Curso: **ENGENHARIA DE COMPUTAÇÃO**
Disciplina: **SISTEMAS COMPUTACIONAIS DISTRIBUÍDOS**
Professor: **KLAUSNER VIEIRA GONÇALVES**

SISTEMAS DISTRIBUÍDOS - SINCRONIZAÇÃO

Curso: **ENGENHARIA DE COMPUTAÇÃO**
Disciplina: **SISTEMAS COMPUTACIONAIS DISTRIBUÍDOS**
Professor: **KLAUSNER VIEIRA GONÇALVES**

SINCRONIZAÇÃO

A sincronização entre processos é importante em diversas situações:

- Para que vários processos não acessem ao mesmo tempo um recurso compartilhado
  - Processos devem cooperar para garantir um acesso temporário exclusivo => exemplo: impressora

- Para que os processos tenham a mesma visão da ordenação de eventos
  - Se uma mensagem m1 do processo P foi enviada antes ou depois de m2 do processo Q

Curso: **ENGENHARIA DE COMPUTAÇÃO**
Disciplina: **SISTEMAS COMPUTACIONAIS DISTRIBUÍDOS**
Professor: **KLAUSNER VIEIRA GONÇALVES**

SINCRONIZAÇÃO

A sincronização em sistemas distribuídos costuma ser mais difícil do que em sistemas monoprocessadores ou multiprocessadores.

Em um sistema centralizado, o tempo não é ambíguo:

- Quando um processo quer saber a hora, ele faz uma chamada ao sistema, e o núcleo responde
- Se A pergunta a hora, e um pouco mais tarde B perguntar a hora, o valor obtido por B será maior do que o obtido por A

Em um SD, conseguir acordo nos horários não é trivial

Curso: **ENGENHARIA DE COMPUTAÇÃO**
Disciplina: **SISTEMAS COMPUTACIONAIS DISTRIBUÍDOS**
Professor: **KLAUSNER VIEIRA GONÇALVES**

PROBLEMAS DE SINCRONIZAÇÃO

- Cada processador (nó do SD) tem um componente chamado **relógio**.

- Relógio em SD podem:
  - Acusar horas diferentes entre si (**defasagem interna**)
  - Acusar hora diferente do horário real (**defasagem externa**)
  - Ter velocidades diferentes: clocks acelerados (**defasagem variável**)

Curso: **ENGENHARIA DE COMPUTAÇÃO**
Disciplina: **SISTEMAS COMPUTACIONAIS DISTRIBUÍDOS**
Professor: **KLAUSNER VIEIRA GONÇALVES**

PROBLEMAS DE SINCRONIZAÇÃO

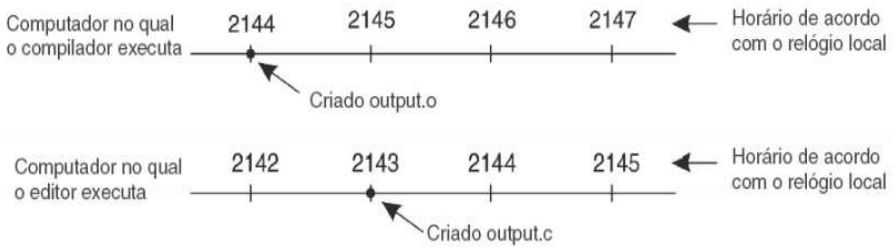
Relógios sincronizados são necessários para uma série de aplicações:

- Identificar atualidade de mensagens (antigas devem ser descartadas)
- Aplicações de tempo real (deve responder em um curto período de tempo)
- Controle de versões

Curso: **ENGENHARIA DE COMPUTAÇÃO**
Disciplina: **SISTEMAS COMPUTACIONAIS DISTRIBUÍDOS**
Professor: **KLAUSNER VIEIRA GONÇALVES**

PROBLEMAS DE SINCRONIZAÇÃO

Defasagem Interna



Curso: **ENGENHARIA DE COMPUTAÇÃO**
Disciplina: **SISTEMAS COMPUTACIONAIS DISTRIBUÍDOS**
Professor: **KLAUSNER VIEIRA GONÇALVES**

PROBLEMAS DE SINCRONIZAÇÃO

- A sincronização de relógios não precisa ser absoluta (Lamport, 1978)
- Se 2 processos não interagem, seus relógios não precisam ser sincronizados
- O que importa não é que todos os processos concordem exatamente em horas são, mas **concordam com a ordem em que os eventos ocorrem**.

O importante é saber o evento que ocorreu antes e o evento ocorreu depois

Curso: **ENGENHARIA DE COMPUTAÇÃO**
Disciplina: **SISTEMAS COMPUTACIONAIS DISTRIBUÍDOS**
Professor: **KLAUSNER VIEIRA GONÇALVES**

SINCRONIZAÇÃO

Relógios Lógicos
- Necessidade não apenas da consistência interna dos relógios
- Possibilita identificar a ondem dos eventos

Relógios Físicos
- Para cenários onde os relógios devem ser exatamente iguais
- E não devem desviar do tempo real

Curso: **ENGENHARIA DE COMPUTAÇÃO**
Disciplina: **SISTEMAS COMPUTACIONAIS DISTRIBUÍDOS**
Professor: **KLAUSNER VIEIRA GONÇALVES**

RELÓGIOS LÓGICOS

Curso: **ENGENHARIA DE COMPUTAÇÃO**
Disciplina: **SISTEMAS COMPUTACIONAIS DISTRIBUÍDOS**
Professor: **KLAUSNER VIEIRA GONÇALVES**

RELÓGIOS LÓGICOS - LAMPORT

Em 1978, Lamport mostrou que, embora a sincronização de relógios seja possível, não precisa ser absoluta:

- Se dois processos não interagem entre si, não é necessário que seus relógios sejam sincronizados
- Não é necessário que todos os processos concordem com a hora exata, mas com a ordem em que os eventos ocorrem

Curso: **ENGENHARIA DE COMPUTAÇÃO**
Disciplina: **SISTEMAS COMPUTACIONAIS DISTRIBUÍDOS**
Professor: **KLAUSNER VIEIRA GONÇALVES**

RELÓGIOS LÓGICOS - LAMPORT

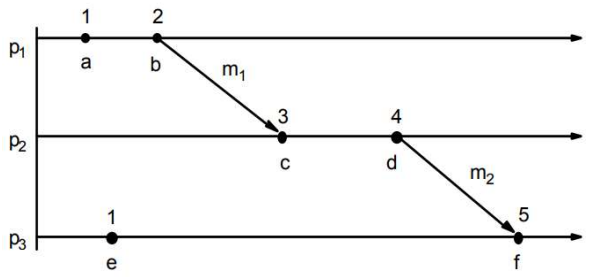
Definida a relação "**acontece antes**":

- Se ***a*** e ***b*** são eventos do mesmo processo:
  - Se ***a*** acontece antes de ***b***: ***a*** => ***b***
  - Sendo ***a*** o envio da mensagem e ***b*** o recebimento da mensagem então: ***a*** => ***b***
  - Se ***a*** => ***b*** e ***b*** => ***c***, então ***a*** => ***c*** (propriedade transitiva)

- Se x e y acontecem em processos diferentes e tanto x => y quanto y => x são falsas, os processos x e y são ditos concorrentes.

Curso: **ENGENHARIA DE COMPUTAÇÃO**
Disciplina: **SISTEMAS COMPUTACIONAIS DISTRIBUÍDOS**
Professor: **KLAUSNER VIEIRA GONÇALVES**

RELÓGIOS LÓGICOS - LAMPORT



- Nem todos os eventos podem ser relacionados através da relação "**acontece antes**"
- Consideremos ***a*** e ***e*** (processos sem a existência de mensagens entre os processos)
- São definidos como processos concorrentes: ***a*** || ***e***

Curso: **ENGENHARIA DE COMPUTAÇÃO**
Disciplina: **SISTEMAS COMPUTACIONAIS DISTRIBUÍDOS**
Professor: **KLAUSNER VIEIRA GONÇALVES**

RELÓGIOS LÓGICOS - LAMPORT

- Cada processo tem um contador (relógio lógico)
- Inicialmente o relógio lógico tem valor 0
- Processo incrementa seu contador quando um evento de envio ou processamento é realizado
- Contador é atribuído a um evento como seu marcador de hora
- Um evento de envio de mensagem carrega seu marcador de hora
- Em um evento de recebimento de mensagem o contador é atualizado por:

max(contador local, marcador de hora da mensagem) + 1

Curso: **ENGENHARIA DE COMPUTAÇÃO**
Disciplina: **SISTEMAS COMPUTACIONAIS DISTRIBUÍDOS**
Professor: **KLAUSNER VIEIRA GONÇALVES**

RELÓGIOS LÓGICOS - LAMPORT

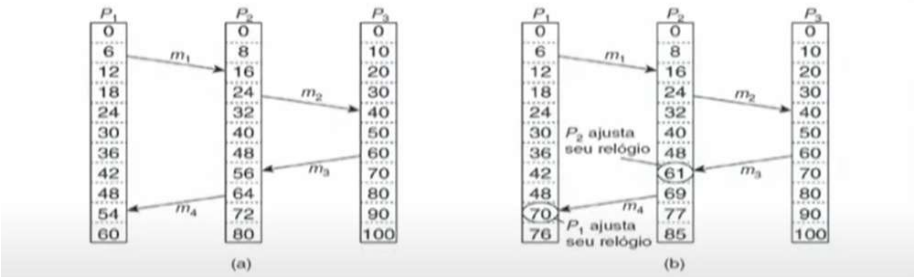


Figura 6.9 (a) Três processos, cada um com seu próprio relógio. Os relógios funcionam a taxas diferentes. (b) O algoritmo de Lamport corrige os relógios.

Curso: **ENGENHARIA DE COMPUTAÇÃO**
Disciplina: **SISTEMAS COMPUTACIONAIS DISTRIBUÍDOS**
Professor: **KLAUSNER VIEIRA GONÇALVES**

RELÓGIOS FÍSICOS

Curso: **ENGENHARIA DE COMPUTAÇÃO**
Disciplina: **SISTEMAS COMPUTACIONAIS DISTRIBUÍDOS**
Professor: **KLAUSNER VIEIRA GONÇALVES**

RELÓGIOS FÍSICOS

Computadores tem circuitos para monitorar a **passagem de tempo => relógios => temporizador**

- Um temporizador é um cristal de quartzo lapidado, que quando mantidos sob tensão, oscilam em uma frequência bem definida

- Associado a cada cristal
  - Contador
  - Registrador de Retenção

Curso: **ENGENHARIA DE COMPUTAÇÃO**
Disciplina: **SISTEMAS COMPUTACIONAIS DISTRIBUÍDOS**
Professor: **KLAUSNER VIEIRA GONÇALVES**

RELÓGIOS FÍSICOS

Funcionamento do Temporizador:

- Cada oscilação do cristal reduz uma unidade do contador
- Quando o contador chega a zero é gerada uma interrupção e o contador é recarregado pelo registrador de retenção

- É possível programar o temporizador para gerar uma interrupção em uma determinada frequência

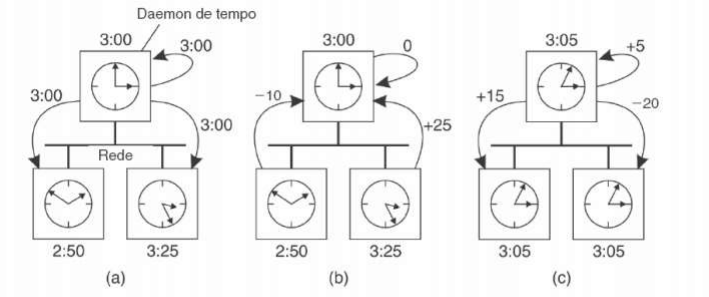
- Cada interrupção é denominada ciclo do relógio

- A cada ciclo de relógio é somada uma unidade a hora armazenada => relógio é mantido atualizado

Curso: **ENGENHARIA DE COMPUTAÇÃO**
Disciplina: **SISTEMAS COMPUTACIONAIS DISTRIBUÍDOS**
Professor: **KLAUSNER VIEIRA GONÇALVES**

RELÓGIOS FÍSICOS - ALGORITMO DE BERKELEY

- No algoritmo Berkeley o servidor consulta todas as máquinas de tempos em tempos, obtendo o horário de cada máquina
- Gera uma média de todas as horas, e informa a todos os computadores o deslocamento de tempo a ser feito

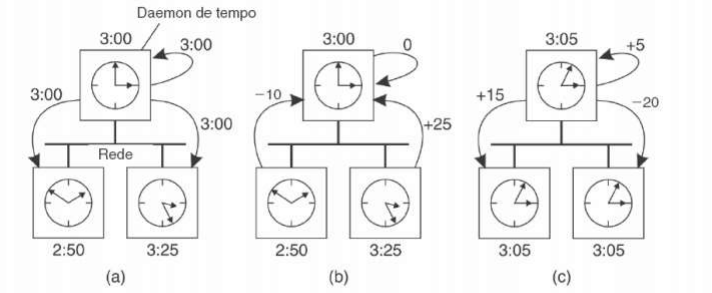


(a) O daemon de tempo pergunta a todas as outras máquinas os valores marcados por seus relógios.
(b) As máquinas respondem. (c) O daemon de tempo informa a todas como devem ajustar seus relógios.

Curso: **ENGENHARIA DE COMPUTAÇÃO**
Disciplina: **SISTEMAS COMPUTACIONAIS DISTRIBUÍDOS**
Professor: **KLAUSNER VIEIRA GONÇALVES**

RELÓGIOS FÍSICOS - ALGORITMO DE BERKELEY

- No algoritmo Berkeley o servidor consulta todas as máquinas de tempos em tempos, obtendo o horário de cada máquina
- Gera uma média de todas as horas, e informa a todos os computadores o deslocamento de tempo a ser feito



(a) O daemon de tempo pergunta a todas as outras máquinas os valores marcados por seus relógios.
(b) As máquinas respondem. (c) O daemon de tempo informa a todas como devem ajustar seus relógios.

Curso: **ENGENHARIA DE COMPUTAÇÃO**
Disciplina: **SISTEMAS COMPUTACIONAIS DISTRIBUÍDOS**
Professor: **KLAUSNER VIEIRA GONÇALVES**

RELÓGIOS FÍSICOS

Problemas

- O tempo do relógio físico não pode voltar (atrasar) => Solução é diminuir a frequência do clock

- Em um rede é difícil medir com precisão o delay da rede
- Atraso na rede local mal calculado para computar o tempo certo nos ajustes dos relógios
- Algoritmos como o de Berkeley tentam atenuar esse problema

Curso: **ENGENHARIA DE COMPUTAÇÃO**
Disciplina: **SISTEMAS COMPUTACIONAIS DISTRIBUÍDOS**
Professor: **KLAUSNER VIEIRA GONÇALVES**

ELEIÇÃO E COORDENAÇÃO

Curso: **ENGENHARIA DE COMPUTAÇÃO**
Disciplina: **SISTEMAS COMPUTACIONAIS DISTRIBUÍDOS**
Professor: **KLAUSNER VIEIRA GONÇALVES**

ALGORITMOS DE ELEIÇÃO

Usados quando há necessidade de um nó agir como coordenador.

Premissas:
- Cada processo possui um número único
- Todo processo conhece o número do processo de qualquer outro
- Os processos não sabem quais processos estão “vivos” e quais estão “mortos”

São exemplos tradicionais de algoritmos de eleição:
- Algoritmo do valentão
- Algoritmo de anel

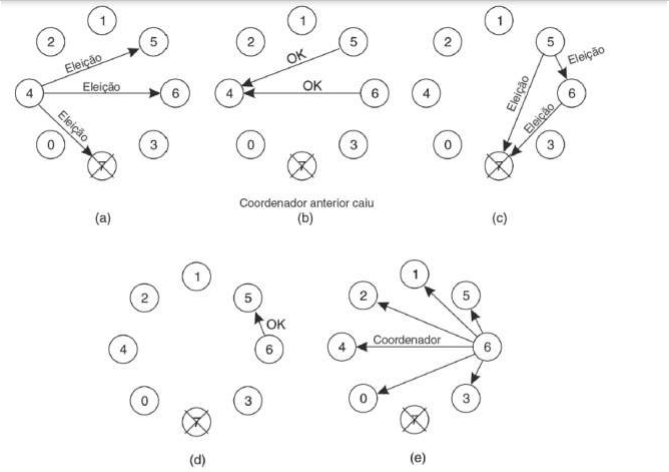
Curso: **ENGENHARIA DE COMPUTAÇÃO**
Disciplina: **SISTEMAS COMPUTACIONAIS DISTRIBUÍDOS**
Professor: **KLAUSNER VIEIRA GONÇALVES**

ALGORITMOS DE ELEIÇÃO - VALENTÃO

- Todos nós possuem um identificador
- Sempre que um nó P percebe que o coordenador não responde, P inicia uma eleição:
  1. P envia uma mensagem ELEIÇÃO a todos os processos de números mais altos
  2. Se nenhum responder, P vence a eleição e se torna o coordenador
  3. Se um dos processos de número mais alto responder, ele toma o poder e o trabalho de P está concluído
- Dessa forma é eleito o nó com maior identificador

Curso: **ENGENHARIA DE COMPUTAÇÃO**
Disciplina: **SISTEMAS COMPUTACIONAIS DISTRIBUÍDOS**
Professor: **KLAUSNER VIEIRA GONÇALVES**

ALGORITMOS DE ELEIÇÃO - VALENTÃO



Algoritmo de eleição do valentão. (a) O processo 4 convoca uma eleição. (b) Os processos 5 e 6 respondem e mandam 4 parar. (c) Agora, cada um, 5 e 6, convoca uma eleição. (d) O processo 6 manda 5 parar. (e) O processo 6 vence e informa a todos.

Curso: **ENGENHARIA DE COMPUTAÇÃO**
Disciplina: **SISTEMAS COMPUTACIONAIS DISTRIBUÍDOS**
Professor: **KLAUSNER VIEIRA GONÇALVES**

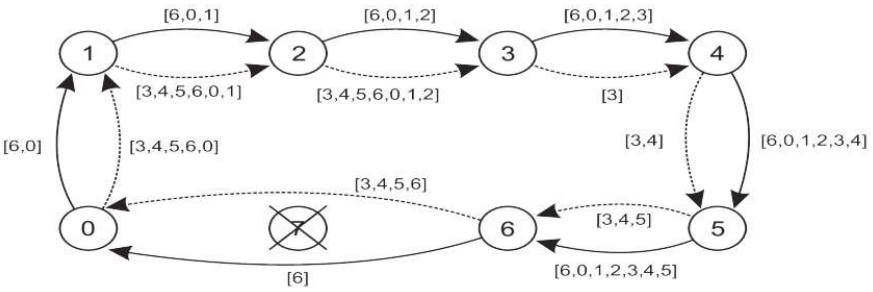
ALGORITMOS DE ELEIÇÃO - ANEL

- Baseado na utilização de anel (físico ou lógico), mas não usa ficha:
  - Quando qualquer processo nota que o coordenador não está funcionando, monta uma mensagem ELEIÇÃO com seu número
  - A mensagem é enviada a seu sucessor ou próximo que esteja em funcionamento
  - A cada etapa, o remetente adiciona seu número de modo a se tornar também um candidato à eleição de coordenador
  - Quando a mensagem retorna ao processo que iniciou a eleição, este envia a mensagem COORDENADOR com o número mais alto da sua lista

Curso: **ENGENHARIA DE COMPUTAÇÃO**
Disciplina: **SISTEMAS COMPUTACIONAIS DISTRIBUÍDOS**
Professor: **KLAUSNER VIEIRA GONÇALVES**

ALGORITMOS DE ELEIÇÃO - ANEL

Nós 3 e 6 iniciam a eleição simultaneamente



Curso: **ENGENHARIA DE COMPUTAÇÃO**
Disciplina: **SISTEMAS COMPUTACIONAIS DISTRIBUÍDOS**
Professor: **KLAUSNER VIEIRA GONÇALVES**

EXCLUSÃO MÚTUA

Curso: **ENGENHARIA DE COMPUTAÇÃO**
Disciplina: **SISTEMAS COMPUTACIONAIS DISTRIBUÍDOS**
Professor: **KLAUSNER VIEIRA GONÇALVES**

EXCLUSÃO MÚTUA

- Uma questão fundamental em SD é a concorrência e a colaboração entre vários processos
- Como garantir que o acesso concorrente de recursos não gere situações de inconsistência de dados?
- São necessárias soluções que garantam o acesso mutualmente exclusivo pelos processos
- Algoritmos distribuídos de exclusão mútua podem ser classificados em duas categorias
  - Baseados em ficha e Baseados em permissão

Curso: **ENGENHARIA DE COMPUTAÇÃO**
Disciplina: **SISTEMAS COMPUTACIONAIS DISTRIBUÍDOS**
Professor: **KLAUSNER VIEIRA GONÇALVES**

EXCLUSÃO MÚTUA

Abordagens baseadas em fichas

- Existe uma ficha disponível no sistema e apenas o processo que possuir a ficha pode acessar o recurso compartilhado
- Evita inanição (completa falta de consumo) - Starvation
- Fácil evitar deadlocks
- O problema é se a ficha for perdida, como recriá-la?

Abordagens baseadas em permissão

- Processo que quer recursos deve antes pedir permissão aos demais

Curso: **ENGENHARIA DE COMPUTAÇÃO**
Disciplina: **SISTEMAS COMPUTACIONAIS DISTRIBUÍDOS**
Professor: **KLAUSNER VIEIRA GONÇALVES**

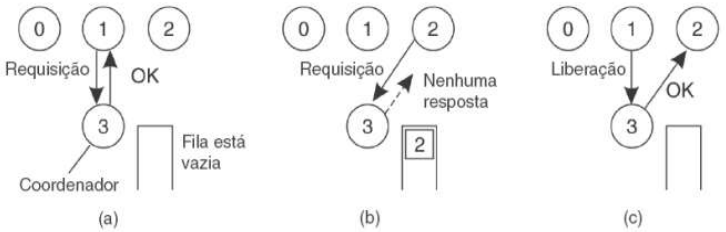
EXCLUSÃO MÚTUA - ALGORITMO CENTRALIZADO

Simula o que é feito em um sistema monoprocessador:

- Um processo é eleito como coordenador
- Sempre que um processo quiser acessar determinado recurso, é necessário pedir permissão ao coordenador, através de uma mensagem
- O coordenador permite acesso ao recurso através de uma mensagem de concessão, desde que nenhum outro processo esteja acessando o recurso naquele momento
- Para usar uma região crítica: REQUISIÇÃO, CONCESSÃO e LIBERAÇÃO

Curso: **ENGENHARIA DE COMPUTAÇÃO**
Disciplina: **SISTEMAS COMPUTACIONAIS DISTRIBUÍDOS**
Professor: **KLAUSNER VIEIRA GONÇALVES**

EXCLUSÃO MÚTUA - ALGORITMO CENTRALIZADO



(a) O processo 1 solicita ao coordenador permissão para acessar um recurso compartilhado. A permissão é concedida. (b) Depois, o processo 2 solicita permissão para acessar o mesmo recurso. O coordenador não responde. (c) Quando o processo 1 libera o recurso, informa ao coordenador, que então responde a 2.

Curso: **ENGENHARIA DE COMPUTAÇÃO**
Disciplina: **SISTEMAS COMPUTACIONAIS DISTRIBUÍDOS**
Professor: **KLAUSNER VIEIRA GONÇALVES**

EXCLUSÃO MÚTUA - ALGORITMO DISTRIBUÍDO

Requer ordenação total de todos os eventos no sistema

- Para isso será usado... Algoritmo de Lamport!

Funcionamento:

- Quando processo deseja acessar um recurso compartilhado, monta uma mensagem que contém o nome do recurso, seu número de processo e a hora corrente (lógica).
- Envia mensagem para todos processos, inclusive ele mesmo

EXCLUSÃO MÚTUA - ALGORITMO DISTRIBUÍDO

Funcionamento (continuação):

- Quando um processo recebe uma mensagem de requisição de outro, executa uma ação de acordo com seu próprio estado em relação ao recurso:
    1. Se o receptor não estiver acessando o recurso nem quer acessá-lo, devolve "OK" ao remetente
    2. Se já tem acesso ao recurso, não responde e coloca requisição em uma fila
    3. Se receptor também quer acessar o recurso, mas ainda não possui a permissão, compara a marca de tempo da mensagem que chegou com a marca de tempo da mensagem que enviou a todos. Se a que recebeu é mais baixa envia OK, senão coloca requisição na fila e não envia

Curso: **ENGENHARIA DE COMPUTAÇÃO**
Disciplina: **SISTEMAS COMPUTACIONAIS DISTRIBUÍDOS**
Professor: **KLAUSNER VIEIRA GONÇALVES**

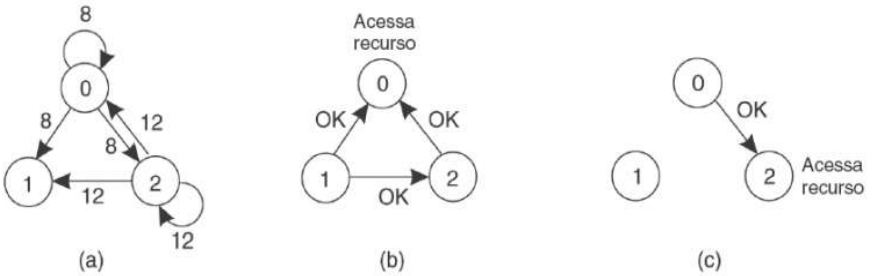
EXCLUSÃO MÚTUA - ALGORITMO DISTRIBUÍDO

Funcionamento (continuação):

- Após enviar requisições que peçam permissão, um processo aguarda recebimento de todas as respostas
- Quando houver permissão de todos, processo acessa o recurso
- Processo libera o recurso enviando um "OK" a todos os processos que estão em sua fila

Curso: **ENGENHARIA DE COMPUTAÇÃO**
Disciplina: **SISTEMAS COMPUTACIONAIS DISTRIBUÍDOS**
Professor: **KLAUSNER VIEIRA GONÇALVES**

EXCLUSÃO MÚTUA - ALGORITMO DISTRIBUÍDO

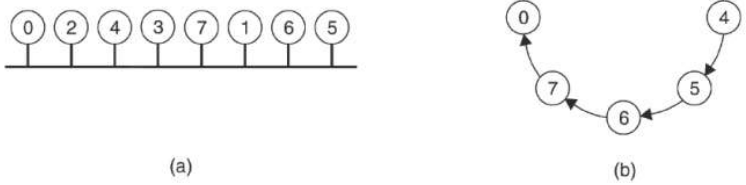


(a) Dois processos querem acessar um recurso compartilhado no mesmo momento. (b) O processo 0 tem a marca de tempo mais baixa, portanto vence. (c) Quando o processo 0 conclui, também envia uma mensagem *OK*, portanto, agora, 2 pode seguir adiante.

Curso: **ENGENHARIA DE COMPUTAÇÃO**
Disciplina: **SISTEMAS COMPUTACIONAIS DISTRIBUÍDOS**
Professor: **KLAUSNER VIEIRA GONÇALVES**

EXCLUSÃO MÚTUA - ALGORITMO TOKEN RING

- Algoritmo baseado em ficha que circula ao redor do anel lógico
- Se o processo quer acessar recurso e tem a ficha, usa recurso e quando acaba passa ficha
- Se não quer usar recurso, passa a ficha



Curso: **ENGENHARIA DE COMPUTAÇÃO**
Disciplina: **SISTEMAS COMPUTACIONAIS DISTRIBUÍDOS**
Professor: **KLAUSNER VIEIRA GONÇALVES**

EXCLUSÃO MÚTUA - ALGORITMO TOKEN RING

- Garante a exclusão mútua
- Não há deadlock, nem starvation
- A dificuldade é recuperar uma ficha perdida
- Como saber o tempo razoável para um token não aparecer?
- Se um processo morreu, um vizinho que tenta enviar o token para ele pode detectar, e anel pode ser reconfigurado

Curso: **ENGENHARIA DE COMPUTAÇÃO**
Disciplina: **SISTEMAS COMPUTACIONAIS DISTRIBUÍDOS**
Professor: **KLAUSNER VIEIRA GONÇALVES**

COMPLEMENTO

Sistemas Distribuídos - Aula 11 – Sincronização
https://www.youtube.com/watch?v=v5oEn3UhMVY

Sistemas Distribuídos - Aula 12 - Abordagens de sincronização
https://www.youtube.com/watch?v=qY6o64vaVQ8

Simulação do Algoritmo de Lamport em JAVA
https://github.com/joao8tunes/RelogiosLamport