Algoritmos de Exclusão Mútua em Sistemas Distribuídos

_ NOTAS DE AULAS _

Prof. Tiago Garcia de Senna Carneiro

DECOM/UFOP

1.O Problema da Sessão Crítica

- Considere um sistema consistindo de n processos {p1, p2, ..., pn}. Cada processo possui um pedaço do código, chamado **seção crítica,** onde o processo pode alterar dados compartilhados. Para garantir que estes processos executem corretamente, o sistema deve assegurar que, quando um processo estiver executando sua seção crítica, nenhum outro processo possa executar a sua sessão crítica. Portanto, o sistema deve garantir **exclusão mútua** entre os processos executando uma seção crítica.
- A seguir vemos a estrutura típica de um processo pi:

repeat

entrada da seção crítica;

seção crítica;

saída da seção crítica;

resto do processo;

until *false;*

- O **problema da seção crítica** consiste no projeto de um protocolo que possa ser utilizado pelos processos para garantir a exclusão mútua nas suas seções críticas.
- Para garantir que dois ou mais processos cooperantes executem corretamente e de forma eficiente, esse protocolo deve satisfazer os seguintes requisitos:

- Exclusão mútua: dois ou mais processos não podem estar executando suas seções críticas simultaneamente.
- Progresso: nenhum processo executando fora da sua seção crítica pode impedir que outro processo entre na sua seção crítica.
- Espera limitada: nenhum processo pode esperar eternamente para entrar na sua seção crítica.
- Nada deve ser assumido sobre a velocidade ou número de processadores no sistema.

2. Algoritmos Centralizados

(fig, 11.9 pág. 326 do livro texto)

Nos algoritmos centralizados, sempre que um processo precisar executar sua seção crítica, ele deve enviar uma mensagem a um processo coordenador informando em qual seção crítica ele deseja entrar e solicitando permissão para fazê-lo.Se nehum outro processo estiver executando a seção crítica cujo acesso foi requisitado, o processo coordenador envia uma permissão ao solicitante. Ao receber a resposta do coordenador o processo requisitante entra na seção crítica.

Quando um processo solicita ao coordenador a entrada em uma seção crítica e já existe um outro processo executando-a, o coordenador nega o acesso e coloca a requisição numa fila. Quando o processo que está na seção crítica acabar de executá-la, ele envia uma mensagem ao coordenador abrindo mão do seu privilégio e então, o acesso à seção crítica é cedido para a primeira requisição pendente.

Características:

- o algoritmo garante a exclusão mútua.
- o algoritmo é justo, as requisições são atendidas na ordem de chegada.
- nenhum processo espera indefinidamente pela seção crítica (Espera Limitada)
- fácil implementação, somente três tipos de mensagens (solicitação, permissão de uso, liberação).

Desvantagens:

- baixa tolerância à falhas, se o coordenador falhar, possivelmente, todo o sistema falhará.
- o coordenador pode se tornar um gargalo do sistema.

Conclusão:

-uma vez que um único ponto de falha poderá derrubar todo o sistema, este algoritmo é **inaceitável** para a construção de sistemas distribuídos.

3. Um Algoritmo Distribuído

(fig, 11.10 pág. 327 do livro texto)

Neste algortimo, quando um processo deseja entrar em uma seção crítica ele deve construir uma mensagem contendo o nome da seção crítica, o seu próprio número e o tempo corrente. Depois, ele envia, de forma confiável, a mensagem para todos os processos do sistema, inclusive ele próprio.

Quando um processo recebe de outro processo a requisição de acesso a uma determinada seção crítica, sua ação vai depender da sua situação relativa a esta seção:

a) se o receptor não estiver na seção crítica e não desejar executá-la ele envia um OK ao transmissor;

b) se o receptor estiver na seção crítica, ele não responde nada, e coloca a requisição numa fila;
c) se o receptor também deseja entrar na seção crítica, mas ainda não o fez, ele deve comparar a informação de tempo da mensagem entrante com a informação de tempo da mensagem que ele enviou aos demais processos. O valor mais baixo ganha. Se o valor da mensagem entrante for mais baixo o processo deve enviar um OK ao processo transmissor. Se o valor da sua própria mensagem for mais baixo, o receptor deve colocar a mensagem entrante em uma fila e não responde nada..

Um processo que solicitou uma seção crítica só pode executar a mesma após receber o concentimento de todos os processos do sistema. Ao terminar a execução de uma seção crítica, ele deve enviar uma mensagem OK a todos os processos cujas requisições estão em sua fila e esvaziar a fila.

Características:

- desenvolvido por Lamport em 1978 e posteriormente melhorado por Ricart e Agrawalw em 1981;
- exige a ordenação total de todos os eventos do sistema, ou seja, todas as máquinas do sistema devem concordar sobre a ordem na qual os eventos do sistema ocorreram;
- a espera limitada é garantida;
- são necessárias 2(**n**-1) mensagens, onde **n** é o número processos no sistema;
- todas as mensagens, solicitação, permissão de uso e liberação, devem ser confiáveis

Desvantagens:

- gera um tráfego muito alto na rede;
- se não existirem primitivas de comunicação grupal disponíveis, cada processo terá que manter sua própria lista de processos;
- um único ponto de falha do algoritmo centralizado foi substituído por **n** pontos de falha. Se um processo falha e não responde a uma solicitação seu silêncio será interpretado como uma negativa à tentativa de acesso à seção crítica, bloqueando assim todas as tentativas subsequentes de qualquer outro processo do sistema. Portanto, no critério tolerância a falhas este algoritmo é **n** vezes pior que o primeiro.

Melhorias:
- ao invés de não responder nada, todos os processo devem responder de forma negativa quando não permitirem que um determinado processo entre na seção crítica.
- ao invés de esperar o concentimento de todos os processos do sistema, um processo que deseje entrar numa seção crítica passa a esperar pela resposta de somente 50%+1 processos no sistema. Para garantir que esta melhoria funcione, um processo nunca pode enviar um OK a dois ou mais processos do sistema, ele precisa sempre esperar que o primeiro processo libere a seção crítica para enviar um OK para o segundo e assim por diante.

Conclusão:
- este algoritmo é mais lento, mais complicado, mais caro e menos robusto que o algoritmo centralizado, mas mostra que é possível a implementação de algoritmos distribuídos para garantir a exclusão mútua numa seção crítica.

4. Um Algoritmo em *Token Ring*

Neste algoritmos, os processos do sistema são organizados em um anel e numerados segundo a ordem em que aparecem no anel. Quando o anel é inciado, o processo **0** recebe uma ficha (*token*) que lhe dá permissão para entrar na seção crítica. A ficha circula pelo anel. Quando um processo recebe a ficha ele verifica se ele mesmo deseja entrar numa seção crítica, se estiver ele a executa. Caso contrário, ele passa a ficha para o processo que o sucede no anel. Um processo só pode entrar em uma única seção crítica a cada vez que recebe a ficha.

Características:
- a espera limitada é garantida.

Desvantagem:
- a ficha pode ser perdida e é difícil detectar esta perda. Um processo pode ficar por uma hora sem receber a ficha e ainda assim não pode afirmar que ela foi perdida, pois outro processo pode estar executando sua seção crítica por todo este tempo.
- se um processo falhar o anel é quebrado. Soluciona-se este problema obrigando-se que cada processo confirme o recebimento da ficha, o processo que falhou será identificado no momento que o processo que o antecede tentar lhe entragar a mesma. Assim, o processo que falhou é retirado do anel.
- se nehum processo quiser entrar na seção crítica a ficha permanecerá circulando pelo anel em alta velocidade e desperdiçando recurso, mesmo que seja pouco recurso.
- um processo pode esperar por um longo intervalor de tempo para conseguir entrar numa seção crítica.

Conclusão:

- este é o melhor algoritmo.