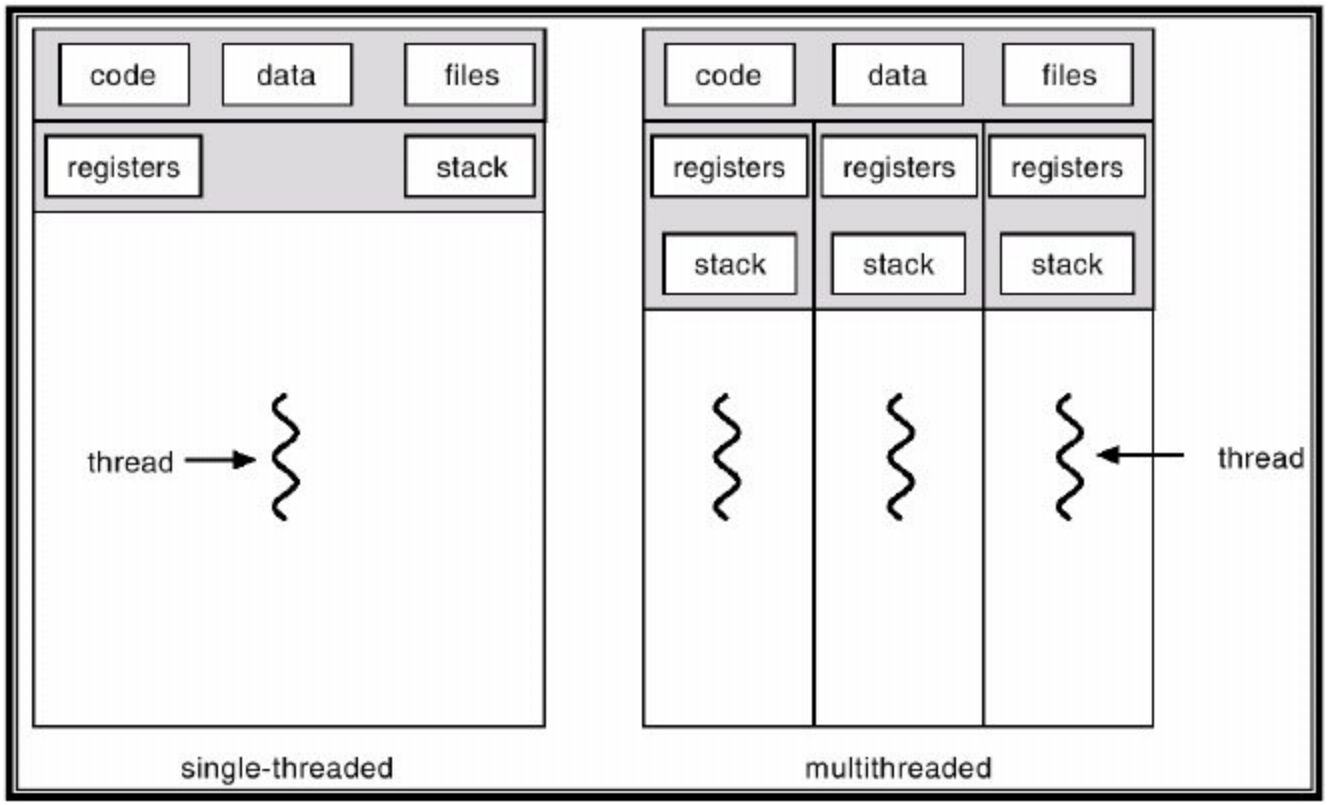
Insper

Sistemas Hardware-Software

Aula 21 - Semáforos II e Modelos de Concorrência

Engenharia
Fabio Lubacheski
Maciel C. Vidal
Igor Montagner
Fábio Ayres

Processos e threads



Insper

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

Mutex – Representação Conceitual

```
Lock(Mutex s)
{
  if( s = 1 )
     s = 0
  else
    “Bloqueia a thread”
}
```

```
Unlock(semaforo s)
{
     if(“existe uma thread
        bloqueada” )
        “desbloqueia a
         thread”
     else
        s = 1
}
```

Semáforos

"Inteiro especial que nunca fica negativo"

Só pode ser manipulado por duas operações atômicas

WAIT:

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

POST:

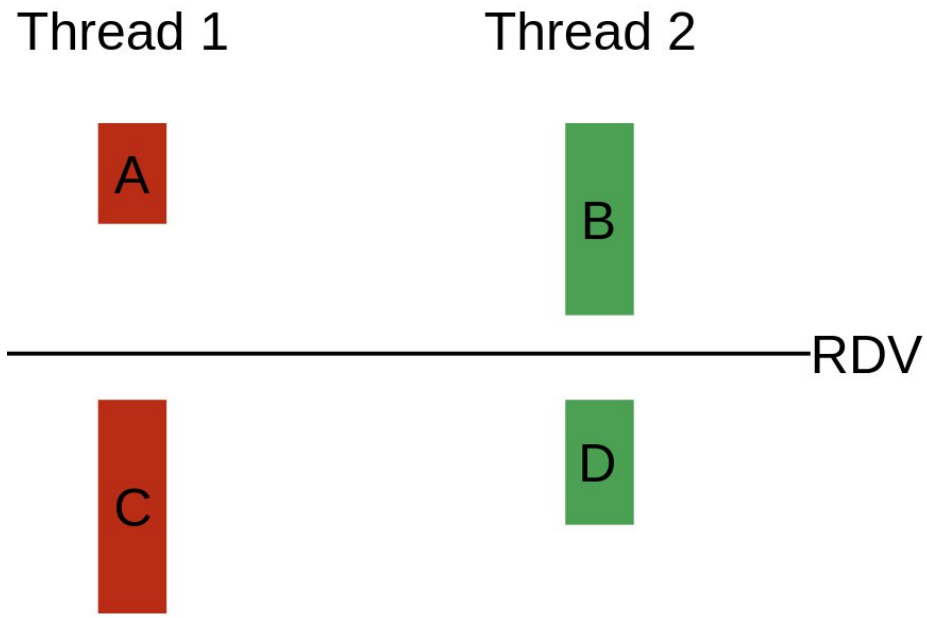
- Se tiver thread bloqueada libera a thread
- Aumenta o valor em 1

Semáforos – Representação Conceitual

```
Wait(semaforo s)
{
  if( s > 0 )
     s = s - 1
  else
    “Bloqueia a thread”
}
```

```
Post(semaforo s)
{
     if(“existe uma thread
        bloqueada” )
        “desbloqueia a
         thread”
     else
        s = s + 1
}
```

Relembrando RDV



Correção

Veja a correção do problema de RDV da aula passada usando semáforos

Correção

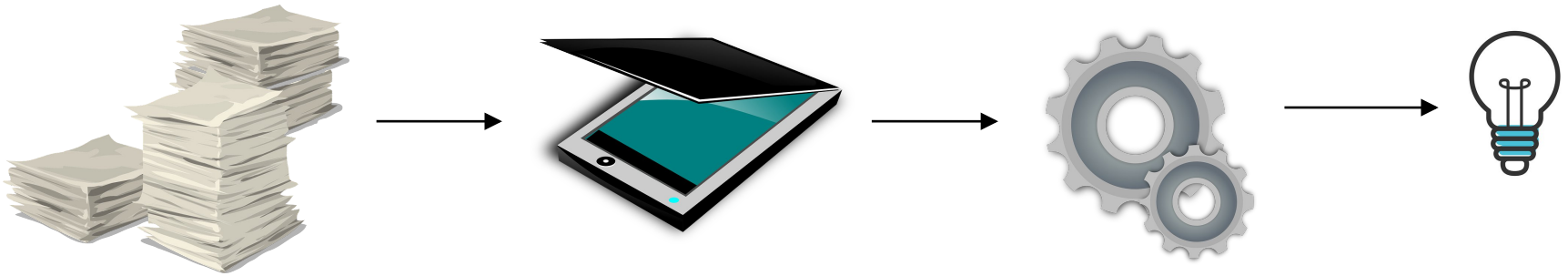
Correção do problema usando barreira da aula passada.

Atividade prática

Aplicação de Semáforos POSIX (20 minutos)

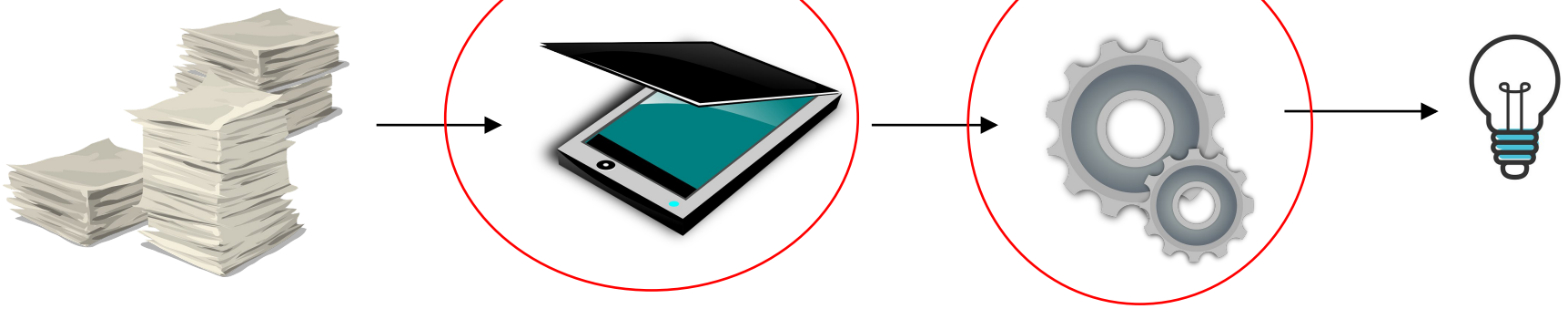
1. Limitar a **N** threads o acesso a um recurso

Problema – leitura de informações



Exemplo 1 – produtor consumidor

Produtor: Escaneia e devolve imagem a ser processada



Consumidor: transforma imagem em informação

Sincronização

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

Exemplo 1 – produtor consumidor

Dois conjuntos de threads

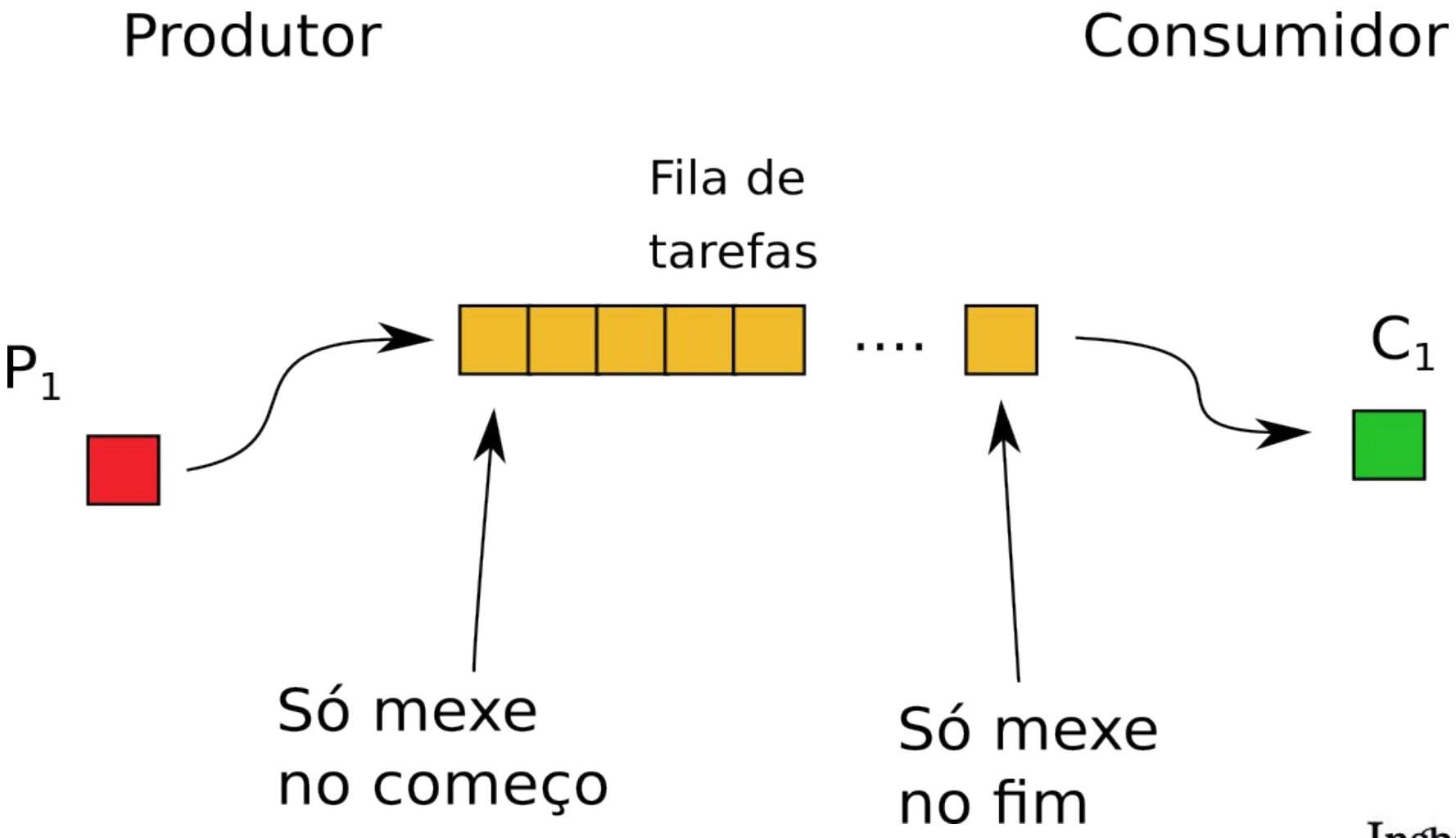
- Produzem tarefas a serem executadas

Pode depender de um recurso compartilhado controlar tamanho das tarefas.

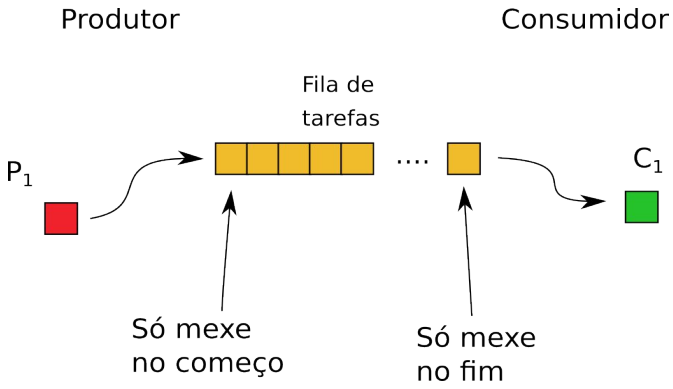
- Consomem as tarefas e as executam.

Cada consumidor não depende dos produtores nem de outros consumidores.

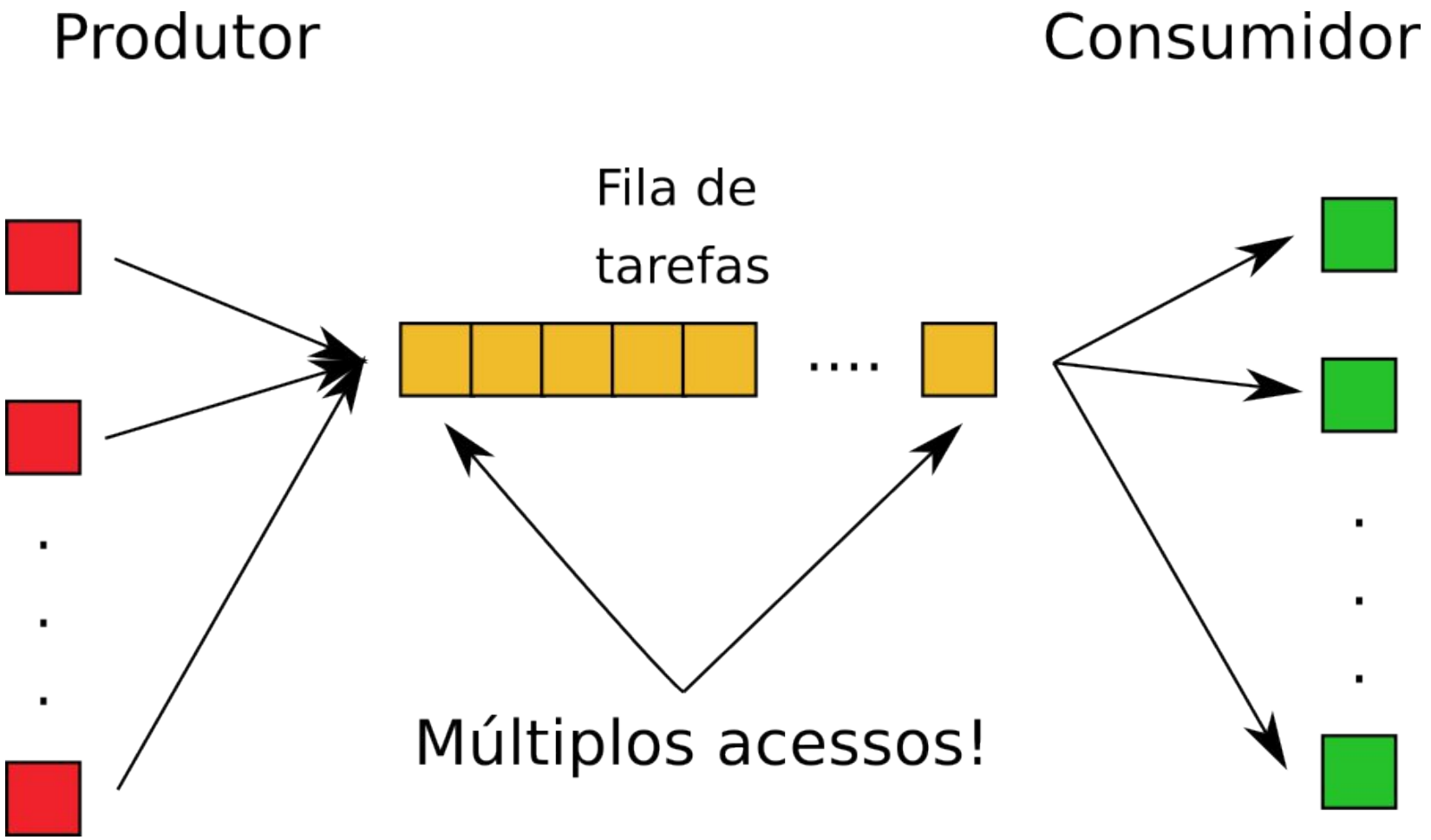
Exemplo 1 – produtor consumidor



Exemplo 1 – produtor consumidor



Exemplo 1 – produtor consumidor



Insper

www.insper.edu.br