UNIVERSIDADE FEDERAL ALFENAS (UNIFAL)
Bacharelado em Ciência da Computação

Disciplina	Método de entrega	Data de entrega
DCE540 - Computação Paralela e Distribuída	Moodle da disciplina	22/09/2021 às 8h00

Professor
Iago Augusto de Carvalho (iago.carvalho@unifal-mg.edu.br)

Prova 02

Cada aluno deverá submeter um único arquivo .pdf com a resolução da prova.
A prova pode ser realizada de duas maneiras:

- Com papel e caneta, sendo posteriormente escaneada e enviada
- Digitada em algum editor de texto, e.g., Word ou LaTeX

A prova deverá ser entregue no Moodle da disciplina até a data limite.

- Atrasos não serão tolerados

Exercício 1 (10 %)

Quando estamos trabalhando com algoritmos de sincronização de *clock*, devemos garantir uma acurácia e uma precisão mínima para todos os *clocks* de um sistema distribuído. Neste contexto, explique o que é

a) Acurácia

b) Precisão

Exercício 2 (15 %)

Clocks físicos são baseados em cristais de quartzo, sendo que o passar do tempo é registrado através da captura das oscilações deste cristal. Sabe-se que estes cristais possuem uma frequência de oscilação ideal F. Entretanto, é muito comum que a oscilação destes cristais se desvie por, até mesmo, ρ unidades da frequência ideal F devido a fatores externos, como desgaste do cristal, alterações na bateria de alimentação do cristal, mudanças extremamente bruscas de temperatura e/ou pressão do ambiente, dentre outros fatores

Assuma que estes fatores externos são aleatórios e não exista nenhuma tendência sobre a alteração da frequência de oscilações do cristal. Deste modo, qual é o desvio de *clock* esperado para uma máquina qualquer do seu sistema distribuído após um intervalo de tempo t qualquer? Justifique sua resposta e apresente um histograma (montado a partir da observação do desvio de *clock* de n dispositivos do seu sistema distribuído em um mesmo instante de tempo t), indicando o valor mais provável para o desvio de *clock* de um dispositivo qualquer e a maior difereça que deverá ser observada entre o desvio de *clock* de dois diferentes dispositivos. Assuma que n é um valor grande o suficiente e que todos os dispositivos do seu sistema distribuído são iguais.

Exercício 3 (15%)

Qual é a principal ideia por trás da utilização de *clocks* lógicos? Explique, com suas próprias palavras, as principais vantagens e desvantagens da utilização deste método de sincronização.

Exercício 4 (20%)

Compare os três algoritmos de exclusão mútua apresentados em aula (centralizado, distribuído e *token-ring*). Apresente as principais vantagens e desvantagens de cada um. Além disto, exponha uma situação onde a utilização de cada um dos algoritmos é vantajosa.

Exercício 5 (30%)

Algoritmos de exclusão mútua coordenam o acesso a recursos compartilhados. Estes algoritmos devem garantir que não ocorram *deadlocks*, que não exista *starvation* e que o acesso aos recursos compartilhados sejam realizados com equidade e justiça. Explique, com suas próprias palavras, como os três algoritmos de exclusão mútua apresentados em aula (centralizado, distribuído e *token-ring*) garantem estas três propriedades.

Exercício 6 (10%)

Algoritmos de eleições são utilizados para eleger processos coordenadores em sistemas distribuídos, isto é, aqueles processos que são responsáveis por gerenciar o acesso aos recursos compartilhados. Em sistemas relativamente pequenos, podemos utilizar o algoritmo de bully e o algoritmo do anel para eleger os processos coordenadores. Entretanto, a utilização destes algoritmos em sistemas de larga escala, com dezenas de milhares de nós, é indesejável. Deste modo, explique os problemas da utilização do algoritmo de bully e do algoritmo do anel em sistemas distribuídos de larga escala.

Gabarito

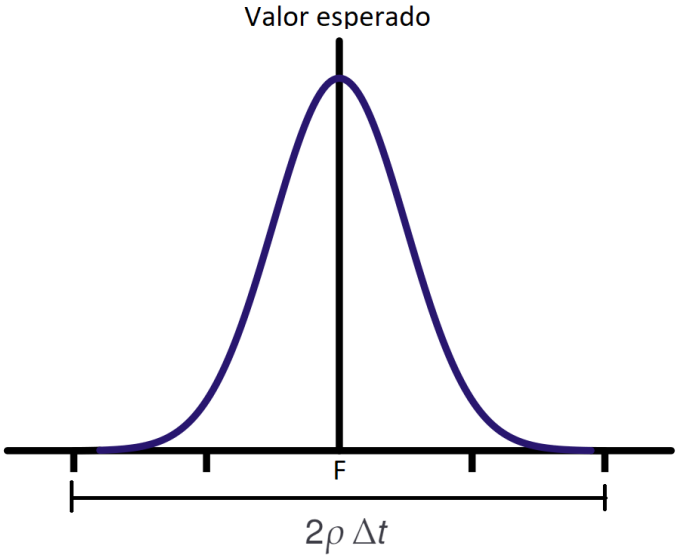
Exercício 1

a) Acurácia refere-se a diferença entre o *clock* de quaisquer dois dispositivos pertencentes a um mesmo sistema distribuído. Normalmente, deseja-se uma acurácia mínima α

b) Precisão refere-se a diferença entre o *clock* de um dispositivo de um sistema distribuído e um *clock* de acreditação global, como o UTC. Normalmente, deseja-se uma precisão mínima π

Exercício 2

Levando em consideração que a frequência varia de forma aleatória e não exista nenhuma tendência, espera-se que as oscilações mais rápidas compensem as mais lentas. Desta forma, é esperado que o desvio de *clock* de um dispositivo seja praticamente nulo. Podemos representar estes desvios através de um histograma, onde os desvios de *clock* são representados como uma curva gaussiana (distribuição normal) e a maior diferença entre dois desvios quaisquer seria igual a $2\rho\,\Delta t$, onde Δt representa o intervalo de tempo decorrido desde a sincronização anterior dos clocks. Este histograma é representado na figura abaixo.



Exercício 3

Clocks lógicos tentam estabelecer uma ordem de ocorrência de eventos em sistemas distribuídos. Desta forma, não é necessário que o *clock* de todos os dispositivos estejam sincronizados, mas só que todos concordem com a ordem das ações uns dos outros. A principal vantagem de *clocks* lógicos é que não é necessário sincronizar todos os dispositivos de um sistema distribuído entre si, mas somente aqueles que compartilham um mesmo recurso compartilhado. Além disso, *clocks* lógicos podem ser utilizados em ambientes onde um servidor de *clock* confiável não está disponível. Já a principal desvantagem deste método de sincronização (principalmente quando usamos o algoritmo de Lamport) é que podem existir enormes espaços entre dois *clocks* sequenciais em um mesmo dispositivo. Estes espaços são provenientes da alteração do valor do *clock* corrente quando chega uma mensagem atualizando o valor de um *clock* a fim de estabelecer a ordem lógica dos acontecimentos. Por causa disso, faz-se necessário sincronizar o *clock* de todos os dispositivos do sistema distribuído de tempos em tempos.

Exercício 4

Algoritmo centralizado: O algoritmo centralizado tem como principal característica a existência de um único processo coordenador que gerencia o acesso a todos os recursos compartilhados. A principal vantagem deste algoritmo é sua simples implementação e eficiência, pois poucas trocas de mensagens são necessárias e o *delay* para acesso a recursos compartilhados geralmente é pequeno. Entretanto, este algoritmo tem como

principal desvantagem o fato de que o processo coordenador não pode nunca falhar. Este algoritmo deve ser utilizado quando existe um nó da rede dedicado para a tarefa de coordenação, sendo que este nó tem fortes mecanismos de prevenção e recuperação a falhas.

Algoritmo distribuído: O algoritmo distribuído baseia-se na ideia de *clocks* lógicos, sendo que o acesso a recursos compartilhados é realizado de acordo com a ordem lógica dos pedidos de acesso. Entre as vantagens deste método, podemos apontar o reaproveitamento do protocolo de *clocks* lógicos para realizar, também, o gerenciamento de acesso a recursos compartilhados. Entretanto, ele possui diversas desvantagens, como o elevado número necessário de troca de mensagens e o grande número de pontos de falhas (pois se qualquer dispositivo/aplicação do sistema distribuído falhe, o protocolo deixa de funcionar). Este algoritmo pode ser aplicado em um sistema distribuído de pequena escala que já utilize algoritmos de sincronização de *clock* lógicos e deseja-se implementar o gerenciamento de acesso a recursos compartilhados com pouco esforço de desenvolvimento.

Algoritmo *token-ring*: O algoritmo *token-ring* constrói uma estrutura de rede virtual conectando todos os processos/aplicações/dispositivos do sistema distribuído na forma de um ciclo. Para cada recurso compartilhado, ele cria um *token* e faz este *token* circular na rede virtual, sendo que quem tem a posse do *token* pode realizar o acesso ao recurso. As vantagens deste algoritmo são sua simples implementação e a desnecessidade de termos um processo coordenador, além do fato de que o gerenciamento e acesso a diferentes recursos compartilhados é realizado separadamente (cada um com seu *token*). Já em relação as desvantagens, podemos citar a perda de *tokens* na rede: quanto ocorre uma falha em uma aplicação que está em posse de um *token*, é difícil para o *middleware* e o sistema distribuído descobrirem que alguém caiu e o *token* está perdido. Este algoritmo de exclusão mútua é indicado para sistemas distribuídos onde não há a possibilidade de criarmos um processo coordenador e não é necessário centralizar as ações de coordenação de acesso em um único dispositivo.

Exercício 5

Algoritmo centralizado

- *Deadlock*: O processo coordenador consegue interpretar quando ocorre um *deadlock* observando as mensagens de requisição de acesso a recursos compartilhados. A partir daí, ele consegue tratar da maneira que achar mais eficiente
- *Starvation*: Sempre que um processo realiza uma requisição, ele é enfileirado para que possa acessar o recurso compartilhado. *Starvation* nunca ocorre pois a fila sempre anda.
- Igualdade e justiça: Todos os processos tem a mesma chance de acessar o recurso compartilhado, basta que eles enviem uma requisição. Além disso, a justiça vem do fato de que, quem pediu primeiro, consegue acessar primeiro.

Algoritmo distribuído

- *Deadlock*: *Deadlocks* não ocorrem devido a utlização dos tempos de *clock* nas requisições para decidir quais processos tem acesso a quais recursos compartilhados. Requisições com valores de *clock* mais baixos tem prioridade sobre as de valores mais altos.
- *Starvation*: Sempre que um processo requer acesso a um recurso compartilhado, o tempo de seu *clock* é armazenado. Esta requisição é inserida na fila de todos os outros processos interessados neste recurso compartilhado e, quando o recurso for liberado, o primeiro processo é notificado. Assim, garante-se que não exista *starvation*
- Igualdade e justiça: Todos os processos tem a mesma chance de acessar o recurso compartilhado, basta que eles enviem uma requisição. Além disso, a justiça vem do fato de que, quem pediu primeiro, consegue acessar primeiro.

Algoritmo *token-ring*

- *Deadlock*: Não existe uma maneira clara de garantir que não existam *deadlocks*.
- *Starvation*: Os *tokens* circulam entre todos os processos do sistema distribuído. Desta forma, em algum momento, um processo que queira acessar um recurso compartilhado (simbolizado por um *token*), vai tomar posse deste *token* e poderá realizar o acesso.
- Igualdade e justiça: Os *tokens* circulam por toda a rede e passam por todos os processos com a mesma prioridade.

Exercício 6

Tanto o algoritmo de *bully* como o algoritmo do anel trocam um número elevado de mensagens dentro do sistema distribuído. Desta forma, sua utilização em sistemas de larga escala é muito custosa computacionalmente, pois implica na troca de centenas de milhares de mensagens sempre que deseja-se eleger um novo coordenador, o que pode causar instabilidades na rede do sistema distribuído. Além disso, estes algoritmos elegem somente um único processo coordenador, sendo que sistemas de larga escala normalmente demandam diversos processos coordenadores distribuídos igualmente através do sistema para garantir decisões de acesso mais rápidas