Sistemas Distribuídos
Aula 8: VCube, Propriedades & Versão 2

Prof. Elias P. Duarte Jr.
Universidade Federal do Paraná (UFPR)
Departamento de Informática
www.inf.ufpr.br/elias/sisdis

Sumário

- O Algoritmo VCube: relembrando, praticando
- A Latência do VCube no pior caso: $\log^2 N$ rodadas
- Pior caso do número de testes
- A Versão 2 do VCube: NlogN testes a cada logN rodadas

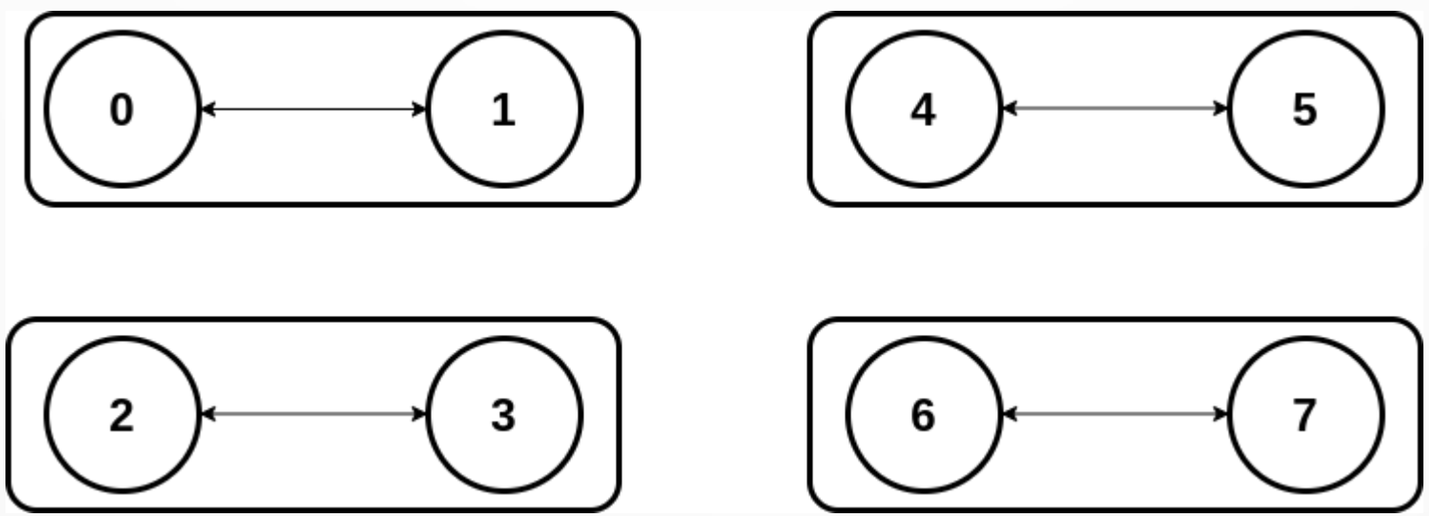
O Algoritmo VCube

- O VCube é um algoritmo distribuído hierárquico
- Os N processos são organizados em *clusters*
- Cada processo tem logN clusters
  - 4 processos: 2 clusters, 8 processos: 3 clusters,

    16 processos: 4 clusters, ... 1024 processos: 10 clusters...
- Os clusters começam com 2 nodos, e vão dobrando a cada rodada de testes...
- O maior cluster tem metade (N/2) dos processos

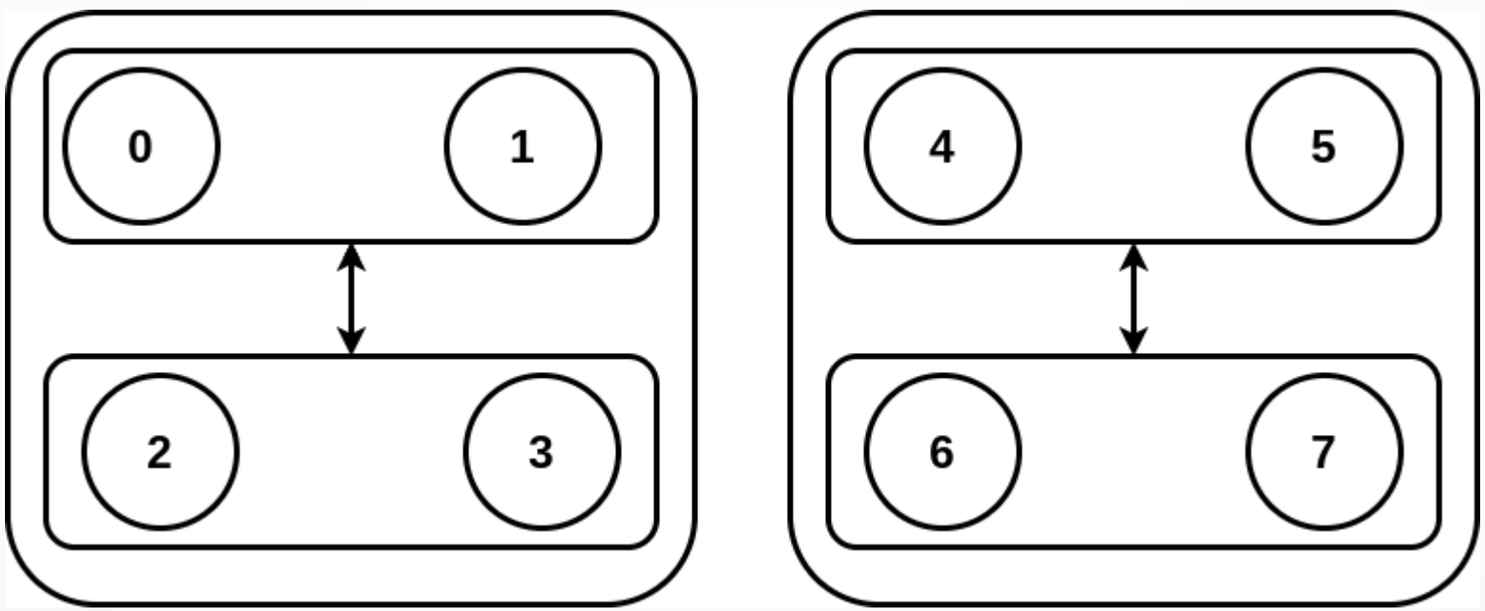
VCube: Estratégia de Testes

- No início os testes são executados em clusters com dois nodos



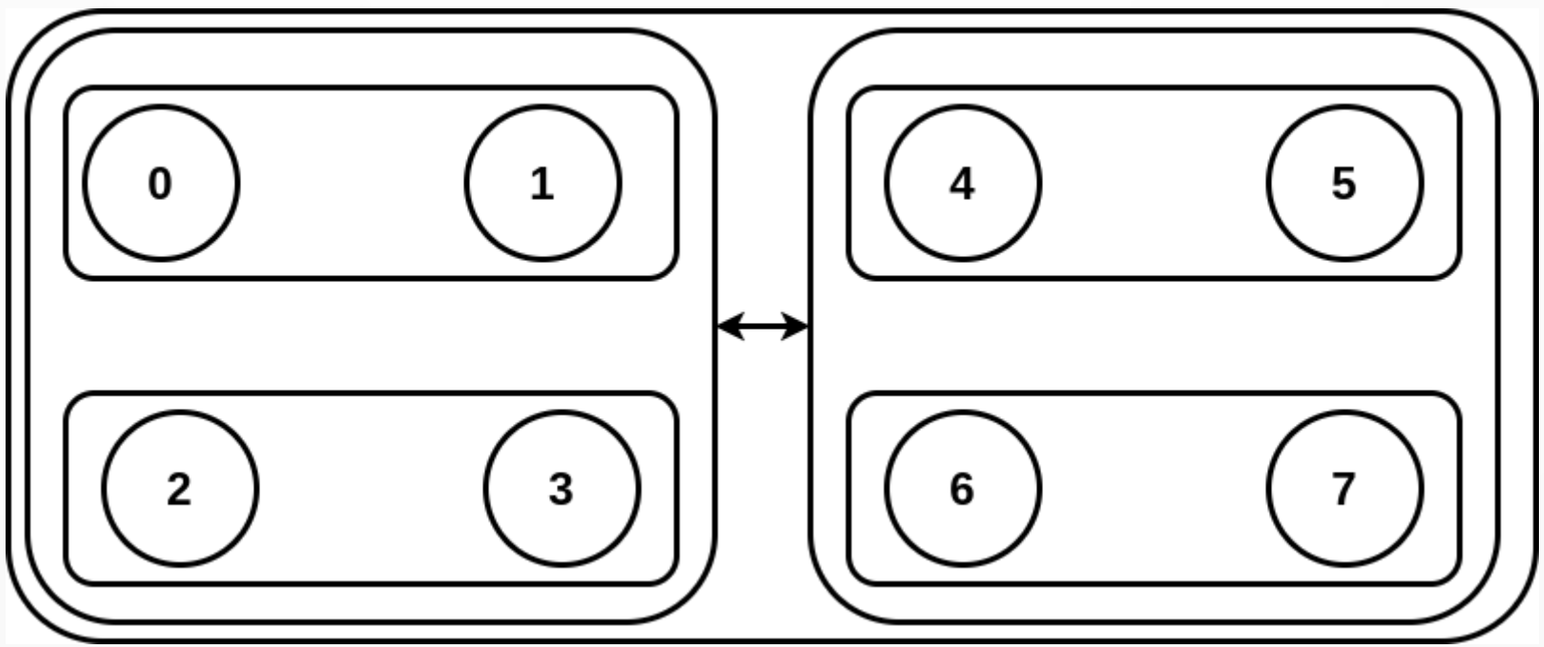
VCube: Aumenta o Cluster

- Na segunda rodada os clusters têm 4 processos



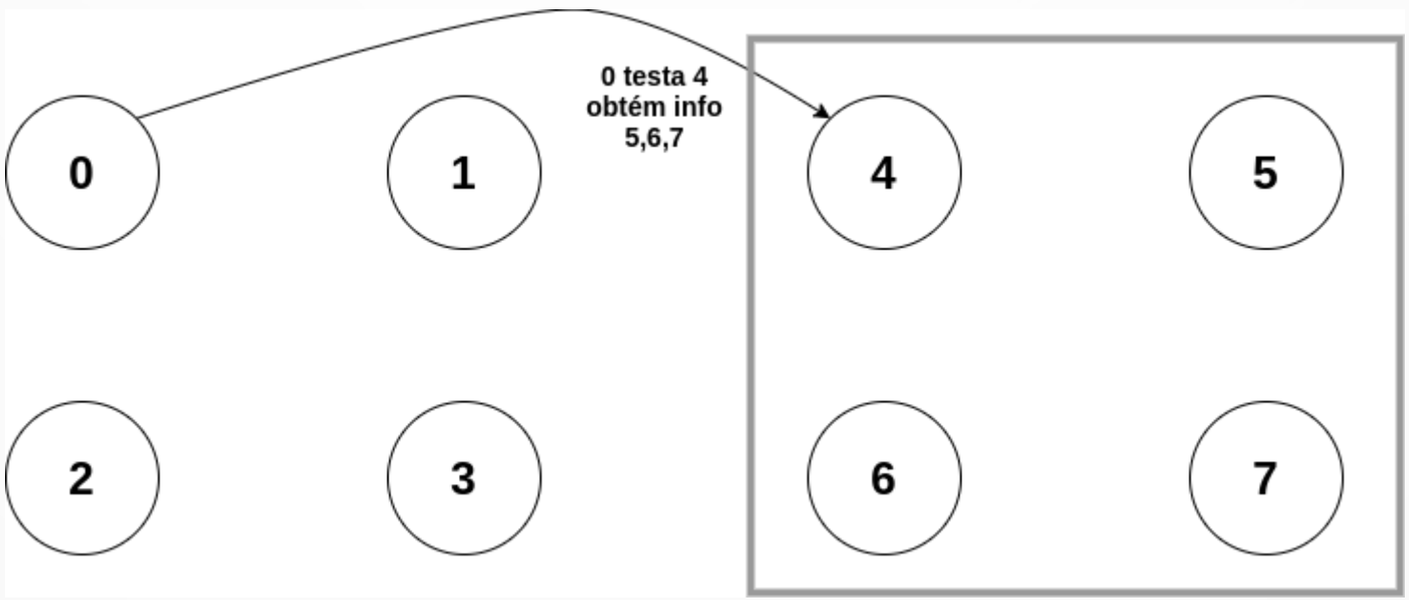
VCube: O Maior Cluster

- Na última rodada o cluster tem N/2 processos, na seguinte tudo recomeça no cluster de 2 processos



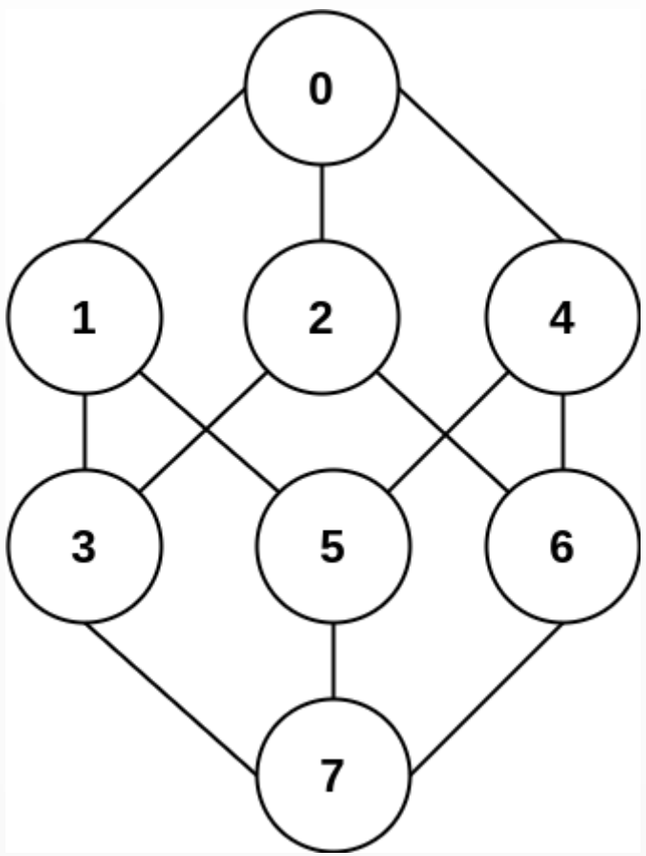
VCube: Obtenção de Informações

- Quanto um processo correto é testado, o testador obtém as informações do cluster testado



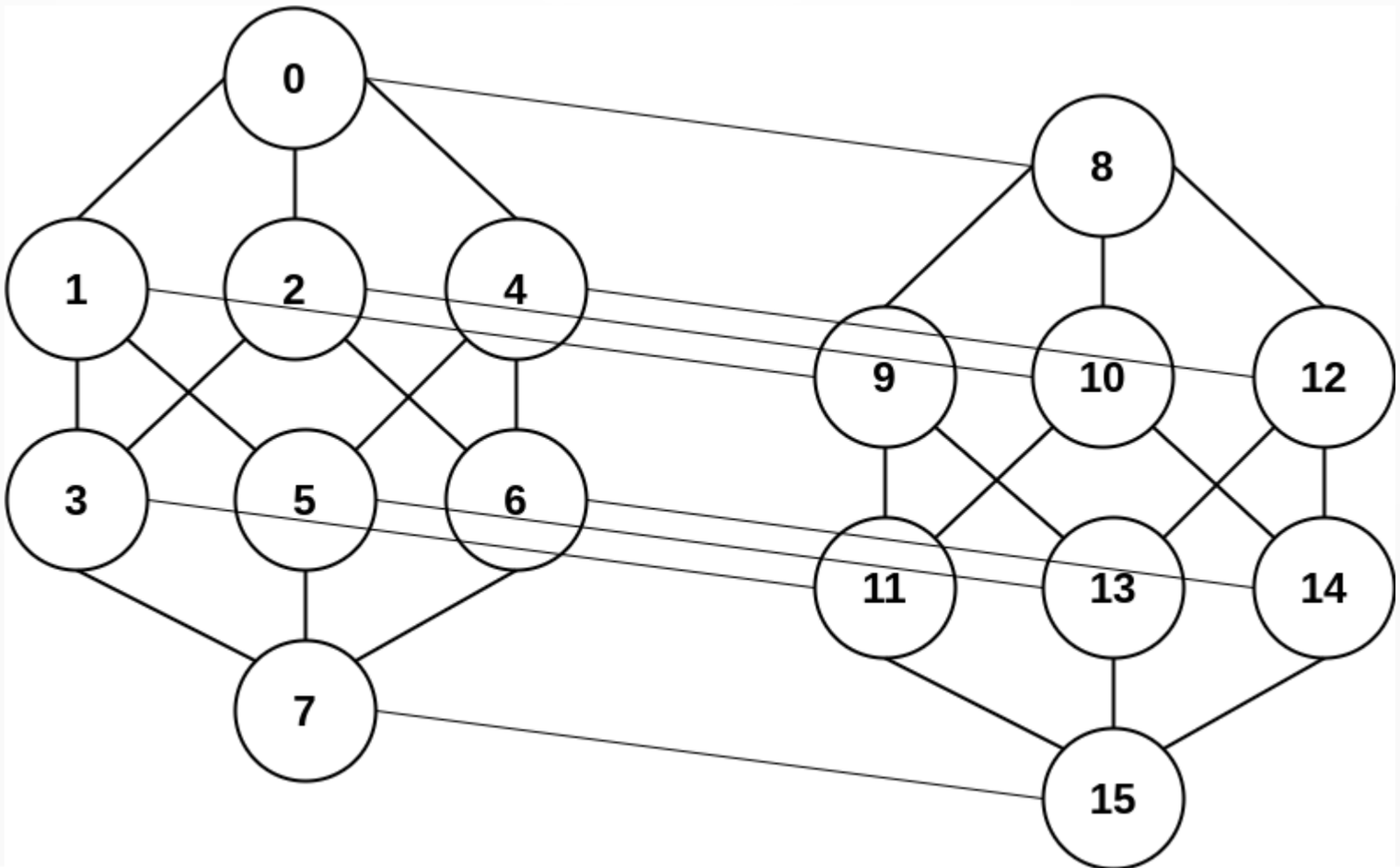
VCube com Todos os Processos Corretos: Hipercubo

- Este é um hipercubo formado por 8 processos
- Consegue ver o cubo?



Hipercubo de 4 Dimensões

- Veja que dobramos o número de nodos, são dois hipercubos de 8 nodos



Hipercubo: Topologia Escalável

- O hipercubo têm diversas propriedades logarítmicas
  - logN – logaritmos base 2, sempre
- Grau de cada nodo, diâmetro (distância máxima)
- Quando dobra o número de nodos, o incremento é de 1 unidade
- Propriedades logarítimicas correspondem a escalabilidade: funciona bem quando N cresce

VCube: Muito Mais que Hipercubo

- VCube: corresponde a um hipercubo virtual quando todos os processos estão corretos
- O que acontece quando processos falhos?
- Lembre-se: topologia subjacente: *fully-connected*
- VCube: quando processos falham a topologia se reorganiza, mantendo diversas propriedades logarítmicas
- O testador continua testando no cluster, até encontrar processo correto ou testar todos os processos falhos

VCube: A Função do Cluster

- A função C(i,s): s-ésimo cluster no processo i
- O tamanho do cluster é sempre uma potência de 2, há logN clusters
- A função C(i,s) retorna a lista de processos (em ordem) que devem ser testados pelo testador *i* nos clusters s=1,2,...,logN

A Função C(i,s), N=8 processos

s	C(0,s)	C(1,s)	C(2,s)	C(3,s)	C(4,s)	C(5,s)	C(6,s)	C(7,s)
1	1	0	3	2	5	4	7	6
2	2,3	3,2	0,1	1,0	6,7	7,6	4,5	5,4
3	4,5,6,7	5,6,7,4	6,7,5,4	7,4,5,6	0,1,2,3	1,2,3,0	2,3,0,1	3,0,1,2

O Algoritmo Distribuído VCube

- Processos executam testes periodicamente em intervalos de testes (p.ex.30s ou 10ms)
- Cada processo i mantém o vetor local $State_i[N]$
- Em cada intervalo, um processo testa um dos seus logN clusters
- Testes são executados sequencialmente em cada cluster, até um processo correto ser testado
  - versão 1, hoje vamos ver uma outra versão
- Ao testar um processo correto: testador obtém informações sobre todos os demais processos do cluster que não testou neste intervalo

VCube: Rodada de Testes

- Uma rodada de testes do algoritmo VCube acontece quando todos os processos corretos testaram pelo menos um de seus clusters

VCube: Rodada de Testes

- Uma rodada de testes do algoritmo VCube acontece quando todos os processos corretos testaram pelo menos um de seus clusters
- Não há relógio global, cada processo usa seu relógio local, que não está sincronizado com os demais
- Os processos podem começar seus intervalos de testes em instantes distintos, podem testar clusters de tamanhos distintos ao mesmo tempo
- Em um cluster podem haver um ou mais processos falhos: processos demoram tempos diferentes testando 1 cluster
- O processo mais lento determina a rodada

VCube: Premissas

- No VCube os processos falham por parada (*crash*)
- Enlaces não falham: sistema *fully-connected*
- Assume que um processo correto executa testes perfeitos: determina corretamente o estado do processo testado (100% perfeito!)
- Portanto: o sistema é síncrono

Algoritmo VCube executado pelo processo i:

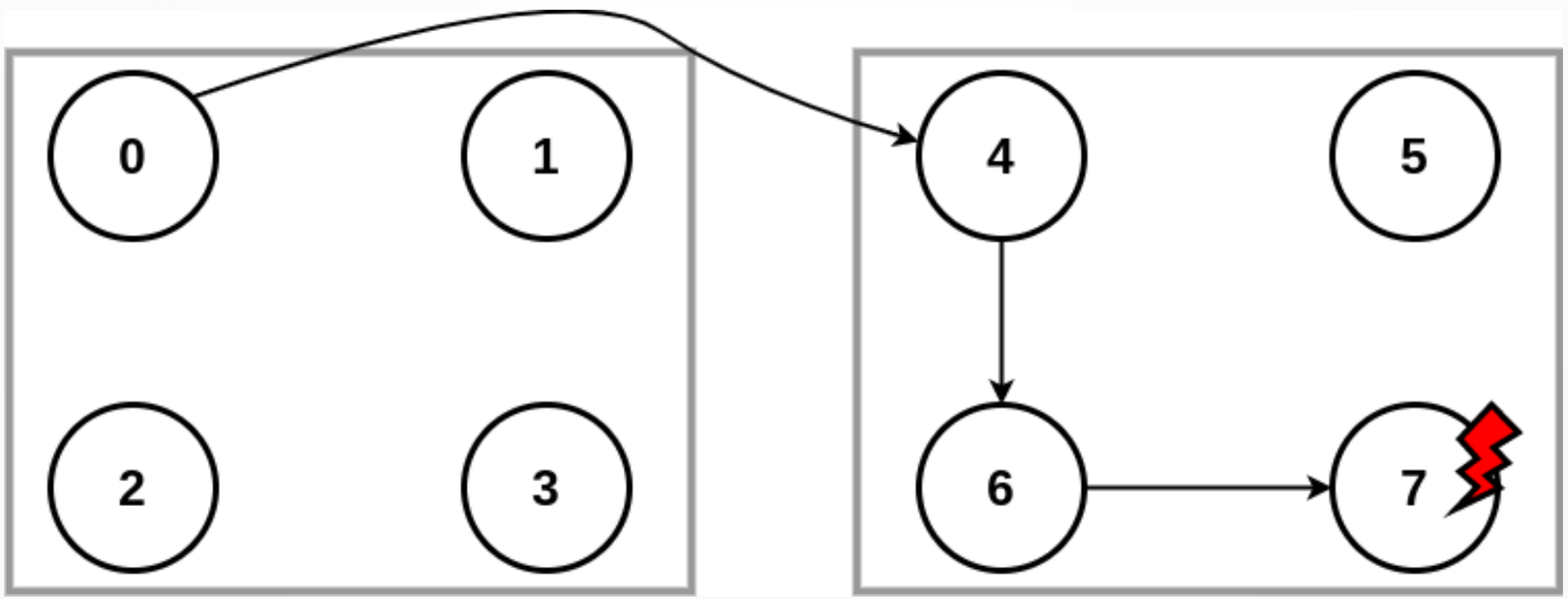
```
repita
   para s de 1 até logN e voltando a 1 faça
     repita
      execute um teste no próximo processo de C(i,s);
      se o processo foi testado correto
      então obtenha info cluster s e atualize Statei[N];
     até ter testado um processo correto ou todos falhos;
     durma até o próximo intervalo de testes;
   fim-para;
para-sempre;
```

Propriedades do VCube: Latência e Número de Testes

- Qual a latência do VCube no pior caso?

Propriedades do VCube: Latência e Número de Testes

- Qual a latência do VCube no pior caso?
- Não é logN, mas $\log^2 N$ rodadas de testes: vamos provar agora



Prova da Latência do VCube

- Definição: o grafo de testes G=(V,T) é construído pelo algoritmo VCube ao longo de logN rodadas de testes consecutivas
- Os vértices do grafo são os processos do sistema
- As arestas do grafo correspondem aos testes executados sobre processos corretos
- Desta forma, existe uma aresta de cada processo correto para um processo em cada um dos logN clusters testados

Prova da Latência do VCube

- Lema 1: Dado um processo i, um cluster qualquer C(i,s) e um instante de tempo qualquer: em no máximo logN rodadas de testes i testa C(i,s)
  - Prova: pela definição do algoritmo, em 1 intervalo de testes um processo correto testa um cluster até encontrar um processo correto ou testar todos os processos falhos
  - Em 1 rodada de testes todos os processos corretos completam o teste de 1 de seus clusters

Prova da Latência do VCube

- Continuando o Lema 1
  - Cada processo tem logN clusters, portanto em logN rodadas de testes todos os processos testaram todos os seus clusters
  - Assim, no pior caso, para um instante de tempo imediatamente após um processo testar um cluster, serão necessárias no máximo logN rodadas de testes para aquele processo testar aquele cluster novamente.

Prova da Latência do VCube

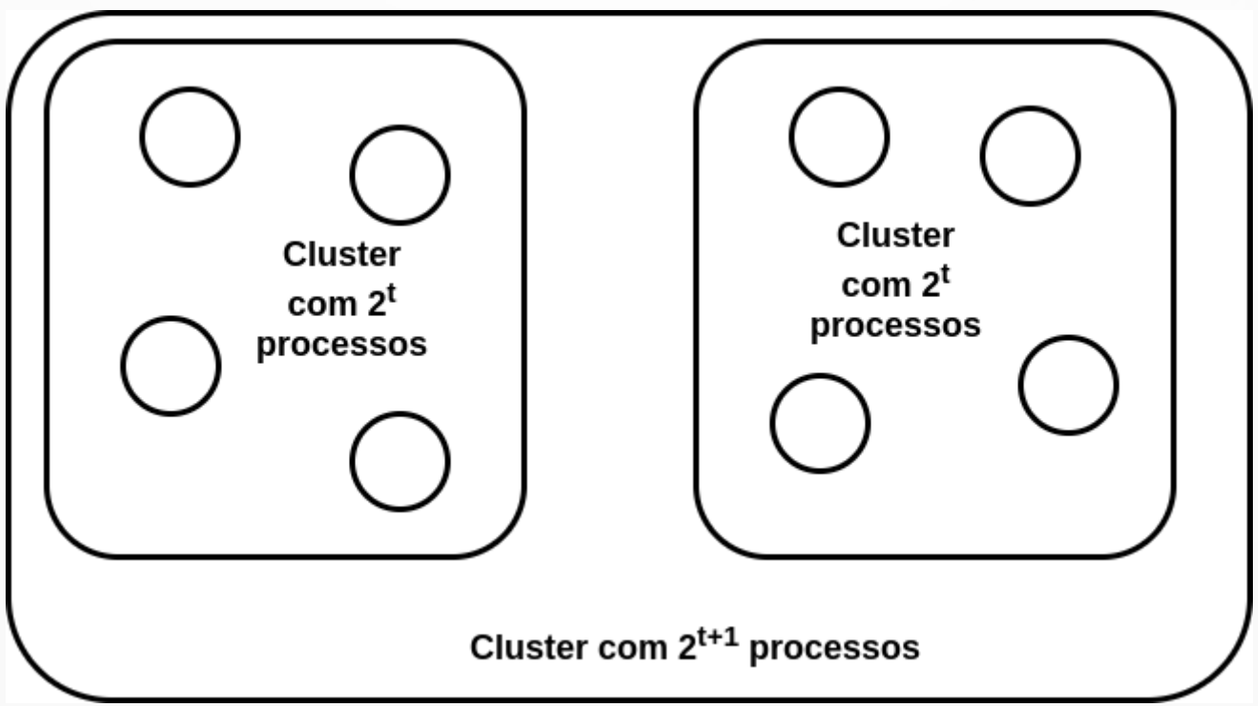
- Teorema 1: O menor caminho entre dois vértices do grafo de testes G=(V,T) contém no máximo logN arestas.
  - Prova: Indução em t, considerando $N = 2^t$
  - Base da indução: considere um sistema com $2^1 = 2$ processos. Cada processo testa o outro em cada rodada testes, portanto o menor caminho entre eles tem sempre 1 aresta.
  - Hipótese da Indução: em um sistema com 2^t processos o maior caminho tem t arestas

Prova da Latência do VCube

- Continuação do Teorema 1
  - Passo da Indução: vamos mostrar que, com a hipótese da indução verdadeira, em um sistema de 2^{t+1} processos o caminho máximo é de t+1 arestas
  - Um sistema com 2t+1 processos corresponde a dois sitemas com 2t processos

Prova da Latência do VCube

- Continuação do Teorema 1
  - Passo da Indução: Um sistema com 2t+1 processos corresponde a dois sitemas com 2t processos

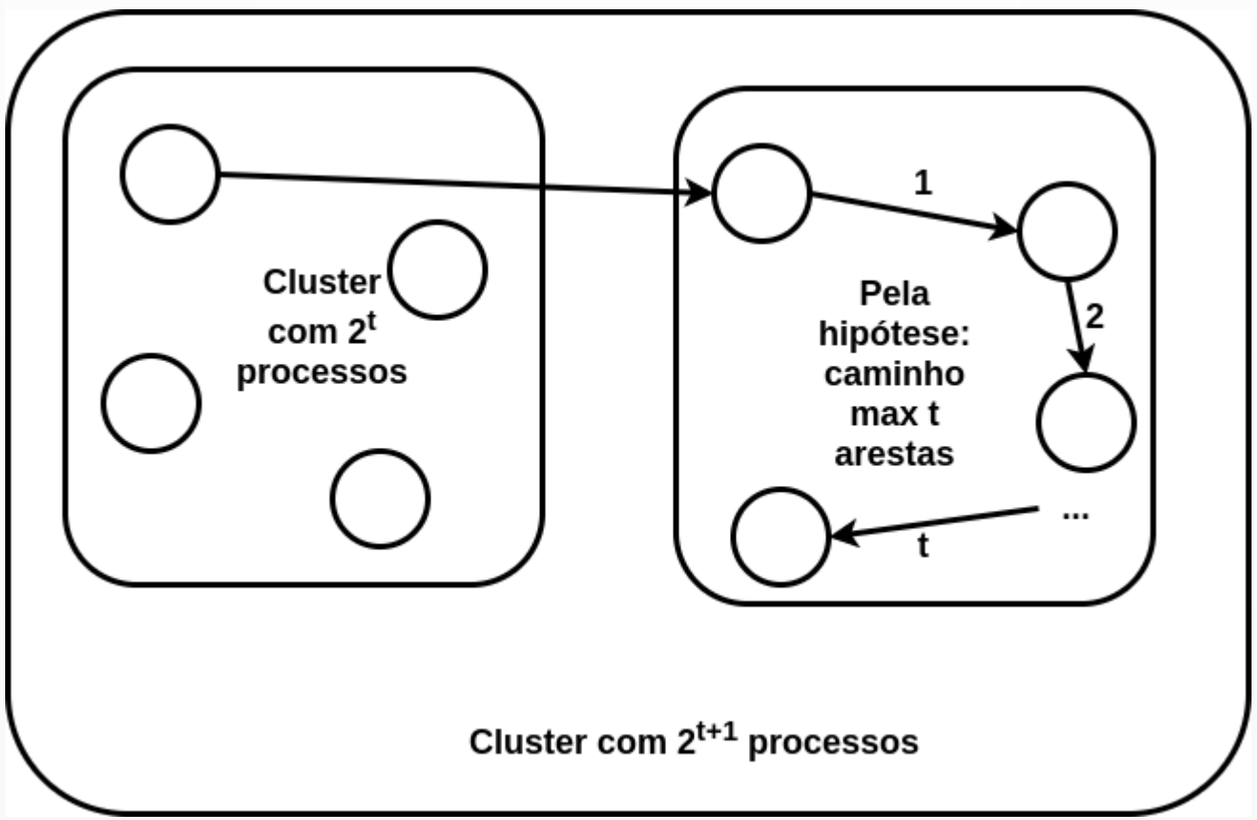


Prova da Latência do VCube

- Continuação do Teorema 1
  - Sem perda de generalidade, considere o cluster da direita
  - De acordo com a hipótese da indução, o caminho mínimo dentro deste cluster com 2^t processos tem, no máximo t arestas
  - Cada processo do cluster da esquerda testa 1 processo do cluster da direita
  - Portanto a distância mínima entre qualquer processo do cluster da esquerda e qualquer processo do cluster da direita é de t+1 arestas

Prova da Latência do VCube

- A distância máxima em um sistema com 2^{t+1} processos tem t+1 arestas

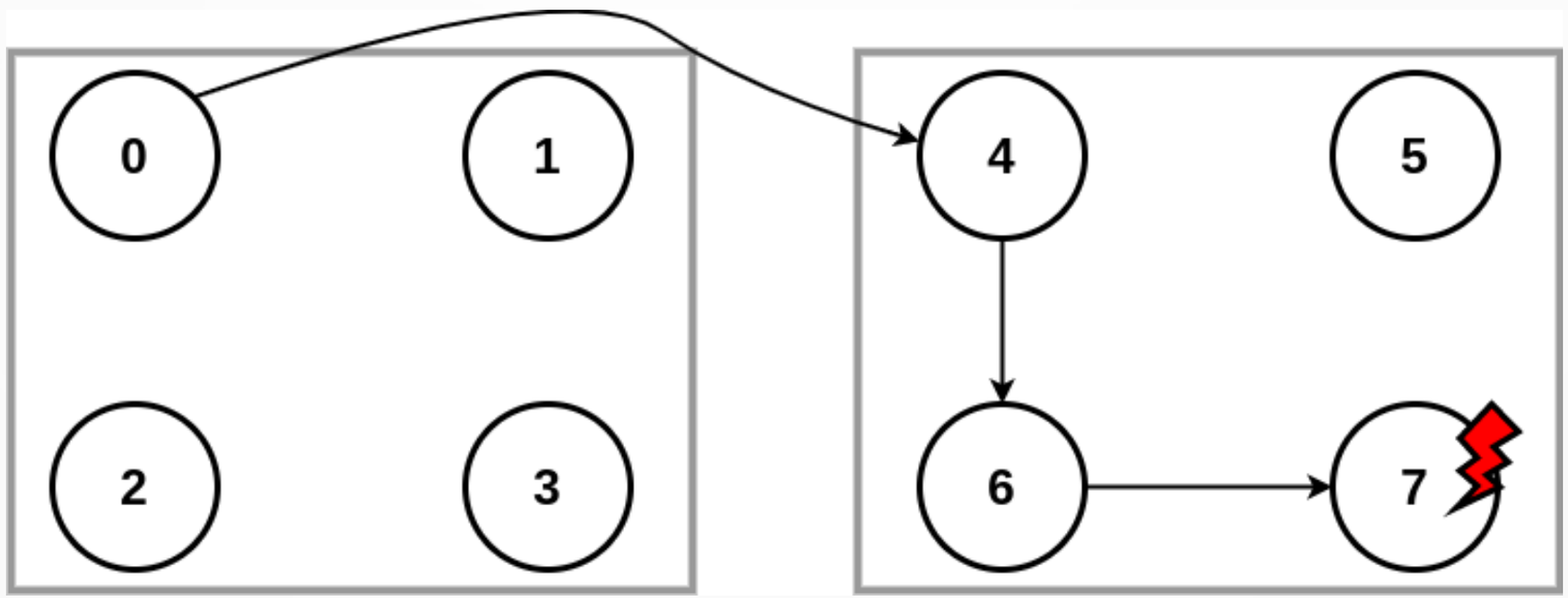


Propriedades do VCube: Latência

- Teorema 2: Considere o sistema em uma determinada situação de falhas. Em no máximo $\log^2 N$ rodadas de testes todos os processos corretos completam o diagnóstico.
  - Prova: De acordo com o Lema 1, podem ser necessárias logN rodadas de testes para um processo testar um de seus clusters. De acordo com o Teorema 1, o caminho entre dois processos no grafo de testes tem no máximo logN arestas (testes). Desta forma: para a execução de cada um destes testes podem ser necessárias logN rodadas. Assim: no total são necessárias logN*logN = $\log^2 N$ rodadas para completar o diagnóstico.

Propriedades do VCube: Latência e Número de Testes

- A latência do VCube no pior caso: $\log^2 N$ rodadas de testes

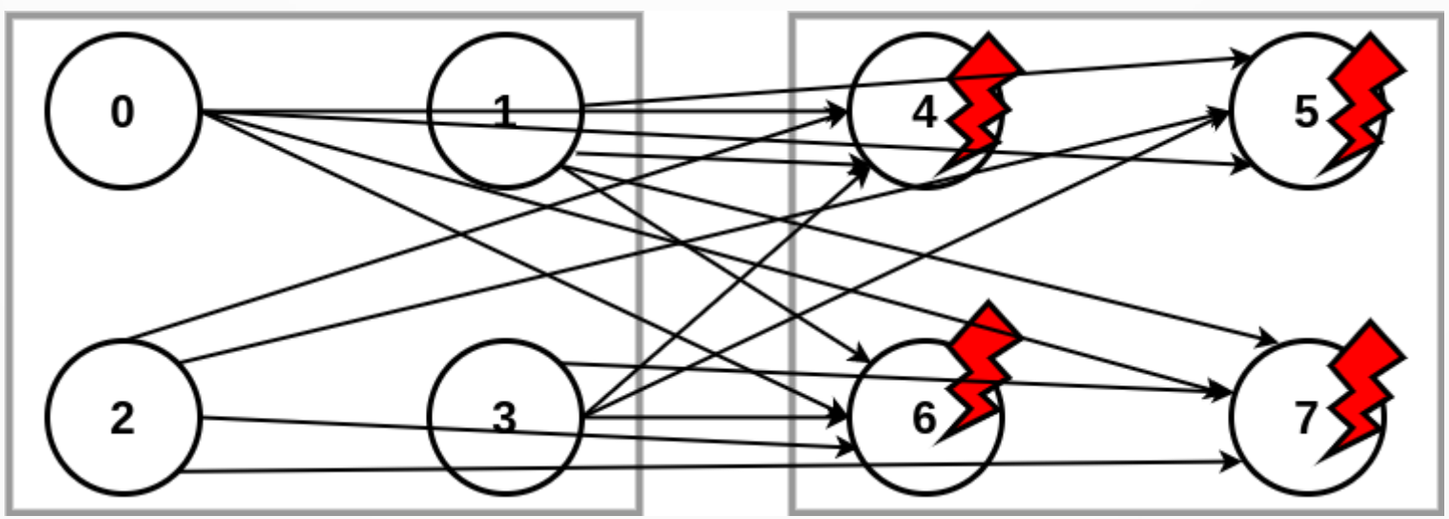


Propriedades do VCube: Número de Testes

- Qual o número máximo de testes executados pelo VCube?

Propriedades do VCube: Número de Testes

- Qual o número máximo de testes executados?
- Considere que todos os processos do maior cluster (com N/2 processos) estão falhos
- Quantos testes executa? Os N/2 testadores testam todos os N/2 falhos: $N^2/4=O(N^2)$



VCube: Escalabilidade de Testes

- Este problema foi resolvido com uma nova versão do VCube
- Esta versão determina quem é o *testador* de cada processo em cada cluster
- Veja que no algoritmo original a função C(i,s) é usada para determinar quem são os nodos testados
- Aqui: o testador do nodo j no cluster s é o primeiro nodo sem-falha em C(j,s)

VCube: Escalabilidade do Número de Testes

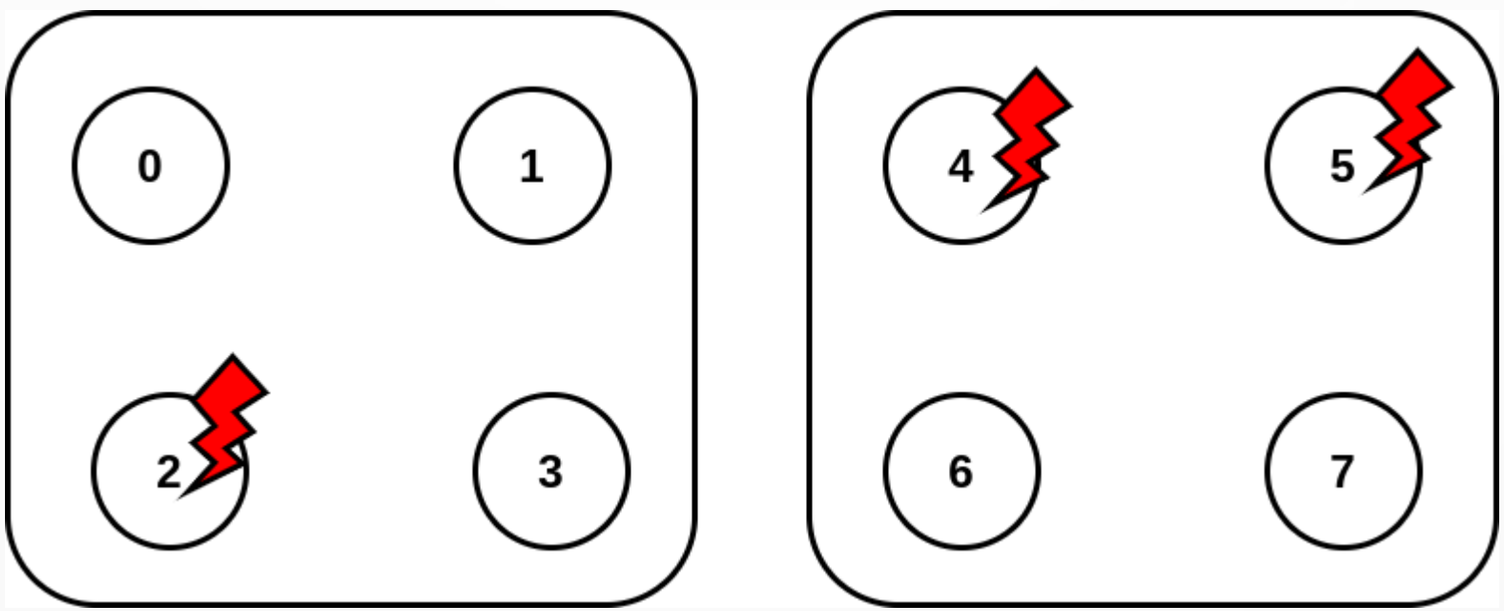
- Assim considerando o sistema que leva ao pior caso do número de testes no algoritmo original
- Os testadores são:
  - Testador do processo 4: primeiro processo correto em C(4,3)= 0
  - Testador do processo 5: primeiro processo correto em C(5,3)= 1
  - Testador do processo 6: primeiro processo correto em C(4,3)= 2
  - Testador do processo 7: primeiro processo correto em C(5,3)= 3
- Apenas 1 testador por processo, ao invés de todos do outro cluster!

Outra Diferença da Nova Versão do VCube: Obtenção de Informações

- A versão 2 do VCube tem uma outra diferença: ao testar um processo correto, o testador obtém qualquer “novidade” que o testado tenha
  - Na versão anterior obtém apenas informações do cluster testado
- Esta nova estratégia acelera fortemente o diagnóstico na média – o pior caso continua sendo $\log^2 N$
- Para implementar na prática: basta o testado guardar o vetor State[] que enviou para cada testador, e no próximo teste só manda o que mudou
  - Na simulação basta comparar os vetores State[] do testado e testador ;-)

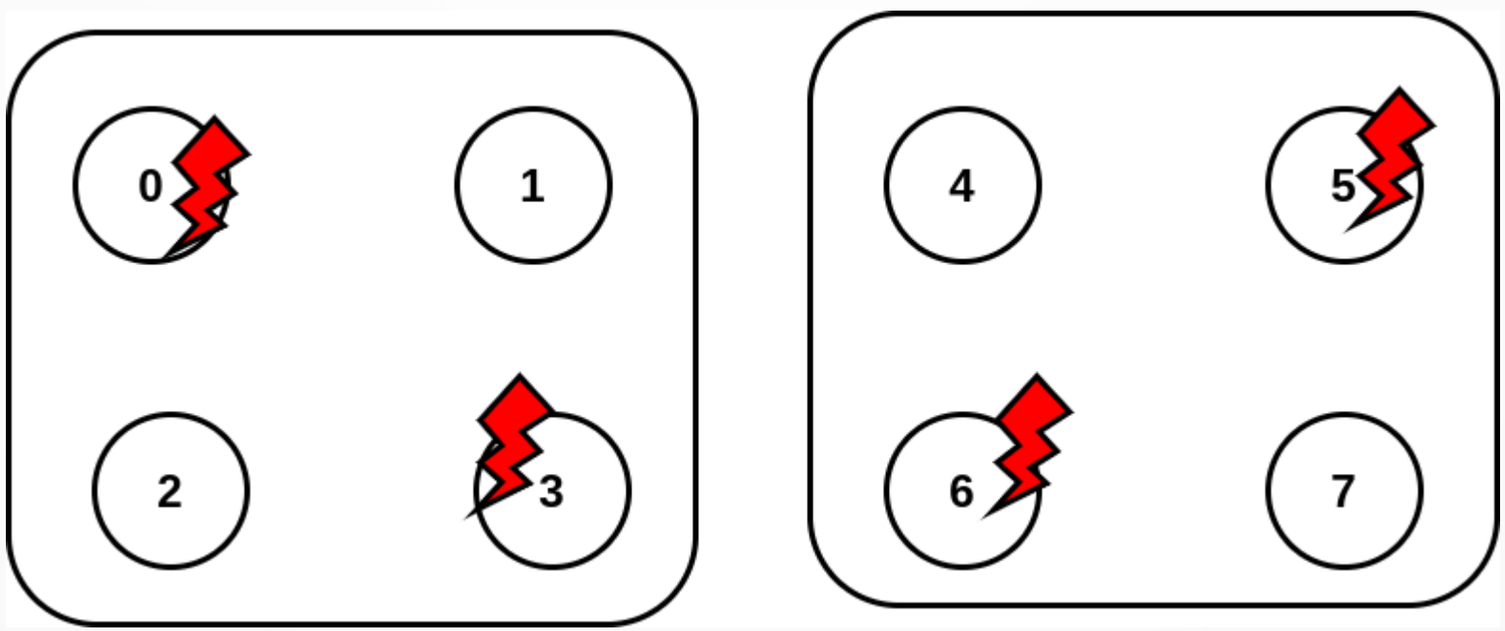
Vamos Fazer Alguns Exercícios

- Exercício 1: mostre os testes executados e as informações transmitidas para o VRing e as duas versões do VCube



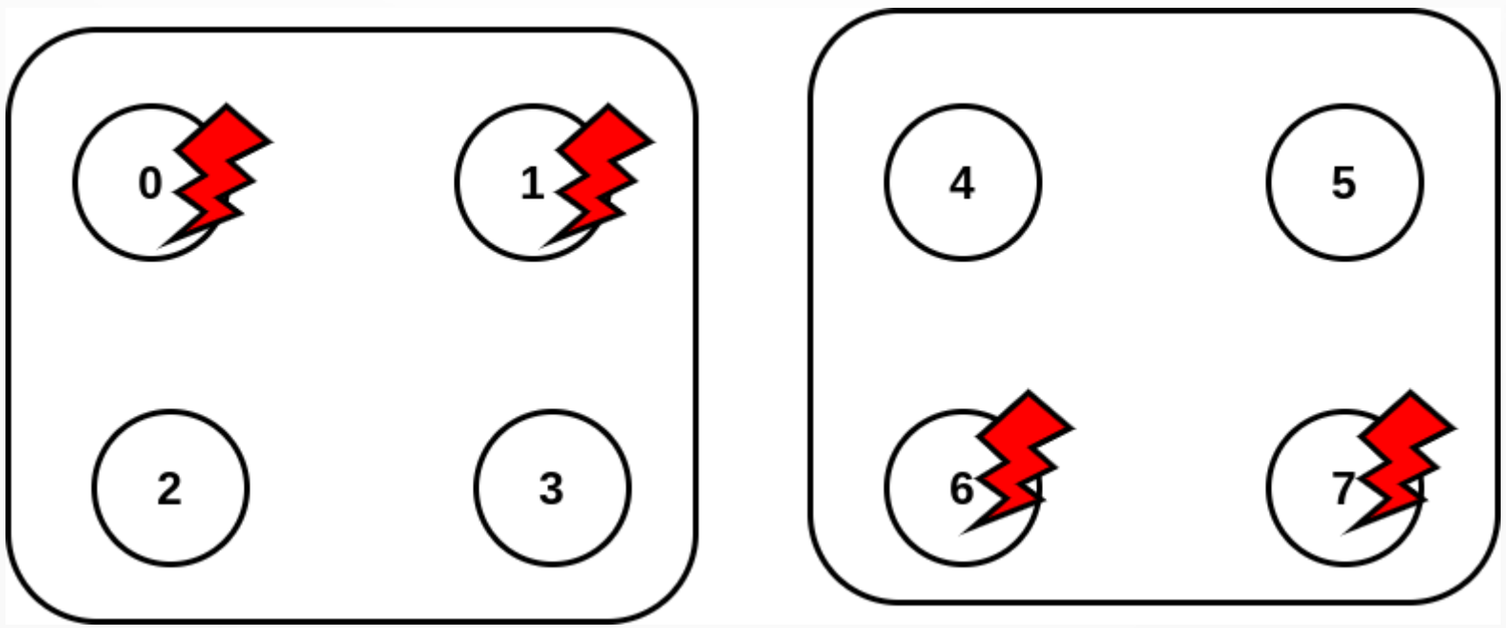
Vamos Fazer Alguns Exercícios

- Exercício 2: mostre os testes executados e as informações transmitidas para o VRing e as duas versões do VCube



Vamos Fazer Alguns Exercícios

- Exercício 3: mostre os testes executados e as informações transmitidas para o VRing e as duas versões do VCube



Referência original do VCube:

“A Hierarchical Adaptive Distributed System-Level Diagnosis Algorithm,” *IEEE Transactions on Computers*, Vol. 47, No. 1, 1998.

Referência da Versão 2 do VCube:

"VCube: A Provably Scalable Distributed Diagnosis Algorithm," Workshop on Latest Advances in Scalable Algorithms for Large-Scale Systems (ScalA), at The Supercomputing Conference 2014 (SC'2014), pp. 1-8, New Orleans, USA, 2014.

Conclusão

- Nesta aula continuamos o estudo sobre o algoritmo distribuído hierárquico VCube
- Provamos a latência no pior caso
- Vimos outra versão: garante NlogN testes a cada logN rodadas

Obrigado!
Página da Disciplina
Sistemas Distribuídos:
www.inf.ufpr.br/elias/sisdis