Sistemas Distribuídos
Aula 17

Aula passada

- Exclusão mútua
- Algoritmo centralizado
- Algoritmo de Lamport
- *Token Ring*

Aula de hoje

- Eleição de líder
- Algoritmo do valentão
- Algoritmo em anel

Coordenação

- Muitos sistemas (e algoritmos) distribuídos necessitam de um coordenador
  - algoritmo de Berkeley para sincronizar relógios
  - exclusão mútua centralizada
  - *tracker* no Bittorrent
- Em muitos casos, coordenador escolhido manualmente (*hardcoded*)
  - ponto único de falha, inflexibilidade

- Como construir sistemas distribuídos com coordenador?

Coordenação Dinâmica

- **Ideia**: determinar coordenador dinamicamente
  - escolha faz parte do sistema distribuído
  - arquitetura com coordenador deixa de ser centralizada
  - papel do coordenador pode ser realizado por outros processos/máquinas

- Como determinar o coordenador?
  - eleição de líder (coordenador)
  - algoritmo distribuído que ao final todos concordam no líder

Eleição de Líder

- **Ideia**: escolher um processo para ser o coordenador
  - todos os outros processos precisam concordar
- Assumir que todos processos possuem identificador único
- **Ideia**: Líder é o processo com maior identificador

- Como encontrar o líder?
- Ideia 0: todos enviam mensagem para todos, para que todos conheçam os identificadores
  - proibitivo!

Algoritmo do Valentão (*Bully Algorithm*)

- Proposto em 1982 por Garcia-Molina
  - processos tem ID único, rede confiável e FIFO
  - três tipos de mensagem: *eleição, OK, coordenador*
- Processo P que detecta falha do coordenador inicia eleição (visando substituir o coordenador)

  1) Envia mensagem eleição com seu ID a todos os processos

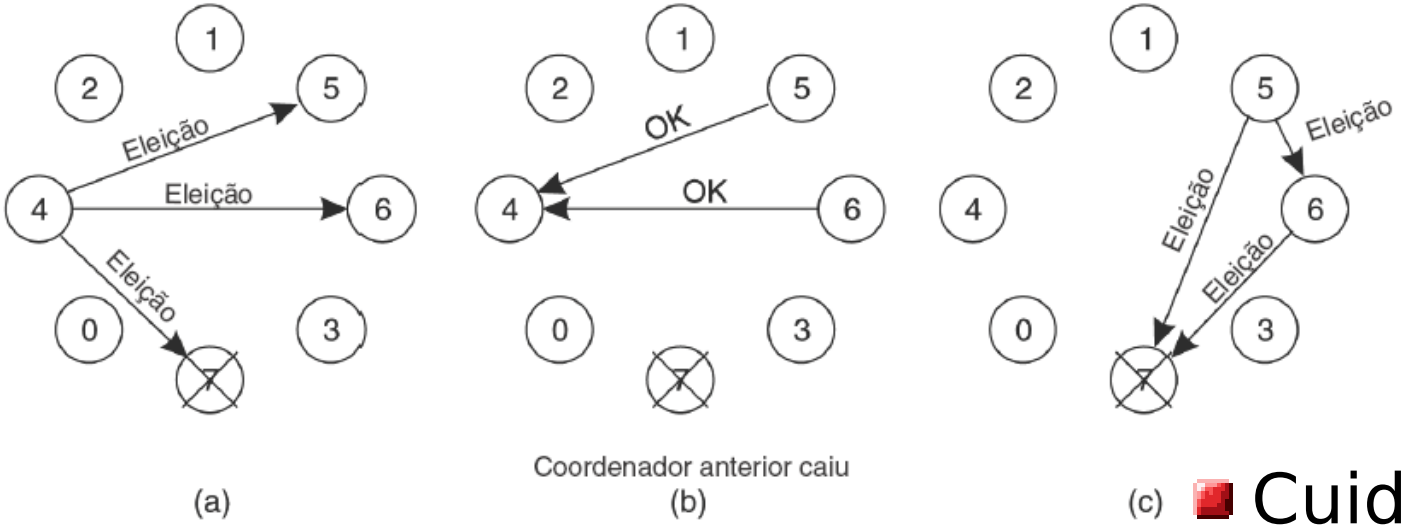
  2) Se não recebe resposta, P envia mensagem coordenador a todos processos (se torna coordenador)

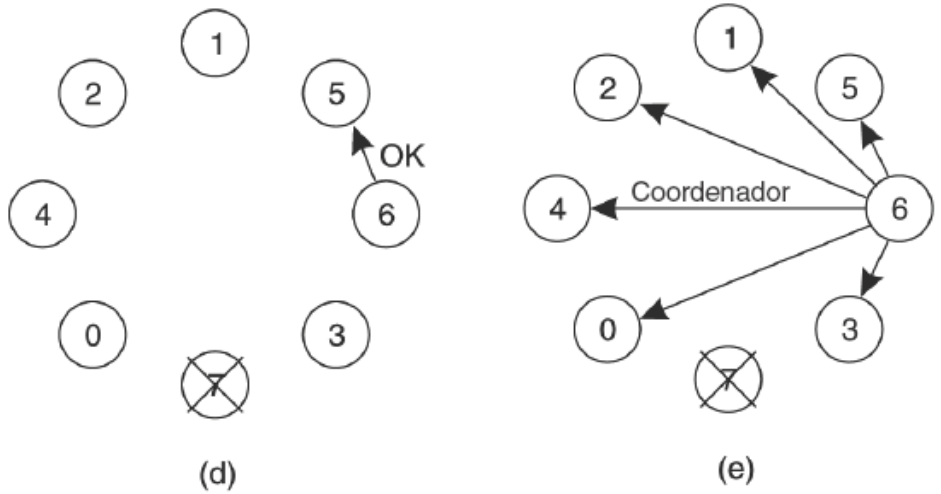
  3) Se recebe mensagem OK de algum processo, P desiste

Algoritmo do Valentão (*Bully Algorithm*)

- Processo Q que recebe mensagem de eleição
  1) Se ID for menor, não faz nada
  2) Se ID for maior, envia mensagem OK a P
  3) Inicia uma eleição (faz o papel de P)
- Eleição eventualmente termina, pois algum processo possui maior ID
  - todos desistem, e ganha o de maior ID (valentão)

Exemplo do Valentão





- Cuidado: exemplo ilustrativo não mostra todas as mensagens enviadas!
- Mensagem de eleição sempre enviada a todos
- Processo 4 inicia eleição;

Complexidade do Valentão

- Quantas mensagems são enviadas?
- Depende de quem inicia a eleição
- Pior caso: menor ID inicia eleição
  - $\Theta(n^2)$ mensagens
- Melhor caso: maior ID inicia eleição
  - $\Theta(n)$ mensagens

- Como tentar evitar o pior caso?
- Retardar início de nova eleição, se ID for pequeno
  - tanto ao detectar falha, quanto quando receber ID maior

Eleição em Anel

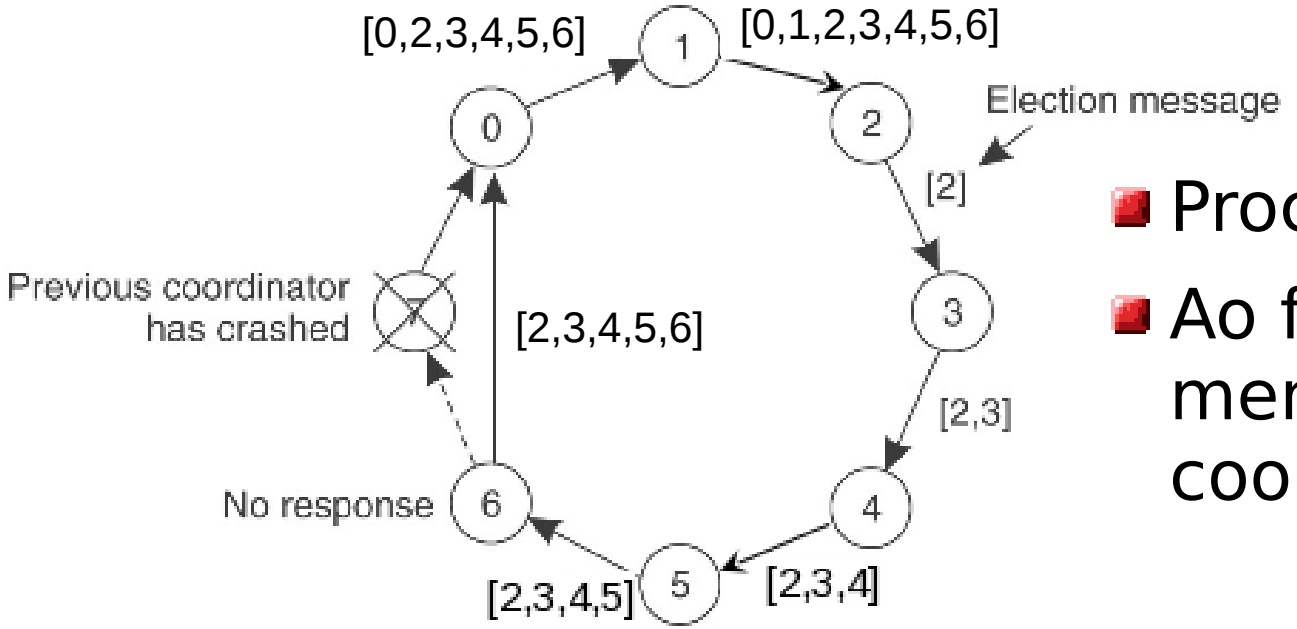
- Organizar processos em topologia de anel lógico (já vimos isto antes)
  - mas todos sabem contactar todos diretamente
- Dois tipos de mensagem: eleição, coordenador
- Processo P que detecta falha do coordenador inicia uma eleição

  1) envia mensagem de eleição com seu ID ao próximo processo operacional no anel

  2) ao receber mensagem que circulou o anel, envia mensagem coordenador a todos (maior ID recebido)

Eleição em Anel

- Processo Q que recebe uma mensagem de eleição
  - 1) adiciona seu ID na mensagem
  - 2) envia mensagem de eleição ao próximo processo operacional no anel
- Exemplo



- Processo 2 inicia eleição
- Ao final, envia mensagem coordenador com 6

Propriedades e Complexidade

- Quantas mensagems são enviadas?
- 2 voltas no anel: $2n = \Theta(n)$
- Tamanho das mensagens?
- Cresce a cada passo, total de n identificadores ao final

- O que acontece se dois processos começam eleição, simultaneamente?
  - eleições independentes, funciona mas dobra número de mensagens
- O que acontece se processo falha durante eleição?