Departamento de Informática

Roteamento em Redes Móveis Ad Hoc (Mobile Ad hoc NETworks - MANET)

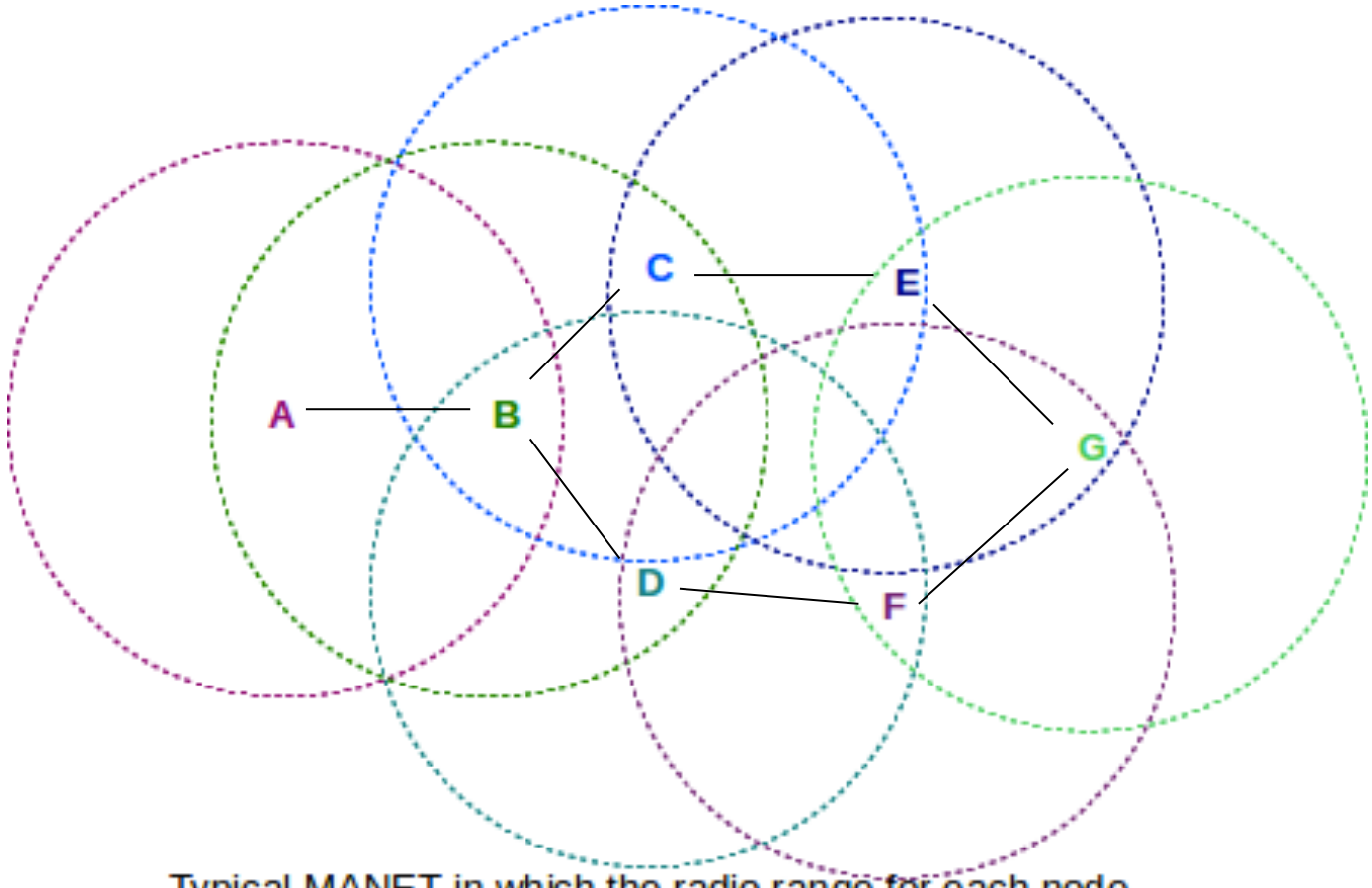
Agenda

- Principais Características de MANETs
- Aplicações
- Roteamento
  - Protocolos Table driven
  - Protocolos Source-initiated, On-demand
- Comparação
- Roteamento em redes Mesh

Redes MANET - Principais Características

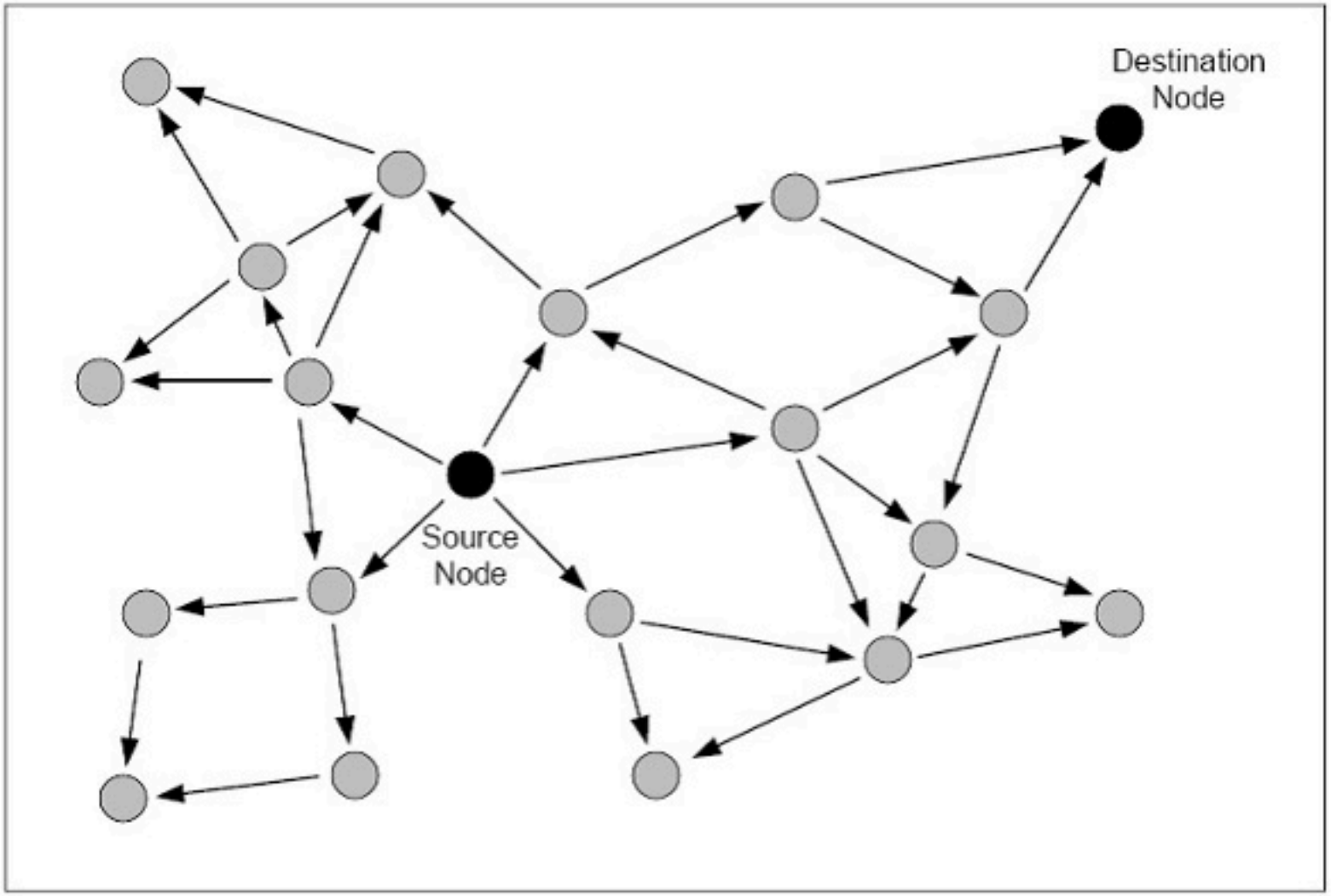
- Nós podem fazer o papel de consumidor ou produtor de dados, e todos fazem o papel de roteador
- Transmissão é por broadcast, onde alcance depende da potência de transmissão (em alguns casos, é variável), mas destinatário (e next hop) é determinado pela mensagem
- Cada nó apenas possui conectividade com um subconjunto dos nós mais próximos
- O serviço prestado por um nó (roteamento, processamento na rede, etc.) em um momento pode não estar mais acessível no momento seguinte
- Recurso energia é um aspecto central

Uma MANET simples



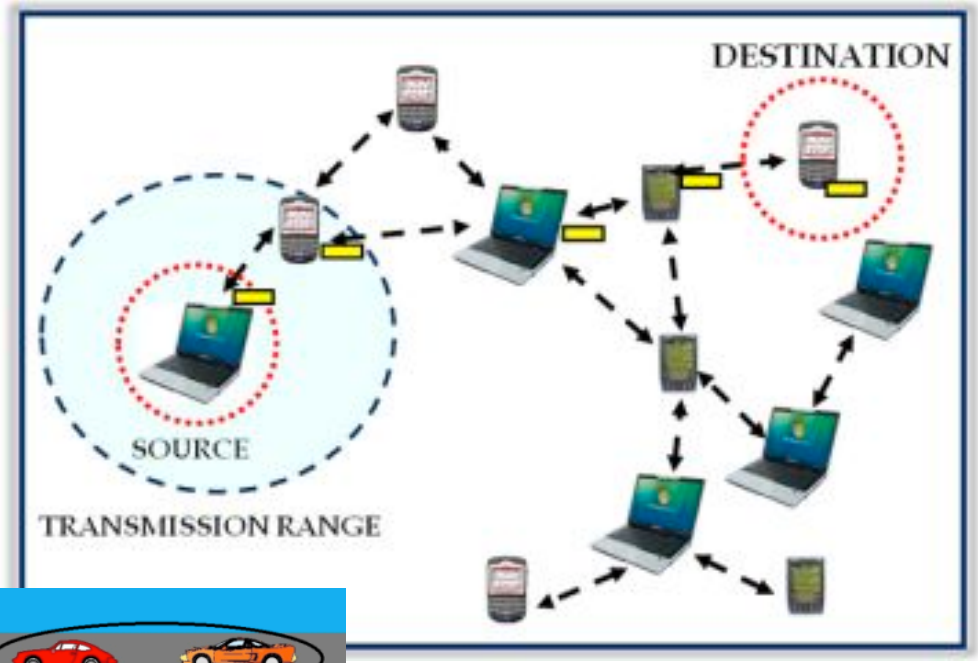
Typical MANET in which the radio range for each node is represented by a circle around that node. A can reach G either by the route A—B—C—E—G or by A—B—D—F—G.

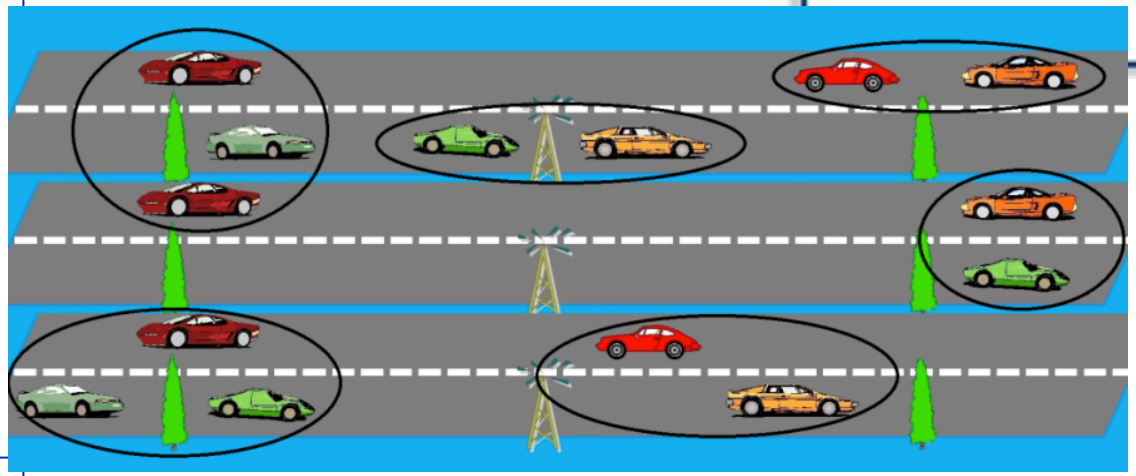
Exemplo



Exemplo

Rede WiFi modo Ad hoc





Rede V2V
(IEEE 802.11p/ WAVE)

Tipos de Aplicações

- Em geral: Formação espontânea de uma rede, sem necessidade de administração & configuração.

- Por exemplo:
  - Comunicação em áreas remotas, sem infra-estrutura de rede sem fio (redes mesh)
  - Comunicação direta entre usuários co-localizados (reunião, aula)
  - Interação entre nós que se movem (veículos: V2V e V2R, pessoas)
  - Dispositivos *wearable* (Personal Area Network)
  - Redes de sensores

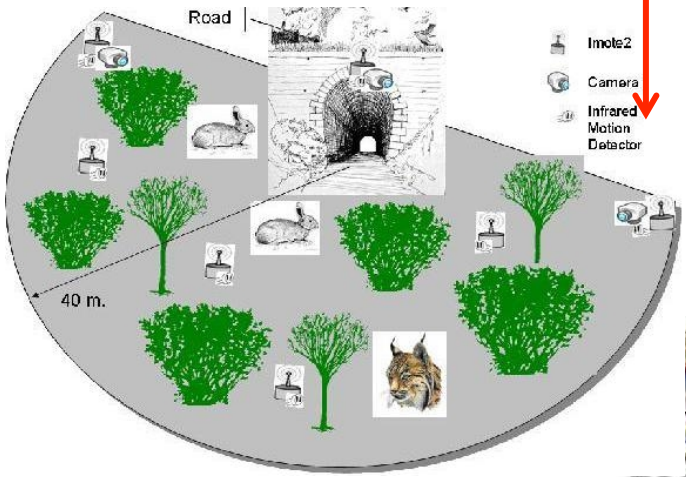
- A maioria das MANETs possuem (pelo menos) um nó, Gateway, conectado a uma rede WWAN ou WLAN, com acesso à Internet

Exemplo de Aplicações

- Monitoramento ambiental em regiões isoladas (ou sem infra-estrutura de comunicação)

  Exemplos:
  - Disseminação de vídeos, dados topográficos, condições climáticas, audio
  - Rastreamento de objetos e/ou pessoas
- Coleta e processamento de informações de redes de sensores
- Colaboração espontânea entre usuários
- Coordenação de tarefas entre robôs móveis

ZigBee™ Wifi Bluetooth®



Departamento de Informática

Principais desafios de MANETs

1. Roteamento em uma toplogia dinâmica, com baixa perda de pacotes e latência fim-a-fim
2. Maximizar utilização da banda de RF para a transmissão de dados da aplicação (minimizar o overhead de controle)
3. Auto-monitoramento e auto-adaptação da topologia
   - mobilidade, topologia variável, e nível de energia em cada nó
4. Atribuição dinâmica de papéis aos nós
5. Implementação eficiente de multicast
6. Gerenciamento de grupos de nós
7. Segurança

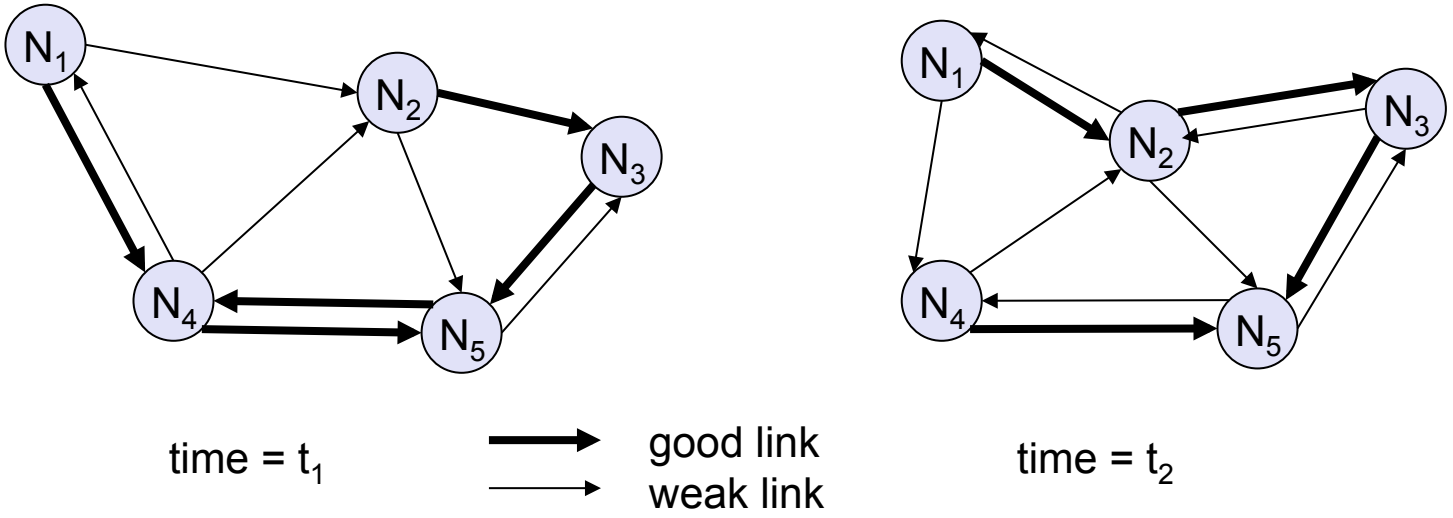
PUC RIO

Transmissão wireless em Redes móveis Ad hoc

- **Enlaces assímétricos** – intensidade de sinal (qualidade da transmissão) entre N1 e N2 difere em entre em cada uma das direções
- **Enlaces redundantes** – a partir de cada nó, geralmente há vários enlaces (e caminhos) possíveis
- **Interferência em transmissões** – frequente colisão, pois não há coordenação no acesso ao meio de transmissão
- **Topologia dinâmica** – lista de nós vizinhos alcançáveis muda frequentemente, afetando a manutenção das tabelas, ou o desempenho do roteamento
- **Níveis de energia variáveis**– e capacidade de encaminhar mensagens

Departamento de Informática

Exemplo: Assimetria de Enlaces e Topologia dinâmica



Seja uma comunicação Request-Reply entre N1 e N3, e assuma que todos os nós tenham conhecimento da conectividade global da rede.

Em t_1: Request-Reply poderia ser encaminhado via N4 e N5

Em t_2: Request-Reply só pode ser encaminhado via N2

Na maioria dos protocolos de roteamento, assume-se que os enlaces são sempre bi-direcionais.

PUC RIO

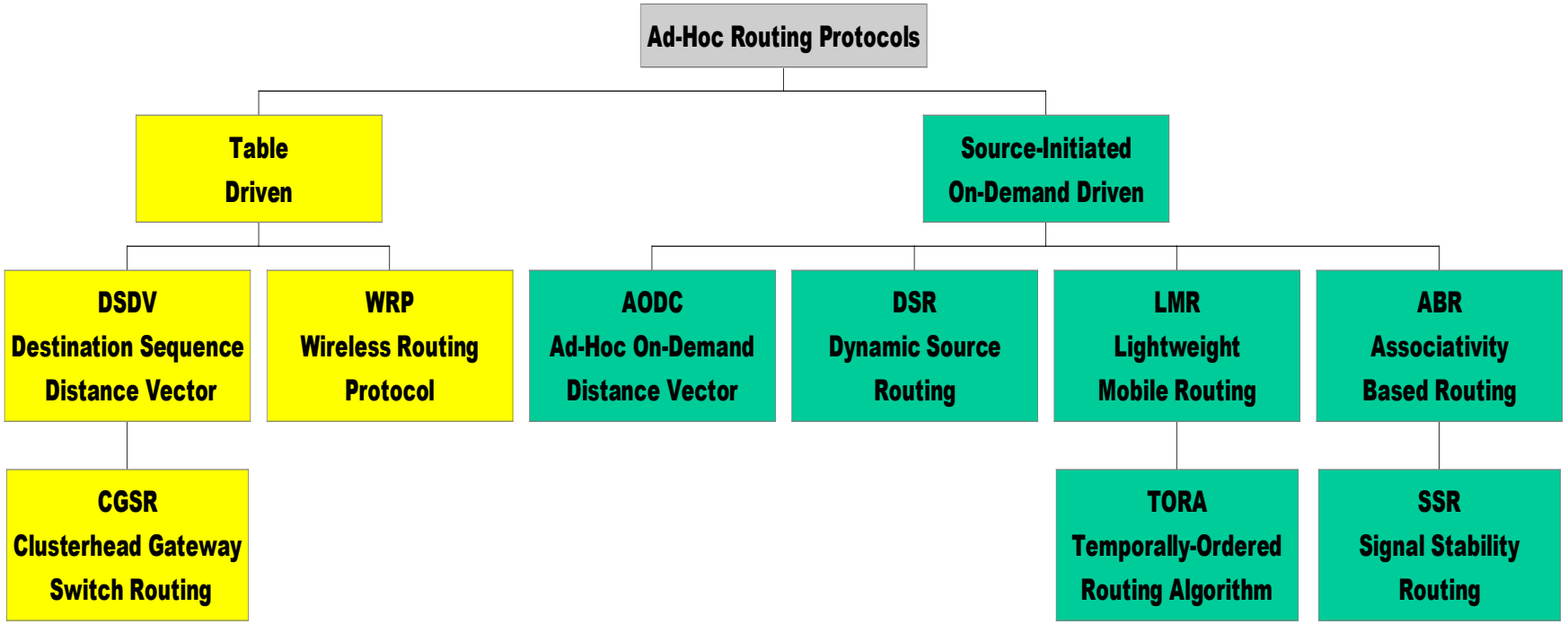
Classificação de Protocolos de Roteamento

Em MANETs, o roteamento precisa ser dinâmico e descentralizado, mas pode ser:

- Pró-ativo (*Table-driven*) x Reativo (*On demand*)
- Single-path x Multiple-path
- Plano x Hierárquico
- *Source-initiated* x *Router-intelligent*
- Intra-domínio x Inter-domínio
- Com informação sobre posição absoluta ou relativa (p.ex. geo-routing, directional routing) x sem informação
- Otimização global x local de recursos

Departamento de Informática

Taxonomia de Protocolos de Roteamento

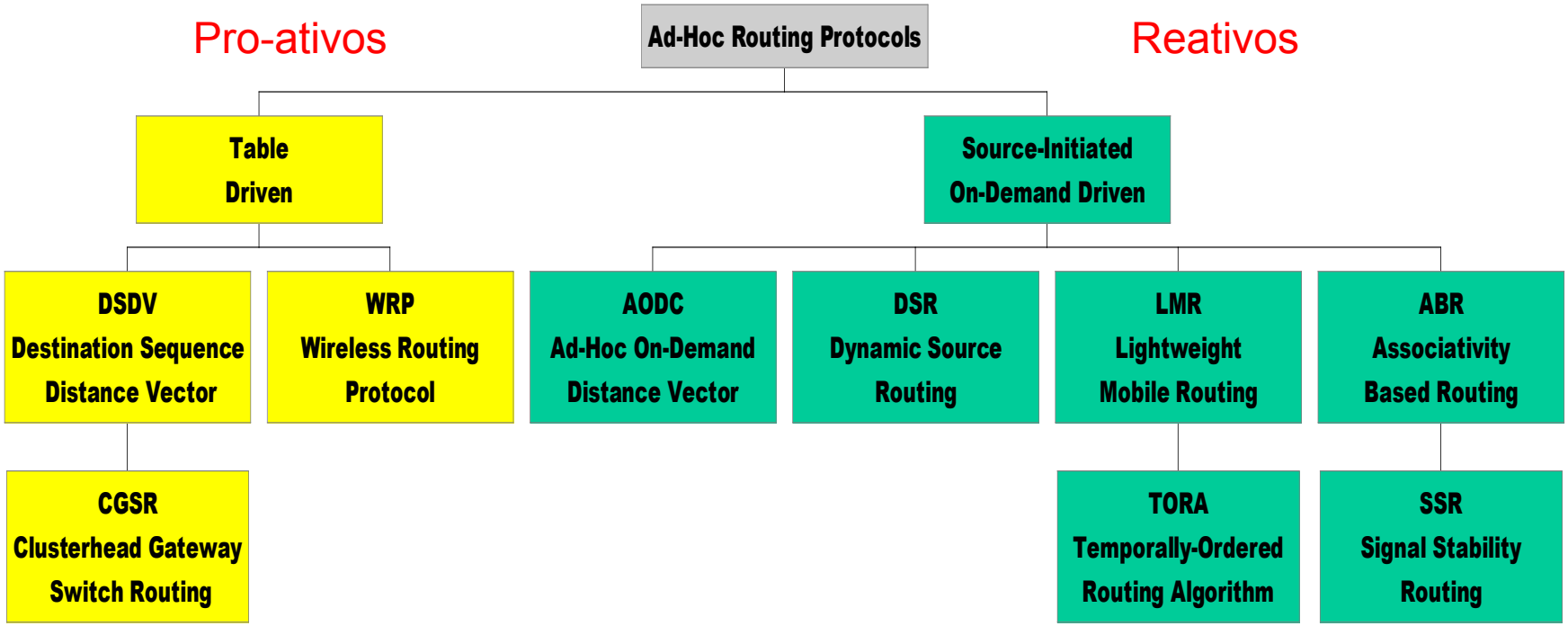


…e muitos outros

…e muitos outros

Departamento de Informática

Taxonomia de Protocolos de Roteamento



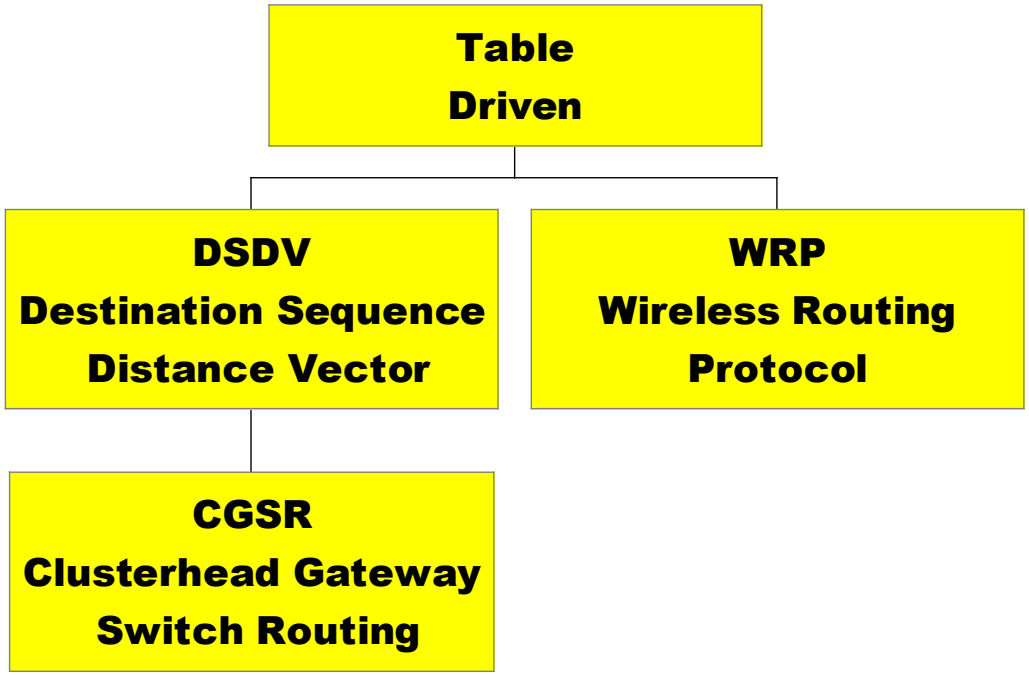
- Baseado em algs. distribuidos tradicionais de menor caminho
- Mantém sempre as melhores rotas entre cada par de nós,
- Atualizacões periódicas das tabelas de roteamento

- Processo de procura uma rota sempre que necessário;
- Manutenção de rota: quando um enlace quebra, parte da rota é re-descoberta
- Nós guardam em cache rotas que vão aprendendo a partir de pacotes que estão passando;

PUC RIO

Departamento de Informática

Algoritmos Table Driven: uma Taxonomia



DSDV é uma extensão do **Distance Vector Routing** (para redes cabeadas)

PUC RIO

Protocolos baseados em Tabela (Table driven)

- Tentativa de manter informação de roteamento (de qualquer nó para qualquer outro nó) consistente e atualizada
- Objetivo: reagir à mudanças da topologia de rede, propagando as atualizações (anúncios de rota), a fim de obter uma visão consistente das distâncias/custos de roteamento na rede.
- Apropriado para aplicações que geram um tráfego constante e bem distribuído entre os nós da rede ad hoc.

Departamento de Informática

Distance Vector Routing - Revisão

Usa o algoritmo Bellman-Ford:

- Cada roteador mantém uma tabela com a distância (# hops) de sí para todos os demais nós na rede;
- Periodicamente transmite pacotes de atualização a cada um dos roteadores vizinhos.
- Cada roteador atualiza cada entrada de sua tabela para destino se a distância anunciada for menor distância do que a que já possuia.

- Problema: se as tabelas não refletirem exatamente o estado das conexões, em caso de desconexões, pode ocorrer roteamento em loop e contagem infinita de hops ("*count-to-infinity*")

C. E. Perkins and P. Bhagwat, "Highly Dynamic Destination-Sequenced Distance- Vector Routing (DSDV) for Mobile Computer," Comp. Commun. Rev., Oct. 1994, pp. 234-244.

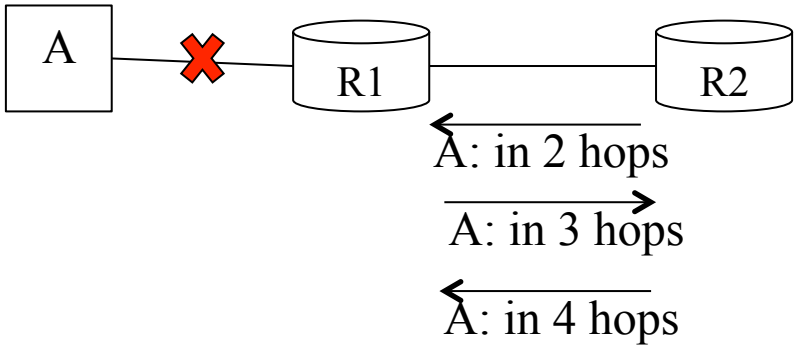
PUC RIO

Departamento de Informática

Problema de contagem ao infinito "*count-to-infinity*"

Ocorre quando um nó (ou subrede) A se desconecta de roteador R1:

- Após certo tempo, R1 difunde Métrica = ∞ para A
- Pode receber de roteador vizinho "R2" a informação (desatualizada) sobre o nr. de hops para "A", e que passa pelo próprio R1, mas ele desconhece isso
- R1 então acha que descobriu uma nova rota para A, através de R2,…
- Exemplo:



- Um solução: evitar informar uma atualização para o vizinho de de quem recebeu a atualizacão
- Outros algoritmos, como o DSDV resolvem isso com um SeqNr, que indica a atualidade da informação

PUC RIO

DSDV (Destination Sequenced Distance Vector)

DSDV difere de DV principalmente nos seguintes pontos:

- Usa números de sequencia para todas as entradas na tabela (e pacotes de atualização)
- Além de atualizações periódicas, gera nova atualização assim que houver mudança significativa na tabela de roteamento
- Mas espera até que as rotas se estabilizem (evitando anúncios precipitados sobre distâncias incorretas para nós que se reconectaram)
- Usa dois tipos de atualizacões de rota:
  - Full Dump Update: envia toda a tabela de roteamento
  - Incremental Update: somente com as entradas modificadas

DSDV (Destination Sequenced Distance Vector)

Entradas da Tabela:

Destination	Next	Metric	Seq. Nr	Install Time	Stable Data
A	A	0	A-550	001000	Ptr_A
B	B	1	B-102	001200	Ptr_B
C	B	3	C-588	001200	Ptr_C
D	B	4	D-312	001200	Ptr_D

Next: próximo hop

Metric: nr de hops até destino

Seq_Nr: gerada por cada destino (DestinoID-Nr) e indica a atualidade da rota
InstallTime: quando a entrada da tabela foi gerada

StableData Pointer: para uma tabela com informação sobre a estabilidade da rota

Anúncio de Rotas

Contendo a tabela de roteamento (parcial ou integral) com:

- DestinoID
- Métrica: Nr de hops para destino
- Destination SeqNr

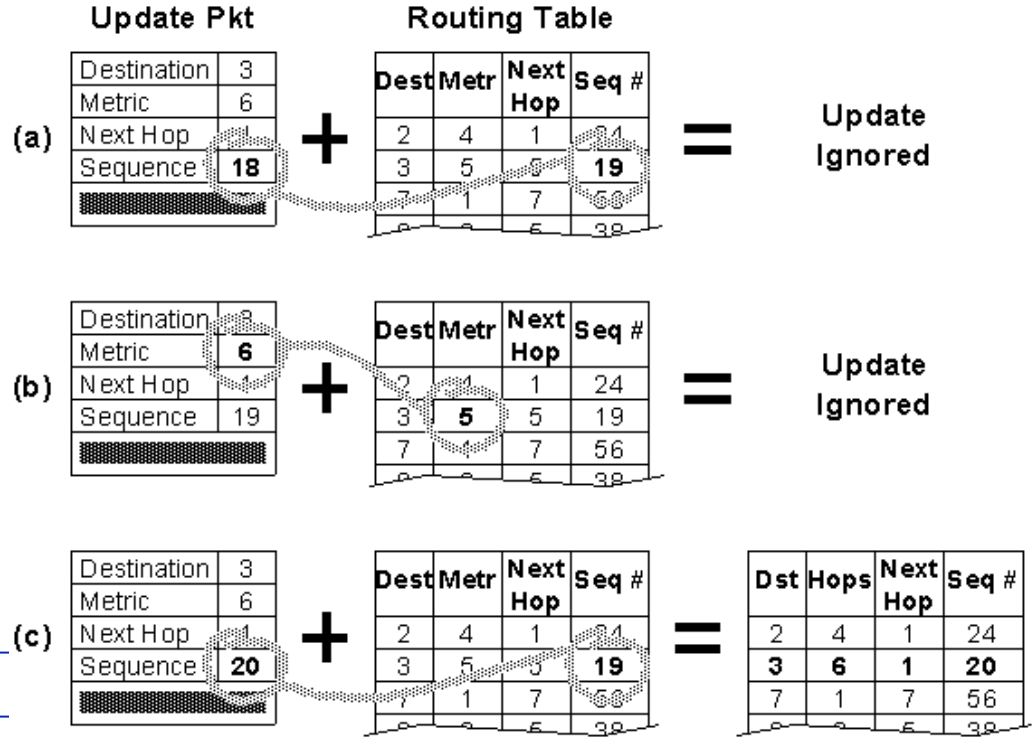
Setando o SeqNr em próprio anúncio:

- Incremente o SeqNr em 2 para cada entrada anunciada
- Se um nó não é mais alcançável, incremente o SeqNr de 1 e coloque Metrica em ∞

DSDV – uso de SeqNr

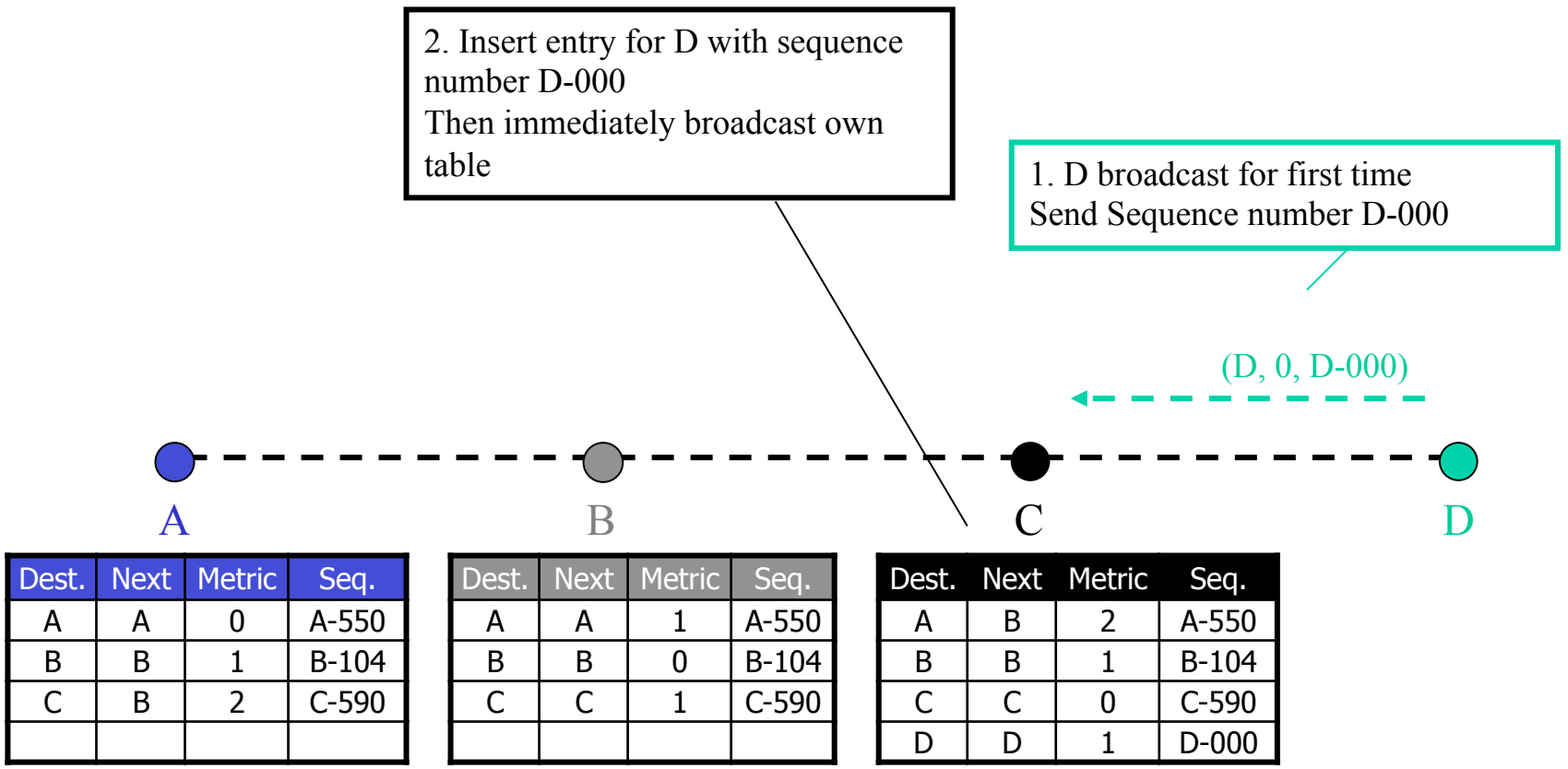
Comparação de rotas para um destino:

- Só se o SeqNr da atualização for igual a SeqNr que consta na sua tabela, o roteador irá comparar as distâncias e armazenar a menor delas
- Se SeqNr do anúncio for maior, adota a nova distância recebida
- Isso evita “count-to-infinity”, pois evita usar anúncios antigos



Departamento de Informática

DSDV – Exemplo
(Conexão de nó D)



Dest.	Next	Metric	Seq.
A	A	0	A-550
B	B	1	B-104
C	B	2	C-590

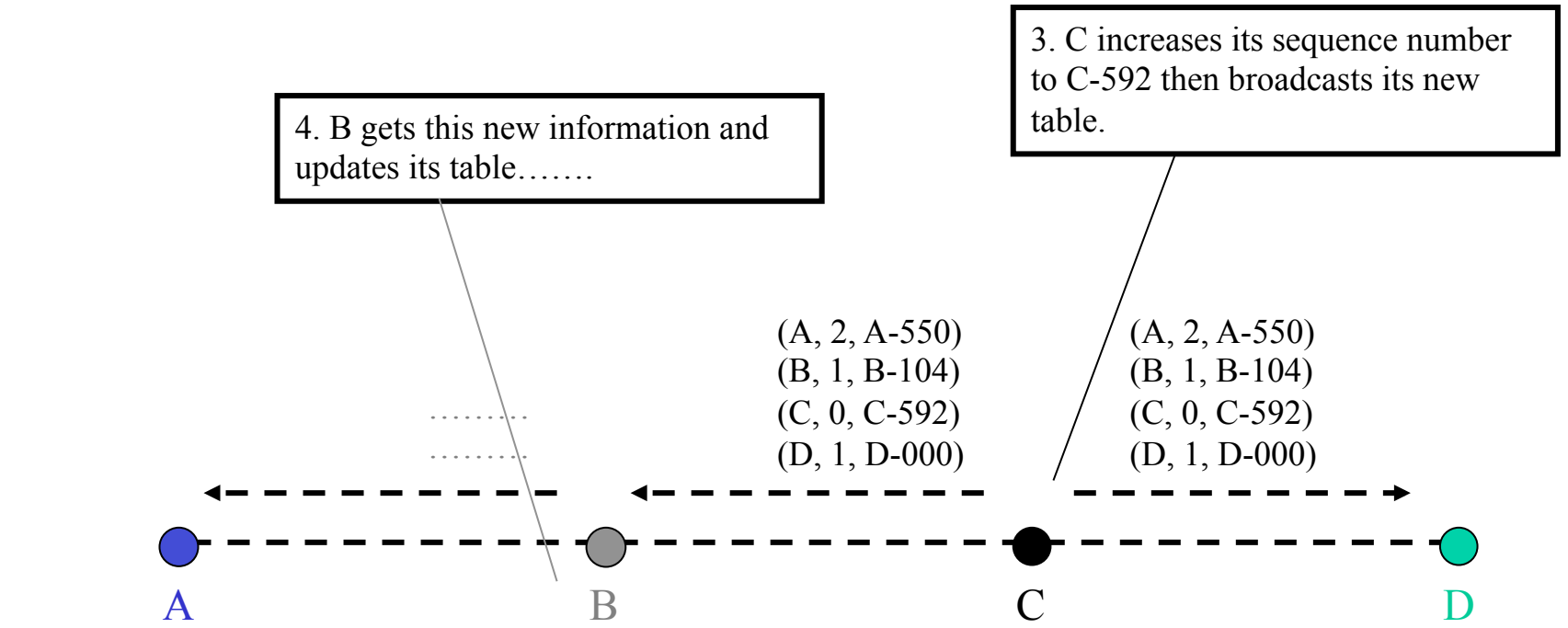
Dest.	Next	Metric	Seq.
A	A	1	A-550
B	B	0	B-104
C	C	1	C-590

Dest.	Next	Metric	Seq.
A	B	2	A-550
B	B	1	B-104
C	C	0	C-590
D	D	1	D-000

Fonte: A. Kesharwani (SlideShare)

PUC RIO

DSDV – Exemplo
(Conexão de nó D)



Dest.	Next	Metric	Seq.
A	A	0	A-550
B	B	1	B-104
C	B	2	C-590

Dest.	Next	Metric	Seq.
A	A	1	A-550
B	B	0	B-104
C	C	1	C-592
D	C	2	D-000

Dest.	Next	Metric	Seq.
A	B	2	A-550
B	B	1	B-104
C	C	0	C-592
D	D	1	D-000

Fonte: A. Kesharwani (SlideShare)

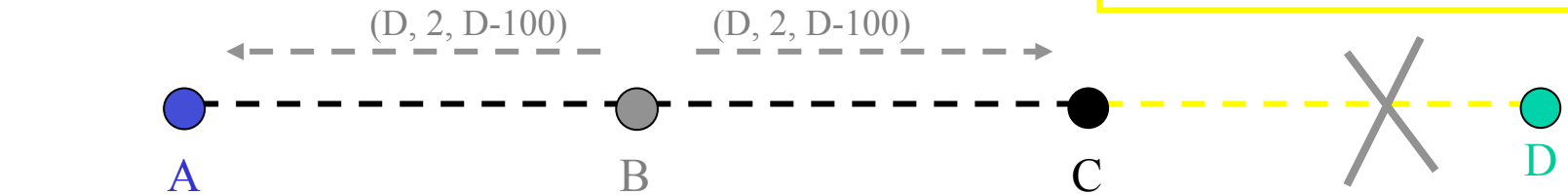
Departamento de Informática

DSDV - Exemplo

Sem o efeito “count-to-infinity”

2. B does its broadcast
-> no affect on C (C knows that B has stale information because C has higher seq. number for destination D)
-> no loop -> no count to infinity

1. Node C detects broken Link:
-> Increase Seq. Nr. by 1
(only case where not the destination sets the sequence number -> odd number)



Dest.	Next	Metric	Seq.
...	...	...	
D	B	3	D-100

Dest.c	Next	Metric	Seq.
...	...	...	
D	C	2	D-100

Dest.	Next	Metric	Seq.
...	...	...	
D	D	∞	D-101

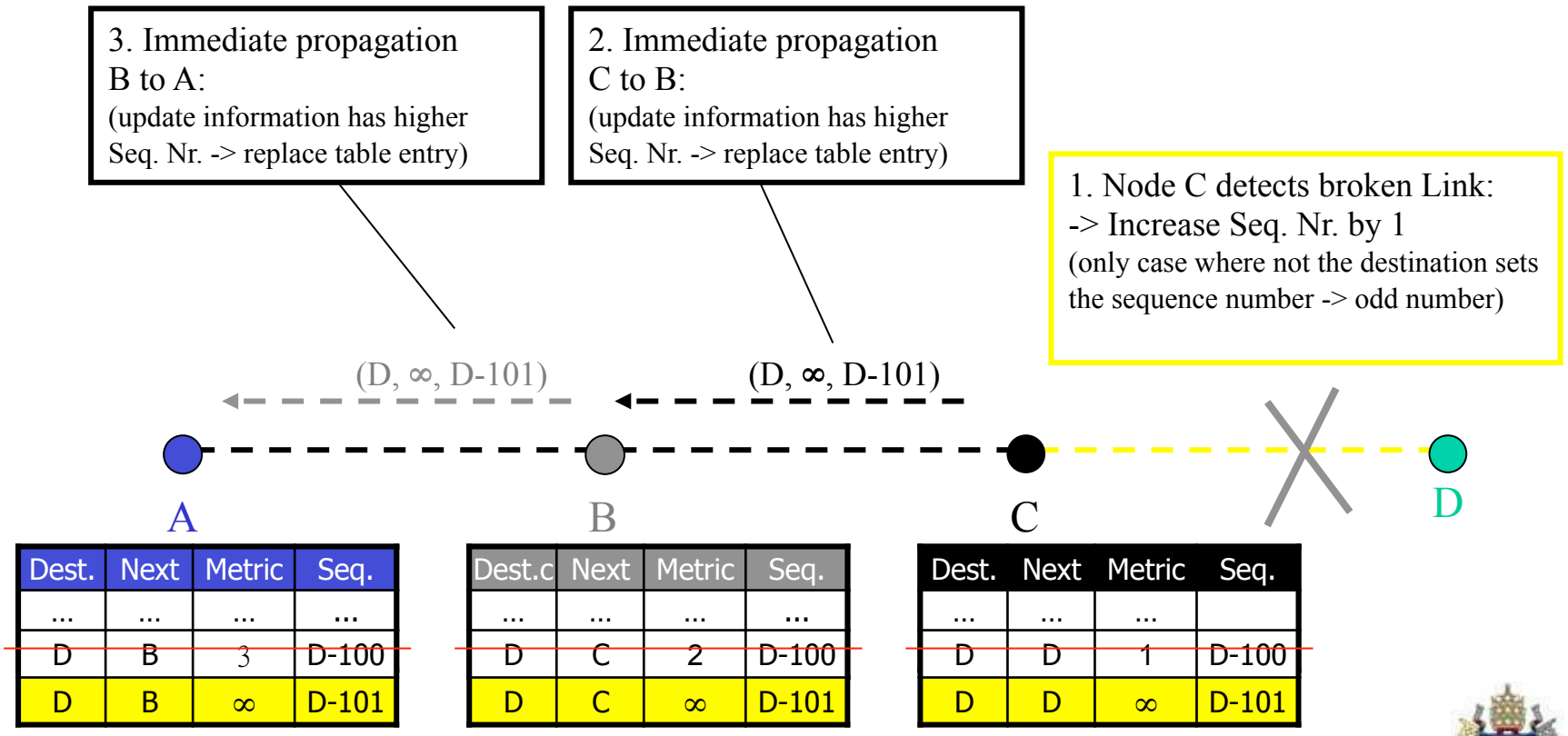
Fonte: A. Kesharwani (SlideShare)

PUC RIO

Departamento de Informática

DSDV - Exemplo

Anúncio imediato da desconexão do nó D



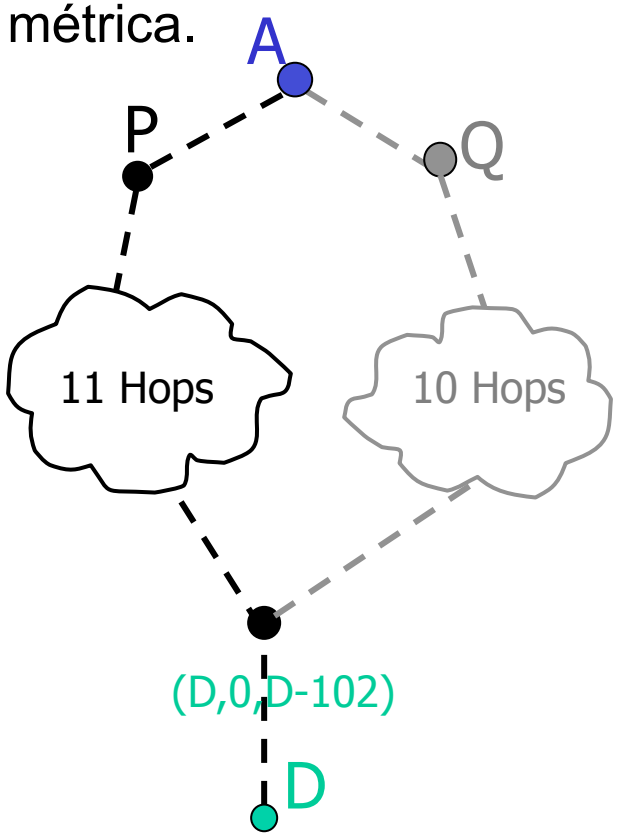
Fonte: A. Kesharwani (SlideShare)

PUC RIO

Departamento de Informática

DSDV – Esperando Rotas se estabilizarem

Problema da Flutuação: A pode receber anúncios de rota para D por várias partes da rede, e deve aguardar até que conheça a rota de menor métrica.



- Entry for D in table of A: [D, Q, 14, D-100]
- D enters the network
- D makes a broadcast with Seq. Nr. D-102
- A receives from P Update (D, 15, D-102)
  -> Entry for D in A: [D, P, 15, D-102]
  A must propagate this route immediately!

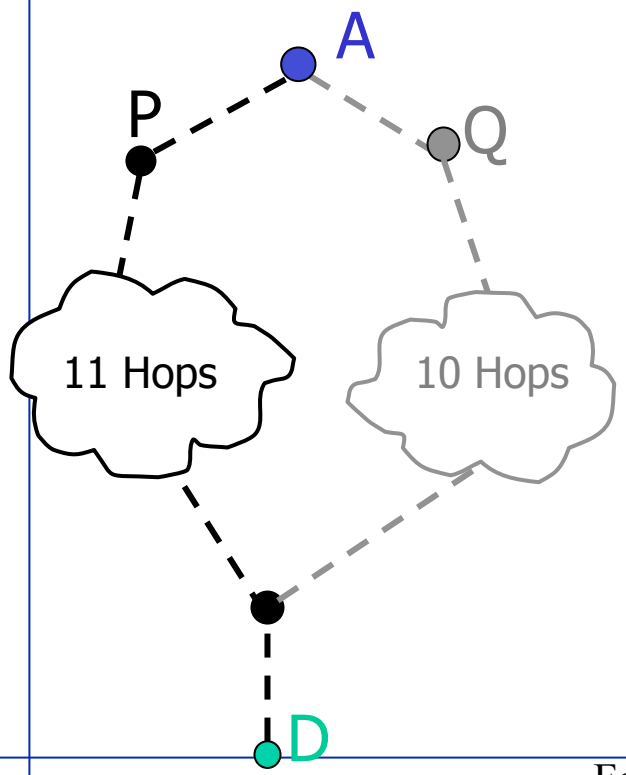
- A receives from Q Update (D, 14, D-102)
  -> Entry for D in A: [D, Q, 14, D-102]
  A must propagate this route immediately!

Fonte: A. Kesharwani (SlideShare)

PUC RIO

DSDV – Esperando Rotas se estabilizarem

- A mede o tempo médio entre o primeiro anúncio de rota e o anúncio da melhor rota (para D)
- Durante esse tempo, inibe atualizações para os vizinhos (a fim de "confirmar" se a mudança de topologia se estabilizou)



- Record last and avg. Settling Time of every Route in a separate table. (Stable Data)
  Settling Time = Time between arrival of first route and the best route with a given seq. nr.
- A still must update his routing table on the first arrival of a route with a newer seq. nr., but will wait to advertise it.
- Time to wait is proposed to be 2*(avg. Settling Time).

Fonte: A. Kesharwani (SlideShare)

DSDV - Resumo

Vantagens:

- Simplicidade (simples inclusão do SeqNr permitiu estender DV para MANETs)
- Rotas são mantidas sempre atualizadas, todos os nós podem utilizá-las

Desvantagens:

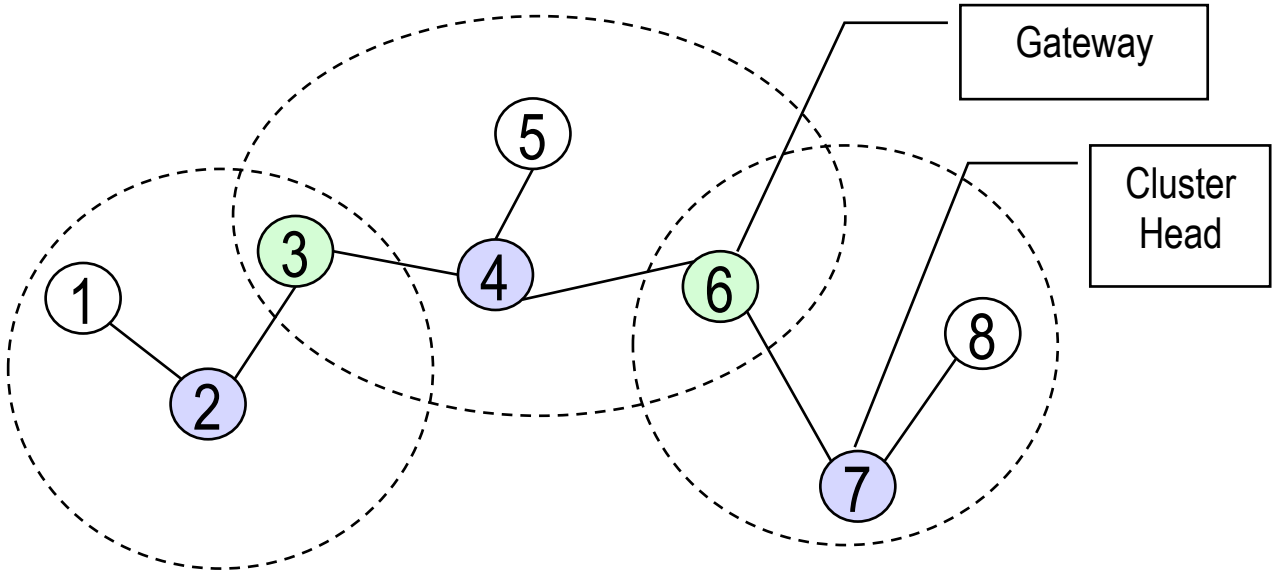
- Muito trafego do anúncios de rotas
- É inviável para MANETs com alta taxa de mobilidade ou muitos nós

Clusterhead Gateway Switch Routing (CGSR)

Rede é particionada em clusters, cada cluster tem um **cluster-head** (que contém a tabela de roteamento para outros clusters)

Gateways são nós que estão no alcance de pelo menos 2 cluster-heads

- Ambos devem ser nós com pouca mobilidade (Least Cluster Change Algorithm)



Clusterhead Gateway Switch Routing (CGSR)

Cada nó contém 2 tabelas:

- O Cluster-head Member Table: o cluster-head vizinho que deve ser usado para roteamento;
- Tabela de roteamento DSDV: próximo hop para alcançar o destino;

Roteamento:

1. Nós enviam pacotes para o cluster-head de seu cluster
2. Cluster-head encaminha para outros cluster-head através de nós gateways (por DSDV)
3. Em algum momento, o cluster-head encaminha para o nó destino

Departamento de Informática

Wireless Routing Protocol (WRP)

Proposto por J.J.Garcia-Luna-Aceves e S. Murthy:

- É protocolo proativo e table-driven;
- da classe de algoritmos *Path Finding Algorithms* (PFA);
- Usa informacão sobre o "nr de hops ou custo de rotas" e o penúltimo hop no caminho mais econômico para cada destino.

S.Murthy and J.J.Garcia-Luna-Aceves, "An Efficient Routing Protocol for Wireless Networks", ACM Mobile Networks and Applications Journal, pp 183-197, 1996.

PUC RIO

Path Finding Algorithms

- Cada nó mantém a árvore geradora de caminho mínimo Shortest Path Spanning Tree (SPST) de cada um de seus vizinhos.
- Com essa informação, cria a sua árvore shortest-path-spanning.
- Obs: Shortest path → Shortest Weighted Path
- Periodicamente cada nó difunde atualizações (broadcasts para seus vizinhos a 1 hop) de sua SPST, contendo:
  - Endereço (ou ID) destino
  - Distância até o destino
  - O ID do nó que é o penúltimo da sua rota (pela SPST) até esse destino.

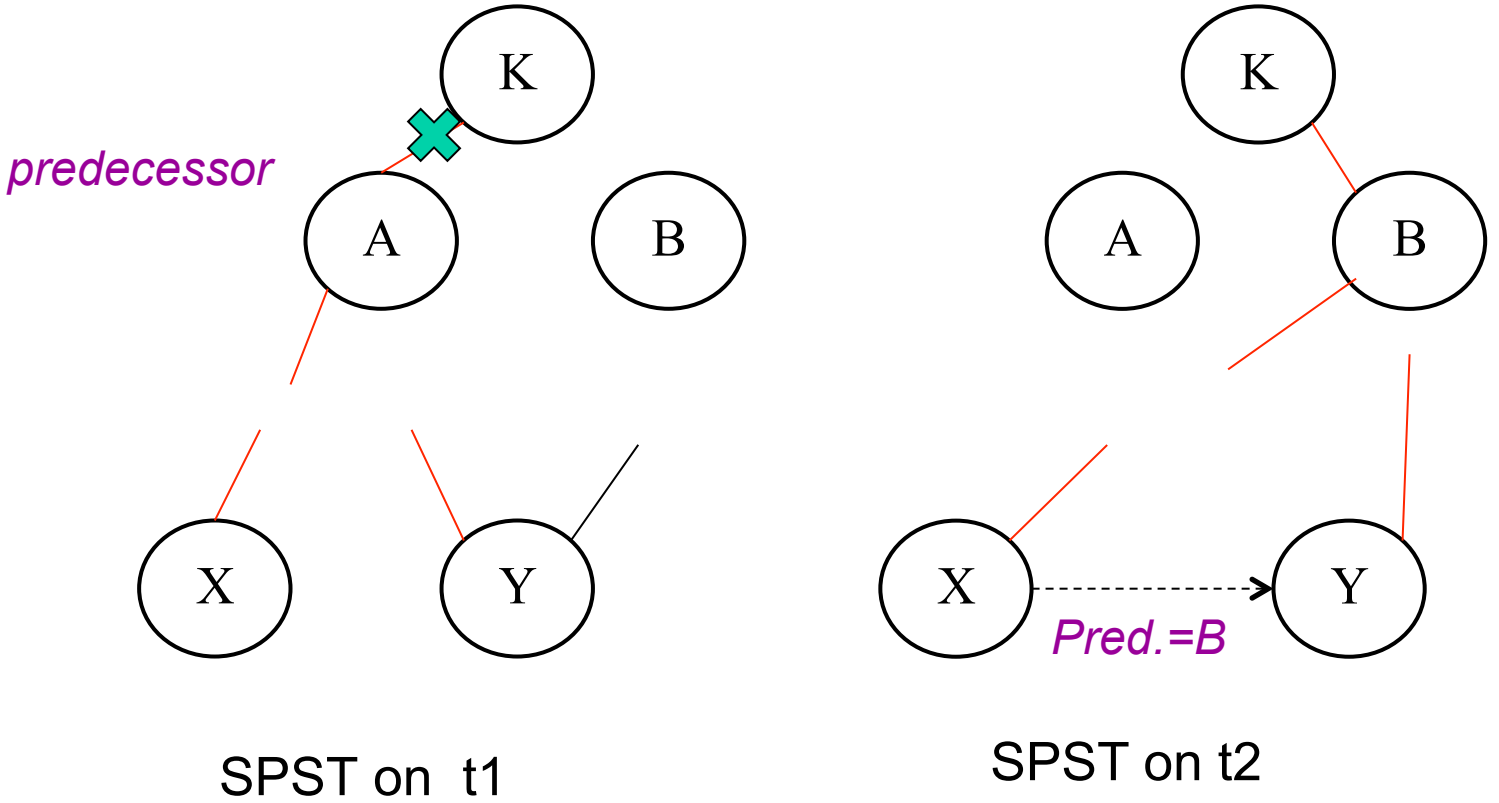
Departamento de Informática

Wireless Routing Protocol

- Usa a ideia geral de PFA
- Chamemos o penúltimo nó de *predecessor*.
- Quando um nó recebe um update de um vizinho irá verificar a consistência da informação do predecessor informada por todos os seus vizinhos.
- A partir da árvore SPST sabe qual é o próximo hop para encaminhar mensagens na direção do destino (chamemos esse de nó *successor*)
- Quando um nó recebe uma atualização de um nó vizinho, compara o seu *predecessor* com o *predecessor* do Update. Se forem diferentes, sabe que houve uma mudança do SPST perto do destino.

PUC RIO

WRP: Atualização da árvore SPST



Departamento de Informática

Wireless Routing Protocol

Cada nó mantém 4 tabelas:

- Distance Table
- Routing Table
- Link-Cost Table
- Message Retransmission List (MRL) Table

PUC RIO

WRP: Routing e Link-cost Tables

Vejamos as tabela para um nó específico:

Tabela de roteamento:

- Para cada destino K, contém:
  - Endereço (ou ID) de K;
  - Distância mais curta até K.
  - O sucessor no caminho mais curto até K.
  - O predecessor nesse caminho mais curto até K.

Link-cost table:

- O custo de transmitir a mensagem através do link para cada um dos vizinhos.
- E o tempo que transcorreu desde que recebeu uma mensagem válida do vizinho.

WRP: Distance Table

Distance Table contém:

- Para cada destino K, nó mantém a distância até K através de cada um de seus vizinhos e o ID do nó predecessor ao longo dessa rota.
- Essa tabela é usada para construir (e manter) a STSP.

Message Retransmission List (MRL) Table contém:

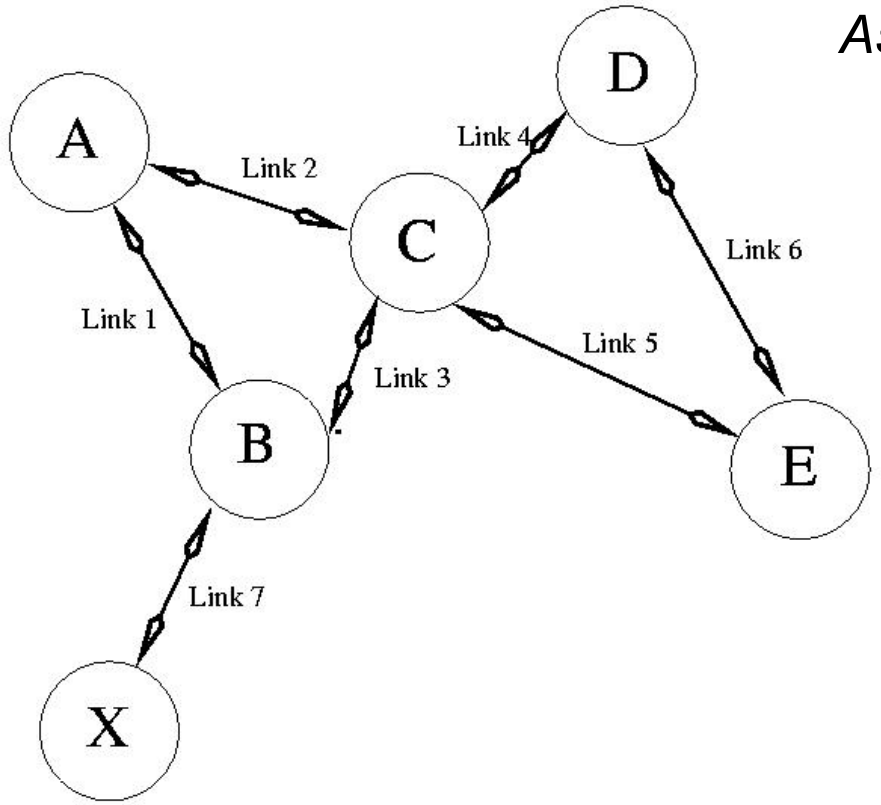
- SeqNr das próprias msgs de Update
- Contador de Retransmisões de cada Update
- Flag que indica, para cada vizinho, se recebeu um Ack para uma msg de Update
- A lista de entradas modificadas enviadas em cada msg de Update

WRP: Mensagens

- Após processar as atualizacões dos vizinhos, ou quando detectar uma mudança no enlace para algum vizinho, envia uma msg de Update
- Em resposta a msg de Update, um nó envia um ACK
- Nós também trocam msgs Hello para verificar a conectividade com vizinhos
- Também usa números de sequência nas entradas das tabelas e nas mensgens de Update para comparar a atualidade da informação

Departamento de Informática

WRP: um exemplo



Assuma que o custo do Link 2 é 10.

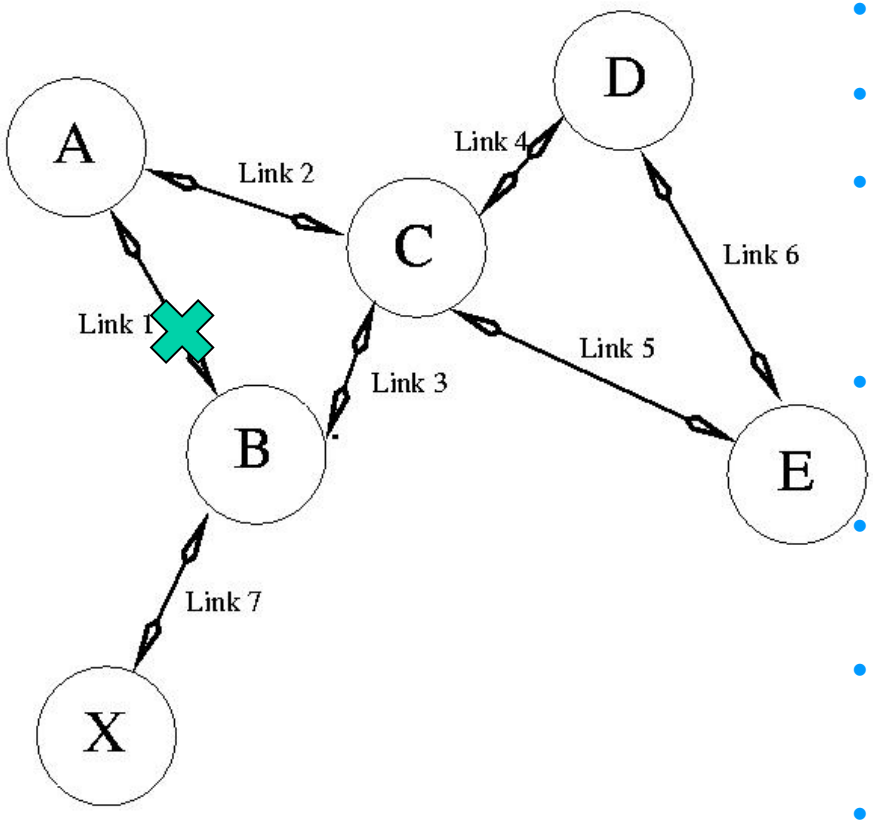
Tabela de roteamento em A

Dest	Cost	Pred	Succ
A	0	A	A
B	1	A	B
C	2	B	B
D	3	C	B
E	3	C	B
X	2	B	B

PUC RIO

Departamento de Informática

WRP: um exemplo

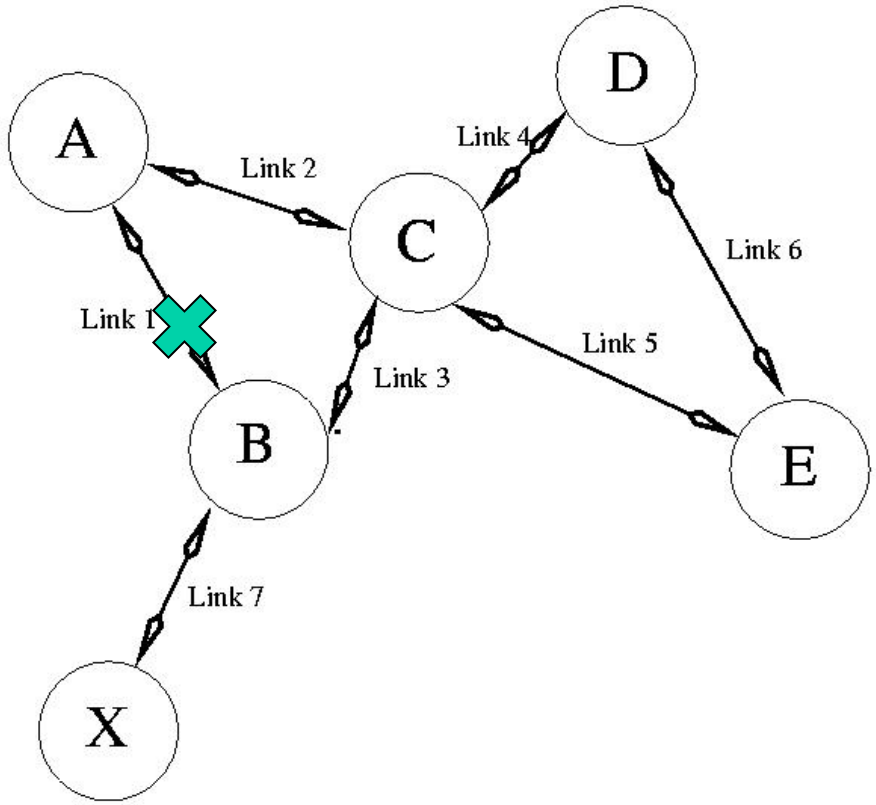


Quando Link 1 quebra:

- A irá perceber a falha
- Seja X o nó destinatário
- A atualiza Custo_para X = ∞ e seus valores de predecessor = NULL & successor = NULL
- A difunde msg Update para vizinhos (o nó C).
- C sabe que sua rota para X é através de B.
- Essa informação é difundida para todos vizinhos, incluindo A
- A então percebe pela SPSP de C, que consegue alcançar X através de C.
- Atualiza Custo_para (X)= 12, preceessor = B e sucessor = C

PUC RIO

WRP: um exemplo



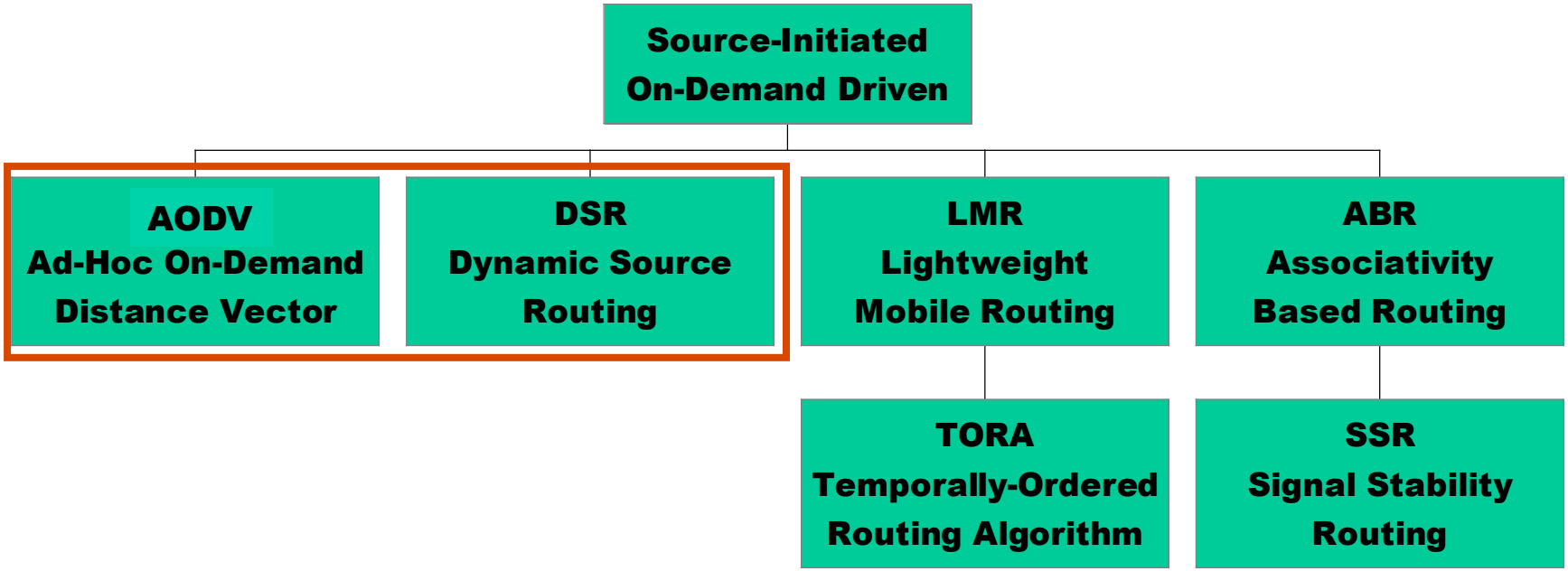
A Tabela de Roteamento de A então ficaria assim:

Dest	Cost	Pred	Succ
A	0	A	A
B	11	C	C
C	10	A	C
D	11	C	C
E	11	C	C
X	12	B	C

Algoritmos iniciados na fonte (Source initiated)

- Uma rota somente é criada quando o nó fonte precisa enviar para o destino:
  - Quando um nó precisa de uma rota, inicia o processo de descoberta de rota.
  - Este processo termina quando a rota é descoberta, ou quando todos os possíveis caminhos tiveram sido analisados.
- Uma vez que a rota é criada, usa-se um protocolo de manutenção de rota
- Não há necessidade de atualizações periódicas!
- É adequado para MANETs com:
  - demanda de comunicação esporádica e seletiva (apenas entre alguns pares de nós) e
  - a topologia muda com baixa frequência

A taxonomia de Algoritmos iniciados na fonte



D. B. Johnson and D. A. Maltz, “Dynamic Source Routing in Ad-Hoc Wireless Networks,” Mobile Computing, T. Imielinski and H. Korth, Eds., Kluwer, 1996, pp. 153–81.

C. E. Perkins and E. M. Royer, “Ad-hoc On-Demand Distance Vector Routing,” Proc. 2nd IEEE Wksp. Mobile Comp. Sys. and Apps., Feb. 1999, pp. 90–100.

Princípios Gerais de Protocolos Reativos (On Demand)

Descoberta de Rotas:

- Nó fonte difunde (broadcast para os vizinhos diretos) um RouteRequest (RREQ) com ID do destino BcastID único. Cada RREQ contém um *route record*, a ser preenchido com os nós percorridos durante o processo de descoberta.
- Quando um nó recebe este broadcast do RREQ:
  - Se o próprio é o destinatário então envia ao remetente uma resposta RouteReply RREP (contendo a rota armazenada em RREQ)
  - Se RREQ já foi recebido anteriormente, então simplesmente ignora
  - Senão, adiciona o próprio endereço no route record e também difunde RREQ para seus vizinhos
- O nó fonte em algum momento recebe RREP contendo a rota descoberta

Otimizações

- Se o diâmetro da rede é conhecido, pode-se limitar o número de difusões (através de TTL)
- A difusão pode ser controlada também no número de vizinhos

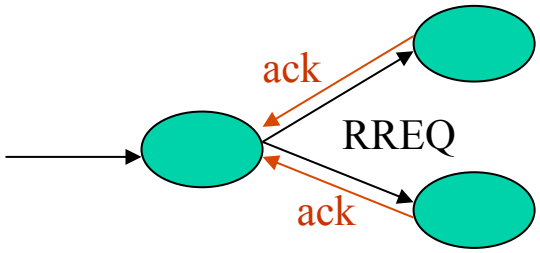
Princípios Gerais de Protocolos Reativos

- Cada nó móvel guarda localmente (em cache) as rotas que aprendeu, i.e. sequência de nós visitados por todos pacotes que passaram por ele;
- Cada entrada no cache tem um tempo de validade
- Antes de re-enviar, o nó primeiro verifica se possui em seu cache uma rota para o destino.
- Se possui, utiliza/informa essa rota; senão tenta descobrir uma rota usando o protocolo de descoberta de rotas (difundindo para os vizinhos)

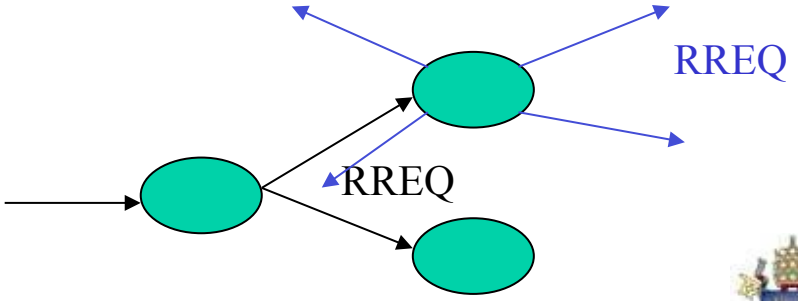
Source-initiated: Princípios Gerais

Manutenção de rotas:

- enquanto uma rota está em uso, o procedimento de manutenção de rota monitora o fluxo de pacotes, e informa ao nó remetente sobre eventuais quebras de rota
  - Ao enviar qualquer pacote (RREQ ou RREP) o nó remetente pode:
    a. Esperar por um Ack da camada de enlace (quando disponível)
    b. Escutar o meio para detectar se outros nós estão re-encaminhando este pacote
    c. Solicitar explicitamente um Ack da camada de rede
  - Se o nó detectar problemas de encaminhamento, pode:
    - informar isso ao nó de quem recebeu o pacote, e/ou
    - tentar descobrir outra rota



(A) Espera por Ack dos seguintes



(B) Ouvir se os seguintes estão propagando o RREQ

Ad-Hoc On Demand Distance Vector (AODV)

- Baseado em DSDV, mas cria rotas sob demanda (on demand);
- Cada nó mantém uma tabela de roteamento que contém o próximo hop para cada destino;
- Nó que quer enviar pacotes para nó D (e desconhece qualquer rota para D), inicia o processo de descoberta
  - Faz broadcast de um Route Request (RREQ) para os vizinhos;
  - Em algum momento, o próprio D, ou um nó intermediário com uma rota suficientemente recente em sua tabela) é alcançado por RREQ, que enviam um RREP como unicast;
  - A medida que RREQ é propagada através da rede, nós intermediários usam a mensagem para atualizar suas tabelas de roteamento (na direção do nó fonte –reverse routes)
- Cada nó mantém um número de sequência e um broadcast-ID para cada destino
  - SeqNr é para avaliar a atualidade da informação
  - BcastID é para filtrar RREQs repetidos

Departamento de Informática

AODV – Tipos de Mensagem

- RREQ – Route request
- RREP – Route reply
- RERR – Route error
- HELLO – msg periódica, para monitoramento de alcançabilidade de vizinhos diretos

PUC RIO

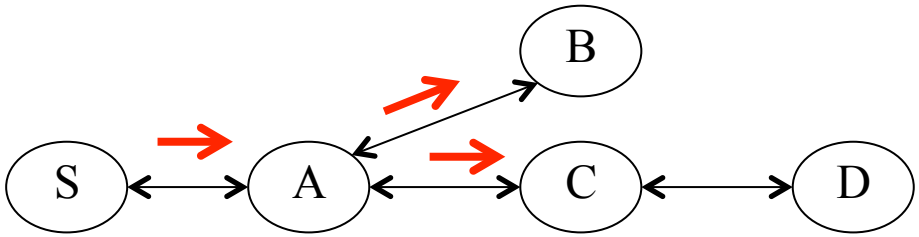
Route Request (RREQ)

Route Requests (RREQ) contém:

- Source identifier (SrcID)
- Destination identifier (DestID)
- Source sequence number (SrcSeqNum)
- Destination sequence number (DestSeqNum)
- Broadcast identifier (BcastID)
- Time to live(TTL)

- **BcastID**: é incrementado pelo nó fonte para cada RREQ, e juntamente com com SrcID identifica unicamente um RREQ
- **DestSeqNum**: é usado para garantir que todas as rotas não possuem loops e têm a informação mais atual
- **SrcSeqNum**: é o número mais recente que o nó fonte possui do Destino (um nó intermediário só responde RREP se possuir uma rota mais recente)

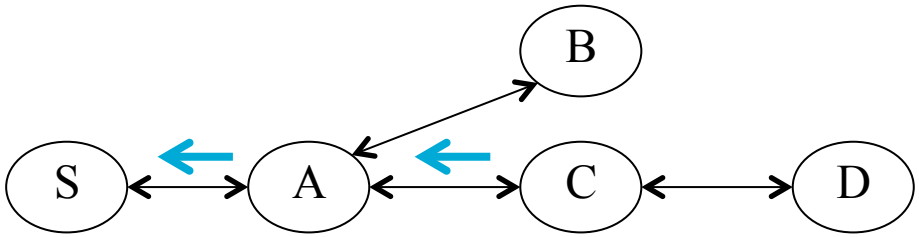
AODV: descoberta de Rota



- S precisa de rota para D
  - S cria um RREQ com (Id(D), BcastId, seqNr(D), Id(S)), e faz broadcast para seus vizinhos (ou seja, A)
- A recebe RREQ
  - Cria entrada na tabela p/ S [Dest=S, nexthop=S, hopcount =1] (reverse route)
  - Como não possui rota para D, faz broadcast RREQ com Id(D), BcastId, seqNr(D), id(S)) para B e C

Departamento de Informática

AODV: descoberta de Rota



- C recebe RREQ:
  - cria nova entrada na tabela p/ S [Dest=S, nexthop = A, hopcount =2] (reverse route)
  - Se tem D em sua tabela de roteamento, e seu SeqNr para D > SeqNr(D) em RREQ, então:
    - Envia para A unicast de RouteReply (RREP) com Id(D), maior SeqNr(D), Id(S)
    - Senão não reage.
- A recebe unicast RREP
  - Repassa Unicats RREP para S
  - Cria uma entrada na tabela p/ D com [Dest=D, nexthop=C, hopcount=2] (forward route)
- S recebe RREP
  - Cria uma entrada p/ D na tabela com [Dest=D, nexthop=A, hopcount=3] e SeqNr (D) atualizado (forward route)
  - Envia pacotes para D

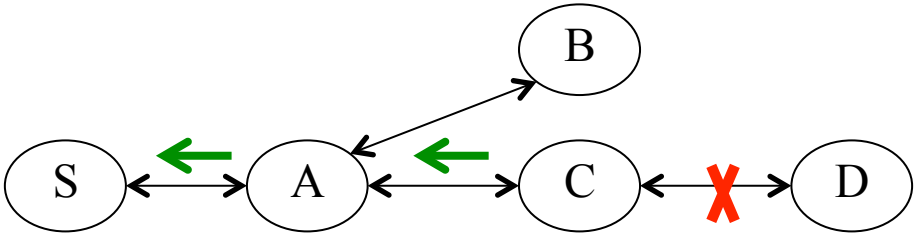
PUC RIO

Mobilidade no AODV

- Quando um nó fonte, S, se move, pode iniciar a descoberta de uma nova rota
- Se um nó intermediário ao longo de uma rota se move e torna-se inalcançável, o seu vizinho anterior (no caminho) percebe isto e propaga uma mensagem RouteError, para cada um de seus vizinhos anteriores,
  - Esta propagação prossegue até atingir o nó fonte, S
- Nós que fazem parte de uma rota ativa (em uso) emitem periodicamente mensagens “Hello” para informar sobre sua alcançabilidade

Departamento de Informática

AODV: manutenção de Rota



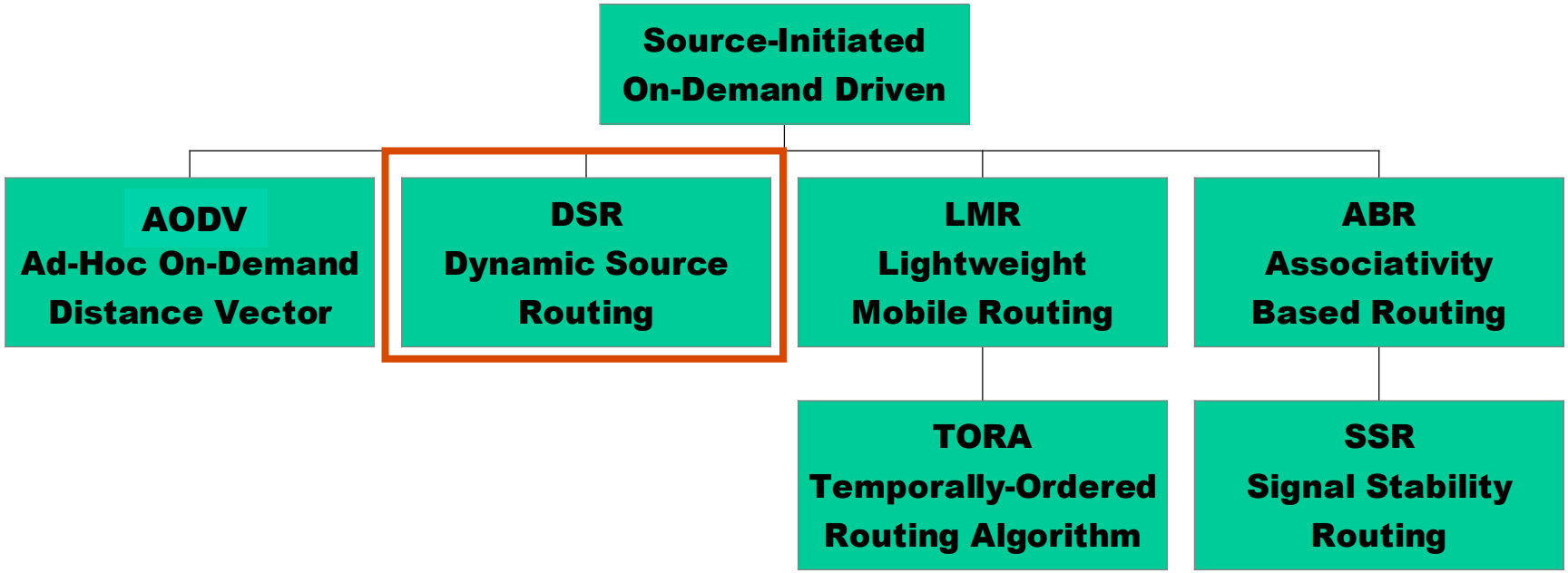
- C detecta não-alcançabilidade de D:
  - Remove rota para D
  - Cria uma mensagem RouteError (RERR) com Unreachable(D), e envia para todos os vizinhos “fluxo acima” (ou seja, para A)
- A recebe RERR de C:
  - Como na entrada para D o nexthop=C, remove entrada para D, e
  - Repassa RERR para S
- S recebe RERR de A:
  - Como na entrada para D o nexthop=A, e remove rota para D
  - Se necessário, inicia novo processo de descoberta de rota para D

PUC RIO

AODV: Controle de congestionamento

- Se nó fonte, S, não recebe RREP do destino D em período de tempo T, inicia nova descoberta de rota, mas repete isso até o máximo de RREQ_RETRIES;
- Para cada nova tentativa de descoberta de rota, o tempo de espera (pelo RREP) é dobrado.

A taxonomia de Protocolos inciados pela fonte



D. B. Johnson and D. A. Maltz, “Dynamic Source Routing in Ad-Hoc Wireless Networks,” Mobile Computing, T. Imielinski and H. Korth, Eds., Kluwer, 1996, pp. 153–81.

Departamento de Informática

DSR: Dynamic Source Routing

- É um outro protocolo reativo e source routing;
- Descoberta de Rotas é similar ao de AODV - com mensagens RREQ (broadcast) e RREP (unicast)
- **Source routing**: O remetente inclui a rota para o nó destino no cabeçalho do pacote Assim, o próximo nó para onde o pacote deve ser transmitido é identificado pelo próximo endereço na lista no cabeçalho do pacote. Não há tabelas de roteamento.
  - Rotas parciais em pacotes RREQ
  - Rotas completas em pacotes de dados
- Cada nó registra em seu cache as rotas (forward e reverse path) que vai aprendendo na passagem de RREQ, RREP e pacotes de dados.- A sequência de hops) copiada do cabeçalho de todos os pacotes recebidos

PUC RIO

Departamento de Informática

DSR: Descoberta de Rota

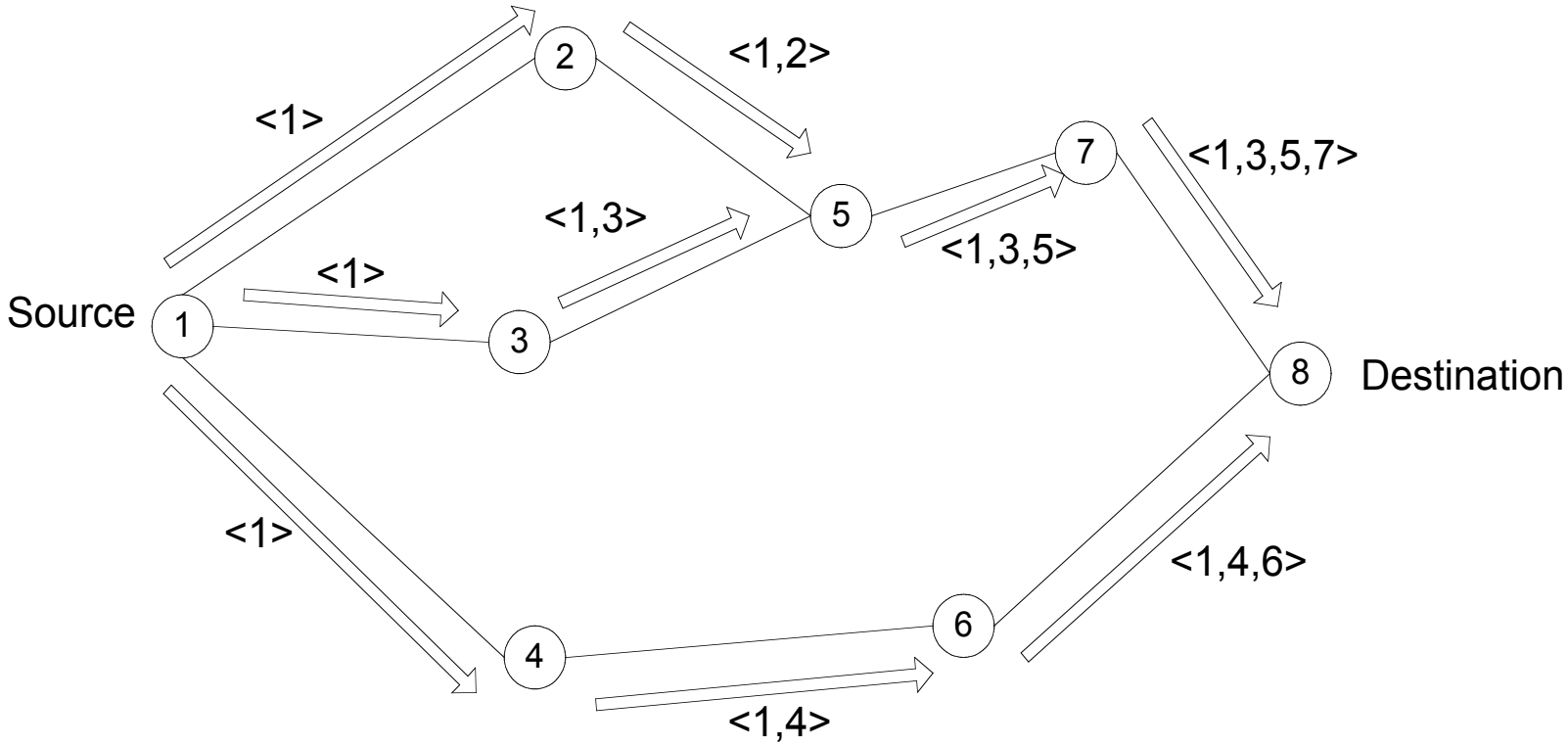
Ao receber RREQ:

- Se é o nó destino:
  - Cria um RREP para o nó fonte (copiando a rota acumulada de RREQ)
- Se é um nó intermediário:
  - Descarta a mensagem se já recebeu uma com mesmo (SenderID, BcastID) ou se descobre seu próprio endereço no cabeçalho
  - Se não, adiciona seu endereço ao cabeçalho da RREQ e difunde a mensagem para os vizinhos

PUC RIO

Departamento de Informática

DSR: Descoberta de Rota

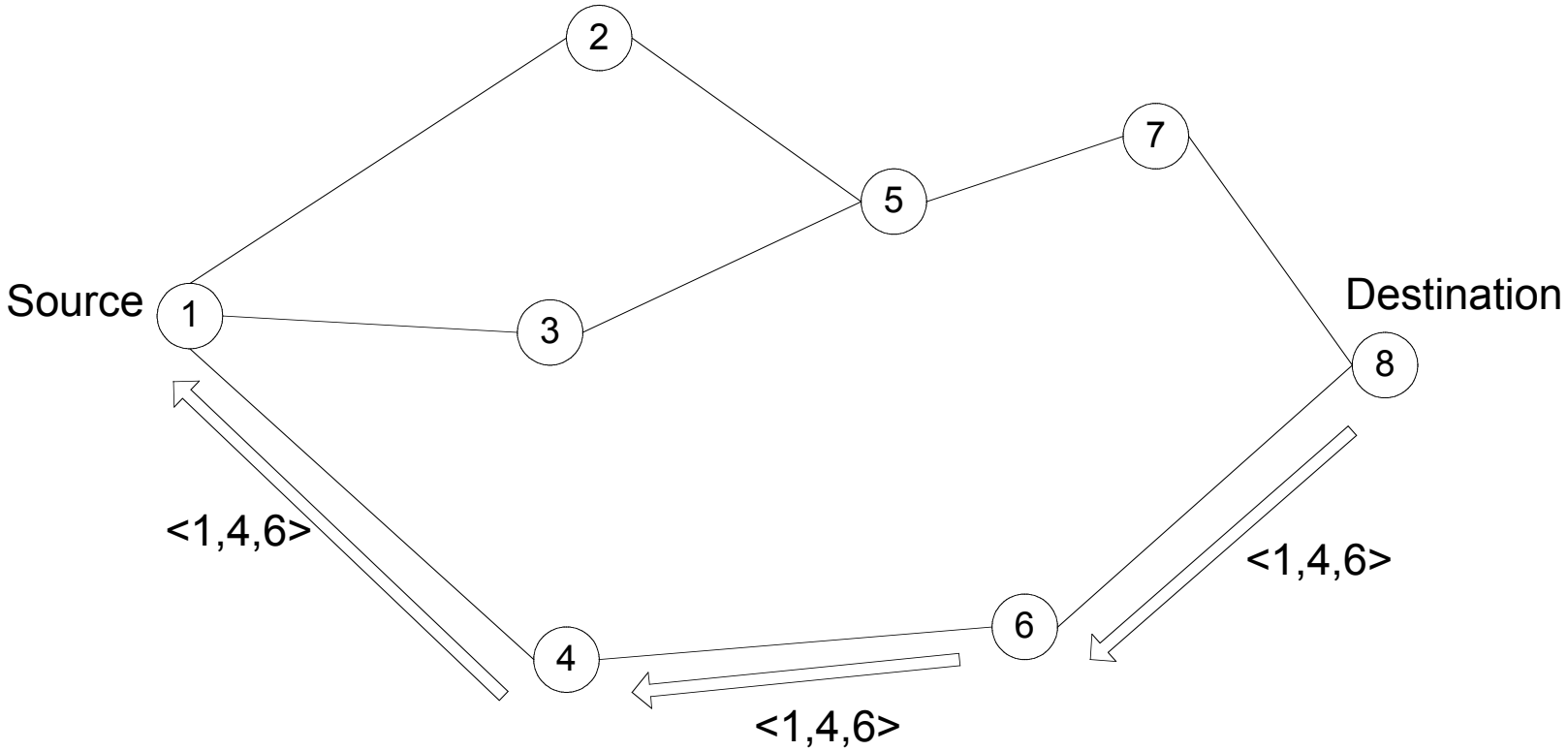


Propagação de RREQ: construindo o route record de 1 para 8

PUC RIO

Departamento de Informática

DSR: Descoberta de Rota

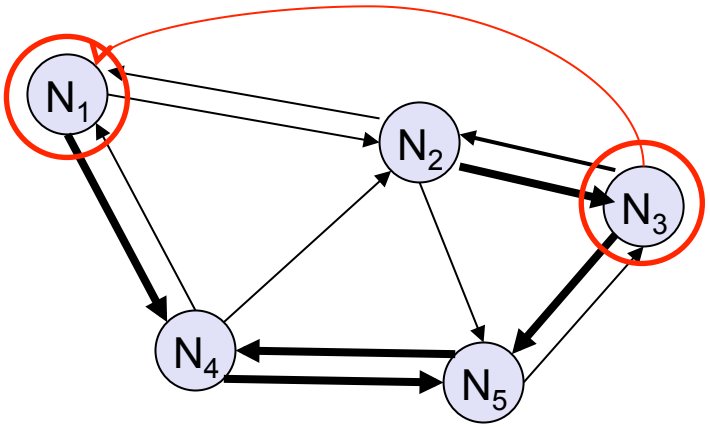


Envio de Route Reply com o Route Record

PUC RIO

Departamento de Informática

Exemplo de Descoberta de Rota
de N1 para N3



@N1: broadcast RREQ (N1,ID,N3), recebida por N2 e N4

@N2: broadcast RREQ ((N1,N2), ID, N3), recebida por N1, N3 e N5

@N4: broadcast RREQ ((N1,N4), ID, N3), recebida por N1, N2, N5, que descartam o pacote

@N5: broadcast RREQ ((N1,N2,N5), ID, N3), recebido por N3 e N4

@N3: reconhece que é o destinatário e escolhe o caminho (N1,N2) por ser mais curto

@N3: envia a resposta RREP para N1 informando a rota (N1,N2,N3)

PUC RIO

Departamento de Informática

DSR: Dynamic Source Routing

Manutenção de rotas:

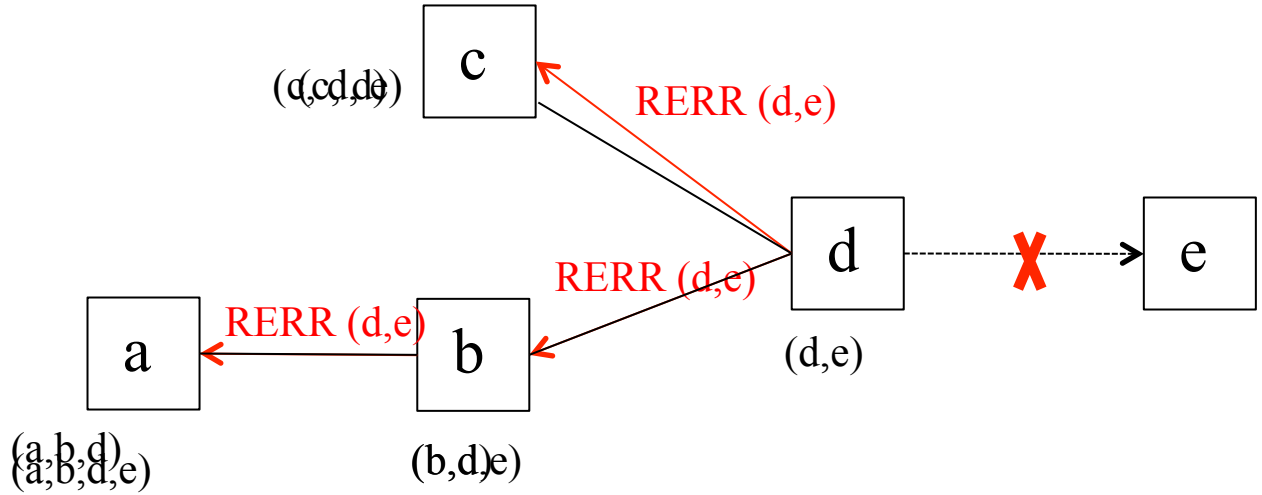
- Quando um nó intermediário da rota (ou o destino) deixa de ser alcançável pelo nó anterior:
  - uma rota alternativa precisar ser descoberta, e
  - todos os nós da rota ainda intacta são notificados.
- Usa-se Pacotes de Erro de Rota (RERR):
  - RERR são criados quando a camada de enlace detecta um erro de transmissão;
  - O RERR inclui os endereços dos dois nós do enlace quebrado;
  - RERR são enviados pelo caminho reverso para o nó fonte;
  - Todos os nós ao longo deste caminho removem/truncam essa parte da rota em seus caches.
- O nó fonte (ou qualquer nó notificado) pode iniciar nova descoberta usando RREQ.

PUC RIO

Departamento de Informática

DSR: Propagação do RERR

- Atualização dos caches quando há quebra de um enlace



PUC RIO

Departamento de Informática

Protocolos Reativos

- Usa meio sem fio de forma mais eficiente (só quando há demanda)
- Mas a descoberta de rotas é baseada em broadcast/flooding na rede ➔ há necessidade de controlar/limitar broadcast
- No DSR: o tamanho do header dos pacotes de dados cresce com do comprimento da rota;
- No AODV: route record só é usado no RREQ e RREP
- Possível colisão (interferência no meio) de RREQ propagados por nós vizinhos
- Problema do RouteReply Storm: quando vários nós intermediários conhecem a rota para o nó destino estes podem gerar muitas mensagens de RREP para o nó fonte
- Assume que os enlaces são bi-direcionais
- Funciona bem somente se demanda por comunicação é pouco frequente, e quando há baixa mobilidade (mudança de vizinhos de um nó próximo do nó destino).

Comparação

Protocol property	DSDV	DSR	AODV	WRP
Loop free	Yes	Yes	Yes	Yes
Multicast routes	No	Yes	No	No
Distributed	Yes	Yes	Yes	Yes
Unidirectional link support	No	Yes	No	No
Multicast	No	No	Yes	No
Periodic broadcast	Yes	No	Yes	Yes
QoS support	No	No	No	Yes
Routes maintained in	Route table	Route cache	Route table	Route table
Route cache/table timer	Yes	No	Yes	Yes
Reactive	No	Yes	Yes	No

Métricas Alternativas

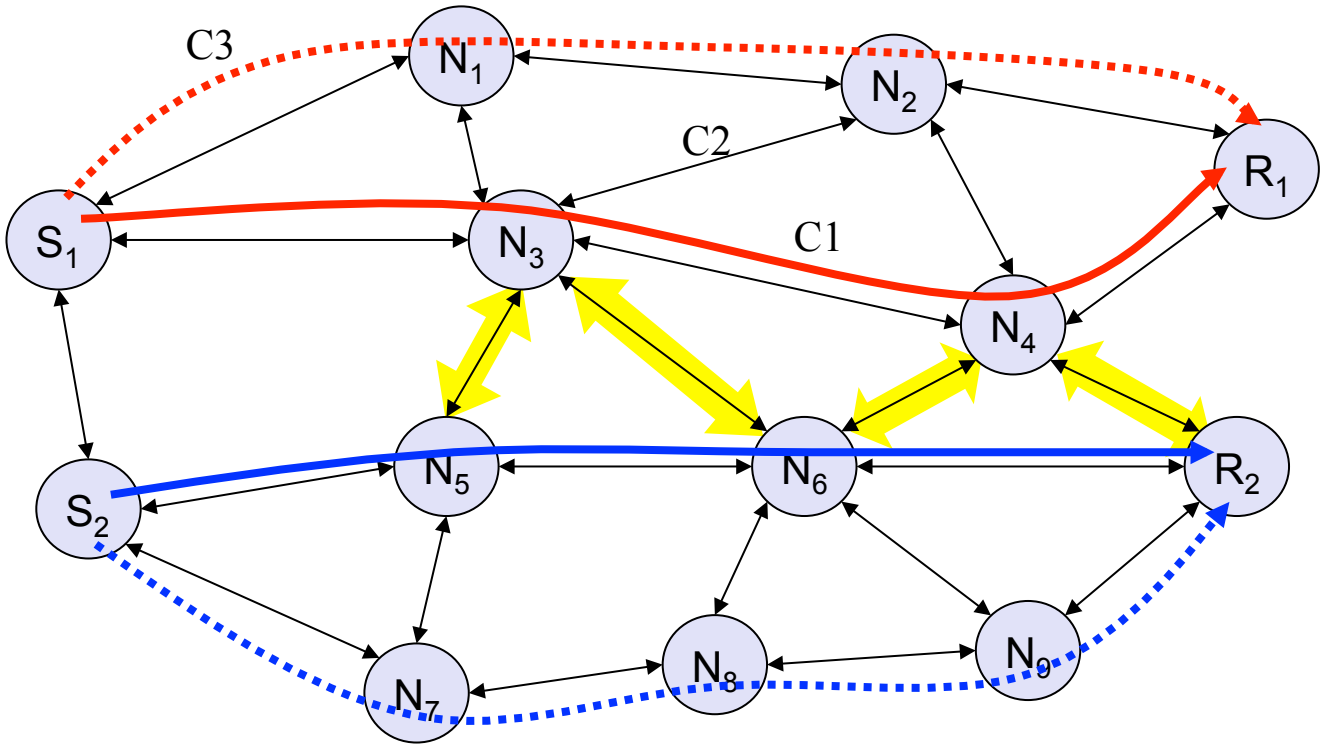
- O número de hops é somente uma das possíveis métricas para determinar a melhor rota
- Geralmente, é aconselhável levar em conta também a qualidade dos enlaces
- Poderia-se usar um critério que tentasse combinar:

  Número mínimo de hops com maximização da qualidade média dos enlaces
- A qualidade do enlace é fortemente determinada pela probabilidade de interferência em transmissões simultâneas de vizinhos

  ➔ Nesse caso, o quanto menor for o número de vizinhos diretos, melhor
- A métrica de menor interferência é usada no algoritmo **Least Interference Routing (LIR)**

Departamento de Informática

LIR: um Exemplo



Interferência

vizinhos

Custo de cada nó = número de seus vizinhos diretos
Custo da rota = Σ custo dos nós na rota
C1 = custo(S1,N3, N4, R1) = 16
C2 = custo(S1, N3, N2, R1) = 15
C3 = custo(S1,N1,N2, R1) = 12

PUC RIO

Exemplos de Algoritmos que levam em conta interferência

- Least Interference Routing (LIR)
  - Custo de um nó é o numero de vizinhos que podem receber o pacote de broadcast
  - Custo do caminho = soma do custo de seus nós
  - Implementação simples: basta descobrir/manter atualizado o número de vizinhos
- Max-Min Residual Capacity Routing (MMRCR)
  - Custo do caminho baseado em uma função da probabilidade de uma transmissão bem-sucedida e interferência (que depende da “atividade” dos vizinhos)
- Least Resistance Routing (LRR)
  - Custo do caminho baseado em estimativa da interferência com vizinhos, rotas que passam pelo nó (gargalos) e outras transmissões de controle

Departamento de Informática

Alguns Outros Protocolos de Roteamento

- Pro-ativos (Table driven)
  - FSLS – Fuzzy Sighted Link State
  - FSR – Fisheye State Routing
  - OLSR – Optimized Link State Routing Protocol
  - TBRPF – Topology Broadcast Based on Reverse Path Forwarding
- Reativos
  - AODV – Ad hoc On demand Distance Vector
  - DSR – Dynamic Source Routing
- Estrutura Hierarquica
  - CGSR – Clusterhead-Gateway Switch Routing
  - HSR – Hierarchical State Routing
  - LANMAR – Landmark Ad Hoc Routing
  - ZRP – Zone Routing Protocol
- Com auxílio de informação de localização
  - DREAM – Distance Routing Effect Algorithm for Mobility
  - GeoCast – Geographic Addressing and Routing
  - GPSR – Greedy Perimeter Stateless Routing
  - LAR – Location-Aided Routing

MANETs na Prática

Modelo teórico de MANETs é muito geral. Onde todos os nós:

1. estão sujeitos à mobilidade irrestrita;
2. são dispositivos de propósito geral (e são igualmente aptos a fazer o papel de intermediários/roteadores);
3. possuem recursos (p.ex. energia) escassos;
4. participam do roteamento para a mesma aplicação, e não têm demanda "local" por recursos – e.g. de usuários locais;
5. usam o mesmo protocolo/tecnologia de comunicação;
6. São homogêneos (a menos de quantidades de recursos) e possuem o mesmo raio de cobertura de rádio
7. desconhecem as suas posições (não possuem GPS)

Referências:
M. Conti, S. Giordano, Multihop Ad Hoc Networking: **The Theory**, IEEE Comm. Magazine, 2007
M. Conti, S. Giordano, Multihop Ad Hoc Networking: **The Practice**, IEEE Comm. Magazine, 2007

MANETs na Prática

Na prática (mundo real), há várias variantes de MANETs orientadas a tipos de aplicação bem específicos, onde há restrição a uma (ou mais) característica(s) do modelo geral:

- Redes em malha (Mesh networks)
  - Rede heterogêneos, com nós (roteadores ou provedores de acesso à Internet)
  - Mobilidade pouco frequente/inexistente;
- Delay Tolerant Networks (DTNs) – redes oportunísticas
  - Encaminhamento store-and-forward (Delay Tolerant Networks) ou redes com alta densidade de dispositivos
  - Mobilidade não é o problema, mas a solução p/ encaminhamento; dispositivos multi-rede e tolerante a desconexões
- VANETs – redes ad hoc veiculares
  - Nós sem limitação de recursos, alto grau de mobilidade; banda de RF é escassa
- WSN – Redes de Sensores sem Fio
  - Energia é o recurso escasso, baixa mobilidade, nós deixam a rede quando não têm energia

Departamento de Informática

Protocolos para Redes Mesh

Reactive Protocols:

- Dynamic Source Routing Algorithm (DSR):
- Ad-hoc On-Demand Distance Vector Routing Algorithm (AODV):
- Link Quality Source Routing Algorithm (LQSR)
- SrcRR Routing Algorithm:

Proactive Routing Protocols:

- Destination Sequenced Distance Vector Routing Algorithm (DSDV):
- Optimized Link State Routing Protocol (OLSR):
- Scalable Routing using heat Protocol:

Hybrid Routing Protocols:

- Hazy-Sighted Link State Routing Algorithm (HSLS)

Fonte: http://confine-project.eu/2012/04/17/confine-and-wmn/

PUC RIO

Departamento de Informática

Tabela Comparativa

Os principais protocolos de roteamento para redes Mesh

	Type of Ptotocol	Hello msg	Multicasting	Routing metrics	Loop free	Scalability	Reliability	Load balancing
DSR	Reactive	NO	NO	Shortest path	YES	NO	YES	NO
AODV	Reactive	YES	YES	Fastest & Shortest path	YES	NO	YES	NO
LQSR	Reactive	YES	YES	Hop count , RTT, ETX	YES	NO	YES	YES
SRCRR	Reactive	NO	NO	ETX, Hop count, Shortest path	YES	NO	YES	YES
DSDV	Proactive	NO	NO	Shortest path	YES	NO	YES	NO
OLSR	Proactive	YES	NO	Shortest path	YES	NO	YES	NO
MRP	Proactive	NO	YES	Hop count , QoS Metrics	YES	YES	YES	YES
Scalability Routing	Proactive	NO	YES	Hop count	YES	YES	YES	NO
HSLS	Hybrid	YES	YES	Hop count , RTT, ETX	YES	YES	YES	YES

Fonte: http://confine-project.eu/2012/04/17/confine-and-wmn/

PUC RIO