**Curso de Ciência da Computação**
**Pontifícia Universidade Católica de Minas Gerais**

# Sistemas Operacionais

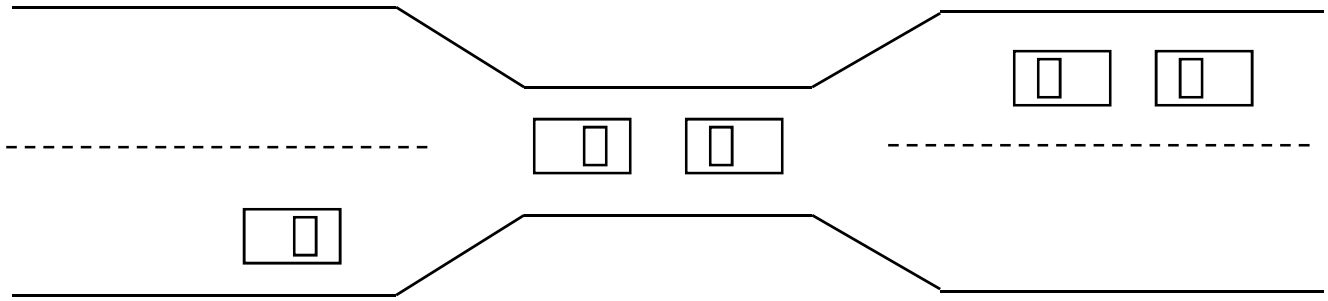
# Capítulo VIII - Deadlocks

# O Problema de Deadlock

- Um conjunto de processos bloqueados, cada qual mantendo um recurso e esperando para receber um recurso mantido por outro processo do conjunto.
- Exemplo
  - Sistema com 2 unidades de fita.
  - $P_1$ e $P_2$ monopolizam um dispositivo cada e necessitam do outro.
- Exemplo
  - semáforos *A* e *B*, iniciados em 1

| $P_0$ | $P_1$ |
|---|---|
| *wait (A);* | *wait(B)* |
| *wait (B);* | *wait(A)* |

# Exemplo da Travessia da Ponte



- Tráfego em um sentido.
- Cada seção da ponte pode ser vista como um recurso.
- Se um deadlock acontece, pode ser resolvido se um carro recuar.
- Vários carros podem ter que recuar se o deadlock acontecer.
- Starvation é possível.

# Modelo do Sistema

- Tipos de recurso $R_1$, $R_2$, . . ., $R_m$
  *ciclos de CPU, espaço de memória, dispositivos de I/O*
- Cada tipo de recurso $R_i$ tem $W_i$ instâncias.
- Cada processo utiliza um recurso como se segue:
  - requisição
  - uso
  - liberação

# Caracterização do Deadlock

Um Deadlock pode ocorrer se 4 condições existirem ao mesmo tempo:

- **Exclusão Mútua:** apenas um processo por vez pode usar o recurso.
- **Posse e espera:** um processo em posse de pelo menos um recurso está esperando por recursos adicionais mantidos por outros processos.
- **Não preempção:** um recurso só pode ser liberado voluntariamente pelo processo que o mantém, após ter completado sua tarefa.
- **Espera circular:** existe um conjunto $\{P_0, P_1, \ldots, P_n\}$ de processos em espera tal que $P_0$ espera por recurso mantido por $P_1$, $P_1$ espera por recurso mantido por $P_2$, ..., $P_{n-1}$ espera por recurso mantido por $P_n$, e $P_n$ espera por recurso mantido por $P_0$.

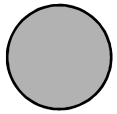
# Grafo de Alocação de Recursos

Um conjunto de vértices $V$ e um conjunto de arestas $E$.

- V é particionado em 2 tipos:
  - $P = \{P_1, P_2, \ldots, P_n\}$, o conjunto de todos os processos no sistema
  - $R = \{R_1, R_2, \ldots, R_m\}$, o conjunto de todos os tipos de recursos no sistema
- aresta de pedido (direcionada) $P_1 \rightarrow R_j$
- aresta de atribuição (direcionada) $R_j \rightarrow P_i$
- Se o grafo não contém ciclos $\Rightarrow$ não há deadlock.
- Se grafo contém ciclo $\Rightarrow$
  - se apenas uma instância por tipo de recurso, há deadlock.
  - Se várias instâncias por tipo de recurso, há possibilidade de deadlock.

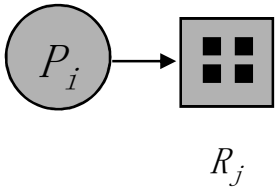
# Grafo de Alocação de Recursos

- Processo

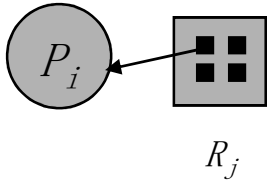


- Tipo de recurso com 4 instâncias

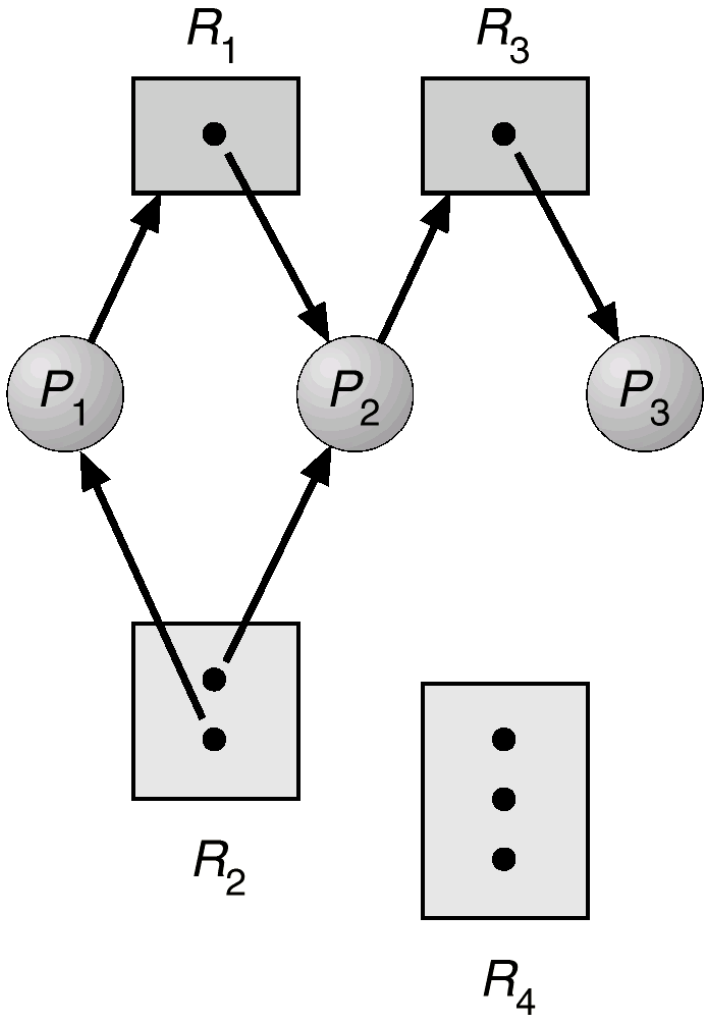
- $P_i$ *solicita instância de* $R_j$



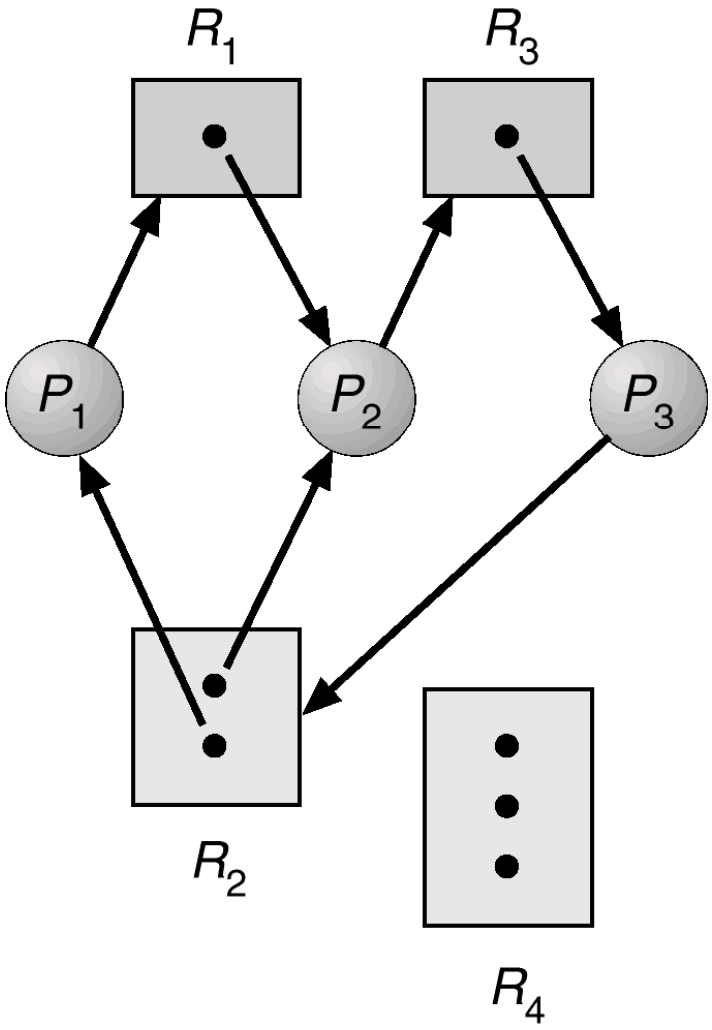

- $P_i$ mantém instância de $R_j$



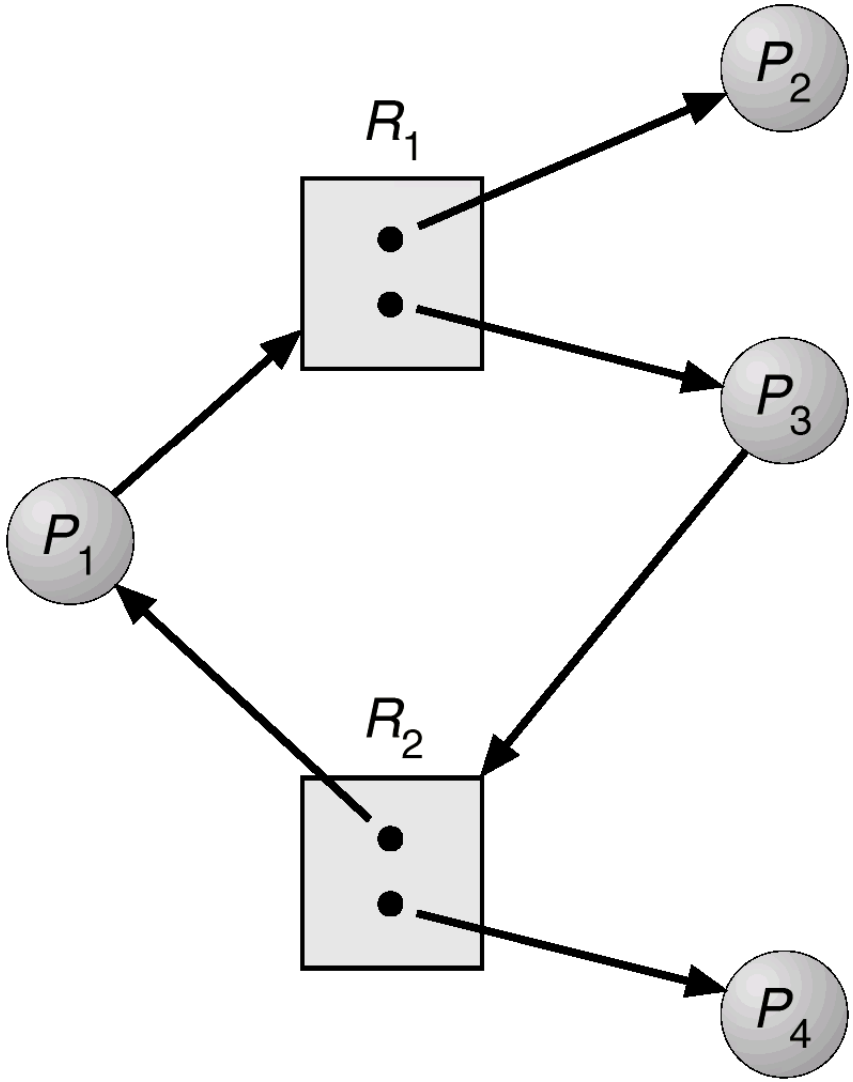
# Grafo de Alocação de Recursos: Exemplo



# Grafo de Alocação de Recursos com Deadlock



# Grafo com Ciclo sem Deadlock



# Métodos para Tratar Deadlocks

- Garantir que o sistema nunca entrará em estado de deadlock.
- Permitir que o sistema entre em deadlock e este seja recuperado.
- Ignorar o problema e fingir que deadlocks nunca acontecerão; usados pela maioria do S.O.s, incluindo o UNIX!!!