UM ALGORITMO DISTRIBUÍDO PARA REDES DE TOPOLOGIA DINÂMICA

Leandro Pacheco de Sousa
Professor: Aldri Luiz dos Santos

Departamento de Informática
Universidade Federal do Paraná

28 de Junho de 2007

Roteiro da Apresentação

Sistemas de Gerência de Redes

- Permitem controloar e monitorar os elementos de uma rede
- Uma função destes sistemas é a gerência de falhas
  - Os algoritmos de diagnóstico distribuído podem ser utilizados como base destes sistemas

Algoritmos de Diagnóstico

Modelo de Diagnóstico

- Centralizado
- Distribuído

Tipos de Rede

- Representáveis por um grafo conexo
- Redes de topologia arbitrária

O Algoritmo DNR (Modificado)

Características do algoritmo *Ditributed Network Reachability* modificado

- Diagnóstico distribuído
- Redes de topologia arbitrária
- Diagnóstico *on-line*
- Composto de 3 fases: teste, disseminação e diagnóstico
  - Disseminação modificada para inundação
  - Não suporta particionamento na rede

O Algoritmo Proposto

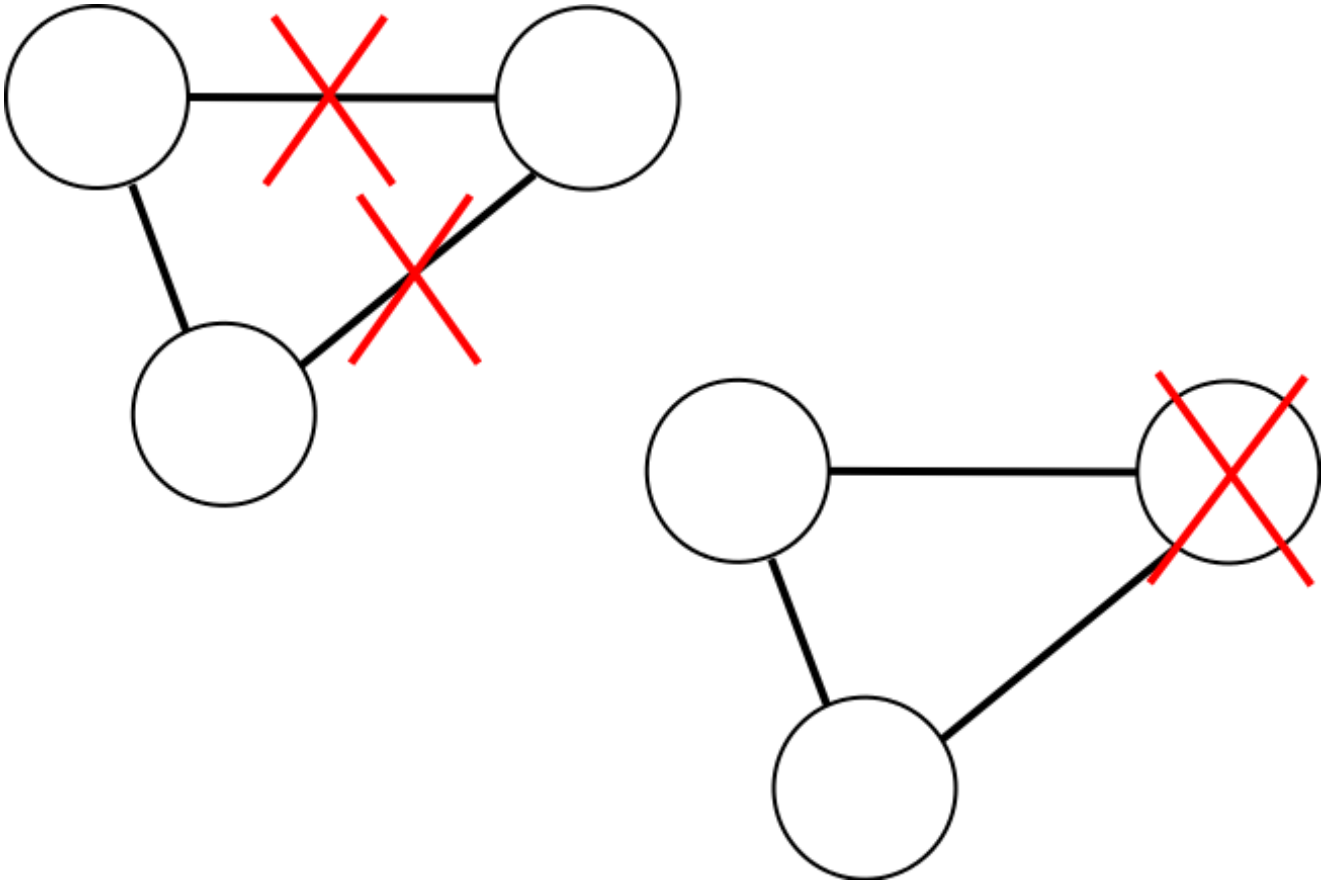
- Modificação do algoritmo DNR com inundação
  - Suporte ao particionamento, como no DNR original
  - Topologias dinâmicas

O Algoritmo DNR com Disseminação por Inundação

Algoritmo de diagnóstico distribuído para redes de topologia arbitrária

- Tem como objetivo determinar o estado das unidades da rede
- Um enlace pode estar
  - Sem-falha
  - Não-respondendo
  - Inatingível
- Um nó pode estar
  - Sem-falha
  - Inatingível

Ambiguidade de Falhas



As Fases do Algoritmo

O Algoritmo possui três fases

- Fase de Testes
- Fase de Disseminação
- Fase de Diagnóstico

A Fase de Testes

- Executa continuamente em todos os nós sem-falha
- Testes são executados durante os intervalos de teste
- Testes são realizados somente entre nós vizinhos
- Estratégia de *Token-Test*

O Token-Test

- Estratégia de passagem de bastão
- Um *token* para cada enlace da rede
- O nó que possui o *token* realiza o teste
- *Token* passa pra o outro nó ao fim do teste
- Testes *two-way*

O Token-Test

Detecção de novos eventos

- Cada nó possui uma tabela contendo todos enlaces da rede
- Todo enlace é associado a um contador chamado de *timestamp*
  - *Timestamp* par indica um enlace sem-falha
  - *Timestamp* ímpar indica um enlace não-respondendo
- Ao fim de um teste, o nó verifica se houve uma falha ou recuperação e incrementa o *timestamp* se necessário

A Fase de Disseminação

É iniciada quando um nó detecta um novo evento

- A informação sobre o evento é distribuída pela rede
- Estratégia tolerante à novas falhas
- Disseminação por Inundação de Mensagens

Inundação de Mensagens

- Mensagens contém informação sobre apenas um evento
- O nó que descobre o evento envia a mensagem para todos os vizinhos
- Quando um nó recebe um evento desconhecido, este envia a mensagem para todos os vizinhos, exceto para o nó de quem ele a recebeu
- Se a mensagem de disseminação não é nova, ignora
  - Chamamos este caso de mensagem redundante

A Fase de Diagnóstico

Pode ser iniciada em qualquer nó a qualquer instante

- São determinados os estados de nós e enlaces da rede
- Cálculo de alcançabilidade feito com o uso de um algoritmo de conectividade em grafos
- Não é o foco deste trabalho

Limitações do Algoritmo

Duas limitações deste algoritmo são

- Não suporta o particionamento da rede
  - Se a rede sofre particionamento e depois se recupera, não garante a consistência da informação nos nós
- Ao início do algoritmo, a topologia da rede já está definida e permanece estática durante a execução

O Algoritmo Proposto

Algoritmo baseado no DNR com inundação

- Composto também de três fases
- Modificações que visam eliminar as limitações citadas anteriormente
  - Suportar o particionamento e posterior recuperação da rede
  - Suportar a inserção e remoção de enlaces e nós na rede

Particionamento na Rede

A solução para este problema é simples

- Relacionada à recuperação de enlaces
- Se este enlace une duas componentes conexas
  - Após a recuperação, as duas extremidades do enlace devem trocar informação sobre toda a rede
- Seria necessária a detecção do particionamento
- Solução
  - Disseminar informação sobre toda a rede em toda recuperação de enlace

Inserção e Remoção de Nós e Enlaces

A principal modificação no algoritmo é

- Modificação na estrutura dos eventos
  - Inclusão de um novo campo, *status*
  - O status pode assumir os valores, "não-respondendo", "não-falho"e "removido"
  - Dois novos eventos: entrada e saída de enlaces
  - Disseminação não se modifica

Inserção e Remoção de Nós e Enlaces

Todo nó do algoritmo começa isolado. Nós se comunicam para criar um "enlace do algoritmo"

- Um dos nós inicia o processo enviando um pacote CONNECT
- Quando um nó recebe um CONNECT
  - Devolve um pacote CONNACK
  - Adiciona o evento sobre o novo enlace
  - Dissemina este evento para seus outros vizinhos
  - Envia todos os eventos que ele possuí através do novo enlace
- Quando o nó iniciador recebe o CONNACK
  - Adiciona o evento sobre o novo enlace
  - Dissemina este evento para seus outros vizinhos
  - Envia todos os eventos que ele possuí através do novo enlace
  - Inicia os testes no enlace
- Garante o suporte ao particionamento
- Não suporta o caso de falha do enlace durante a comunicação de criação do mesmo

Inserção e Remoção de Nós e Enlaces

Para remover um enlace do algoritmo, os nós das pontas se comunicam

- Um dos nós inicia o processo enviando um pacote DISCONNECT
- Quando um nó recebe um pacote DISCONNECT
  - Devolve um pacote CONNACK
  - Altera o estado do enlace para "removido"
  - Dissemina este evento para seus outros vizinhos
  - Interrompe os testes no enlace
- Quando o nó iniciador recebe o CONNACK
  - Altera o estado do enlace para "removido"
  - Dissemina este evento para seus outros vizinhos
  - Interrompe os testes no enlace
- Manter a informação sobre nós removidos é fundamental para suportar o particionamento na rede
  - Enlaces podem ser removidos durante um particionamento
- Não suporta o caso de falha no enlace durante a comunicação de remoção do mesmo

Inserção e Remoção de Nós e Enlaces

Inserção e remoção de nós

- Não existe uma operação explícita de inserção ou remoção de nó
- Só existem a criação e remoção de enlaces
- Para um nó ser inserido na rede
  - Criar um enlace com qualquer um dos nós da rede
- Para um nó ser removido da rede
  - Remover todos os enlaces com os nós da rede

O Simulador ns-2

O ns-2 é um simulador de eventos discretos focado na área de redes de computadores. Algumas de suas características:

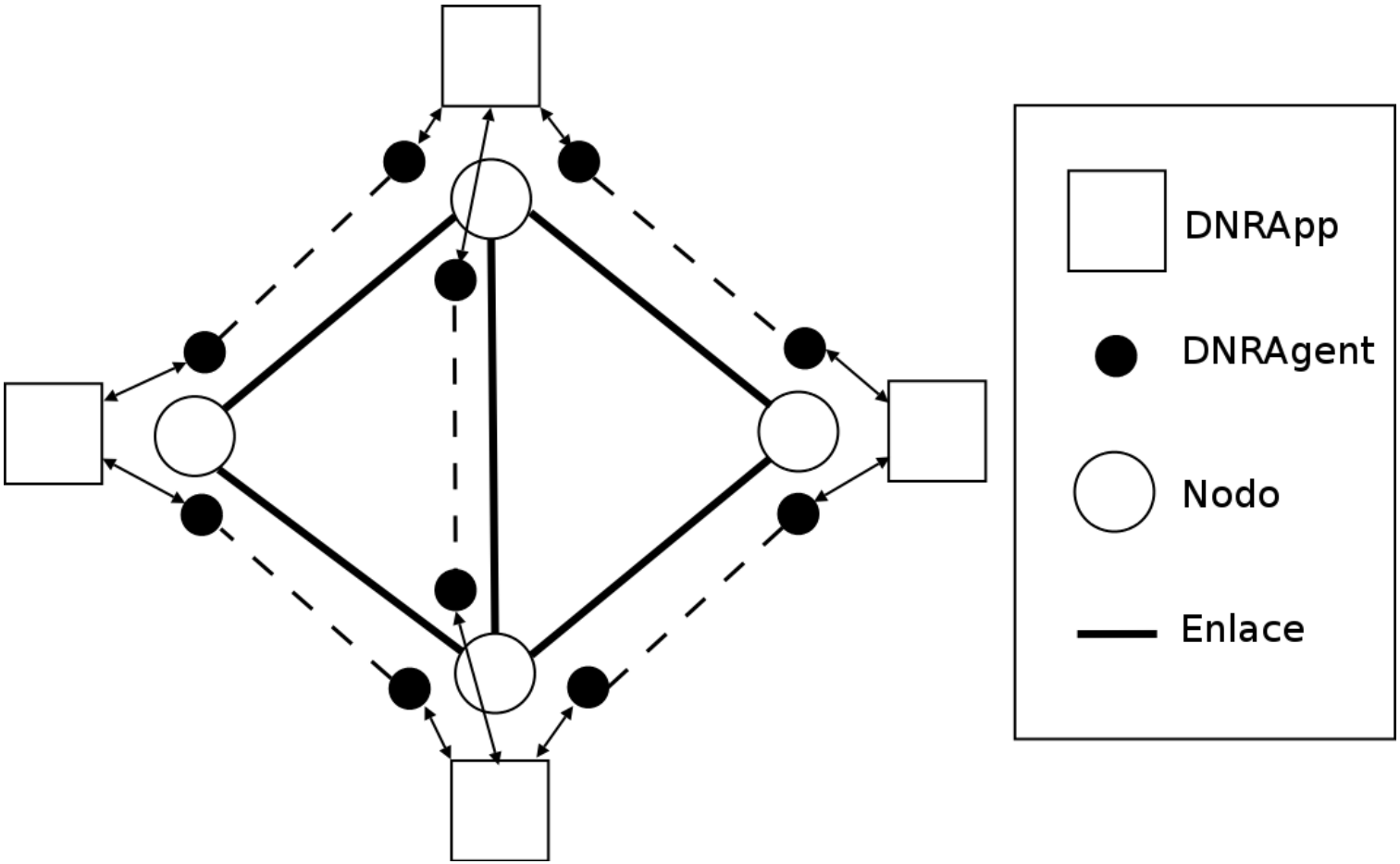
- Implementação nas linguagens C++ e oTcl
- Suporte à simulação de protocolos de comunicação, roteamento e gerência
- Simulação definida em um script oTcl e executada pelo simulador
- Tem sido usado pela comunidade científica para avaliação de propostas

Implementação do Algoritmo DNR Modificado

São definidas algumas novas entidades

- DNRAgent: Comunicação entre nós
- DNRApp: Execução do algoritmo
- DNRIntervalTimer: Controle de intervalos de teste
- DNRTestTimer: Controle de *timeout* dos testes

Exemplo de Configuração para Simulação do Algoritmo



Suporte ao Particionamento

Modificação restrita à DNRApp

- Na detecção de uma recuperação ou inserção de novo enlace, dissemina toda informação que possuí através do DNRAgent deste enlace

Suporte à Remoção e Inserção de Enlaces

Modificações

- No pacote DNR
  - Inclusão de um campo Status
- No agente DNRAgent
  - Novos métodos *sendConnect()*, *sendDisconnect()* e *sendConnAck()*

Suporte à Remoção e Inserção de Enlaces

Modificações na aplicação DNRApp

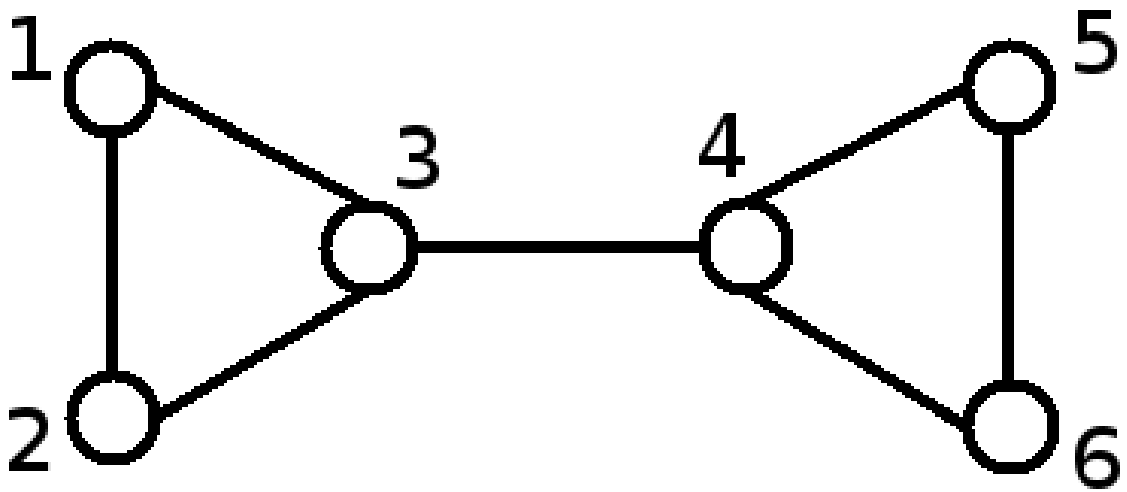
- Foram as modificações mais complexas
- Implementação do algoritmo de entrada e saída de enlaces mostrado anteriormente

Suporte à Remoção e Inserção de Enlaces

Este é o funcionamento da simulação, de maneira simplificada

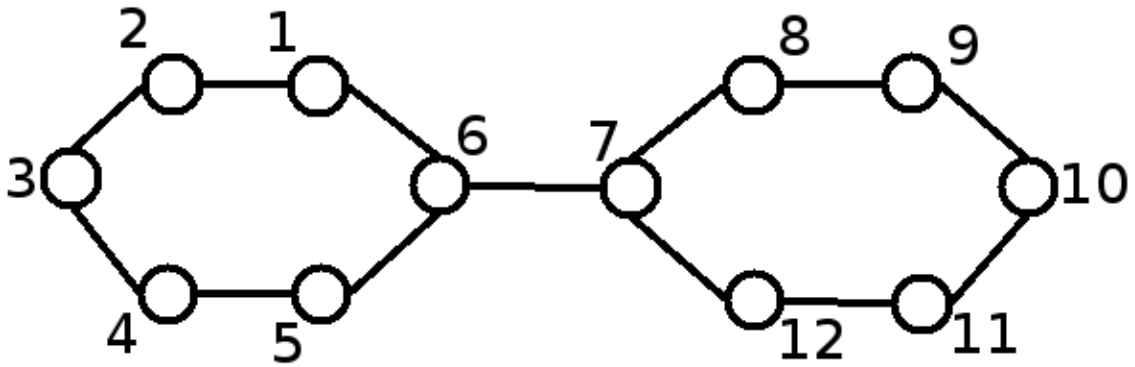
- No início da simulação, a topologia "física"da rede ja está definida
- Já existem os nós, enlaces físicos, agentes, e aplicações
- Cada aplicação começa isolada com relação ao algoritmo. Não existe nenhum "enlace do algoritmo"
- Aplicações precisam se comunicar para criar os enlaces e iniciar os testes

Exemplos de Topologias Utilizadas



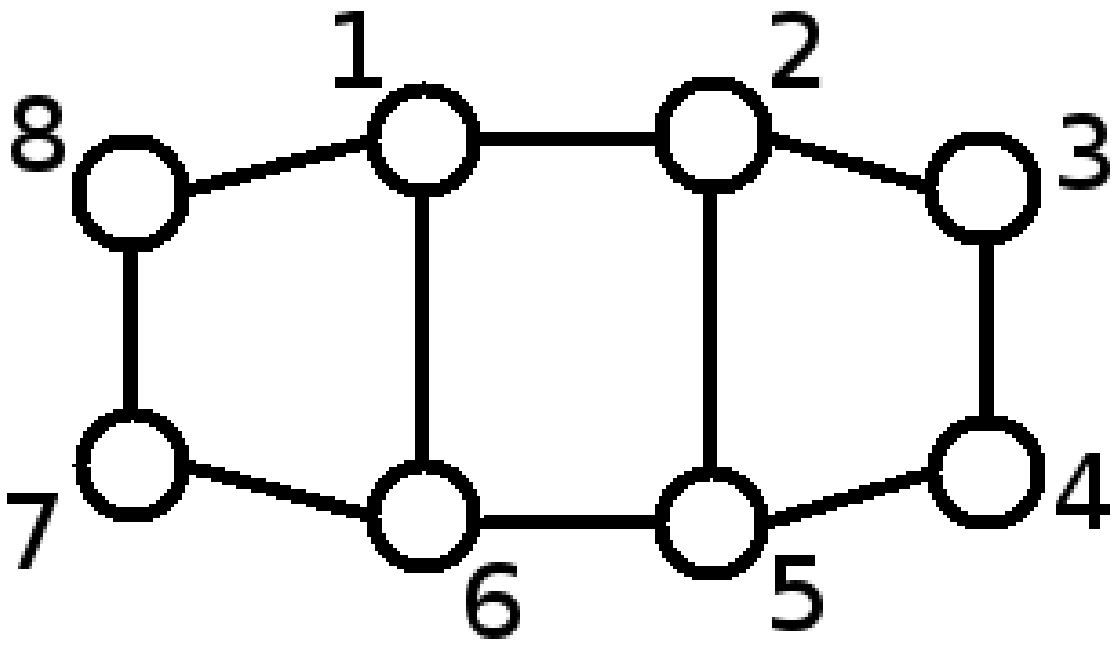
Testes e Resultados

Exemplos de Topologias Utilizadas



Testes e Resultados

Exemplos de Topologias Utilizadas



Testes e Resultados

Com base nos testes executados, foi possível verificar a corretude do algoritmo.

- Foram realizados testes com particionamento
- O algoritmo gerou os resultados esperados tanto no caso da recuperação de falhas, quanto na criação de novos enlaces

Com relação ao numero de mensagens

- Número relativamente alto de mensagens trocadas
- Mensagens tem tamanho pequeno, o impacto na rede não é tão grande
- Nos testes realizados, cerca de 30% das mensagens trocadas eram redundantes
  - Este resultado irá variar de acordo com o número de eventos detectados nas rodadas de teste. Quanto mais eventos detectados, mais mensagens redundantes.

Conclusão

- Apresentamos neste trabalho uma modificação do algoritmo DNR com inundação
- Modificações visaram
  - Incluir o suporte ao particionamento da rede
  - Adaptar o algoritmo para funcionamento em topologias dinâmicas
- O algoritmo proposto foi implementado e testado no ns-2
  - Algoritmo se comportou como o esperado
  - Alto número de mensagens trocadas
  - Mensagens de tamanho pequeno. Não causa grande impacto na rede