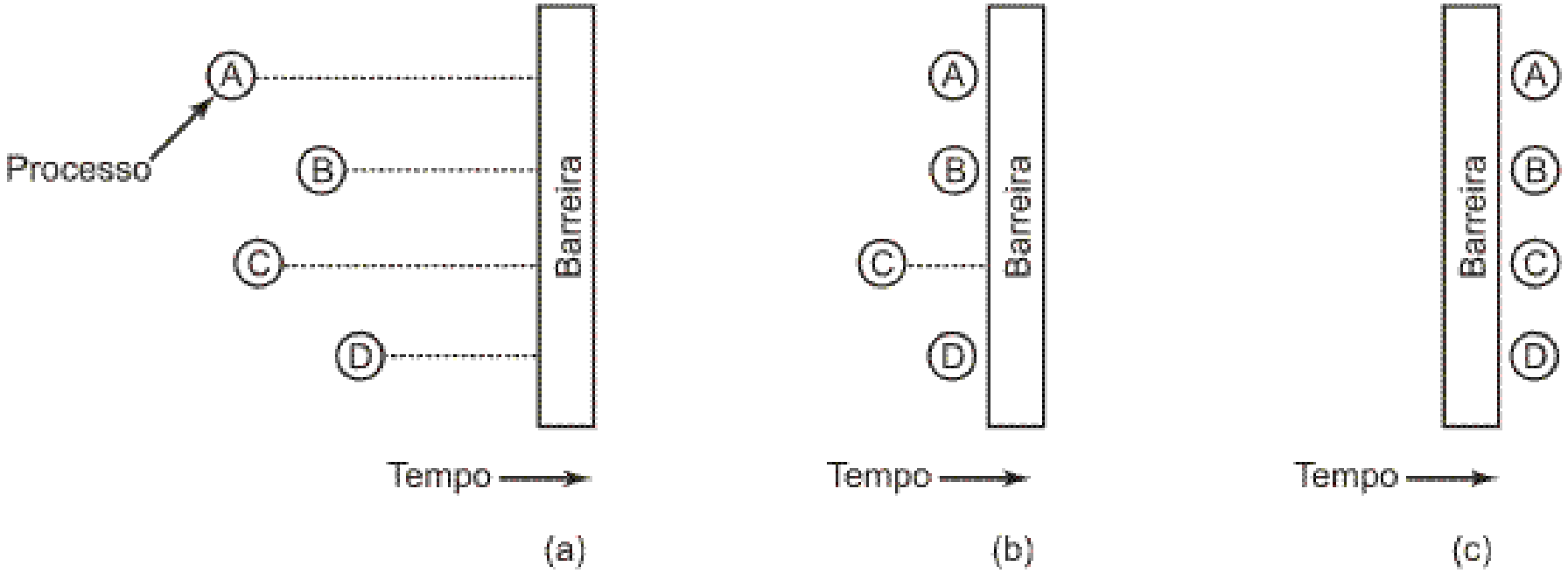
Exclusão mútua

Quatro condições necessárias para prover exclusão mútua:

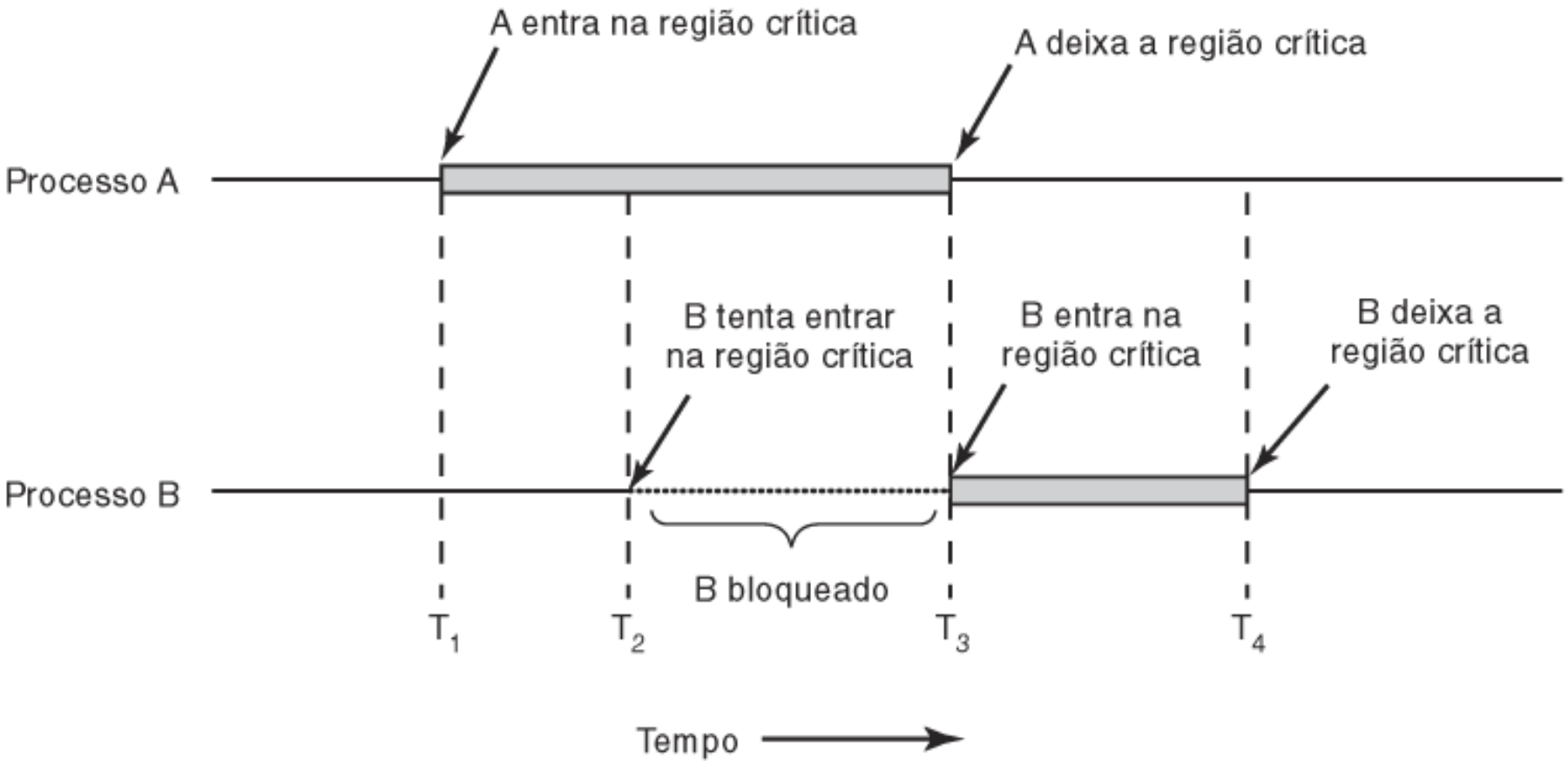
- Nunca dois processos simultaneamente em uma região crítica;
- Nenhuma afirmação sobre velocidades ou números de CPUs;
- Nenhum processo executando fora de sua região crítica pode bloquear outros processos;
- Nenhum processo deve esperar eternamente para entrar em sua região crítica;

Barreira

Aplicações divididas em fase e tem como regra que nenhum processo pode avançar para a próxima fase até que todos os processos estejam prontos;



Exclusão mútua



Problemas Clássicos de IPCs

Prof. Me. Orlando Saraiva Júnior

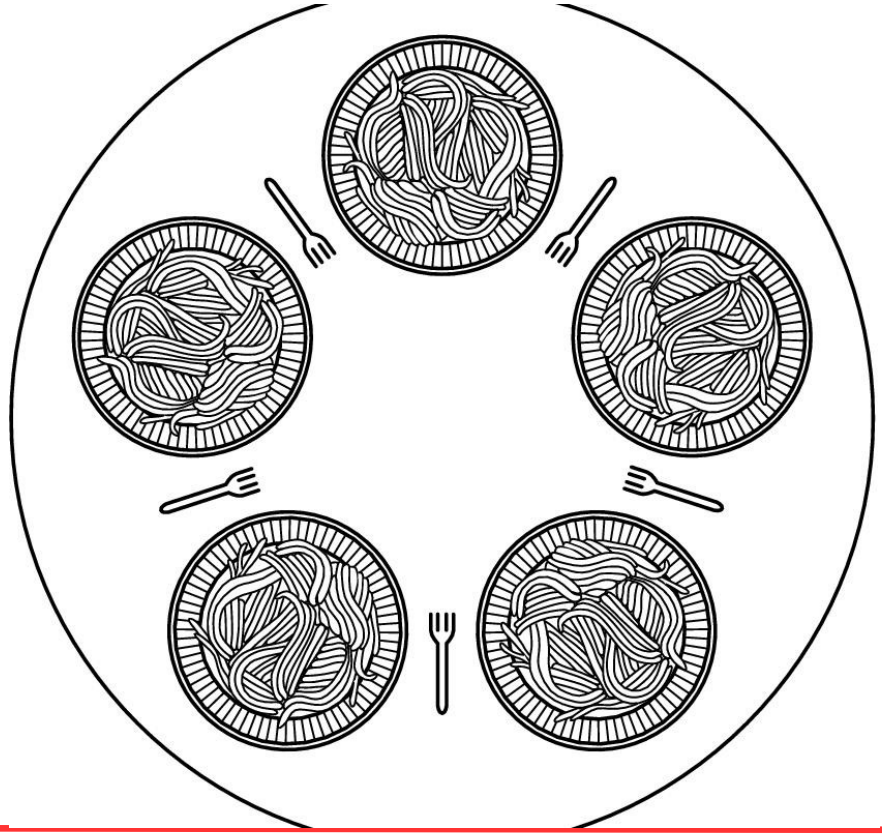
Jantar dos Filósofos

Fatec Araras

Filósofos comem/pensam

Cada um precisa de dois garfos para comer

Pega um garfo por vez

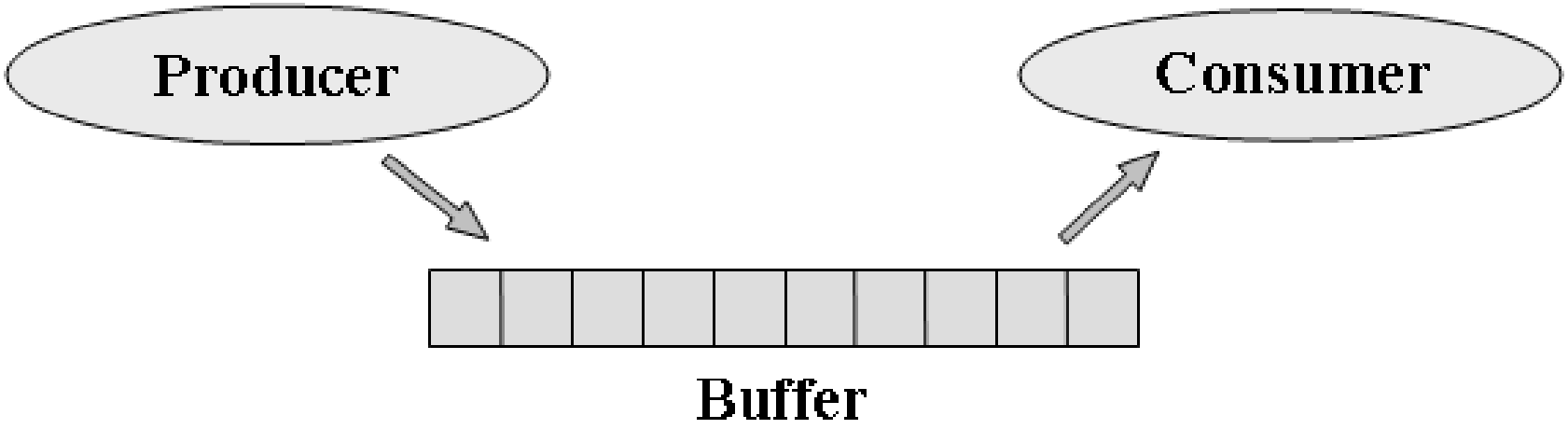


Produtor / Consumidor

Fatec Araras

Dois processos compartilham um buffer de tamanho fixo

- O produtor insere informação no buffer
- O consumidor remove informação do buffer

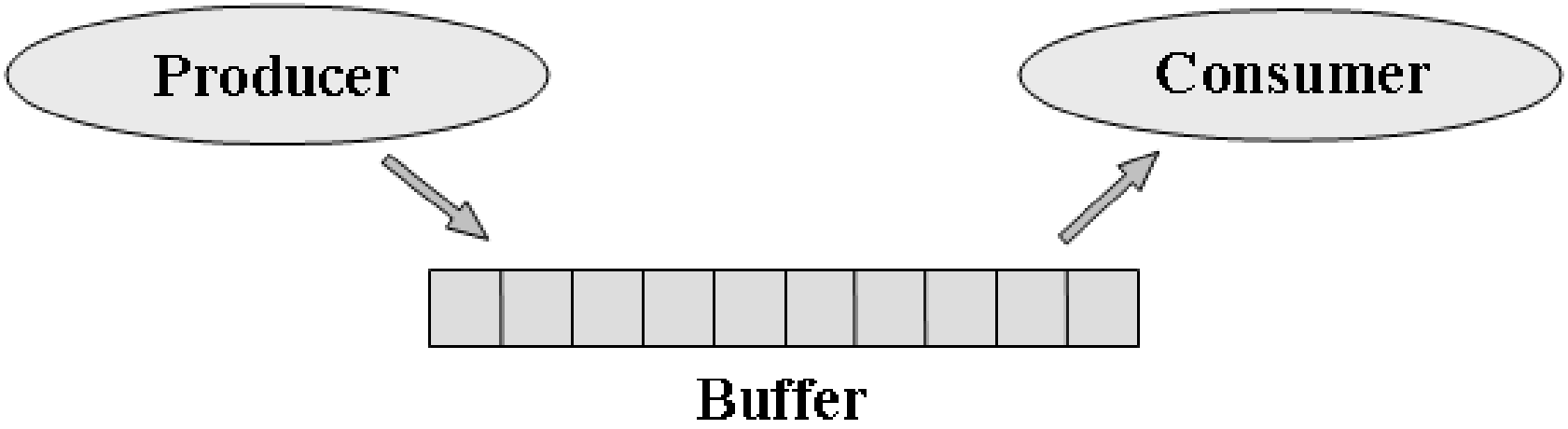


Produtor / Consumidor

Fatec Araras

Dois processos compartilham um buffer de tamanho fixo

- O produtor insere informação no buffer
- O consumidor remove informação do buffer



O Barbeiro sonolento

O Barbeiro sonolento

O barbeiro sonolento dorme toda vez que seu salão está vazio, mas assim que alguém chega, o barbeiro volta ao trabalho.

Desse modo, ele não "perde tempo" acordado.

Em seu seu salão cabem até N clientes na fila de espera. Os clientes que chegam e vêem que o salão está cheio vão embora.

Escalonamento

Prof. Me. Orlando Saraiva Júnior

Escalonamento
de processos

Quando dois processos estão no estado pronto, alguém deverá escolher qual será o próximo processo a executar;

A parte do Sistema Operacional que faz esta escolha se chama **escalonador**, e o algoritmo que ele usa se chama **algoritmo de escalonamento**.

Escalonamento

Fatec Araras

Diferentes ambientes requer diferentes algoritmos de escalonamento;

Três ambientes merecem distinção:

- Lote;
- Interativo;
- Tempo Real;

Escalonamento
de processos

Fatec Araras

Todos os sistemas

Justiça — dar a cada processo uma porção justa da UCP
Aplicação da política — verificar se a política estabelecida é cumprida
Equilíbrio — manter ocupadas todas as partes do sistema

Sistemas em lote

Vazão (throughput) — maximizar o número de jobs por hora
Tempo de retorno — minimizar o tempo entre a submissão e o término
Utilização de UCP — manter a UCP ocupada o tempo todo

Sistemas interativos

Tempo de resposta — responder rapidamente às requisições
Proporcionalidade — satisfazer as expectativas dos usuários

Sistemas de tempo real

Cumprimento dos prazos — evitar a perda de dados
Previsibilidade — evitar a degradação da qualidade em sistemas multimídia

Fatec
Araras

Hora da prática

Prof. Orlando Saraiva Júnior
orlando.saraiva@unesp.br