07 - Sincronização

- Sincronização não existiria caso o sistema fosse local;
- Como os processos cooperam e sincronizam uns com os outros em Sistemas Distribuídos?
- Sistemas Locais usamos os Semáforos (única CPU).
- Esses métodos não funcionarão em Sistemas Distribuídos, porque? Porque dependem implicitamente da existência de memória compartilhada.

07 - Sincronização

Problemas da Sincronização:

- Dois processos que interagem usando um semáforo;
- Devem ser capazes de acessar o semáforo;
- Se ocorrerem dois eventos em um Sistema Distribuído;
- Como decidir sobre a ordem relativa dos eventos.
- Um evento precede outro evento?
- Difícil determinar se os eventos ocorrem em máquinas diferentes.

07 - Sincronização

Problemas da Sincronização:

- Cada processador (nó de um sistema distribuído) tem um componente chamado relógio;
- Relógios em Sistemas Distribuídos podem:
  - Acusar horas diferentes entre si (defasagem interna);
  - Acusar hora diferente da hora real externa (defasagem externa);
  - Ter velocidades diferentes (defasagem variável);

07 - Sincronização

Problemas da Sincronização:

- Relógios sincronizados são necessários para uma série de aplicações;
- Identificar atualidade de mensagens (antigas devem ser descartadas);
- Aplicações de tempo real;
- Controle de versões;

07 - Sincronização

Parte I – Relógios

- Como sincronizar eventos com base no tempo real?
- Sincronização do relógio;
- Como determinar a ordem relativa?
- Relógios lógicos.

07 - Sincronização

Parte II – Estado e eleições globais

- O que é o “estado global” de um sistema distribuído?
- Como determinamos o “coordenador” de um sistema distribuído?

07 - Sincronização

Parte III – Como sincronizar para compartilhar

- Exclusão mutua;
- Transações distribuídas;

07 - Sincronização

Sincronização de Relógio

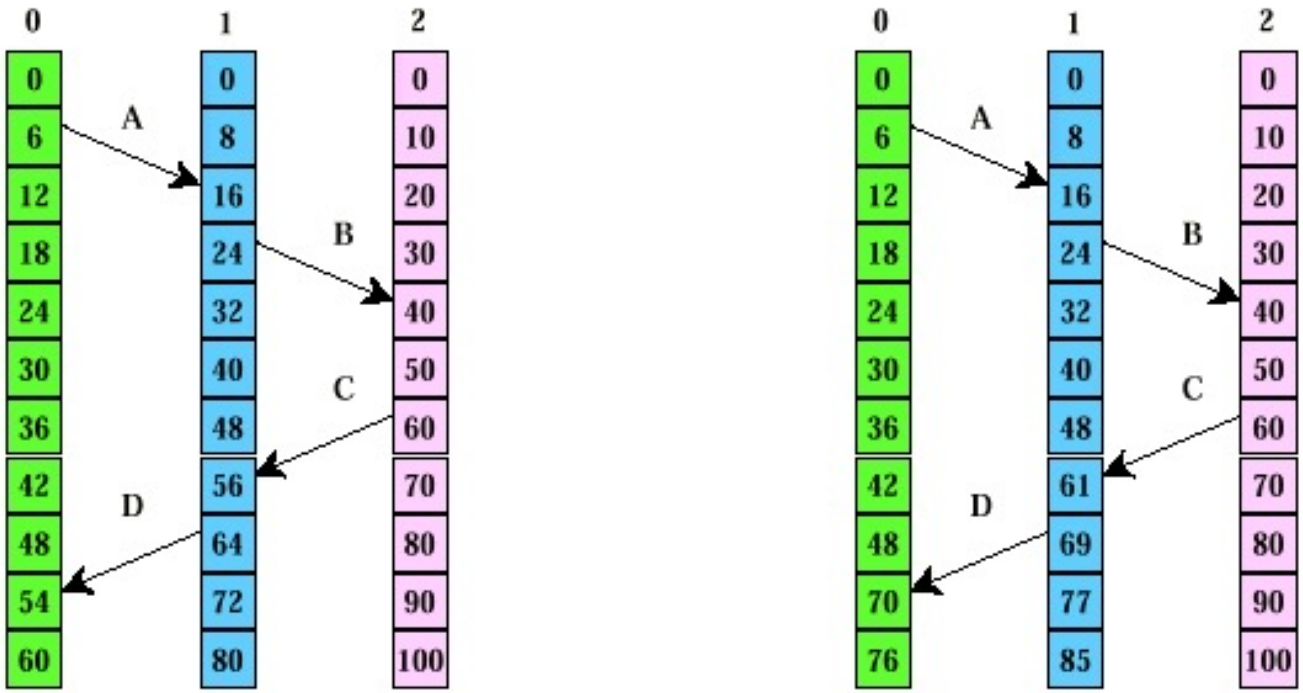
- A sincronização do relógio não precisa ser absoluta! (Lamport, 1978).
- Se dois processos não interagem, seus relógios não precisam ser sincronizados.
- O que importa não é que todos os processos concordem exatamente em que horas são.
- Mas concordam com a ordem que os eventos ocorrem.

07 - Sincronização

Sincronização de Relógio

- Relógios Lógicos:
  - Necessidade apenas da consistência interna dos relógios?
  - Prover identificar a ordem dos eventos;
  - Não se os relógios estão próximos do tempo real;
- Relógios Físicos:
  - Para cenários onde os relógios devem ser apenas iguais;
  - Não devem se desviar do tempo real;

07 - Sincronização



07 - Sincronização

Relógio Lógico de Lamport:

- Eventos são ordenados usando relógios lógicos;
- Em um mesmo processo, relação “acontece – antes” é garantida para eventos que ocorrem;
- No mesmo processo e eventos que se relacionam através de mensagens trocadas;

07 - Sincronização

Relógios Físicos:

- GMT: Greenwich Mean Time;
- BIH: Bureau International de l´Heure;
- TAI: International Atomic Time;
- UTC: Universal Coordinated Time;
- NIST: National Institute of Standart Time;
- WWV: Estação de rádio de ondas curtas;
- GEOS: Geostationary Environment Operational Satellite;

07 - Sincronização

Algoritmos pra Sincronizar Relógios: Algoritmo de Berkeley

- Não dispõe de uma máquina com receptor de Tempo Universal Coordenado;
- O servidor de tempo requer periodicamente de cada máquina, o tempo de seu relógio;
- O servidor de tempo calcula a média e diz para cada máquina como ajustar o relógio;

07 - Sincronização

Relógios Físicos: Problemas.

- O tempo do relógio físico não podem voltar (atrasar) – solução é diminuir a frequência do clock;
- Em uma rede, é difícil medir com precisão o delay da rede;
- Atraso na rede mal calculado para computar o tempo certo no ajuste dos relógios;
- Algoritmo como o de Berkeley tentam atenuar esse problema;

07 - Sincronização

Parte II – Eleição e Coordenação

Muitos algoritmos requerem um processo para atuar como coordenador;

- Exemplo: Coordenador no algoritmo centralizado de exclusão mútua;
- Em geral, não importa qual processo irá atuar como coordenador, mas é preciso elege-lo;

- Como eleger um coordenador?

07 - Sincronização

Parte II – Eleição e Coordenação

Premissas:

- Cada processo possuí um número único (por exemplo, seu endereço de rede) sepadado;
- Todo processo conhece o número do processo de qualquer outro processo;
- Os processos não sabem quais processos estão “vivos” e quais estão “mortos”;

07 - Sincronização

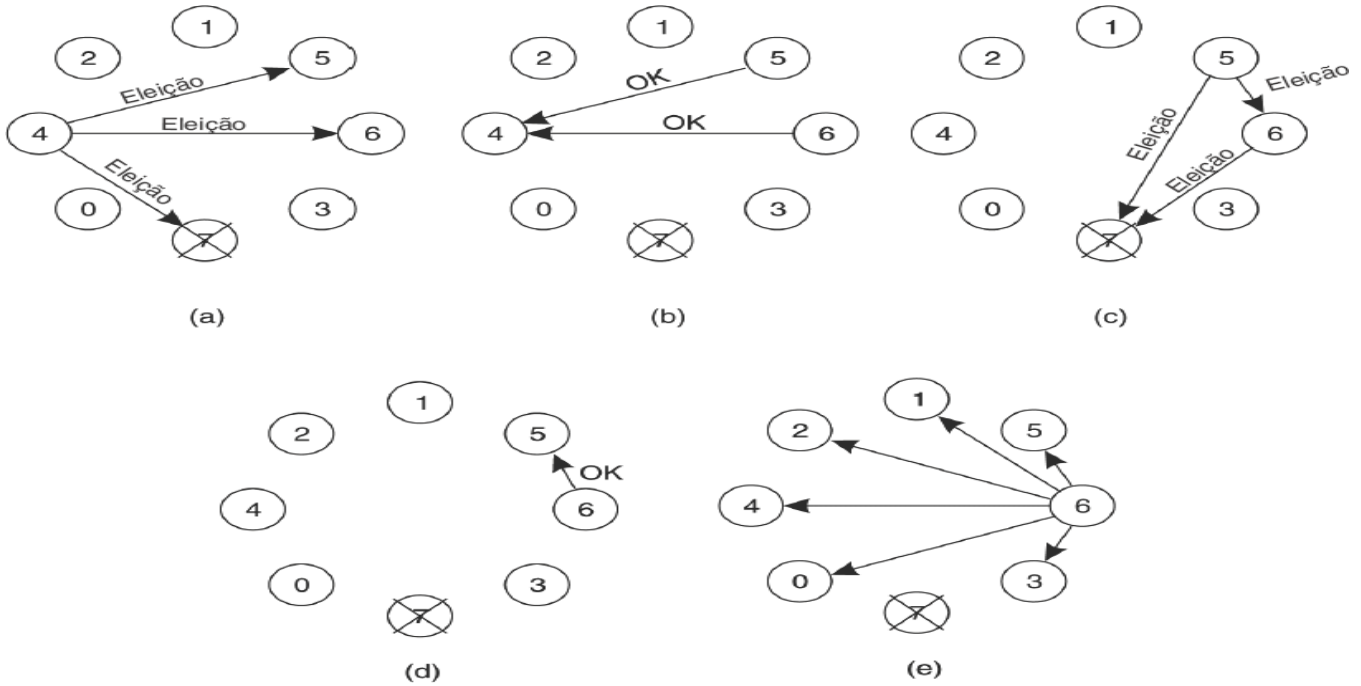
Parte II – Eleição e Coordenação

Abordagem:

- Localize um processo com o número de processo maior e eleja-o como coordenador;
- Os algoritmos de eleição diferem na forma como fazem a localização;

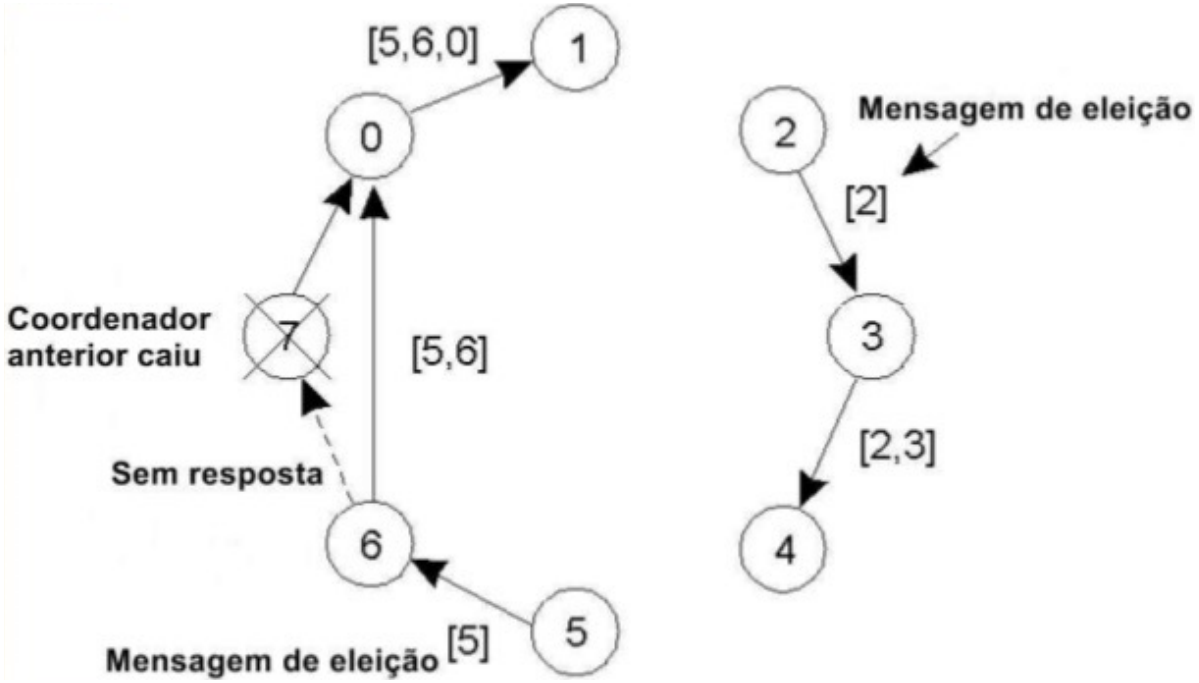
07 - Sincronização

Algoritmo do Valentão



07 - Sincronização

Algoritmo do Anel



07 - Sincronização

Parte III – Exclusão mútua em Sistemas Distribuídos

- A melhor maneira de se conseguir exclusão mútua em um Sistema Distribuído é imitar o que foi feito em um sistema local;
- Um processo é eleito como coordenador;
- Quando um processo quer entrar na região crítica, ele envia uma mensagem requisitando ao coordenador;
- Se nenhum outro processo esta na região crítica, o coordenador da o OK;

07 - Sincronização

Parte III – Exclusão mútua em Sistemas Distribuídos

Algoritmo Centralizado:

- Processo 1 pede permissão ao coordenador para entrar em uma região crítica. A permissão é concedida.
- Processo 2 então pede permissão para entrar na mesma região crítica. O coordenador não responde.
- Quando o Processo 1 saí da região crítica, ele avisa o coordenador, quando então responde a 2.

07 - Sincronização

Parte III – Exclusão mútua em Sistemas Distribuídos

Algoritmo Centralizado: Vantagens.

- Obviamente garante exclusão mútua;
- É justo (os pedidos são concedidos na ordem em que são recebidos);
- Sem inanição (um processo não espera para sempre);
- Fácil de implementar (apenas 3 mensagens: solicitação, concessão e liberação);

07 - Sincronização

Parte III – Exclusão mútua em Sistemas Distribuídos

Algoritmo Centralizado: Desvantagens.

- Coordenador: um único ponto de falha;
- Traz um gargalo no desempenho;
- Se os processos normalmente bloqueiam após fazer um pedido, eles não podem distinguir um coordenador morto da “permissão negada”;

07 - Sincronização

Parte III – Exclusão mútua em Sistemas Distribuídos

Algoritmo Centralizado: Desvantagens.

- Coordenador: um único ponto de falha (falhou o coordenador, parou o sistema até que eleja um novo coordenador);
- Traz um gargalo no desempenho;
- Se os processos normalmente bloqueiam após fazer um pedido, eles não podem distinguir um coordenador morto da “permissão negada”;

07 - Sincronização

Parte III – Exclusão mútua em Sistemas Distribuídos

Algoritmo Distribuído

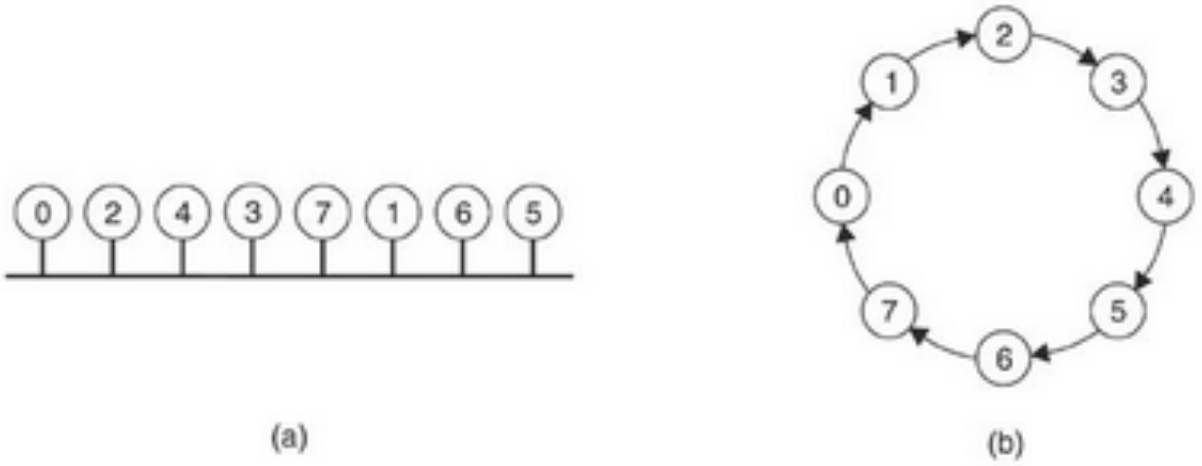
- Dois processos querem entrar na mesma região crítica ao mesmo tempo;
- Processo 0 tem o timestamp mais baixo, então ganha;
- Quando o processo 0 finaliza ele envia um OK, então o processo 2 agora pode entrar na região crítica;

- Problema: Falha de qualquer processo “crashes”;

07 - Sincronização

Parte III – Exclusão mútua em Sistemas Distribuídos

Algoritmo Distribuído: Algoritmo Token Ring



(a) Grupo de processos não ordenados em uma rede. (b) Um anel lógico é construído em software.

07 - Sincronização

Parte III – Exclusão mútua em Sistemas Distribuídos

Algoritmo Distribuído: Algoritmo Token Ring – Vantagens

- Garante a exclusão mútua (apenas 1 processo possui o token a qualquer instante;
- Sem inanição (o token circula entre os processos em uma ordem bem definida;

07 - Sincronização

Parte III – Exclusão mútua em Sistemas Distribuídos

Algoritmo Distribuído: Algoritmo Token Ring – Desvantagens

- Regeneração do Token se ele for perdido;
- Detectar que o token esta perdido é difícil, a rede pode estar sobrecarregada;