Redes de Computadores

Carlos Henrique Barriquello

UFSM - Universidade Federal de Santa Maria
DELC - Departamento de Eletrônica e Computação
barriquello@gmail.com barriquello@gedre.ufsm.br

Santa Maria, RS

Sumário

Algoritmos de roteamento

O **local da decisão** determina que nós são responsáveis por realizar o roteamento, podendo ser:

- **Distribuído**: Cada nó é responsável por selecionar um enlace para a saída dos pacotes;
- **Centralizado**: um nó específico realiza o roteamento;
- **Estação Fonte**: a rota é decidida na estação que gerou a mensagem.

Algoritmos de roteamento

Fonte de Informação: geralmente o cálculo do roteamento necessita de informações da rede, como topologia, tráfego, custo do enlace.

Frequência de Atualização: função da fonte de informação e da estratégia de roteamento:

- Mudança de Topologia;
- Periódica;
- No momento do roteamento.

Algoritmos de roteamento estáticos

- **Algoritmo de caminho mais curto (centralizado)** - uma rota única é estabelecida para cada par fonte-destino. Utiliza um dos algoritmos de mínimo custo para determinar a rota.

- **Inundação ou flooding (distribuído)** - em cada nó, o pacote que chega é retransmitido para todos os nós vizinhos exceto para o vizinho que enviou o pacote originalmente.

Inundação ou *flooding*

Em cada nó, o pacote que chega é transmitido para todos os nós vizinhos exceto para o vizinho que enviou o pacote originalmente

Gera uma vasta quantidade de pacotes duplicados, na verdade um número infinito.

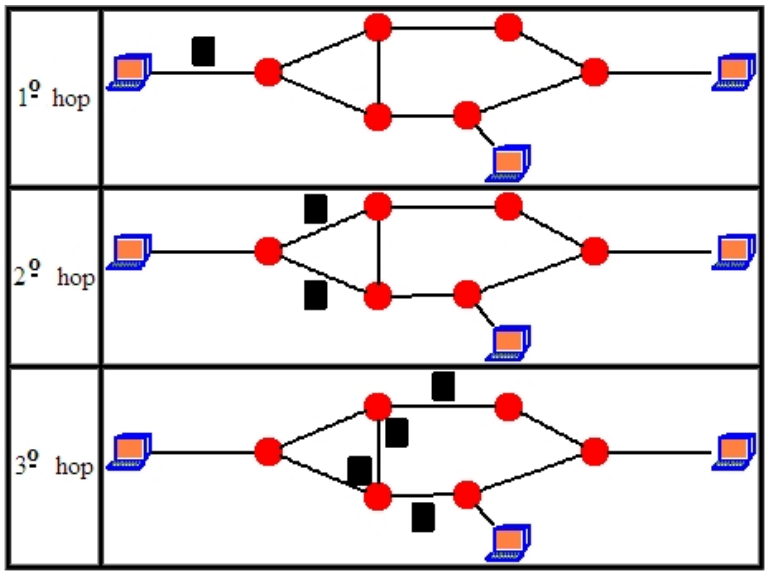

Figura: Estratégia de roteamento denominada "flooding".

Inundação ou *flooding*

Várias cópias de um mesmo pacote poderão chegar ao destino.

Soluções:

- acrescentar ao pacote **um identificador único ou número de seqüência** para descartar duplicatas. Cada roteador precisa de uma lista por roteador de origem informando quais números de seqüência originários desse ponto já foram vistos. Se houver um pacote de entrada na lista, ele não será transmitido na inundação.
- ter um **contador de hops** contido no cabeçalho de cada pacote; o contador é decrementado em cada *hop*, com o pacote sendo descartado quando o contador atingir zero.

Inundação ou *flooding*

Propriedades da inundação:

- Todo as as rotas são experimentadas;
- Muito robusto;
- Pelo menos um pacote chegará pela rota de custo mínimo;
- Pode ser usado para estabelecer a rota ótima;
- Pode ser usado para implementar comunicação *broadcast* – difusão
- Todos os nós são visitados;
- Útil para distribuir informação;
- Tem como desvantagem a carga de tráfego alta.

Inundação ou *flooding*

Aplicações do algoritmo de inundação:

- em aplicações militares, em que muitos roteadores podem ser destruídos a qualquer momento, a grande robustez do algoritmo de inundação é altamente desejável.
- em aplicações de bancos de dados distribuídos, às vezes é necessário atualizar todos os bancos de dados ao mesmo tempo e, nesse caso, o algoritmo de inundação pode ser bastante útil.
- é útil como uma unidade de medida que servirá como base de comparação com outros algoritmos de roteamento.

O algoritmo de inundação sempre escolhe o caminho mais curto, pois todos os caminhos possíveis são selecionados em paralelo.

Roteamento estático de custo mínimo

Este algoritmo de roteamento utiliza um algoritmo para calcular a rota de **custo mínimo** entre fonte e destino no grafo da rede, de acordo com os custos dos enlaces.

O custo pode ser:

- número de saltos (*hops*)
- tempo de atraso na fila
- taxa de transmissão de dados
- distância geográfica
- combinações destes critérios e outros fatores.

São conhecidos diversos algoritmos para se calcular o caminho mais curto entre dois nós de um grafo.

Roteamento estático de custo mínimo

- Menor número de saltos: nós 1-3-6;
- Menor custo: nós 1-4-5-6.

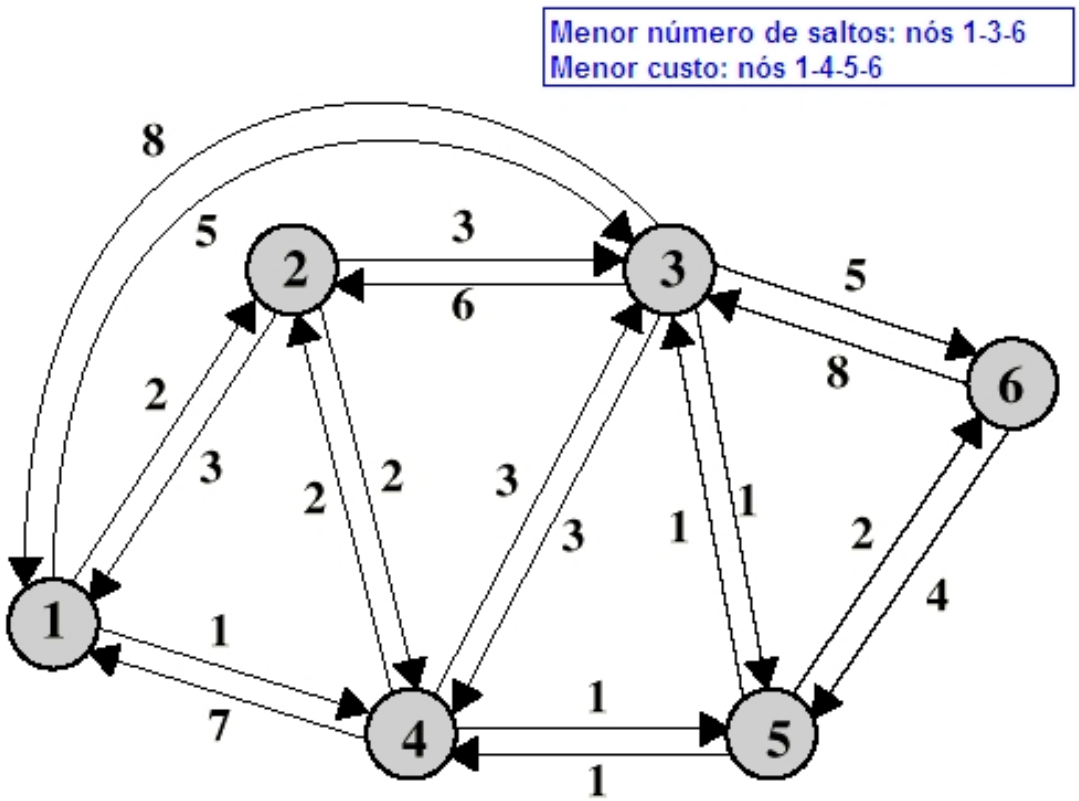

Figura: Exemplo de grafo da rede com custos por enlace.

Algoritmo de Dijkstra (1959)

1. Inicialização do algoritmo de Edsger Djkstra[1]:
   1. $T = s$, conjunto de nós incorporados consiste apenas no nó fonte;
   2. $L(n) = w(i, j)$ para $n \neq s$

   em que, n é o n-ésimo nó na rede e s é o nó fonte; $T =$ conjunto de nós já incorporados pelo algoritmo; $w(i, j) =$ custo direto entre os nós i e j; $L(n) =$ custo até o momento do nó s para o nó n.
2. Incorporação de novo nó: Ache um nó, ainda não incorporado a T, que possua custo mínimo do nó s e o incorpore em T.
   1. Ache $x \notin T$ tal que $L(x) = \min[L(j)]$
3. Atualize os valores de custo:
   1. $L(n) = \min[L(n), L(x) + w(x, n)]$ para todo $n \notin T$.

Algoritmo de Dijkstra (1959)

Tabela: Algoritmo de custo mínimo – Dijkstra aplicado à Figura 2

Iteração	T	L(2)	Path	L(3)	Path	L(4)	Path	L(5)	Path	L(6)	Path
1	{1}	2	1-2	5	1-3	1	1-4	∞	–	∞	–
2	{1,4}	2	1-2	4	1-4-3	1	1-4	2	1-4-5	∞	–
3	{1,2,4}	2	1-2	4	1-4-3	1	1-4	2	1-4-5	∞	–
4	{1,2,4,5}	2	1-2	3	1-4-5-3	1	1-4	2	1-4-5	4	1-4-5-6
5	{1,2,3,4,5}	2	1-2	3	1-4-5-3	1	1-4	2	1-4-5	4	1-4-5-6
6	{1,2,3,4,5,6}	2	1-2	3	1-4-5-3	1	1-4	2	1-4-5	4	1-4-5-6

Exercícios

1. Determine a rota de menor custo para a rede ilustrada na Figura 3. Empregue o algoritmo de Dijkstra.

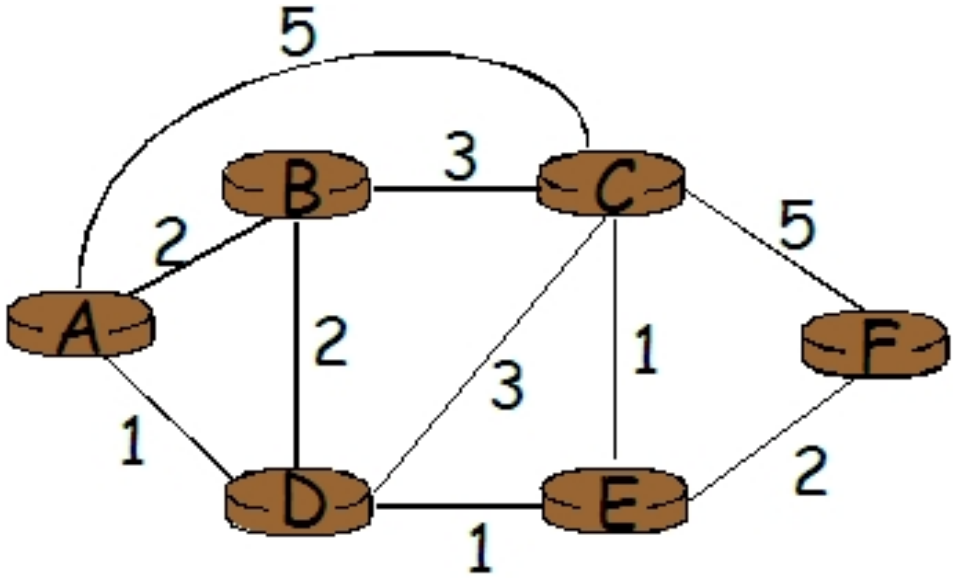

Figura: Representação de uma rede de comunicação de dados.

Exercícios

Considere que as distâncias entre os nós de uma rede estão armazenas em uma matriz *dist*. Escreva, em linguagem C, uma função que determine a menor distância entre dois nós nesta rede com base no algoritmo de Dijkstra.
Ex.: dist[6][6] =

$$\begin{pmatrix} 0 & 2 & 5 & 1 & \infty & \infty \\ 2 & 0 & 3 & 2 & \infty & \infty \\ 5 & 3 & 0 & 3 & 1 & 5 \\ 1 & 2 & 3 & 0 & 1 & \infty \\ \infty & \infty & 1 & 1 & 0 & 2 \\ \infty & \infty & 5 & \infty & 2 & 0 \end{pmatrix}$$

Exercícios

Considere que a matriz abaixo representa o grafo de uma rede. Suponha que ela utilize a inundação como algoritmo de roteamento. Se um pacote enviado por A até D tem uma contagem máxima de *hops* igual a 3, liste todas as rotas que ele seguirá. Além disso, informe quantos *hops* o pacote consome de largura de banda.

dist[9][9] =

$$\begin{pmatrix} & A & B & C & D & E & F & G & H \\ A & 0 & 2 & - & - & - & - & 6 & - \\ B & 2 & 0 & 7 & - & 2 & - & - & - \\ C & - & 7 & 0 & 3 & - & 3 & - & - \\ D & - & - & 3 & 0 & - & - & - & 2 \\ E & - & 2 & - & - & 0 & 2 & 1 & - \\ F & - & - & 3 & - & 2 & 0 & - & 2 \\ G & 6 & - & - & - & 1 & - & 0 & 4 \\ H & - & - & - & 2 & - & 2 & 4 & 0 \end{pmatrix}$$