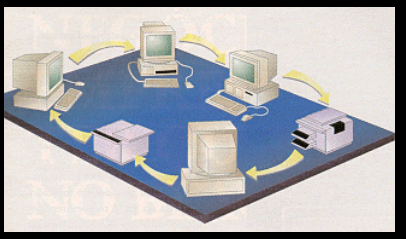
Sistemas Distribuídos

Exclusão Mútua



Referência

- “Sistemas operacionais modernos” Andrew S. TANENBAUM Prentice-Hall, 1995
  - Seção 11.2 pág. 325 - 329

Conteúdo

- Algoritmo centralizado
- Algoritmo distribuído
  (Algoritmo de Ricart e Agrawala)
- Algoritmo token ring
- Comparação dos três algoritmos

Exclusão mútua

- Sistemas construídos com processos concorrentes são programados usando regiões críticas → quando um processo deseja fazer acesso a estrutura ou dado compartilhado deve antes entrar na região crítica para garantir a exclusão mútua (semáforos, monitores etc)
- Como proceder no caso de sistemas distribuídos?

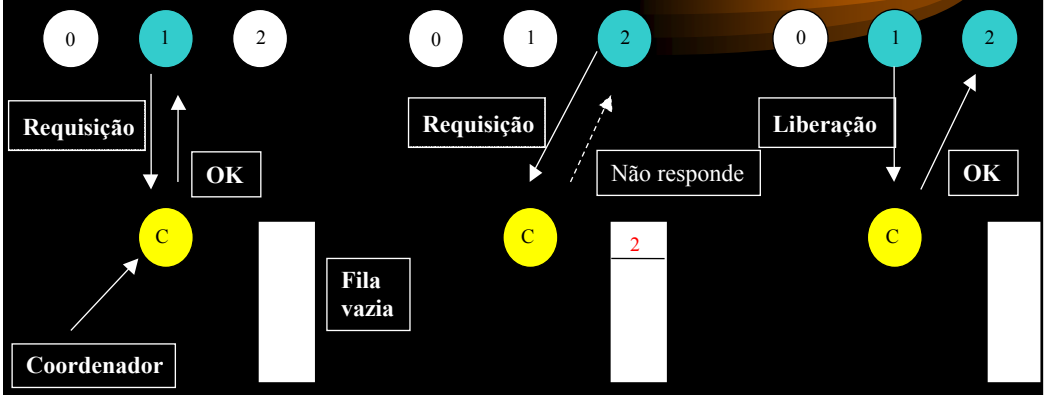
Algoritmo centralizado

- Simulação da metodologia de sistemas com um único processador:
  - Elege-se um processo para ser o coordenador.
  - Sempre que um processo deseja executar sua região crítica, deve antes enviar uma mensagem de solicitação ao coordenador.
  - Se nenhum outro processo estiver executando a região crítica equivalente o coordenador autoriza a execução

Algoritmo centralizado

- São necessárias 3 mensagens:
  - Requisição (REQUEST)
  - permissão de uso (GRANT)
  - Liberação (RELEASE)

Funcionamento do Algoritmo



Algoritmo centralizado

- Garante-se a exclusão mútua, o coordenador só permite que um processo entre na sua região crítica de cada vez (sem *deadlock*).
- Algoritmo justo: as requisições são atendidas na ordem de chegada. Nenhum deles espera indefinidamente (sem *starvation*).
- Lembre-se: são necessárias 3 mensagens

Algoritmo centralizado - Problemas

- O coordenador é um ponto crítico, se ele falhar o sistema pára
- Os processos não sabem distinguir "coordenador fora do ar" e "permissão de acesso negada"
- Em sistemas grandes um único coordenador significa queda de desempenho

Algoritmo distribuído

- Algoritmo de Ricart e Agrawala (1981)
- Exige-se a coordenação dos eventos do sistema, ou seja, qual a ordem dos eventos ocorridos
  - Pode-se utilizar o algoritmo de Lamport para relógio lógico
- Quando um processo deseja entrar em sua região crítica ele envia uma mensagem contendo:
  - a identificação da região crítica,
  - seu número e
  - o tempo corrente a todos os outros processos

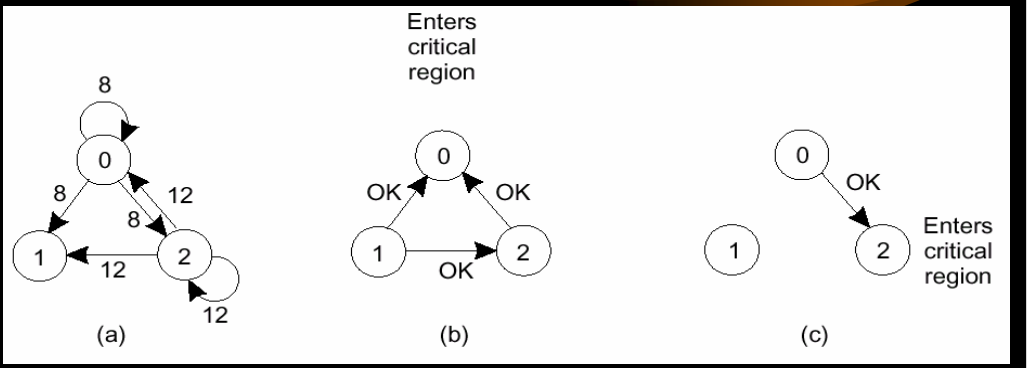
Algoritmo distribuído

- Quando um processo recebe a mensagem, sua ação depende da situação:
  1. Se o receptor não está executando a região crítica e não pretende executá-la, ele envia uma mensagem de OK.
  2. Se o receptor estiver executando a região crítica, ele não deve responder, guardando a mensagem em uma fila.

Algoritmo distribuído

  3. Se o receptor também deseja entrar na região crítica, mas ainda não o fez, ele deve comparar a informação de tempo da mensagem, com a da mensagem que ele enviou. O valor mais baixo ganha. Se o valor da mensagem for mais baixo que o da sua própria mensagem ele deve enviar um OK. Se for mais alto coloca a mensagem na fila e não responder.
- Após enviar as mensagens de solicitação, o processo deve aguardar até receber permissões de todos os outros processos

Algoritmo distribuído



Algoritmo distribuído

- Se o processo 2 enviar a mensagem mais cedo obtém o direito de executar sua região crítica e o processo 0 deve aguardar
- O número de mensagens necessárias

  2 * (n-1) n: número de processos
- Se um processo falha os outros interpretam o silêncio como uma negação de execução da região crítica

  → enviar resposta garantindo ou negando execução

Algoritmo distribuído

- Todos os processos estão envolvidos em todas as decisões
  → o desempenho ruim de um deles pode provocar a queda de desempenho do sistema
- Para melhor o algoritmo: um processo pode entrar na região crítica se obter a permissão da maioria dos processos (se um processo garante o acesso a um outro processo, não poderá enviar nenhuma permissão até que este saia da região crítica)

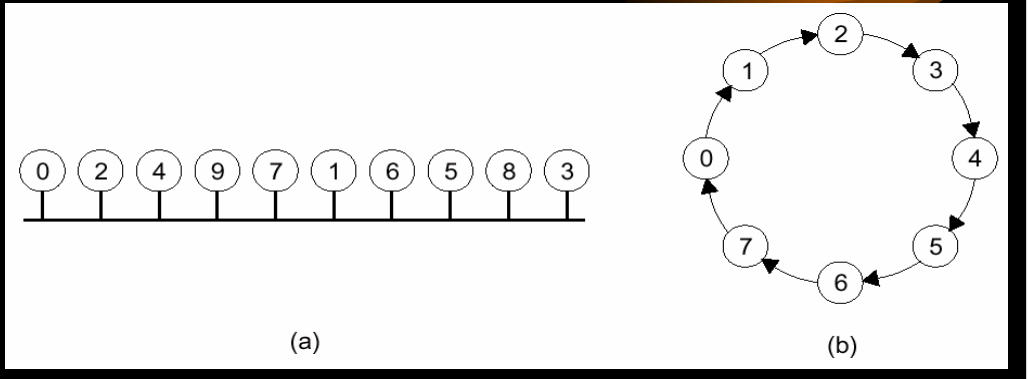
Algoritmo em anel (Token Ring)

- Constrói-se um anel lógico com os processos envolvidos.
- Na inicialização o processo 0 recebe a ficha (token).
- O token deve circular pelo anel passando do processo k para o k+1.

Algoritmo em anel

- Quando o processo recebe o token ele pode executar a região crítica (uma única vez) e, quando sair, passa o token para o próximo processo.
- Se o processo não desejar entrar a região crítica, deve apenas passar o token para o próximo processo.

Algoritmo em anel



Algoritmo em anel

- Correção do algoritmo:
  - Somente o processo com o token executa a região crítica (sem *deadlock*)
  - Todos os processos tem a chance de executar suas regiões críticas (sem *starvation*)
- Problemas:
  - Como detectar que a perda do token?
  - O que fazer quando um processo falha?

Comparação

Algoritmo	Msg por entrada/saída da RC	Retardo antes da entrada (msgs)	Problemas
Centralizado	3	2	Coordenador falha
Distribuído	2 (n-1)	2 (n-1)	Qualquer processo falha
Token ring	1 a ∞	0 a n-1	Perda de token; processo falha

Exercícios

- “Sistemas operacionais modernos” Andrew S. TANENBAUM Prentice-Hall, 1995
  - Capítulo 11 Exercícios 4 - 7 (pág. 345)