

Sistemas Hardware-Software

Aula 22 - Modelos de Concorrência

2021 – Engenharia

[Igor Montagner](#), Fábio Ayres



Correção

Implementação de Barreiras

Barreiras

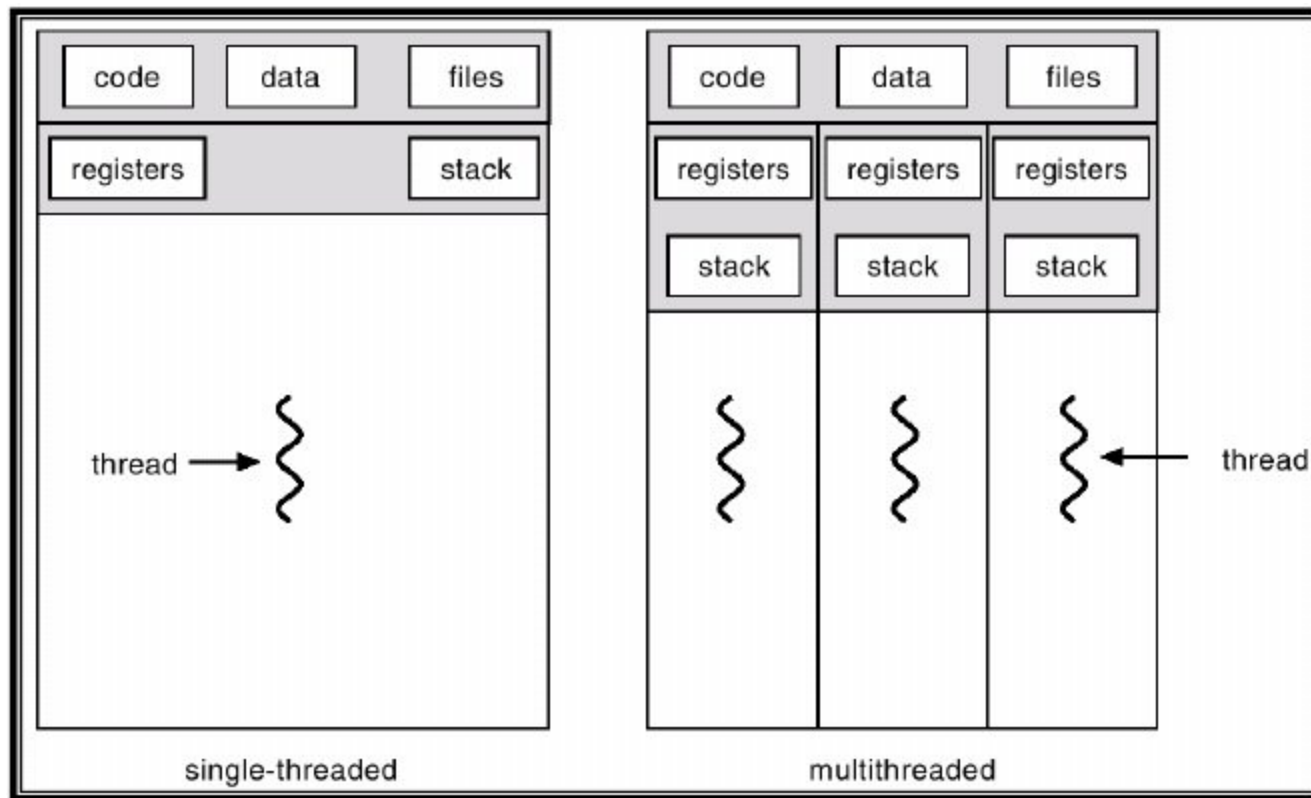
Dados compartilhados:

Mutex:

Semáforos:

Barreiras

Processos e threads



Conceito : Race Condition

"Ocorre quando a saída do programa depende da ordem de execução das threads"

Em geral ocorre quando

- uma variável é usada em mais de uma thread e há pelo menos uma operação de escrita.
- trabalhamos com os mesmos arquivos simultaneamente em várias threads

Conceito : Região Crítica

"Parte do programa que só pode ser rodada uma thread por vez"

- elimina situações de concorrência
- elimina também toda a concorrência e pode se tornar gargalo de desempenho

Mutex (Mutual Exclusion)

Primitiva de sincronização para criação de regiões de exclusão mútua

- Lock – se estiver destravado, trava e continua
 - se não espera até alguém destravar
- Unlock – se tiver a trava, destrava
 - se não tiver retorna erro

Semáforos

"Inteiro especial que nunca fica negativo"

Só pode ser manipulado por duas operações atômicas

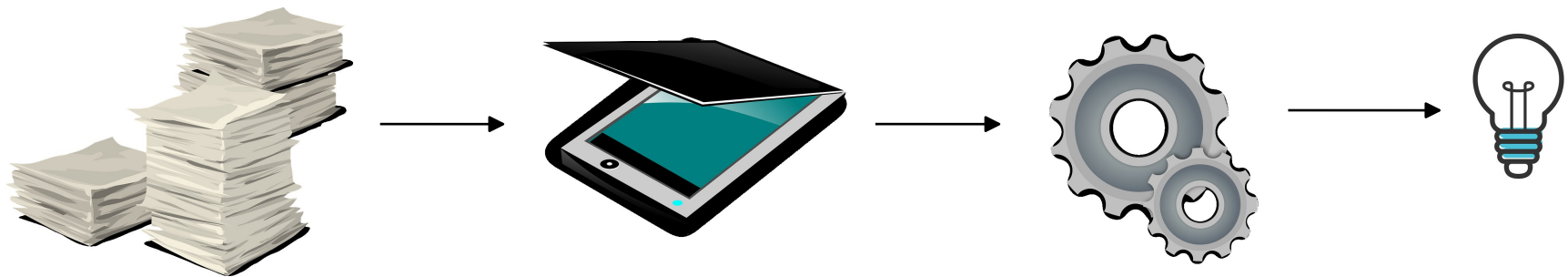
POST:

- Aumenta o valor em 1

WAIT:

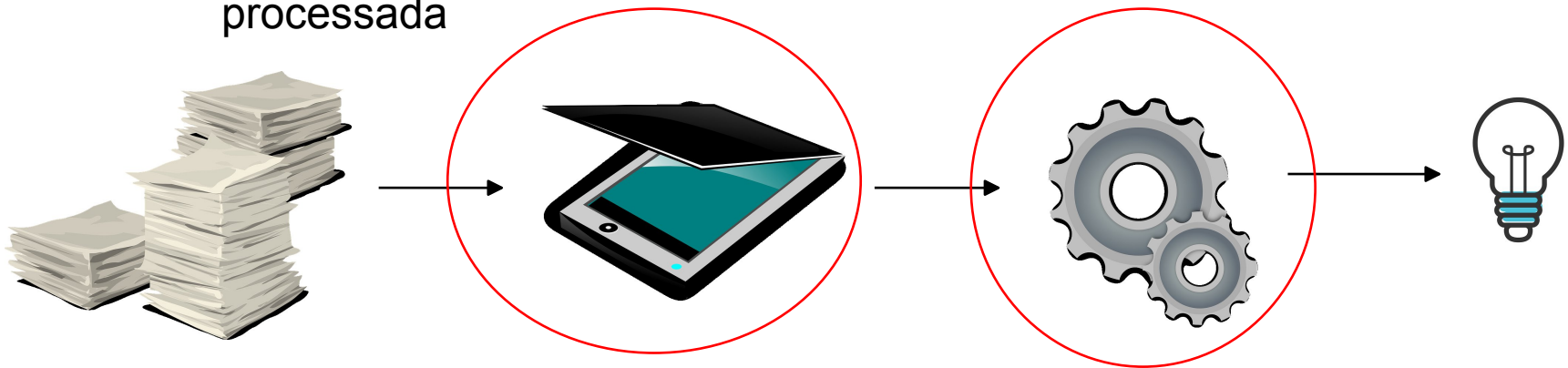
- Se for positivo, diminui em 1
- Se for 0 fica esperando;

Problema – leitura de informações



Exemplo 1 – produtor consumidor

Produtor: Escaneia e devolve imagem a ser processada



Sincronização

Consumidor:
transforma
imagem em informação

1. Consumidor: espera produtor enviar item
2. Produtor: cria item e avisa Consumidor

Exemplo 1 – produtor consumidor

Dois conjuntos de threads

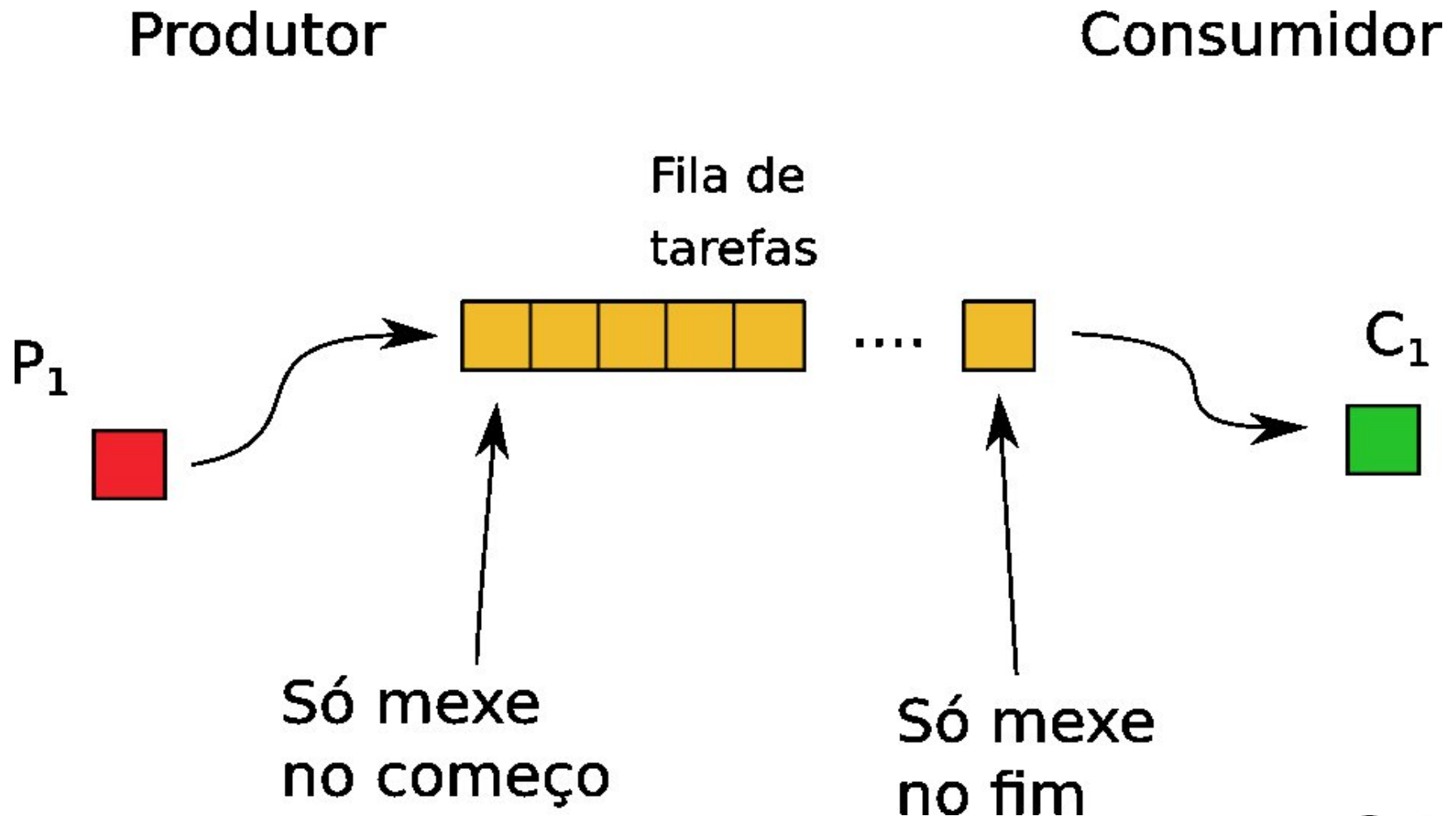
- Produzem tarefas a serem executadas

Pode depender de um recurso compartilhado
controlar tamanho das tarefas.

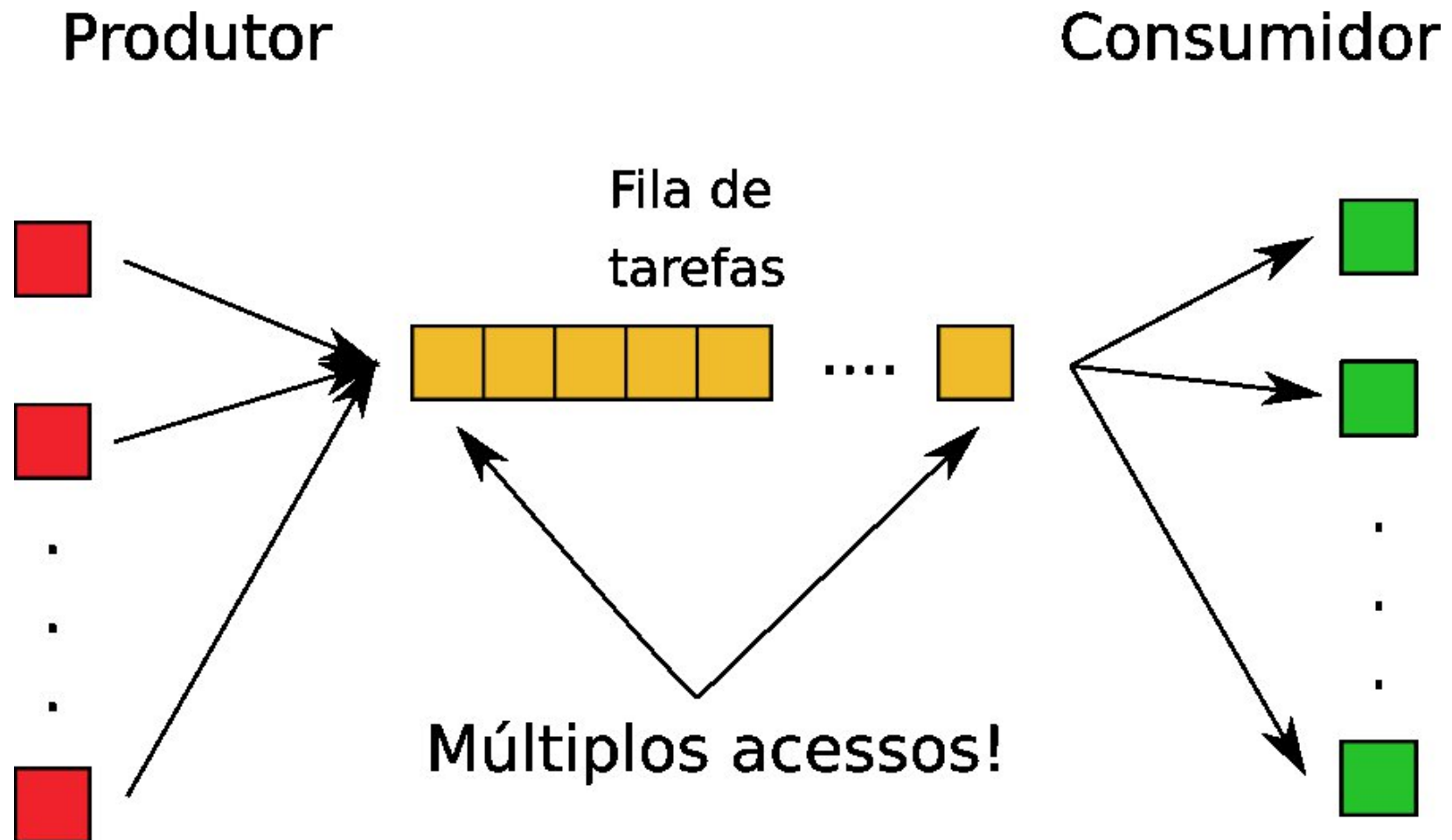
- Consomem as tarefas e as executam.

Cada consumidor não depende dos produtores nem de outros consumidores.

Exemplo 1 – produtor consumidor



Exemplo 1 – produtor consumidor



Produtor Consumidor

Dados compartilhados:

Mutex:

Semáforos:

Produtor Consumidor

Produtor

Consumidor



www.insper.edu.br