Computação Distribuída

Odorico Machado Mendizabal

Universidade Federal de Santa Catarina – UFSC
Departamento de Informática e Estatística – INE

Exclusão Mútua

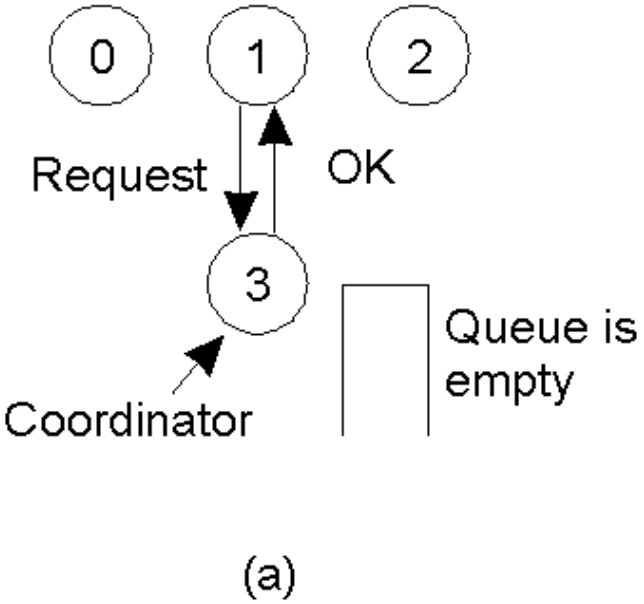
Exclusão mútua em sistemas distribuídos

- Processos em um sistema distribuído devem acessar recursos compartilhados em exclusão mútua

- Algoritmos para exclusão mútua em sistema distribuído podem ser:
  - Centralizados
  - Distribuídos
    - Abordagem baseada em permissão
    - Solução baseada em ficha

- Uma boa solução deve considerar aspectos como ausência de impasse (*deadlock*) ou inanição (*starvation*)

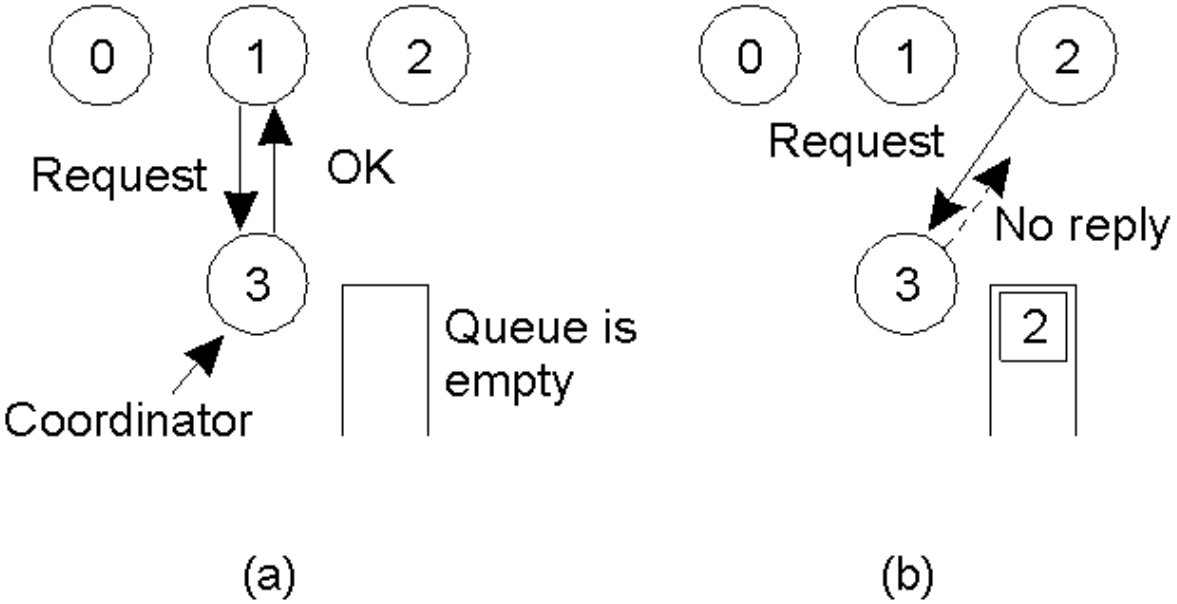
Exclusão mútua – Algoritmo centralizado



(a)

a) Processo 1 solicita ao coordenador (processo 3) permissão para entrar na região crítica. Permissão é dada.

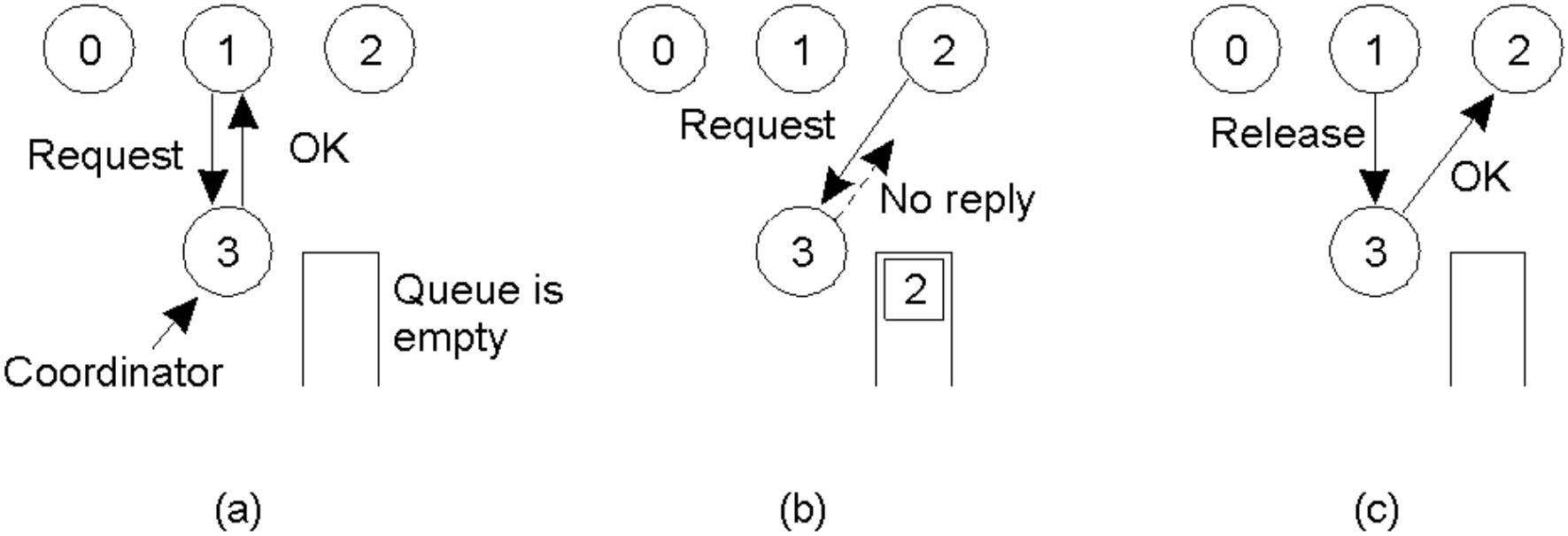
Exclusão mútua – Algoritmo centralizado



a) Processo 1 solicita ao coordenador (processo 3) permissão para entrar na região crítica. Permissão é dada.

b) Processo 2 solicita permissão para entrar na mesma região crítica. O coordenador não responde à requisição, pois a região está ocupada.

Exclusão mútua – Algoritmo centralizado



a) Processo 1 solicita ao coordenador (processo 3) permissão para entrar na região crítica. Permissão é dada.

b) Processo 2 solicita permissão para entrar na mesma região crítica. O coordenador não responde à requisição, pois a região está ocupada.

c) Quando o processo 1 sai da região crítica ele avisa ao coordenador. Então, o coordenador responde ao processo 2, concedendo acesso à região

Algoritmo de Ricart-Agrawala

- Implementa exclusão mútua distribuída entre N processos

- **Premissas**
  - O número de processos participantes no sistema é fixo
  - As mensagens são entregues em ordem total

- **Idéia básica**
  - Processos que querem acessar a região crítica enviam uma requisição para todos os processos e podem acessar a região crítica somente após todos os outros processos responderem a esta requisição

Algoritmo de Ricart-Agrawala

- Cada **processo**
  - Possui um identificador único ID
  - Contabiliza eventos usando relógios lógicos de Lamport
  - Registra o seu estado, sendo:
    - `RELEASED` – fora da região;
    - `HELD` – está acessando a região;
    - `WANTED` – quer acessar a região

- **Mensagens** contêm:
  - Ts (*timestamp* obtido pela leitura do relógio lógico local)
  - Pid (Identificador do processo)

Algoritmo de Ricart-Agrawala

Algoritmo:

Na inicialização:

```
state = RELEASED;
```

Para entrar na seção:

```
state = WANTED;
Ts = timestamp do requisitante;
m = <Ts,Pid>
Difusão atômica de m para todos os processos
Espera até que número de respostas == (N – 1);
state = HELD;
```

Algoritmo de Ricart-Agrawala

Algoritmo:

```
No recebimento de mensagem <t, p> onde p ≠ Pid
Se state = HELD ou (state = WANTED e (Ts, Pid) < (t,p))
   Coloca na fila a requisição de p sem enviar resposta;
Se não
   Responde OK imediatamente para p;
```

Para sair da seção Crítica:

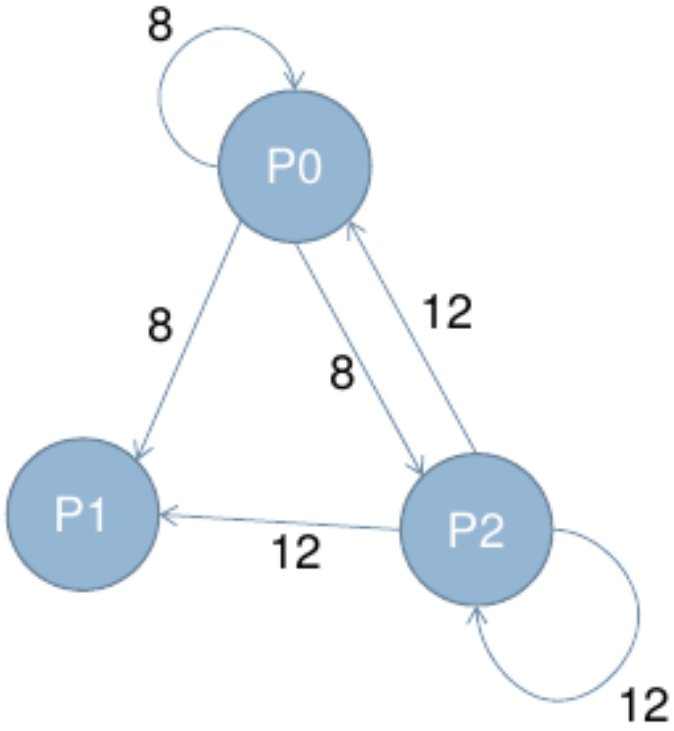
```
state := RELEASED;
Responde OK para todas as requisições enfileiradas
```

Exemplo

P0 e P2 querem acessar um recurso.

P0 e P2 enviam para todos os processos os seus respectivos *timestamps*, inclusive para eles mesmos.

P1 não está interessado no recurso.

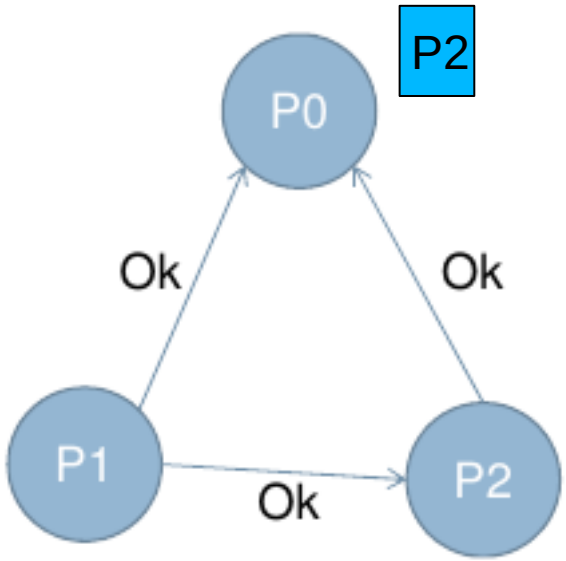


Exemplo

Como o processo P1 não está interessado no recurso envia respostas (Ok) para todos os processos.

P2 envia Ok para P0 pois o *timestamp* de P0 é menor que o seu.

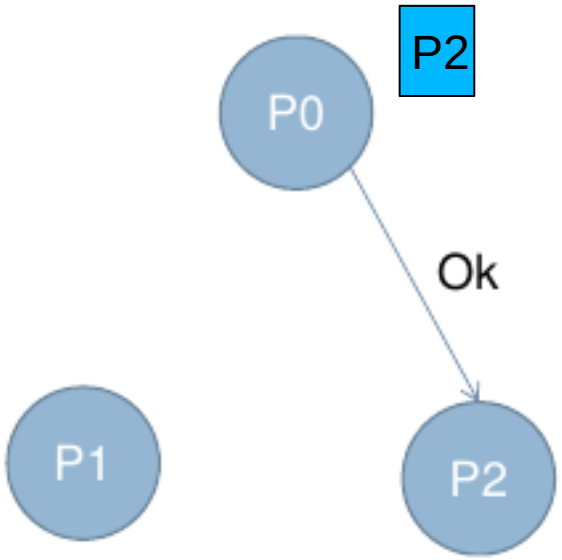
P0 coloca P2 na sua fila para ter acesso ao recurso.



Exemplo

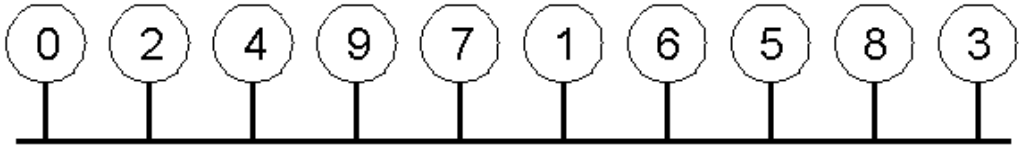
Após P0 acessar o recurso, este envia uma resposta OK para todos os processos em sua fila (neste caso, P2)

P2 terá recebido mensagem OK de todos os processos e pode acessar a região crítica

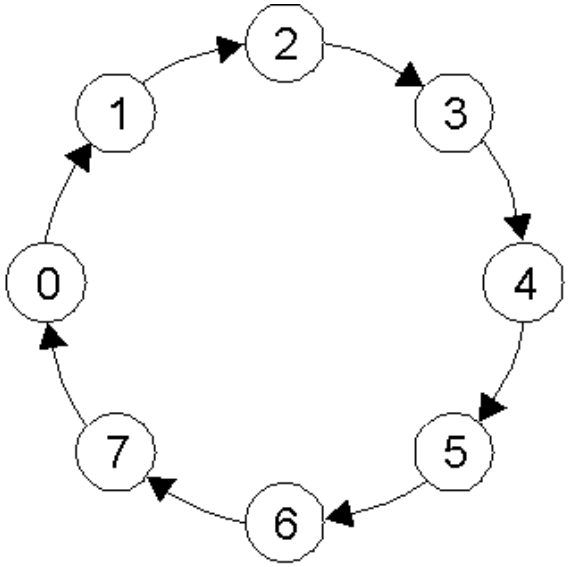


Algoritmo Token Ring

Um grupo não ordenado de processos (Figura a) na rede pode ser organizado como um anel lógico (Figura b)



(a)



(b)

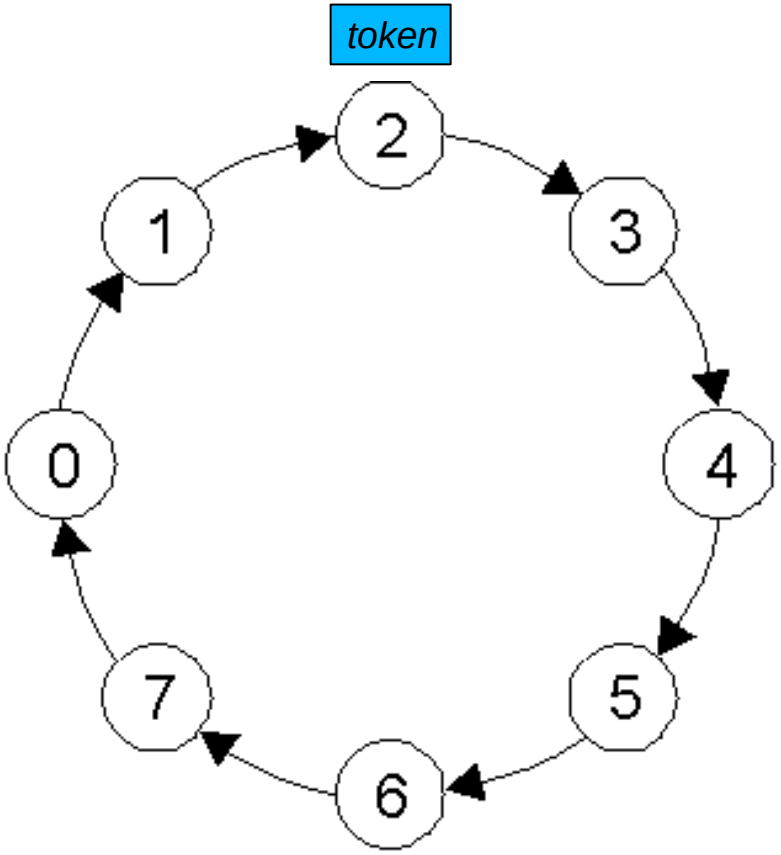
Algoritmo Token Ring

- Um bastão (*token*) circular pelo anel

- Apenas o processo que está com o token pode acessar a região crítica

- Processo sem intenção de acessar a região repassar o token para o próximo processo

P2 pode entrar na região



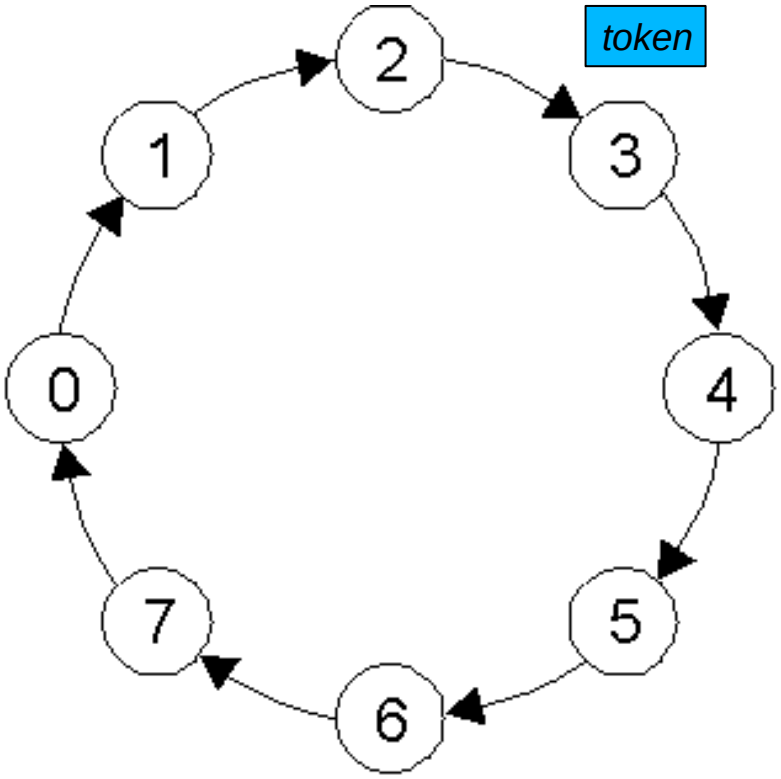
Algoritmo Token Ring

- Um bastão (*token*) circular pelo anel

- Apenas o processo que está com o token pode acessar a região crítica

- Processo sem intenção de acessar a região repassar o token para o próximo processo

P3 pode entrar na região



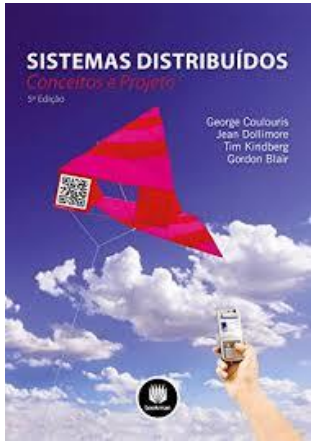
Comparação entre algoritmos de Exclusão Mútua

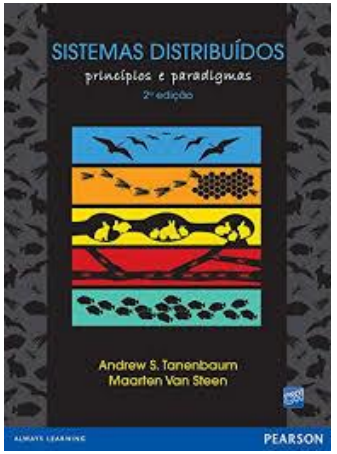
Algoritmo	**Mensagens por acesso/saída da região**	**Atraso antes de entrar (em número de mensages)**	**Problemas**
Centralizado	3	2	Falha do coordenador
Distribuído	2 (n – 1)	2 (n – 1)	Falha de qualquer processo
Token ring	1 to ∞	0 to n – 1	Perda do token, Falha de processo

Referências

Parte destes slides são baseadas em material de aula dos livros:

- *Coulouris, George; Dollimore, Jean; Kindberg, Tim; Blair, Gordon. Sistemas Distribuídos: Conceitos e Projetos. Bookman; 5ª edição. 2013.*
- *Tanenbaum, Andrew S.; Van Steen, Maarten. Sistemas Distribuídos: Princípios e Paradigmas. 2007. Pearson Universidades; 2ª edição.*





- *Imagens e clip arts diversos: https://free-icon-rainbow.com/ , https://www.gratispng.com/*