Universidade Federal do Amazonas
Instituto de Ciências Exatas
Departamento de Ciência da Computação
Programa de Pós-Graduação em Informática

SAULO JORGE BELTRÃO DE QUEIROZ

Avaliação de Roteamento Incremental em Redes em Malha Sem Fio Baseadas em 802.11

Manaus
2009

Saulo Jorge Beltrão de Queiroz

Avaliação de Roteamento Incremental em Redes em Malha sem Fio Baseadas em 802.11

Dissertação de mestrado apresentada ao Programa de Pós-Graduação em Informática do Departamento de Ciência da Computação da Universidade Federal do Amazonas, como parte dos requisitos necessários para a obtenção do título de Mestre em Informática. Área de concentração: **Avaliação de Desempenho em Redes de Computadores**.

Orientador: Prof. Dr.-Ing. Edjair de Souza Mota

Manaus

2009

Saulo Jorge Beltrão de Queiroz

Avaliação de Roteamento Incremental em Redes em Malha sem Fio Baseadas em 802.11

Dissertação de mestrado apresentada ao Programa de Pós-Graduação em Informática do Departamento de Ciência da Computação da Universidade Federal do Amazonas, como parte dos requisitos necessários para a obtenção do título de Mestre em Informática. Área de concentração: **Avaliação de Desempenho de Redes de Computadores**.

Banca Examinadora

Prof. Dr.-Ing. Edjair de Souza Mota (Orientador)
Universidade Federal do Amazonas

Prof. Dr. César Augusto Viana Melo
Universidade do Estado do Amazonas

Prof. Dr. Edson Nascimento Silva Júnior
Universidade Federal do Amazonas

Prof. Dr. Horácio Antônio Braga Fernandes
Universidade Federal do Amazonas

Manaus – 2009

Ao ...

Ao meu Senhor Jesus Cristo, à minha família.

Agradecimentos

Em especial, ao Senhor Jesus Cristo, por ter tido misericórdia de mim antes de tudo e por Ser a Ressurreição e a Vida.

Aos meus amados pai e mãe, meus verdadeiros amigos desta vida com os quais sempre posso contar.

Às minhas irmãs, Sheyla e Sheylane, por toda palavra de carinho, apoio e incentivo.

Ao meu orientador Edjair Mota, por todas oportunidades até então a mim concedidas, em especial a de cursar o mestrado. Também agradeço pelo aprendizado proporcionado enquanto estive na monitoria da disciplina Simulação de Sistemas.

Aos colegas de mestrado pelo companheirismo e ajuda técnica. Em especial para Andréa Giordana, Arlen Nascimento, João Luis, Loide Mara, Kaio Rafael e Valinda Maia.

À professora Andréa Pereira Mendonça, de quem recebi as primeiras aulas de algoritmos e que serviu de modelo para que eu decidisse-me pela carreira em informática.

A todo pessoal que trabalhou na secretaria do PPGI dentre os anos de 2007 à 2009, em especial à Elienai que me ajudou a solucionar vários problemas burocráticos.

Ao meu amor, Dulcina Hernandez, que entrou na minha vida no último ano deste mestrado e que, desde então, não cessou de ser incentivo e ânimo para que eu pudesse prosseguir.

Ao professor Edjard Mota pelo incentivo para realização do curso.

Ao professor Edleno Moura, em particular pelo auxílio prestado durante a viagem

que precisei fazer para fins de pequisa. Também agradeço pela oportunidade de participar na assistência da disciplina de Análise de Algoritmos.

Aos pesquisadores Eduardo Cerqueira e Augusto Neto, por compartilharem suas experiências técnicas comigo e pela grande hospitalidade com que me receberam em Portugal.

Ao professor César Melo, por ter dado início ao "processo de remoção de ruídos" de minha mente no que diz respeito à noção de auto-similaridade. ;)

À FAPEAM, pelo apoio financeiro.

Em especial, ao Senhor Jesus Cristo, por ser o α e o Ω.

Resumo

Neste trabalho abordamos a questão da manutenção da tabela de roteamento de estado de enlace em redes em malha sem fio (*Wireless Mesh Networks*, WMNs) baseadas em enlaces 802.11. Neste contexto, o *Internet Engineering Task Force* (IETF) propôs o padrão RFC 3626, que descreve o protocolo *Optimized Link State Routing* (OLSR). Em linhas gerais, o OLSR mantém uma visão global da rede (i.e. um grafo) por intermédio da troca periódica de mensagens de topologia e, após cada atualização no grafo, utiliza uma variação do algoritmo de Dijkstra para recalcular as rotas até todos os destinos conhecidos (conjunto V).

Devido o potencial de crescimento das WMNs e a presença de roteadores com baixo poder de processamento nestas redes, o tempo de cálculo das rotas no OLSR pode enfrentar sérios problemas de escalabilidade. Neste trabalho nós investigamos o uso do modelo de computação incremental limitada (*Bounded Incremental Computation*, BIC) como uma alternativa à proposta do OLSR para o recálculo da tabela de roteamento em WMNs 802.11. No modelo BIC, o tempo de execução dá-se em termos do conjunto δ de roteadores cujas rotas ótimas foram afetadas por uma mudança no grafo.

Após discutirmos aspectos de modelagem de canais 802.11, recorremos à simulação de horizonte infinito a fim de estudar a dinâmica das WMNs e obter aproximações empíricas para as curvas médias de tempo dos algoritmos de Dijkstra (T_{Dij}) e de Ramalingam e Reps (T_{RRs}), este último baseado no modelo BIC.

Os resultados sugerem que T_{Dij} domina assintoticamente T_{RRs} por um fator variável, i.e. $T_{RRs} = o(T_{Dij})$. Isso deve-se fundamentalmente ao fato de que, a cada recálculo, o OLSR supõe que $|V| = |\delta|$, enquanto que, na prática, $|\delta|$ mostrou-se frequentemente muito menor que $|V|$ em todos os cenários avaliados. Esses resultados indicam que cálculo das rotas pode ser escalável no contexto das WMNs 802.11 se o tempo de execução do procedimento correspondente satisfizer $O(f(|\delta|))$.

Palavras-Chave: redes *mesh*, algoritmos incrementais, roteamento.

Abstract

This work concerns about the routing table maintenance of link state routing protocols in Wireless Mesh Network (WMN) based on links 802.11. In this sense, the Internet Engineering Task Force (IETF) proposes the standard RFC 3626 which describes the Optimized Link State Routing (OLSR). In general words, OLSR keeps an updated view of the whole network (i.e. a graph) by exchanging topology messages periodically and builds the routing table by running a variation of the Dijkstra algorithm for the set V of known routers.

The time OLSR spends to calculate routes in WMNs can be dramatically increased because of the potential of growing of these networks as well as the presence of routers with limited computational resources. These constraints can lead to scalability problems in WMNs. In this work we investigate the Bounded Incremental Computation model (BIC) as an alternative to the OLSR approach for recalculating the routing table in WMNs based on links 802.11. In BIC model the running time of an algorithm is determined in terms of the set δ of routers affected by a changing on the graph.

We investigated important aspects of 802.11 channel modeling and carried out infinite time horizon simulations in order to analyze the WMN dynamism and to obtain empirical curves for the average running time of the Dijkstra algorithm (T_{Dij}) and the Ramalingam and Reps algorithm (T_{RRs}), which is based on BIC.

Results suggest that T_{Dij} dominates T_{RRs} asymptotically by a variable order of

magnitude, i.e. $T_{RRs} = o(T_{Dij})$. This is due to the assumption $|V| = |\delta|$ which is taken by OLSR for every recalculation. On the other hand our results show that $|\delta|$ is frequently much less than $|V|$ for all considered scenarios. Our results also indicate that the problem of routing table maintenance can satisfy scalability issues if the running time of the used algorithm satisfies $O(f(|\delta|))$.

KEYWORDS: *mesh* networks, incremental algorithms, routing

Sumário

Lista de Figuras

Lista de Tabelas

Lista de Algoritmos

Lista de Abreviaturas e Siglas

ANATEL:	*Agência Nacional de Telecomunicações*
AODV:	*Ad-Hoc On-Demand Distance Vector Routing*
AP:	*Access Point*
BATMAN:	*Better Approach To Mobile Adhoc Networks*
BER:	*Bit Error Rate*
BIC:	*Bounded Incremental Computation*
BFS:	*Breadth First Search*
BGP:	*Border Gateway Protocol*
dBm:	*decibels milliwatt*
DAG:	*Directed Acyclic Graph*
DF:	*Distribuição de Frequências*
DFR:	*Distribuição de Frequências Relativa*
DSSSP:	*Dynamic Single Source Shortest Paths Problem*
ETSI:	*European Telecommunications Standards Institute*
ETT	*Expected Transmission Time*
ETX:	*Expected Transmission Count*
FCC:	*Federal Communications Commission*
IC:	*Industry Canada*
IEEE:	*Institute of Electrical and Electronics Engineers*
IETF:	*Internet Engineering Task Force*
IP:	*Internet Protocol*
LD:	*Link Delay*
LOS:	*Line-Of-Sight*

MAC:	*Medium Access Control*
MANET:	*Mobile Ad hoc Network*
MKK:	*Radio Equipment Inspection and Certification Institute*
MPR:	*Multipoint Relay*
MTU:	*Maximum Transfer Unit*
NIC:	*Network Interface Card*
NLOS:	*Non-Line-Of-Sight*
NS-2:	*Network Simulator 2*
OLSR:	*Optimized Link State Routing*
OLSR2:	*Optimized Link State Routing version 2*
olsrd:	*Optimized Link State Routing Daemon*
OLSR-NG:	*Optimized Link State Routing New Generation*
PMP:	*Proactive MANET Protocol*
QoS:	*Quality of Service*
RFC:	*Request For Comments*
RMP:	*Reactive MANET Protocol*
RRs:	*Ramalingan & Reps*
SA :	*Sistemas Autônomos*
SNR:	*Signal to Noise Ratio*
SINR:	*Signal to Interference and Noise Ratio*
SSSP:	*Single Source Shortest Paths*
TTL:	*Time-To-Live*
UDP:	*User Datagram Protocol*
VoIP:	*Voz sobre IP*
WCETT:	*Weighted Cumulative Expected Transmission Time*
WLAN:	*Wireless Local Area Network*
WMN:	*Wireless Mesh Network*

Capítulo 1

Introdução

1.1 Contextualização

Redes em malha sem fio (*Wireless Mesh Network*, WMN) são redes de acesso caracterizadas pela utilização de enlaces de rádio para realizarem o roteamento de forma dinâmica. Por meio dessa característica chave, a instalação de uma WMN pode reduzir substancialmente a infra-estrutura de cabos em um projeto de rede sem fio. Outra vantagem inerente ao funcionamento das WMNs, é a possibilidade ampliá-las gradualmente a partir da adição de novos roteadores sem fio.

Um dos fatores que tem tornado factível o projeto de WMNs é o baixo custo com que tais redes podem ser implementadas. Devido a popularização dos dispositivos sem fio 802.11 padronizados pelo *Institute of Electrical and Electronic Engineers* (IEEE), as WMNs tem se tornado economicamente mais acessível do que outras tecnologias de acesso sem fio, como por exemplo *Wi-Max*.

As características supracitadas favorecem o rápido crescimento das WMNs. Essa possibilidade torna a questão da descoberta e manutenção das rotas ainda mais complexa. Além disso, devido a necessidade de se atender exigências de qualidade de serviço (*Quality of Service*, QoS) ou de se prover serviços personalizados para

diferentes classes de assinaturas de clientes, há interesse de que tais rotas sejam ótimas conforme um dado critério de qualidade.

Neste trabalho, nós abordamos a questão da manutenção da tabela de roteamento de estado de enlace em WMNs baseadas em enlaces 802.11. Neste contexto, o *Internet Engineering Task Force* (IETF) propôs o padrão RFC 3626 [38], que descreve o protocolo *Optimized Link State Routing* (OLSR). Em linhas gerais, o OLSR mantém uma visão global da rede (i.e. grafo) por intermédio da troca periódica de mensagens de topologia e, após cada atualização no grafo, uma variação do algoritmo de Dijkstra é utilizada para recalcular as rotas até todos os destinos conhecidos, representados pelo conjunto V.

Neste trabalho nós investigamos o uso do modelo de computação incremental limitada (*Bounded Incremental Computation*, BIC) como uma alternativa à proposta do OLSR para o recálculo da tabela de roteamento em WMNs 802.11. No modelo BIC, o tempo de execução de um algoritmo dá-se em termos do conjunto δ de roteadores cujas rotas ótimas foram afetadas por uma mudança no grafo, e não necessariamente em termos do conjunto V de roteadores.

1.2 Motivação

O OLSR é atualmente o único representante da filosofia de estado de enlace padronizado pelo grupo IETF *Mobile Ad Hoc Network* (IETF-MANET), que é especializado no roteamento em canais sem fio.

Em sua primeira versão, a idéia chave do OLSR consistia em diminuir a carga de mensagens de controle na rede utilizando o conceito de roteadores *Multi Point Relaying* (MPR). Um roteador MPR é um roteador autorizado a repassar adiante mensagens de controle. Assim, cada roteador OLSR elege dentre seus vizinhos aqueles que irão compor seu *conjunto MPR*. Um vizinho que não é MPR pode

processar para si as mensagens de controle que recebe, mas não pode passá-las adiante. A expectativa em torno dos nós MPRs é contribuir para que "*o OLSR seja escalável em redes grandes e densas*" [38].

Em [22], o grupo OLSR *New Generation* (OLSR-NG), composto por pesquisadores, desenvolvedores e administradores de WMNs responsáveis pela manutenção do OLSRD (*olsr daemon*[1] [94]), reportou uma série de recomendações para contornar problemas de escalabilidade do OLSR evidenciados em WMNs reais, conforme mencionamos a seguir.

O conceito de MPR foi verificado vulnerável em WMNs, uma vez que a falha de um enlace MPR dá margem a que vários roteadores apresentem uma visão inconsistente da rede. Por sua vez, isso pode ocasionar o roteamento dos dados dos usuários via enlaces inválidos. A fim aumentar a garantia de que cada roteador seja devidamente informado acerca das novidades topológicas, em [22] os autores sugeriram o uso da técnica de difusão tradicional, caso em que todos os vizinhos são nós MPRs.

Outra alternativa para assegurar uma visão atualizada da rede em cada roteador é aumentar a frequência de envio de mensagens de controle, conforme sugere o padrão do OLSR. Porém, somado à difusão tradicional, isso comprometeria a escalabilidade da rede por meio de seu rápido congestionamento. No sentido de contornar esse problema foi sugerido o algoritmo *fish-eye* [80]. Sua idéia básica consiste em utilizar diferentes valores de *Time-to-Live* (TTL) para que cada roteador envie informações de topologia com maior frequência aos seus vizinhos e com menor para destinos remotos da rede. Conforme demonstrado pelos autores do algoritmo, esse mecanismo aumenta a confiabilidade das rotas divulgadas ao mesmo tempo que diminui a sobrecarga da rede.

O principal problema de escalabilidade apontado pelo grupo OLSR-NG diz respeito ao (re)cálculo das rotas ótimas. Devido o potencial de crescimento das WMNs e

[1] uma das implementações do OLSR mais usadas do mundo, por exemplo em [4; 6; 78]

a presença de pontos de acesso 802.11 com recursos computacionais escassos (transformado em roteadores em malha pela mudança do *firmware*), verificou-se que o cálculo das rotas pode demorar vários segundos [5], tempo suficiente para a vazão da rede ser prejudicada devido o descarte dos pacotes de entrada. Diante disso, o grupo OLSR-NG anunciou o fim de seu interesse pelo OLSR e voltou suas atenções ao projeto de um novo protocolo independente chamado *Better Approach To Mobile Adhoc Networks* (BATMAN) [22; 5].

1.3 Trabalhos relacionados

A questão do roteamento de estado de enlace em canais sem fio tem sido amplamente discutida na literatura especializada (por exemplo [26; 84]). Em nossas pesquisas, porém, não encontramos nenhum trabalho que aplicasse e avaliasse a abordagem BIC para resolver o problema da manutenção das rotas em WMNs.

O modelo BIC foi proposto por Ramalingam e Reps em [86]. Em [87], os mesmos autores apresentaram um algoritmo no modelo BIC (conhecido por seus nomes) para resolver o problema das gramáticas dinâmicas. Tal problema pode ser visto como uma generalização do problema dos caminhos mínimos dinâmicos de origem única que, por sua vez, pode ser usado para formalizar o problema da manutenção de rotas em WMNs, conforme detalhamos no capítulo 3.

Em nosso trabalho a metodologia de avaliação de desempenho também cumpre um papel extremamente importante, uma vez que interessa-nos estimar a relação assintótica de caso médio entre os algoritmos de Ramalingam e Reps e de Dijkstra no contexto das WMNs.

Enquanto que a complexidade de pior caso dos algoritmos de Dijkstra e Ramalingam e Reps são analiticamente conhecidas (vide [43; 87], respectivamente) até onde investigamos, nada é conhecido a respeito do caso médio de tais algoritmos no

contexto das WMNs. Em [50], Frigioni *et al* propõem uma análise de desempenho médio que contempla ambos os algoritmos, contudo os autores não investigam a relação assintótica entre eles.

Com base em trabalhos prévios [85; 75; 71], verificamos a alta relevância da modelagem do canal sobre as camadas superiores de roteamento e aplicação. Desta forma, o propósito de nosso trabalho requer modelos estocásticos muito mais elaborados para os canais 802.11 do que a distribuição uniforme utilizada por Frigioni *et al* em [50] para valorar os grafos de entrada. Em nosso trabalho, além de provermos uma discussão detalhada sobre distribuição de probabilidades responsável pela valoração dos grafos de entrada, nós recorremos à metodologia de simulação de horizonte infinito, tal como discutida por Mota em [72], como forma de realizar uma aproximação empírica estatisticamente confiável para as curvas médias dos algoritmos avaliados.

1.4 Organização do trabalho

O restante deste trabalho está estruturado como segue-se. No capítulo 2 apresentamos aspectos relevantes para as questões de roteamento e de modelagem de canais 802.11. Ainda no mesmo capítulo fornecemos uma introdução sobre as filosofias adotadas pelo grupo IETF-MANET para roteamento em redes sem fio de múltiplos saltos.

No capítulo 3 oferecemos mais detalhes sobre a operação do protocolo de roteamento OLSR e enunciamos o *problema do caminho mínimo dinâmico de origem única*, o qual pode ser usado para formalizar a questão do roteamento em WMNs. Apresentamos a abordagem do OLSR para uso do algoritmo de Dijkstra e apresentamos uma abordagem alternativa baseada no modelo BIC, em particular através do uso do algoritmo de Ramalingam e Reps.

No capítulo 4 fornecemos todos os detalhes para realizar uma avaliação comparativa tanto entre os algoritmos como entre as abordagens a eles subjacentes. Esses detalhes contemplam a metodologia de simulação para avaliação de desempenho, discussão sobre a modelagem de canais 802.11 e a configuração geral das simulações.

No capítulo 5 apresentamos e analisamos os resultados. Por fim, no capítulo 6 apresentamos as conclusões e os trabalhos futuros.

Capítulo 2

Fundamentação teórica

A possibilidade de conferir *mobilidade* aos usuários das tradicionais redes IP (Internet, redes corporativas, domésticas, etc) tem resultado na grande disseminação de pontos de acesso (AP, do inglês *access points*) baseados nas padronizações da série IEEE 802.11 para acesso sem fio.

Os padrões com prefixo 802.11 de tecnologias de acesso sem fio definem as camadas física (PHY) e de acesso (aleatório) ao meio (MAC - *Medium Access Control*) para as chamadas *redes locais sem fio*, popularmente conhecidas também como *Wireless Local Area Network* (WLAN), ou ainda *Wireless-Fidelity* (Wi-Fi). Tal tecnologia tem sido amplamente adotada pela indústria mundial e, como resultado disso, milhares de equipamentos de uso pessoal baseados em 802.11 (por exemplo *laptops*, PDAs, *smart-phones*) tem sido comercializado a um custo relativamente baixo, em particular nas versões *b* e *g*.

Tradicionalmente, a função básica de um AP 802.11 é servir de *interface* entre o usuário e uma infra-estrutura de rede cabeada. Os APs são conectados a uma rede local a fim de estender os serviços desta por meio de tecnologia sem fio, conforme ilustrado na Figura 2.1a (adaptada de [21]).

A necessidade de ampliar a cobertura sem fio em locais como condomínios (ver-

ticais ou horizontais), grandes aeroportos, hotéis, *campus* de universidades, escolas, cidades, dentre outros, pode conduzir a uma complexa infra-estrutura de rede cabeada. Além disso, a necessidade de expandir a rede pela adição de novos roteadores também pode requerer um novo projeto para a infra-estrutura de rede cabeada.

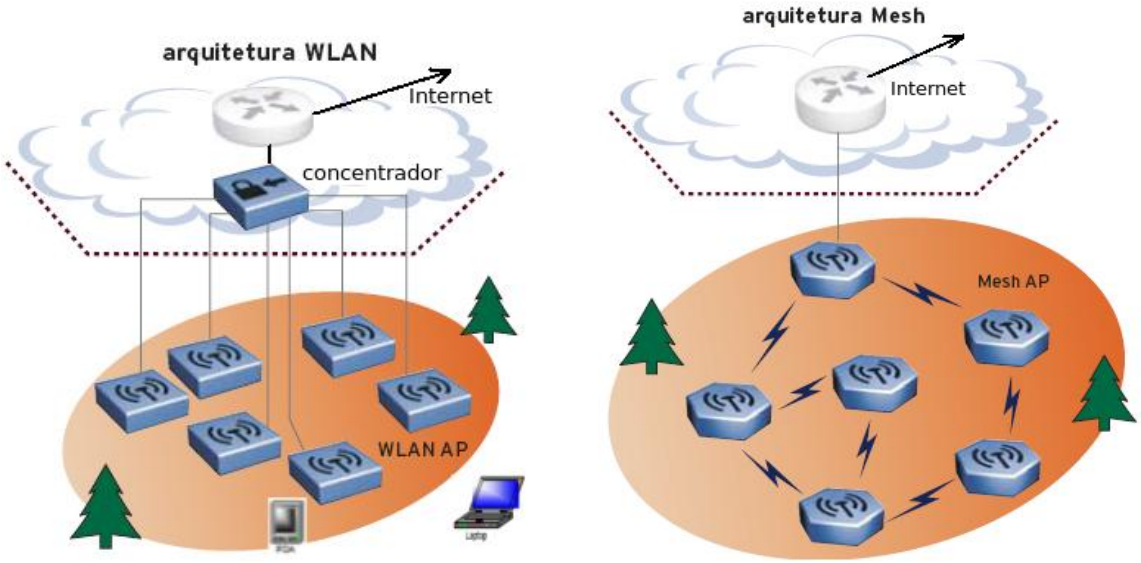

Figura 2.1: WLAN (a) × Redes *Mesh* sem fio (b)

2.1 Redes em malha sem fio

Em face às limitações das WLANs anteriormente expostas, novas propostas conferem aos APs 802.11 a *capacidade de roteamento* (*roteadores mesh* ou *AP Mesh*). Desta forma, além de poder prover acesso aos usuários, os APs também são capazes de se auto-configurar e realizar o roteamento dinamicamente sobre canais sem fio, formando assim uma *rede sem fio de múltiplos saltos* denominada *Wireless Mesh Network* (WMN) ou *backbone em malha sem fio*, conforme mostra a Figura 2.1b (adaptada de [21]).

Em um *backbone* em malha geralmente os roteadores não são móveis mas, caso seja possível, a rede deve ser capaz de continuar provendo o serviço para a qual foi

construída em caso de eventuais mudanças de posicionamento ou falhas de um ou mais roteadores. Quanto à questão de alimentação de energia, em certas ocasiões excepcionais é necessário recorrer a meios alternativos baseados em baterias devido a falta de rede elétrica.

Nas redes em malha sem fio, o local da implementação do roteamento com respeito ao modelo OSI também pode ser feito no nível de enlace, em oposição às implementações no nível de rede. Tal abordagem é uma tecnologia emergente, cujo padrão está sendo desenvolvido pelo grupo de trabalho 802.11s do IEEE [58]. Apesar das diferentes abordagens, um mesmo protocolo de roteamento pode ser implementado na camada de rede ou na de enlace sem que os algoritmos que refletem sua filosofia sejam descaracterizados.

Além da vantagem de possibilitar a simplificação da infra-estrutura de cabos em muitos projetos de rede sem fio (conferir figura 1.1) o uso de canais sem fio para realizar o roteamento também permite flexibilizar o aumento da rede pela adição gradual de novos roteadores sem fio. Conforme ilustrado na figura, também é possível a existência de um *gateway* ou ponte com a Internet ou outras tecnologias de redes (por exemplo IEEE 802.16). Esses motivos têm contribuído para que as redes em malha substituam as tradicionais WLANs em contextos onde há a necessidade imediata ou a possibilidade futura de expansão do raio de alcance do sinal. Assim a tecnologia de redes em malha tem atraído interesses diversos em todo o mundo. Alguns exemplos de projetos reais são citados na tabela 2.1.

Identificação/Ref.	Local	Propósito	Total de APs
Athens wireless network [4]	Grécia	Acesso livre	1019
Berlin FreiFunk.net [6]	Alemanha	Acesso livre	∼600
RoofNet-MIT [18]	EUA	Livre/Pesquisa	∼20
ReMesh [78]	Brasil	Livre/Pesquisa	∼17
WiFly [83]	Taiwan	Comercial	>10000

Tabela 2.1: Exemplos de projetos de redes em malha sem fio ao redor do mundo

Outro fator que tem impulsionado a popularização das redes em malha sem fio é a possibilidade de se acessar a tecnologia a um custo relativamente baixo. Na Internet é possível encontrar comunidades de desenvolvedores de software livre que disponibilizam *firmwares* baseados no sistema operacional Linux para vários modelos de pontos de acesso 802.11a/b/g/n. Uma vez que tais *firmwares* implementam algum tipo de solução em malha, APs tradicionais podem tornar-se roteadores em malha sem fio através do uso de um desses *firmwares*. Alguns exemplos são: Open WRT [17], dd-wrt [8], freifunk [10] e *RoofNet WMN solution* [18]. Adicionalmente também é possível encontrar manuais gratuitos (elaborados a partir de experiência de desenvolvedores e administradores de WMNs) de como projetar e instalar redes em malha sem fio a um custo relativamente baixo, para mais detalhes conferir [24].

2.2 Agravantes de roteamento em canais sem fio 802.11

O crescente sucesso da tecnologia de redes em malha sem fio deve-se, em grande parte, aos princípios que regem seu funcionamento. A auto-configuração (*self-configuration*) permite que a rede se configure automaticamente à medida que novos roteadores são adicionados à rede. A auto-cura (*self-healing*) capacita a rede a tentar contornar problemas diversos, tais como: queda de sinal, falhas de equipamentos em nível de *hardware* ou *software*, desastres em parte da rede, etc.

Por outro lado, o custo para lidar com canais sem fio também representa desafios para o projeto das WMNs, especialmente no quesito de roteamento, onde se incide boa parte da complexidade para se assegurar características como auto-configuração e auto-cura. Neste aspecto, a necessidade de atentarmos à certas peculiaridades dos canais sem fio decorre do fato da atividade de roteamento resumir-se na construção e

manutenção de rotas. Por sua vez, tais atividades podem ser agravadas pela natureza dos enlaces disponíveis. A seguir listamos algumas características dos canais 802.11 que são relevantes para a atividade de roteamento.

2.2.1 Interferência em redes 802.11

O padrão 802.11 define sua operação sobre a faixa de frequência de 2.4 Ghz da banda *Industrial, Scientific and Medical* (ISM) para as versões *b* [56] e *g* [57] e sobre a faixa 5Ghz da banda *Unlicensed National Information Infrastructure* (U-NII) para a versão *a* [55]. A quantidade de canais diferentes nestas faixas, bem como a quantidade de canais ortogonais, variam de acordo com a legislação de cada país. A tabela 2.2 apresenta uma síntese sobre a alocação das faixas de frequência 802.11 conforme a regulamentação aplicada em algumas regiões do mundo.

Regulamentador	Faixa (Ghz)	Total de canais
FCC (EUA), IC (Canadá)	$2,401 - 2,4835$	11
	$5,15 - 5,35$	8
ANATEL (Brasil)	$2,400 - 2,4835$	11
	$5,15 - 5,35$	8
	$5,47 - 5,725$	11
	$5,725 - 5,850$	5
ETSI 1 (Europa)	$2,401 - 2,4835$	13
	$5,15 - 5,35; 5,47 - 5,725$	19
MKK (Japão)	$2,473 - 2,495$	14
	$4,9 - 5; 5,15 - 5,25$	8

Tabela 2.2: Uso das banda ISM e U-NII pelas versões b/g e a do padrão IEEE 802.11 em diferentes países

A faixa de frequência 2.4Ghz é não licenciada, o que isenta seus usuários de solicitar uma licença frente ao órgão regulamentador do país, restando somente

obrigações legais quanto à potência de transmissão e largura de banda de operação no *cartão de interface de rede* (*Network Interface Card*, NIC. Doravante, essa sigla se referirá às interfaces de rede sem fio). No caso do Brasil mais informações neste sentido podem ser encontradas na resolução 397, de 06/04/2005 da Agência Nacional de Telecomunicações (ANATEL) [29].

Um dos problemas mais notáveis decorrentes da escassez de canais ortogonais e da utilização de uma faixa não licenciada é a *interferência*. De uma forma simples, a interferência pode ser descrita como a captação simultânea de sinais em canais não ortogonais por um mesmo NIC de rede sem fio, de sorte que *pelo menos um* dos sinais era endereçado ao receptor. A interferência pode ser classificada de acordo com a fonte geradora dos sinais nela envolvidos, conforme detalharemos nas próximas subseções.

2.2.1.1 Interferência causada por outros equipamentos

Dado que a faixa de frequência 2.4Ghz é definida sobre a banda ISM, ela pode sofrer interferência de outras equipamentos operando sob padrão diferente do 802.11. Alguns exemplos de equipamentos que utilizam a faixa 2.4Ghz são: telefones sem fio, rádios comunicadores, dispositivos *bluetooth* (padrão IEEE 802.15), etc. Além disso ela pode sofrer ruídos de frequências emitidas por dispositivos eletro-eletrônicos (por exemplo forno microondas).

2.2.1.2 Interferência causada por outros sistemas 802.11

Este tipo de interferência ocorre quando os sinais nela envolvidos são oriundos de outros redes 802.11 como por exemplo, pontos de acesso 802.11 tradicionais, ou até de outras redes em malha sob entidades administrativas diferentes. Neste último caso, se os devidos cuidados de instalação e configuração inicial não forem tomados, a interferência pode evoluir do nível físico ao lógico, prejudicando ainda mais o

desempenho do roteamento. Uma vez que redes em malha seguem o princípio da auto-configuração, caso duas redes diferentes implementem a mesma pilha de solução *mesh*, basta que pelo menos um roteador de cada rede esteja no raio de comunicação um do outro para que a camada de roteamento "entenda" a rede como uma só.

Algumas linhas de pesquisa propõem o uso de esquemas de negociação automática de canais entre redes diferentes para evitar a interferência em nível físico (por exemplo [96]). Os ganhos auferidos por tais soluções são limitados pela quantidade de canais ortogonais disponíveis e pelo grau de interferência entre os canais restantes. Esse grau de interferência é determinada pela proximidade das frequências de cada canal. Conforme ilustrado na figura 2.2 (adaptada de [48]) podemos verificar que a largura de banda dos canais é de 22MHz, o espaçamento entre as frequências centrais dos canais adjacentes é de 5MHz (exceto para a distância entre os canais 13 e 14, que é de 12MHz) e a quantidade *teórica* de canais sem sobreposição é de, no máximo, três (por exemplo 1, 6 e 11).

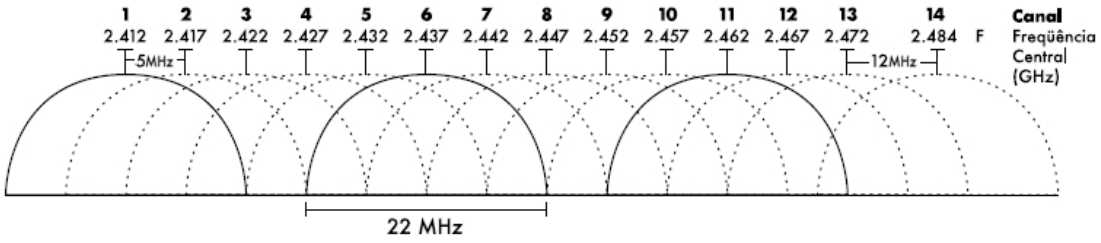

Figura 2.2: Alocações de canais da banda ISM para o padrão IEEE 802.11b/g

O padrão IEEE 802.11 não define mecanismos de negociação automática de canal. Assim a adoção de uma tecnologia de negociação automática de canais entre redes diferentes requer prévia negociação entre seus respectivos administradores a fim de se adotar uma tecnologia em comum. Na prática, esse tipo de solução é utilizada apenas em grandes redes operadas por uma mesma entidade administrativa a fim de amenizar a interferência causada entre seus roteadores (*auto-interferência*) (por exemplo: solução proprietária da empresa Nortel [21]). O fenômeno da auto-

interferência é explanado com mais detalhes na próxima subseção.

Outra estratégia comumente adotada por administradores de rede para amenizar os efeitos da interferência em nível físico ocasionadas por redes 802.11 vizinhas é utilizar ferramentas de monitoração de canal tais como *Network Stumbler* [14] ou *Wi-Spy*[2]. Por meios destes tipo de softwares é possível monitorar a ocupação dos espectro e assim auxiliar na escolha do canal operação. A figura 2.3 (fonte: [93]) ilustra a ocupação dos canais 6 e 11 obtida pelo dispositivo *Wi-Spy* em conjunto com o software *Chanalyser* [7].

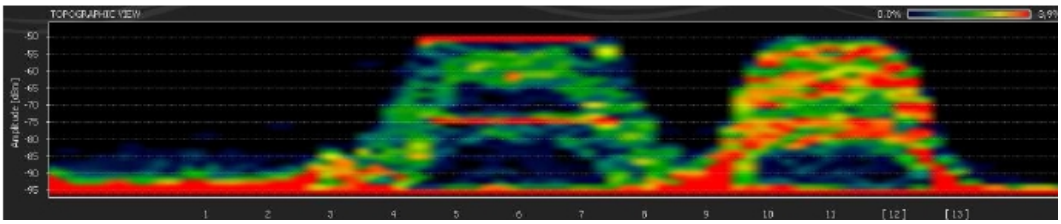

Figura 2.3: Exemplo de ocupação do espectro eletromagnético dos canais 6 e 11 do padrão 802.11b/g

A desvantagem do procedimento acima descrito é que ele deve ser feito regularmente, uma vez que a configuração de ocupação do espectro pode mudar radicalmente.

2.2.1.3 Auto interferência: interferência causada pelo próprio sistema

Esse tipo de interferência ocorre quando um NIC de um dado roteador capta, além do sinal a ela destinado, outros sinais originados em roteadores mesh pertencentes à mesma WMN. Essas portadoras detectadas correspondem ao processo de envio das unidades básicas de transmissão da camada física os quais são denominados *quadros* (ou *frames*).

A *auto-interferência* pode tanto ocorrer devido comunicações paralelas, onde nenhum dos quadros captados apresentam o mesmo endereço de destino (à exceção do quadro endereçado ao NIC), como pode também ser resultado da tentativa de

se enviar vários quadros simultaneamente para um destino em comum. Caso a interferência seja *muito intensa*[1] haverá colisão na interface de recepção, resultando em perda de todos os quadros recebidos. É importante notar que nem sempre os sinais detectados em uma dada interface são passíveis de serem corretamente decodificados em quadros. Nessas condições também é possível ocorrer interferência pois, apesar do receptor não situar-se na *zona de transmissão* ou de *comunicação* da estação transmissora, ele situa-se em sua *zona de interferência*. Esses conceitos explanaremos a seguir.

Podemos definir a zona de transmissão de uma estação A como sendo a região do espaço fora da qual uma outra estação jamais conseguirá detectar a portadora e decodificar com sucesso um quadro emitido por A. Na prática, os limites de tal região podem sofrer frequentes oscilações pois dependem de vários fatores como distância, obstruções, tipo de modulação do quadro, etc. Por sua vez, a zona de interferência de uma estação A é definida como a região do espaço fora da qual uma outra estação jamais sofrerá interferência de A (i.e. a portadora sequer será detectada). Assim a zona de transmissão faz parte da zona de interferência mas o contrário não necessariamente é verdadeiro.

No intuito de evitar colisões, a camada MAC do protocolo 802.11 utiliza o mecanismo CSMA/CA (*Carrier Sense Multiple Access with Collision Avoidance*) [55]. O CSMA/CA permite às estações monitorar o canal antes de tentar um transmissão. Desta forma, se o canal estiver ocioso por um espaço de tempo maior ou igual ao *espaçamento inter-quadros distribuído* (*Distributed Inter Frame Space*, DIFS) a estação põe um quadro de dados (`DATA`) no canal, do contrário ela determina um tempo de espera para enviar o quadro. A transmissão será considerada com sucesso se um quadro `ACK` de resposta for recebido, do contrário um novo tempo de espera é calculado para se reenviar o quadro `DATA`.

[1] a formalização do evento de colisão é uma questão de modelagem, conferir seção 2.3.2

Contudo há algumas situações em que o mecanismo de detecção de portadora do CSMA/CA não funciona de maneira satisfatória. A seguir, descrevemos duas situações clássicas que exemplificam tal limitação.

2.2.2 O problema do terminal escondido

O problema do terminal escondido decorre de uma limitação no mecanismo de detecção de ocupação do canal. Ele ocorre quando duas estações *transmissoras* (A e C, por exemplo) estão fora da zona de interferência uma da outra ao mesmo tempo que ambas estão na zona de transmissão de uma terceira (B, por exemplo). Neste caso diz-se que A e C estão "escondidas" e a comunicação de ambas com B pode sofrer interferência.

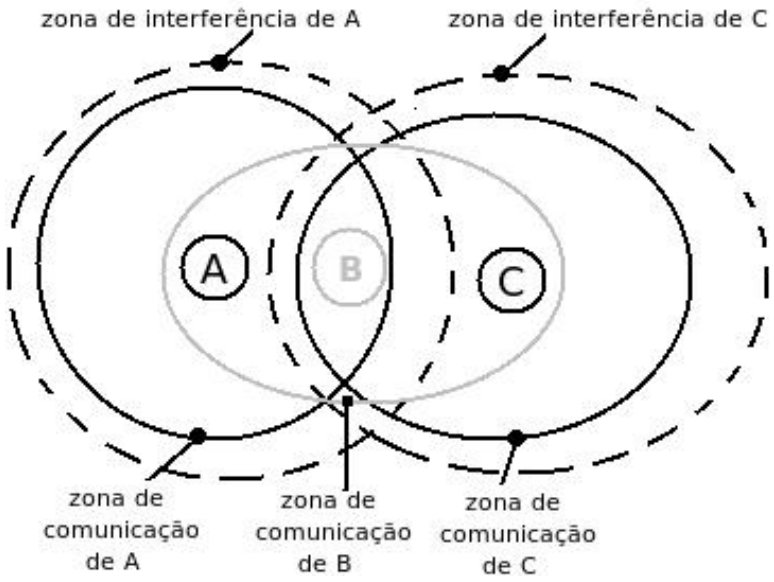

Figura 2.4: Ilustração do problema do terminal escondido

Conforme ilustrado na figura 2.4 o terminal A não consegue detectar a portadora emitida por C durante a comunicação deste terminal com o terminal B. Assim diz-se que C está "escondido" de A. Esta situação pode levar A a transmitir para B e, consequentemente, resultar em colisão.

Uma forma de amenizar consideravelmente o problema do terminal escondido consiste em utilizar um mecanismo de solicitação e reserva de canal conhecido como RTS/CTS (*Request to Send/Clear to Send*). Porém alguns autores ressaltam que o *overhead* introduzido por esta solução pode afetar outros importantes parâmetros de desempenho da rede, como por exemplo a vazão [35].

2.2.3 O problema do terminal exposto

O problema do terminal exposto não está relacionado com a questão da colisão, mas sim com a queda de desempenho da camada MAC no processo de transmissão. Essa queda de desempenho, de alguma forma, reflete-se negativamente nos usuários da camada MAC localizados em camadas superiores.

De forma ilustrada, podemos dizer que o problema do terminal exposto ocorre quando uma estação B é impedida de transmitir para uma estação A por situar-se na zona de interferência de uma estação C que, por sua vez, está enviando (ou tentando enviar) quadros para uma estação D, conforme mostrado na figura 2.5.

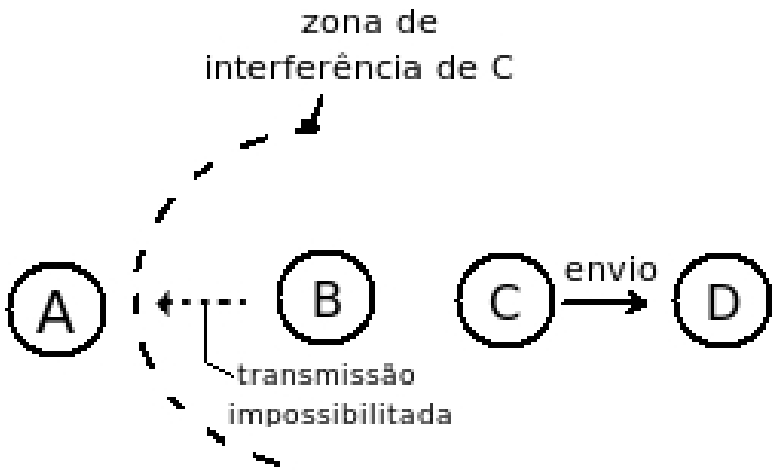

Figura 2.5: Ilustração do problema do terminal exposto

2.2.4 Assimetria de enlaces sem fio

Enlaces sem fio podem sofrer o fenômeno da *assimetria*. Quando isso ocorre somente uma das estações consegue "perceber" a outra. Em outras palavras, somente uma estação encontra-se na zona de transmissão da outra o que impossibilita estabelecer uma comunicação bidirecional.

A assimetria pode ser explicada principalmente pelo fato do conjunto de elementos atenuadores da potência de transmissão de cada estação serem diferentes [91]. Desta forma, mesmo quando os enlaces apresentam simetria há uma possibilidade muito grande de que força do sinal percebida varie com o sentido do enlace.

Em [65], os autores conduziram um experimento em uma WMN onde foi constatado, aproximadamente, 29% de enlaces assimétricos enquanto que cerca de 70% dos enlaces simétricos apresentaram variações da força do sinal dependendo do sentido da comunicação. Em outro estudo reportado por [53], os autores constataram em um experimento em redes móveis sem fio (*Mobile Ad hoc Networks*, MANETs) a presença de 5% a 15% de enlaces assimétricos.

2.2.5 Variabilidade da potência de recepção

A transmissão via canais sem fio se dá por meio de *símbolos*. O símbolo é a unidade básica de transmissão do nível físico e codifica um conjunto de *bits* pelo processo de modulação. A modulação dos *bits* em símbolos ocorre no NIC do lado transmissor do enlace ao passo que a demodulação (possivelmente) ocorre no lado receptor. Um importante fator de desempenho associado aos canais sem fio é a taxa de erros de *bit* (*bit erro rate*, BER), definida como a razão entre o total de *bits* enviados e o total de *bits* corretamente demodulados.

O valor de BER de um canal sem fio é extremamente correlacionado com a *força do sinal* subjacente ao símbolo no instante de sua recepção. O valor da força

de recepção do sinal, por sua vez, é resultado da degradação sofrida pela potência de transmissão do emissor ao longo do caminho de propagação. A seguir serão apresentados alguns dentre os fatores que diminuem a força do sinal.

2.2.5.1 Fatores de atenuação do sinal

A *distância* entre as estações comunicantes é um fator elementar no desvanecimento do sinal transmitido. Essas duas grandezas apresentam uma relação diretamente proporcional. Além da distância, outros fenômenos também *podem* causar atenuação do sinal. A seguir apresentamos brevemente os conceito de reflexão, difração e espalhamento.

- *Reflexão* - Esse fenômeno decorre da incidência das ondas eletromagnéticas sobre uma superfície *refletora* de dimensão maior do que o comprimento da onda irradiada (por exemplo metais, superfície de águas, paredes). A incidência resulta na dissipação de parte da potência do sinal sobre a superfície, e em um sinal refletido com potência menor, formando assim dois ângulos iguais: o ângulo de incidência e o ângulo de refração, conforme ilustrado na figura 2.6 (adaptada de [48]).

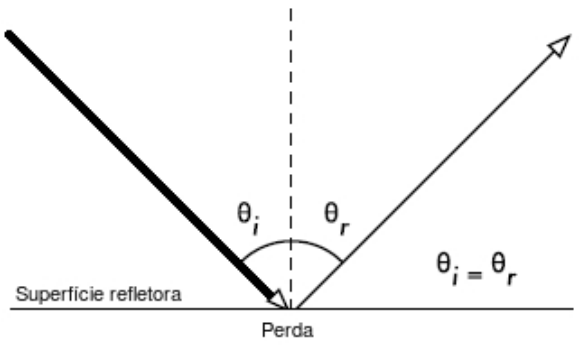

Figura 2.6: Degradação de potência devido reflexão das ondas eletromagnéticas

Em alguns sistemas sem fio a reflexão também pode acarretar no fenômeno do *desvanecimento por múltiplos percursos* (*multipath fading*), no qual diferentes

cópias do mesmo sinal são propagados por diferentes caminhos e alcançam o destino em instantes diferentes.

- *Difração* - Ocorre quando o sinal sofre uma "curva" devido a existência de objetos com bordas no percurso de propagação do sinal (por exemplo: prédios), conforme ilustrado na figura 2.7. Neste caso quanto mais acentuada for a curva, mais fraco será o sinal difratado.

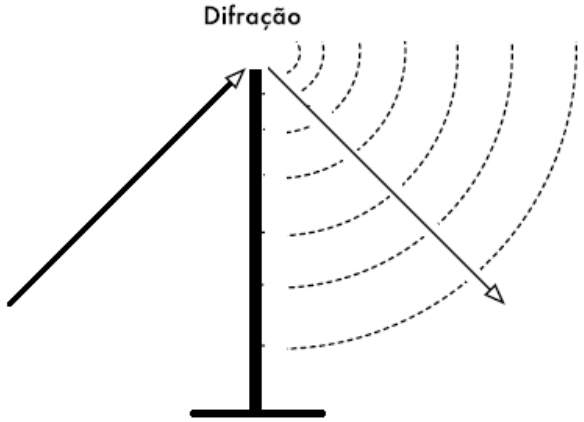

Figura 2.7: Exemplo de difração de onda eletromagnética

- Dispersão ou espalhamento - Resulta da incidência das ondas eletromagnéticas sobre objetos de dimensões menores ou iguais ao comprimento da onda irradiada (por exemplo: mobília de escritório), conforme ilustrado na figura 2.8.

Figura 2.8: Exemplo de espalhamento de onda eletromagnética

Para uma estudo mais aprofundado sobre o impacto destes e outros fenômenos em canais sem fio, consultar [88].

No intuito de possibilitar o uso das redes 802.11 em uma maior diversidade de cenários com variados graus de obstruções, os projetistas do referido padrão disponibilizaram quatro diferentes taxas de transmissão para a versão *b* e oito para as versões *a* e *g*.

Cada uma das taxas 802.11 representa um compromisso entre a vazão em nível de enlace e a condições mínimas de qualidade do sinal para assegurar um BER aceitável. Quanto maior o total de *bits* por símbolo, maior a qualidade de sinal exigida pelo *esquema de modulação* para decodificar os símbolos em níveis aceitáveis de BER. Os seguintes esquemas de modulação são utilizados no padrão 802.11b: BPSK (*Binary Phase Shift Keying*) para a taxa de 1 Mbps, QPSK (*Quaternary Phase Shift Keying*) para a taxa de 2 Mbps e CCK (*Complementary Code Keying*) para as taxas de 5.5 e 11 Mbps. Já as versões *a* e *g* utilizam BPSK para as taxas de 6 e 9 Mbps, QPSK para 12 e 18 Mbps, 16QAM (*Quadrature Amplitude Modulation*) para as taxas de 24 e 36 Mbps e 64QAM para as taxas de 48 e 54 Mbps.

O padrão 802.11 normatiza uma exigência máxima de BER independentemente do esquema de modulação em operação, a saber, BER$\leq 10^{-5}$ (i.e. a máxima taxa de *bits* incorretamente demodulados tolerável). Para atender tal exigência cada esquema impõe suas condições mínimas de operação, conforme mencionamos previamente. Essa condição é conhecida como *sensibilidade de recepção* (*receiver sensitivity*, em dBm) que consiste na força de sinal mínima necessária (medida no conector da antena de recepção) para que um NIC 802.11 consiga detectar a portadora e demodular os símbolos a um BER tolerável pelo padrão.

Para cada taxa de transmissão há uma sensibilidade de recepção diferente. Quanto mais complexo o símbolo (i.e. maior a taxa), maior o *ruído de operação* (*Thermal noise*) inerente ao processo de demodulação e, consequentemente, maior a exigência do esquema. A tabela 2.3 mostra um exemplo de sensibilidades de recepção típicas de NICs 802.11a/b/g para cada taxa de transmissão correspondente (respectivamente

extraídos de [67; 59; 98]).

Taxa (Mbps)	Versão do 802.11	Sens.(dBm)
54 (64QAM)	a	−65
	g	−68
48 (64QAM)	a	−66
	g	−69
36 (16QAM)	a	−70
	g	−73
24 (16QAM)	a	−74
	g	−77
18 (BPSK)	a	−77
	g	−80
12 (QPSK)	a	−79
	g	−82
11 (CCK)	b	−82
9 (BPSK)	a	−81
	g	−84
6 (BPSK)	a	−82
	g	−85
5.5 (CCK)	b	−87
2 (QPSK)	b	−91
1 (BPSK)	b	−94

Tabela 2.3: Sensibilidades de recepção típicas para as taxas de transmissão 802.11a/b/g

A configuração da taxa transmissão adequada de um canal sem fio deve ser feita de maneira coerente com as condições da rede. Altas taxas de transmissão em ambientes demasiadamente afetados por fenômenos de degradação podem comprometer a vazão do enlace de rádio devido a perda de símbolos densos (i.e. com muitos *bits*). Por outro lado, a transmissão a baixas taxas em ambientes com pouco des-

vanecimento também compromete a vazão devido a subutilização do canal. Uma vez que o canal pode experimentar essas diferentes condições ao longo do tempo, o uso de um mecanismo de seleção automática de taxas de transmissão é altamente recomendado.

O padrão IEEE 802.11 não estabelece algum mecanismo de seleção automática, contudo disponibilizou diferentes taxas como uma forma de amparar tais soluções, assim tais mecanismos foram rapidamente difundidos pela indústria. Para um estudo sobre tais mecanismos no contexto de WMNs consultar [62].

2.2.6 Impacto da tecnologia da antena

Em adição aos NICs, *antenas* são utilizadas para aumentar o ganho do sinal de rádio. A escolha da tecnologia de antenas (i.e. da *forma de irradiação*) depende do fim a ela destinado. No caso de WMNs a forma de irradiação que melhor atende às características de expansão não planejada da rede é aquela proporcionada pelas *antenas omni-direcionais*. Nesta modalidade o ângulo de irradiação é de 360°, o que significa que o sinal é propagado para todas as direções do espaço.

Todavia, devido a uma limitação de natureza física das antenas omnis, o ganho proporcionado à potência dos NICs não é homogêneo em todos os ângulos, fazendo com que algumas regiões apresentem lóbulos de irradiação bem menores do que outros. A figura 2.9 ilustra o *diagrama de irradiação* típico de antenas omni-direcionais por meio de cartas polares. Nela podemos verificar que apesar do lóbulo de irradiação horizontal apresentar uma irradiação de 360° quase uniforme, o lóbulo vertical, por sua vez, irradia melhor nas proximidades dos ângulos 0° e 180° (i.e. nos sentidos inferior e superior da antena), enquanto que os demais ângulos apresentam ganhos de irradiação bem menores.

O posicionamento de um dado roteador nas imediações do lóbulo de irradiação

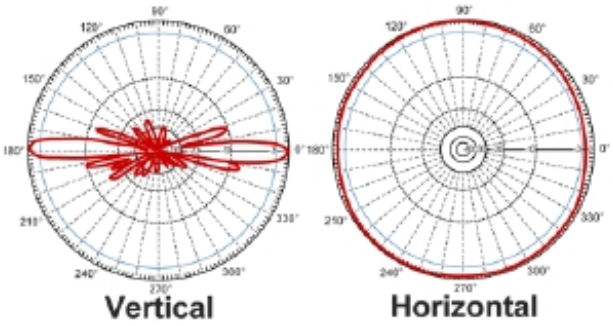

Figura 2.9: Lóbulos de irradiação típicos de antenas omni-direcionais

diagonal superior de um outro pode levar ao surgimento de enlaces mais oscilantes e, consequentemente, afetar o desempenho do roteamento. Esse agravante foi verificado na prática no *backbone* externo do projeto piloto da rede ReMesh, coordenado pela Universidade Federal Fluminense, conforme reportado em [41].

2.2.7 Métricas dinâmicas de roteamento

O primeiro critério de otimização de roteamento empregado nos primórdios dos sistemas autônomos (SA) da Internet foi o de minimizar o número de roteadores intermediários entre o remetente e o destinatário. Entre os sistemas de roteamento sem fio, esse critério, denominado *minimum hop count* ou simplesmente *hop count*, foi inicialmente utilizado por alguns protocolos de roteamento em MANETs. Isso sucedeu-se tanto pela influência das soluções amadurecidas no âmbito da Internet como pela simplicidade de cálculo e manutenção das rotas propiciadas por tal métrica. Outro provável motivo seria o fator de vantagem associado ao uso de MANETs. Dado que tais redes tipicamente operam sob condições críticas (alta mobilidade, limitações de energia, etc), o simples provimento de uma rota de comunicação sem levar em consideração aspectos de QoS, de certa forma, já pode ser considerado satisfatória em muitos contextos.

O critério do menor número de saltos também foi inicialmente adotado em

WMNs, contudo logo percebeu-se que o perfil diferenciado dessas redes exigia métricas mais avançadas. Neste sentido surgiram as métricas dinâmicas de roteamento que são propostas de abstração da qualidade dos enlaces sem fio em WMNs (*quality-aware*). A seguir discorremos sobre alguns dos principais motivos que favorecerem a adoção desta classe de métricas em detrimento do critério *hop count*.

2.2.7.1 Roteadores estáticos

Diferente das MANETs, na grande maioria dos casos os roteadores dos *backbones* em malha sem fio não são móveis e não apresentam restrições de energia. Assim, os fatores que em MANETs tornavam proibitivo uma complexidade adicional à tarefa de roteamento podem ser considerados irrelevantes em WMNs [64].

2.2.7.2 Necessidade de ponderação granularizada

A métrica de menor número de saltos, basicamente, classifica os enlaces como existentes ou não. Desta forma em uma rede baseada na métrica *hop count* todos os enlaces existentes são ponderados de maneira igual.

Na prática, a quantidade de esforço empreendido pela rede pode variar bastante para detectar diferentes enlaces. Isso porque o processo de formação de um enlace na camada de roteamento basicamente decorre da resposta que um roteador recebe às *mensagens de saudação* previamente por ele enviadas. E como as condições de propagação, recepção e decodificação do sinal podem variar consideravelmente entre os roteadores vizinhos, a camada MAC remetente poderá ter que enviar diferentes quantidades de quadros contendo a mesma mensagem de saudação até que um `ACK` ou uma resposta correspondente seja recebida.

A figura 2.10 ilustra uma situação onde a métrica *hop count* pode subutilizar o desempenho do serviço de roteamento. Nela o roteador A registra que a *melhor* forma de se alcançar o roteador C é por meio do enlace direto $A - C$, uma vez

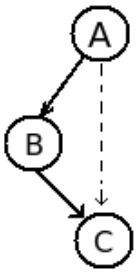

Figura 2.10: Exemplo de limitação da métrica *hop-count*

que ela representa o menor número de saltos. O fato do enlace apresentar sinal fraco é ignorado pela métrica, o que pode aumentar a taxa de perdas do serviço. Uma alternativa que poderia contornar esse problema, seria utilizar a rota A-B-C que, apesar de apresentar um número de saltos maior, apresenta enlaces com maior qualidade de sinal, o que favorece a entrega correta dos quadros, por exemplo.

Os motivos acima apresentados demandam a necessidade de se conceber novas métricas de roteamento, capazes de acompanhar as flutuações inerente aos canais sem fio.

2.2.7.3 Suporte ao tratamento diferenciado de serviços

O uso de métricas dinâmicas permite ao roteamento hierarquizar as rotas disponíveis sob um dado critério de qualidade. Desta forma classes de tráfego com diferentes restrições de QoS podem ser atendidas sob medida. O mesmo raciocínio pode ser empreendido para lidar com diferentes classes de assinaturas de serviços, onde as melhores rotas poderiam ser destinada a um serviço dedicado e as rotas *melhor-esforço* sem garantias de QoS, seriam utilizadas para fins outros diversos.

Também é possível utilizar mais de uma métrica de roteamento para garantir uma personalização ainda maior do serviço. Serviços de tempo real, tais como Voz sobre IP (VoIP) e videoconferência, são sensíveis ao atraso de pacotes, ao passo de que serviços baseados em *Transmission Control Protocol* (TCP), precisam de garantias de entrega sem que, para isso restrições de tempo sejam necessariamente

consideradas.

Adicionalmente, com uma maior granularidade da qualidade proporcionado pelas métricas dinâmicas, é possível fazer um balanceamento racional do tráfego de entrada no roteador, permitindo uma boa seleção de rotas alternativas à medida em que as rotas previamente conceituadas como melhores ficam saturadas com tráfego de usuário.

A missão de abstrair a qualidade de um canal sem fio tendo em conta as flutuações da força do sinal, logo resultou em uma linha de pesquisa na área de WMNs. Em [46], os autores discorrem sobre diretivas gerais para o projeto de métricas dinâmicas em WMNs. Os autores também reiteram que a escolha de uma métrica deve se ajustar as necessidades de aplicações e particularidades da WMN em questão.

A primeira métrica proposta para WMNs é conhecida pela sigla ETX (*Expected Transmission count*) [42]. A métrica ETX pesa os enlaces com o auxílio de mensagens de inspeção de canal denominadas *probes*. Esse esquema possibilita ao ETX *estimar* a quantidade necessária de (re)transmissões em nível de enlace para que um quadro seja recebido com sucesso. Assim, a melhor rota sob a ótica do ETX é aquela que minimiza o total de retransmissões de um mesmo quadro em cada um de seus enlaces.

O ETX serviu de base para o surgimento de outras novas métricas, cada uma das quais procurando incorporar novos aspectos no cálculo proposto pelo ETX ou novas semânticas de interpretação da qualidade, levando a alterações na forma de cálculo. A métrica *Expected Transmission Time* (ETT) estima o tempo necessário para a transmissão de um quadro ocorrer com sucesso. Como esse tempo pode ser afetado pelo tamanho do pacote, o ETT também consideração a taxa de transmissão em adição aos elementos de cálculo propostos no ETX. A métrica *Weighted Cumulative Expected Transmission Time* (WCETT), por sua vez, incorpora aspectos de interferência no cálculo do peso do enlace. Um exemplo de métrica que segue uma

abordagem diferente da ETX é a métrica *Link Delay* (LD), proposta por Cordeiro et al. em [40]. Na LD a camada de roteamento prepara e entrega simultaneamente duas mensagens de controle para a camada inferior. Quando entregues, o destino estima a capacidade do enlace pela diferença de tempo entre as mensagens.

A área de métricas de roteamento para WMNs é relativamente nova. O *draft* mais recente do futuro padrão 802.11s [58], sugere uma métrica chamada *Airtime Link Metric*. A proposta, na verdade, prescreve que três fatores que devem ser considerados para calcular o tempo necessário para se transmitir um quadro com sucesso, a saber, a taxa de transmissão (tamanho do quadro), o *overhead* imposto pela camada física e a probabilidade de retransmissão do quadro [89]. Contudo a forma de calcular este último fator (que pode ser considerado a essência da métrica), é deixado a critério do usuário do padrão [55].

Para uma estudo aprofundado sobre as métrica acima mencionadas, por favor, consulte suas respectivas referências.

Por fim, em [31] os autores apresentam um estudo e uma análise comparativa entre algumas métricas de dinâmicas de roteamento para WMNs e reiteram a superioridade delas em relação à métrica *hop count*.

2.3 Modelagem de canais 802.11

Os aspectos mais relevantes dos canais 802.11 discutidos na seção 2.2, são considerados para efeitos de modelagem na presente seção. De acordo com estudo realizado por Takai em [92], na avaliação de desempenho de protocolos sobre sistemas sem fio os seguintes modelos básicos devem ser considerados: modelo de propagação, modelo de interferência e o ruído de operação e modelo de recepção de sinal. A seguir apresentamos algumas dentre as principais alternativas disponíveis para esses modelos em canais 802.11.

2.3.1 Modelos de propagação

A modelagem da propagação do sinal eletromagnético cumpre um papel fundamental para a abstração dos canais 802.11, uma vez que por meio dela estima-se a potência dissipada ao longo de percurso de propagação do sinal.

Em geral, os modelos de propagação são classificados como modelos de *larga-escala*, os quais estimam a qualidade média do sinal a uma determinada distância do transmissor, nestes casos considera-se a degradação sofrida ao longo do percurso de propagação da onda (*path loss*), e modelos de *pequena-escala* ou modelos de desvanecimento (*fading models*) , os quais capturam as rápidas flutuações da força do sinal (recebido) em curtos espaços de tempo [88]. Nas próximas subseções apresentaremos os modelos tipicamente utilizados para simulação de sistemas 802.11.

2.3.1.1 Modelos determinísticos *Free Space* e *Two-Ray ground*

Os modelos de propagação *Free Space* e *Two-Ray ground* são essencialmente dependentes da distância d, da potência do sinal transmitido P_t, dos ganhos proporcionados pelas antenas comunicantes G_t e G_r e de um fator $L \geq 1$, referente a dissipações não relacionadas à propagação propriamente dita.

Esses modelos assumem a existência de uma linha de visada direta entre o transmissor e o receptor, contudo o *Two-Ray ground* também contempla um percurso de propagação onde ocorre uma reflexão sobre uma superfície plana. Essa consideração confere ao *Two-Ray ground* uma representação ligeiramente mais precisa que o Free-Space.

A estimação da potência P_r no receptor, conforme o *Free Space* e o *Two-Ray ground*, são obtidas a partir das fórmulas 2.1 e 2.2, respectivamente, onde λ é o comprimento de onda em metros e h_t, h_r representam as alturas das antenas em relação à superfície plana:

$$P_r = \frac{P_t G_t G_r \lambda^2}{(4\pi)^2 d^2 L} \tag{2.1}$$

$$P_r = \frac{P_t G_t G_r h_t^2 h_r^2}{d^4 L} \tag{2.2}$$

2.3.1.2 Modelo de desvanecimento: distribuição de Rice

Sistemas de transmissão sem fio podem sofrer oscilações em curtos períodos. Isso deve-se especialmente a fatores como: movimentação dos nós comunicantes, lagura de banda do canal ou *desvanecimento por multi-percursos*.

A *distribuição de Rice* é comumente empregada para modelar fenômenos de desvanecimento por múltiplos percursos. O parâmetro chave desta distribuição é o fator K de Rice. Este valor pode ser definido como a razão entre a potência do sinal de um *percurso com visada* (*LOS*, do inglês *Line-Of-Sight*) e a potência do sinal de um percurso *sem visada* (*NLOS*, do inglês *Non-Line-Of-Sight*). Assim, se $k \rightarrow \infty$ a influência de NLOS tende a zero.

2.3.1.3 Modelos baseados na técnica de traçado de raios

Com base nas equações de Maxwell é possível modelar minuciosamente o comportamento das ondas eletromagnéticas em suas interações com a matéria. Entretanto a computação inerente a tal processo é demasiadamente dispendiosa [91]. Neste sentido a classe de modelos de propagação baseada em *traçado de raios* tenta fornecer uma aproximação assintótica em relação às equações de Maxwell. Com o apoio de aplicativos gráficos modela-se uma região específica (SISP, *SIte SPecific*) em termos de seus obstáculos, por exemplo árvores, prédios, mobília, etc. Em seguida o algoritmo é aplicado para simular os efeitos das ondas eletromagnéticas como reflexões, difrações e espalhamento do sinal. Esse grau de detalhamento confere maior pre-

cisão porém o tempo de processamento pode ainda ser considerado demasiadamente elevado, se comparada aos procedimentos estocásticos tradicionais.

2.3.1.4 Modelo log-normal *Shadowing*

O modelo log-normal *Shadowing* (ou simplesmente *Shadowing*) utiliza um componente determinístico e outro estocástico para descrever o valor médio da potência degradada $PL(d)$ ao longo de uma distância d. O componente determinístico é membro de $\Theta(\log(d))$, ou seja, ele indica que a força média do sinal recebido decai de forma logarítmica com a distância. Todavia, a uma mesma distância do transmissor, o receptor pode experimentar níveis variados de $PL(d)$, bastando para isso haver diferentes níveis de obstrução ao longo do percurso de propagação do sinal. Para representar isso o modelo *Shadowing* utiliza um componente estocástico log-normal o qual, expresso em decibéis, assume a forma de uma variável aleatória normal de média zero e desvio padrão σ, a saber, $X_\sigma \sim N(0, \sigma^2)$.

Assim, no modelo *Shadowing*, a estimação de P_r requer dois parâmetros que refletem o ambiente modelado, a saber, o expoente α de degradação do percurso (*path loss exponent*) e o desvio padrão σ (em dB). A estimativa de degradação da potência de transmissão $PL(d)$ conforme o modelo *Shadowing* é dada pela equação 2.3.

$$PL(d) \quad = \quad \overline{PL}(d_0) + 10\alpha \log\left(\frac{d}{d_0}\right) + X_\sigma \tag{2.3}$$

d_0 é a *distância de referência*, i.e. distância a partir do transmissor onde a degradação pode ser computada deterministicamente pela equação 2.4. Tal equação é referente à propagação no espaço livre. Alternativamente, o valor $\overline{PL}(d_0)$ da equação 2.3 pode ser substituído pela constante c, que corresponde à dissipação *medida* a uma distância d_0 do transmissor.

Ambiente	Valor de α
Espaço livre	2
Rede interna com visada	1.6 à 1.8
Área urbana	2.7 à 3.5
Rede interna com obstruções	4 a 6

Tabela 2.4: Modelo log-normal Shadowing:Expoentes de degradação típicos para alguns ambientes

$$\overline{PL}(d_0) = \begin{cases} 10\log\left[\dfrac{G_t G_r \lambda^2}{(4\pi)^2 d_0^2}\right], \text{ or} \\ c \end{cases} \tag{2.4}$$

A tabela 2.4, apresentada em [88] por Rappaport, apresenta valores típicos de α consolidados para uso no modelo Shadowing. Essa tabela é resultado de diversos trabalhos de medições independentes.

2.3.2 Modelos de interferência e ruído de operação

Com base no modelo de propagação, é possível estimar a intensidade P_r com que a potência do sinal transmitido P_t é detectado pelo NIC receptor. Conforme discutido na subseção 2.2.1, durante a recepção de um quadro, o NIC pode sofrer com a auto-interferência e, consequentemente, experimentar a colisão de quadros. Este fenômeno, por sua vez, é reproduzido com base na modelagem da interferência.

A notação concernente aos modelos explicados nesta seção é dada a seguir. y representa a estação que conseguiu a "atenção" da estação x a fim de enviar quadros para ela por meio da comunicação $(y \rightarrow x)$[2]. I é o conjunto de estações cujas transmissões estão sendo detectadas pelo NIC de x concomitante à comunicação $(y \rightarrow x)$.

[2]ligeiro abuso de notação, uma vez que numa comunicação 802.11, y recebe quadros `ACKs` de x quando esta processa os quadros `DATA` com sucesso

Neste conjunto pode haver também outras estações querendo transmitir para x, mas que serão entendidas como interferência do ponto de vista da comunicação $(y \rightarrow x)$. Por fim $\mathcal{R}_0$ é o ruído correspondente ao esquema de demodulação utilizado por x para decodificar os quadros de y.

Cabe ainda ressaltar que a forma como aplicaremos os modelos a seguir representam apenas o fenômeno da auto-interferência. A interferência de origem desconhecida é o principal fator de "imprevisibilidade" da modelagem do canal 802.11 em WMNs [54], e uma distribuição para tal fenômeno pode variar dramaticamente de acordo com o local da rede. Por estes motivos não a consideraremos neste trabalho. Para uma metodologia de modelagem de interferência externa a partir de medições de uma dada rede, consultar [82].

2.3.2.1 Comparação com valor limiar de colisão

Neste modelo y representa a estação de maior potência em comparação com as demais estações do conjunto I que "desejam" se comunicar com x. Assim o modelo define uma razão entre a potência $P_r^{(y \rightarrow x)}$, correspondente à comunicação $(y \rightarrow x)$, e a soma $\sum_{\forall i \in I} P_r^{(i \rightarrow x)}$, correspondente as demais potências detectadas por x e indesejadas em sua comunicação com y. Se esse valor for superior a um valor limiar de colisão (*Collision Threshold*) então o quadro da comunicação $(x \rightarrow y)$ é declarado livre de colisão e os demais quadros concorrentes são ignorados. Do contrário todos os quadros serão declarados como "perdidos por colisão". Nesse modelo o ruído de operação da taxa de transmissão não é considerado.

2.3.2.2 Interferência cumulativa

Na interferência cumulativa é definida a razão *sinal-ruído* (*Signal to Noise Ratio*, SNR) de uma comunicação $(y \rightarrow x)$ juntamente com a potência subjacente aos demais quadros que a interferem. Disso resulta a razão *Signal to Interference and*

Noise Ratio (SINR), composta pelo sinal ($P_r^{(y \to x)}$) e a soma do ruído de operação com os sinais de interferência ($\sum_{\forall i \in I} P_r^{(i \to x)} + \mathcal{R}_0$) conforme definida na equação 2.5. Ela encerra o ruído de operação e as interferências eem um mesmo fenômeno, denominado simplesmente por *ruído*.

$$SINR_{y \to x} = \frac{P_r^{(y \to x)}}{\sum_{\forall i \in I} P_r^{(i \to x)} + \mathcal{R}_0} \tag{2.5}$$

A interferência cumulativa é o modelo mais adequado para modelar a auto-interferência de um sistema sem fio. Cabe aqui ressaltar que ele não decide sobre perdas devido a colisões. Ele apenas calcula o SINR e o repassa para o *modelo de recepção de sinal* o qual se encarregará de decidir sobre a correta demodulação ou não do quadro.

2.3.3 Modelos de recepção do sinal

Os modelos de recepção do sinal são responsáveis por definir sobre o sucesso da recepção de um quadro. Basicamente, eles tomam tal decisão a partir do SINR subjacente ao quadro. Para os modelos explicados a seguir, as variáveis *CSThreshold* e *RXThreshold* abstraem as zonas de interferência e de transmissão de uma estação, respectivamente.

2.3.3.1 Modelo SNRT: valor limiar de SINR

O modelo SNRT (*SNR Threshold*) baseai-se *somente* em valores limiares para decidir sobre a recepção correta de um quadro. Ele estabelece que se o SINR subjacente a um quadro for maior que o valor de potência indicado em *CSThreshold* então pelo menos a portadora do quadro é detectada. Se, contudo, tal SINR subjacente for

menor do que aquele valor de potência indicado em *RXThreshold* então é impossível ele ser demodulado com sucesso, servindo apenas como interferência para eventuais comunicações paralelas.

2.3.3.2 Modelo baseado em BER

O modelo baseado em BER utiliza o SINR e o esquema de modulação de um quadro como informações para decidir sobre sua correta recepção. Essa decisão é feita de maneira probabilística tendo-se em vista que o sucesso da recepção de um quadro é extremamente correlacionado com o SINR[54]. De forma geral, quanto maior a razão SINR (i.e. maior o sinal e menor o ruído) menor será a taxa de erros BER e, consequentemente, maior será *probabilidade* de se decodificar um quadro corretamente. Essa relação é ainda afetada pela complexidade do quadro que é determinada pelo esquema de modulação utilizado. A figura 2.11 ilustra as curvas BER×SINR típicas para cada esquema de modulação 802.11b. Os dados utilizados referem-se ao NIC Intersil HFA3861B[59].

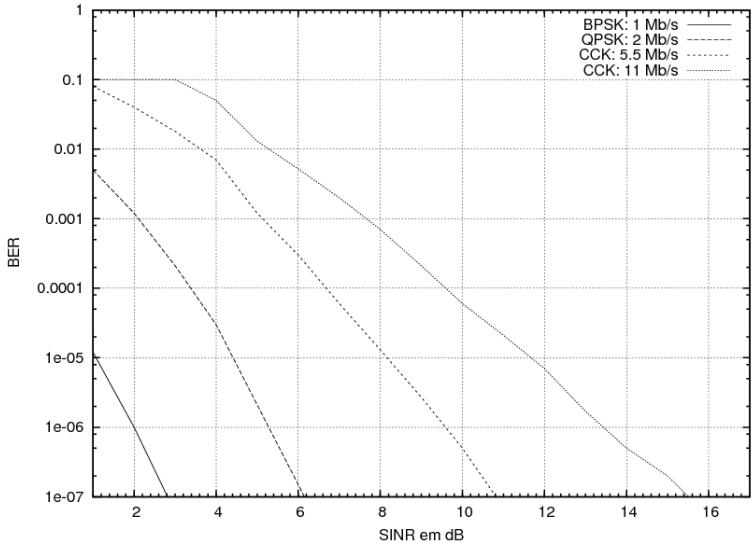

Figura 2.11: NIC Intersil HFA3861B 802.11b: Curvas BER×SINR

O modelo BER não necessariamente elimina o uso das variáveis *CSThreshold*

e *RXThreshold*. Ele apenas permite reproduzir os (relativamente raros) casos em que um quadro apresenta *bits* corrompidos mesmo sob uma potência de transmissão maior que o valor indicado em *RXThreshold*. Além disso ele leva em consideração a sensibilidade de recepção inerente à cada esquema de modulação para ajudar no processo. Por estes motivos ele é considerado um modelo bem mais realístico do que o SNRT [92].

2.4 Fundamentos de análise de algoritmos

Nesta seção revisamos alguns conceitos fundamentais de análise de algoritmos a fim de fornecer um texto auto-contido e amparar a análise dos algoritmos apresentados na seção 3.4 e avaliados no capítulo 5.

2.4.1 Análise assintótica de funções de custo

Um passo fundamental na análise de um algoritmo consiste em obter sua função $f(n)$ correspondente, que denominamos *função de complexidade* ou *função de custo*. No caso da complexidade de tempo, a função f representa a quantidade de operações elementares realizadas pelo algoritmo mediante uma entrada de tamanho n. No processo de análise em geral considera-se mais relevante as operações cujo total de execuções depende da entrada ao passo que operações cujo total de execuções independe da entrada são consideradas de menor relevância. Há contudo casos em que os fatores constantes podem ser um critério decisivo na escolha entre algoritmos diferentes, especialmente quando as funções de custo deles pertencem à mesma classe de complexidade.

As funções de custo servem de base para podermos classificar quão complexo um algoritmo é em relação a um outro de mesmo propósito. As relações de comparação entre funções de custo são estabelecidas no primeiro quadrante do plano cartesiano

através da *notação assintótica*, cujas definições são explanadas a seguir.

- notação O: uma função $f(n)$ *domina assintoticamente* uma função $g(n)$ quando existe um ponto $n_0 \geq 0$ no eixo das abscissas a partir do qual é possível *manter* a curva referente à $f(n)$ *maior ou igual* a curva referente à $g(n)$ com o auxílio de *algum* fator constante positivo, conforme ilustrado na figura 2.12. Formalmente temos que $g(n) = O(f(n)) \equiv g(n) \in O(f(n)) = \{\exists\ c > 0, n_0 \geq 0 \mid g(n) \leq c \cdot f(n),\ \forall n \geq n_0\}$

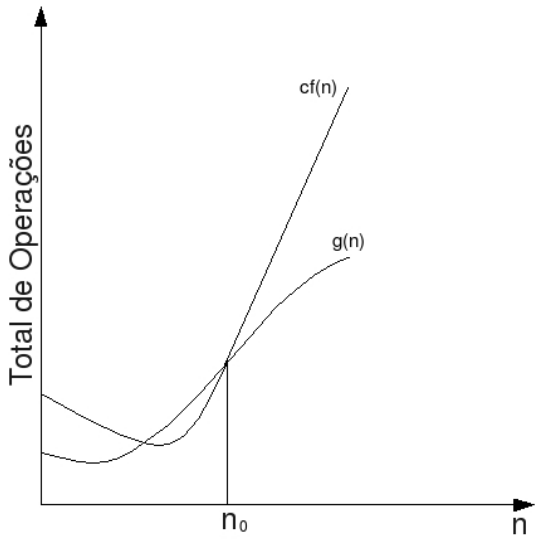

Figura 2.12: Notação O: Dominação assintótica da função $f(n)$ sobre $g(n)$

- notação Ω: uma função $f(n)$ *é assintoticamente dominada* por uma função $g(n)$ quando existe um ponto $n_0 \geq 0$ no eixo das abscissas a partir do qual é possível *manter* a curva referente à $f(n)$ *menor ou igual* a curva referente à $g(n)$ com o auxílio de *algum* fator constante positivo, conforme ilustrado na figura 2.13. Mais formalmente temos que $g(n) = \Omega(f(n)) \equiv g(n) \in \Omega(f(n)) = \{\exists\ c > 0, n_0 \geq 0 \mid g(n) \geq c \cdot f(n),\ \forall n \geq n_0\}$

- notação Θ: uma função $f(n)$ é dita pertencer à mesma classe de complexidade de outra função $g(n)$ se é possível satisfazer condições tais que $f(n)$ tanto possa

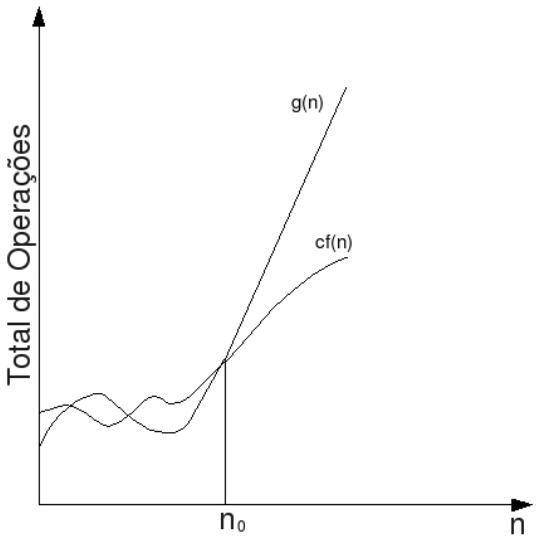

Figura 2.13: Notação Ω: Dominação assintótica da função $g(n)$ sobre $f(n)$

dominar como ser assintoticamente dominada por $g(n)$. Mais formalmente $f(n) = \Theta(g(n)) \Leftrightarrow f(n) = O(g(n)) \wedge f(n) = \Omega(g(n))$

- notação *o*: uma função $f(n)$ *apresenta uma dominação assintoticamente frouxa* sobre uma função $g(n)$ quando, para qualquer constante positiva c_i existe um ponto correspondente $n_i > 0$ no eixo das abscissas a partir do qual a curva $c_i \cdot f(n)$ mantém-se *maior* do que a curva $g(n)$. Formalmente temos $g(n) = o(f(n)) \equiv g(n) \in o(f(n)) = \{\forall c_i > 0, \exists n_i > 0 \mid g(n) < c_i \cdot f(n),\ \forall n \geq n_i\}$, alternativamente essa relação pode ser expressa pelo limite apresentado na equação 2.6:

$$\lim_{n \to \infty} \frac{g(n)}{f(n)} = 0 \tag{2.6}$$

- notação ω: uma função $f(n)$ *é dominada* por uma função $g(n)$ de forma *assintoticamente frouxa* quando, para qualquer constante positiva c_i existe um ponto correspondente $n_i > 0$ no eixo das abscissas a partir do qual a curva $c_i \cdot f(n)$ mantém-se *menor* do que a curva $g(n)$. Formalmente temos $g(n) =$

$\omega(f(n)) \equiv g(n) \in \omega(f(n)) = \{\forall c_i > 0, \exists n_i > 0 \mid g(n) > c_i \cdot f(n),\ \forall n \geq n_i\}$, alternativamente essa relação pode ser expressa pelo limite apresentado na equação 2.7:

$$\lim_{n\to\infty} \frac{g(n)}{f(n)} = \infty \tag{2.7}$$

2.4.2 Análise de casos

Quase sempre o total de operações executadas por um algoritmo é dependente da entrada. Além disso também são frequentes nos algoritmos a presença de trechos de códigos mutuamente excludentes, o que pode refletir numa variação da quantidade de operações executadas pelo algoritmo à cada entrada. Devido a esses motivos a classe de complexidade de um mesmo algoritmo pode variar de acordo com a ótica tomada na análise de custo. Os casos mais comumente considerados são:

- *Melhor caso*: corresponde a menor classe de complexidade de um algoritmo (i.e. menor tempo de execução). Decorre da análise do algoritmo sob a instância do problema que sempre acessa os trechos menos custosos;
- *Pior caso*: corresponde a maior classe de complexidade de um algoritmo (i.e. maior tempo de execução). Decorre da análise do algoritmo sob a instância do problema que sempre acessa os trechos mais custosos;
- *Caso médio* ou caso esperado: *classe de complexidade média*, resultante da média de operações executadas pelo algoritmo a partir de todas as entradas de tamanho n obtidas de uma distribuição de probabilidades suposta [99].

A classe de complexidade referente ao pior caso de um algoritmo fornece-nos um limite superior de desempenho. Ela assegura que um algoritmo não apresentará um tempo de execução ainda pior do que o indicado. Entretanto há situações em

que o pior caso de um algoritmo diverge consideravelmente do desempenho por ele apresentado em seu contexto de aplicação prática. Nestes circunstâncias o estudo do caso médio assume um papel chave no processo na análise de algoritmos [30]. Além disso, no contexto deste trabalho a análise de caso médio também é motivada pelo fato de que os algoritmos a serem avaliados foram projetados sob modelos de custo diferentes, conforme veremos adiante.

Nas próximas seções apresentaremos os algoritmos de Dijkstra e o de Ramalingan & Reps com suas respectivas análises de pior caso. Em seguida, no capítulo 4, apresentaremos uma metodologia para *estimar* as curvas referentes aos casos médios desses algoritmos no contexto das WMNs.

2.5 Protocolos de roteamento para redes em malha sem fio

A dificuldade resultante do uso de canais sem fio 802.11 como enlace de roteamento pode ser verificada, em parte, pelo grande número de protocolos propostos para as primeiras MANETs baseada em tal tecnologia [26]. Em resposta a tal desafio, o principal órgão de propostas de padronizações para a Internet (*Internet Engineering Task Force*, IETF) criou um grupo de trabalho especializado em roteamento para MANETs em nível IP cujos resultados tem sido amplamente adotado para uso em WMNs.

O grupo IETF-MANET adotou duas linhas de padronizações que correspondem à classificação usual para os protocolos de roteamento MANETs, a saber, *Reactive MANET Protocols* (RMP) e *Proactive MANET Protocols* (PMP). Os protocolos reativos só apresentam atividade de roteamento mediante uma solicitação de rota, ao passo que os protocolos proativos procuram manter uma visão atualizada da rede

(i.e. das melhores rotas) independente de solicitações.

Há ainda a classe de protocolos híbridos, que capturam características reativas e proativas. No âmbito do IETF-MANET ainda não há atividades de padronização nesse sentido. Tal grupo atualmente apenas considerado-o como uma possibilidade futura:"*If significant commonality between RMRP and PMRP protocol modules is observed, the WG may decide to go with a converged approach*" [12].

Nesta seção apresentaremos as classes RMP e PMP a partir de dois dentre seus principais representantes padronizados pelo IETF. Mas antes disso, faremos uma breve apresentação das filosofias de roteamento amadurecidas no âmbito das redes cabeadas, em particular na Internet.

2.5.1 Filosofias de roteamento da Internet

Apesar dos trabalhos de análise comparativa de protocolos de roteamento para MANETs frequentemente apontarem as filosofias de roteamento *Vetor de Distâncias* (*Distance Vector*) e *estado de enlace* (*Link State*) como inadequadas para roteamento em canais sem fio, na prática, os principais protocolos de roteamento para MANETs incorporam elementos dessas abordagens. Em geral, alguma alteração é proporcionada para remediar certas limitações das filosofias cabeadas no contexto sem fio. Por este motivo a seguir apresentaremos brevemente tais abordagens.

2.5.1.1 Roteamento estado de enlace

A principal característica associada à abordagem estado de enlace é o fato de que nela os roteadores devem deter o conhecimento *global* da rede em termos de destinos possíveis e enlaces com respectivos pesos. Para isso é necessário a definição de algum mecanismo que capacite os roteadores a investigar o estado de seus enlaces à potenciais vizinhos e, em seguida, difundir essas informações para os outros roteadores. De posse das informações topológicas, o algoritmo realiza um cálculo para

descobrir os caminhos de menor peso até cada destino. Os exemplos mais famosos de protocolos de roteamento globais da Internet são os protocolos OSPF [74] e IS-IS [77].

A principal alegação contra o uso da abordagem de estado de enlace para roteamento em MANETs e WMNs é o *overhead* de mensagens de controle para manter uma visão atualizada da rede nos roteadores. O que significa que protocolos globais para MANETs devem prover alguma maneira de lidar com tal limitação.

2.5.1.2 Roteamento vetor de distância

Em oposição à filosofia de estado de enlace, os roteadores operando sob a abordagem vetor de distâncias não contém informações completas sobre a topologia da rede. Desta forma, a conclusão final sobre os melhores caminhos da rede é atingida após sucessivas trocas de mensagens entre vizinhos a respeito dos melhores caminhos correntemente por eles conhecidos. Após esse processo, os roteadores terão uma *tabela registro de distâncias* na qual cada linha corresponde a um destino na rede, cada coluna a um vizinho e a intercessão da coluna V com a linha D revela o custo de se alcançar o destino D por meio do vizinho V; o menor custo da linha V representa o custo ótimo para se alcançar o destino V. Um dos mais famosos representantes da abordagem de vetor de distâncias na internet é o protocolo *Routing Information Protocol* (RIP)[68].

O principal problema inerente ao roteamento vetor de distância, é o *problema da contagem até o infinito*. Este problema pode ocorrer da seguinte maneira. Primeiramente um caminho ótimo perde tal propriedade devido o aumento de peso de um de seus enlaces. Em seguida, o roteador R adjacente ao enlace afetado E declara um novo vizinho pelo qual pretende alcançar o(s) destino(s) afetado(s) de maneira ótima. Contudo, em algum ponto deste novo caminho, o enlace E aparece novamente. A quantidade de iterações necessárias para se identificar o novo me-

lhor caminho (i.e. que não contém o enlace afetado) é proporcional à mudança de peso sofrida por E. Durante esse processo o tráfego de dados sofrerá com *loops de roteamento*.

2.5.2 Roteamento reativo: AODV

Os protocolos de roteamento *reativos* são caracterizado por construir as rotas somente quando elas são solicitadas, por este motivo eles também são conhecidos como protocolos sob demanda.

O *Ad Hoc On-Demand Distance Vector* (AODV) é um protocolo de roteamento reativo bastante difundido em MANETs. Ele elevado à categoria de RFC em 2003, sendo definido pelo IETF na RFC 3561 [81]. Além de ser reativo, o AODV também utiliza a filosofia vetor de distâncias e um mecanismo adicional para resolver os problemas da contagem até o infinito e de rotas com *loops* desta filosofia, conforme descrito a seguir.

Durante uma comunicação no AODV cada roteador mantém uma tabela de roteamento que informa o próximo salto em direção ao destino e um *número de sequência de destino* que indica a rota mais recente conhecida para o destino desejado. O descobrimento das rotas é realizado por meio da inundação *broadcast* de *mensagens de requisição* (*Route Request*, `RREQ`). As mensagens `RREQ` são encaminhadas de vizinho em vizinho contendo, basicamente, o contador de saltos e os endereços e números de sequência da origem e do destino. Se o roteador que recebeu a mensagem `RREQ` não corresponder ao destino nela solicitado ou não conhecer uma rota válida para o mesmo então ele atualizará a métrica *hop count*, atualizará sua tabela de roteamento para poder rotear a mensagem `RREP` (*Route Reply*) em resposta à requisição da origem e o inundará seus vizinhos com o `RREQ`. Um roteador intermediário que possui uma rota válida para o destino solicitado *não* estará autorizado a responder

a um RREQ se o *bit* D ("*Destination only*") estiver ativado. Nesse caso somente o destino poderá responder com um RREP.

Quanto ao mecanismo de manutenção de rotas, o AODV tanto pode ser configurado para terceirizar a atividade de detecção de falhas (exemplo: uso de quadros ACK da camada MAC do 802.11) como pode utilizar uma solução na própria camada de roteamento, através de pacotes HELLO. As mensagens HELLO são do tipo *broadcast* e são destinadas somente vizinhos imediatos, i.e. possuem TTL igual a 1. Após o recebimento de um HELLO o roteador cria um registro correspondente na tabela de roteamento. Se um novo HELLO não for recebido dentro de um determinado intervalo de tempo, assume-se que o enlace foi perdido e a entrada da tabela de roteamento correspondente é removida.

Quando um roteador AODV intermediário detecta uma falha em uma rota ativa ele pode enviar uma *mensagem de notificação de erro de rota* (*Route Error*, RERR) aos roteadores afetados ou pode tentar fazer um *reparo local*. No primeiro caso a mensagem RERR é enviada para todos os roteadores afetados pela falha, i.e. que precisavam da rota que quebrou. Isso é possível porque o roteador que detecta o enlace falho também mantém a lista de roteadores dependentes dela. Ao receberem a mensagem RERR os roteadores origens afetados armazenarão os tráfegos dos clientes em uma fila e um novo processo de descobrimento de rotas será ativado. Alternativamente, se o roteador que detectou a falha estiver a uma certa *distância máxima de reparo* do roteador destino afetado, ele poderá iniciar um processo de repara do rota de forma transparente aos roteadores origens. Durante esse processo o roteador intermediário irá armazenar os tráfegos de dados e, se uma nova rota não for conseguida após um número de tentativas, o tráfego armazenado será descartado e uma mensagem RERR será gerada. A distância máxima de reparo sugerida pela RFC 3561 é de 10 saltos.

Por um lado, a característica reativa do AODV de não apresentar *overhead* até

que um roteador específico necessite de uma rota é uma das principais vantagens para seu uso em WMNs. Por outro lado, o AODV pode apresentar algumas limitações severas. Além dos atrasos adicionados devido a construção e manutenção de rotas, o armazenamento do tráfego nos roteadores durante o período manutenção de rotas pode se tornar o fator crítico, tendo em vista o perfil típico esperado em WMNs, a saber, redes de acesso à Internet com diferentes classes de tráfegos e composta por roteadores com recursos de memória e processador relativamente escassos [34].

Por fim, as recentes publicações do grupo IETF-MANET não contemplam uma nova RFC para o AODV. Contudo todos os seus mecanismos básicos foram herdados pelo DYMO (*Dynamic MANET On-demand*), o atual foco do grupo RMP do IETF que ainda ainda encontra-se na fase de desenvolvimento de *draft*[32].

2.5.3 Roteamento pró-ativo: OLSR

O *Optimized Link State Routing* (OLSR) é um protocolo de roteamento proativo inicialmente projetado para MANETs. O OLSR foi padronizado pelo IETF através da RFC 3626 [38] e atualmente é o centro das atenções da linha PMP do IETF.

O OLSR segue a filosofia de estado do enlace da Internet assim, conforme previamente visto, o pré-requisito básico para um roteador calcular as melhores rotas é ter à sua disposição uma visão global da rede. Na qualidade de um protocolo proativo, um roteador OLSR monitora *periodicamente* o estado de seus enlaces em busca de calcular e manter rotas até seus vizinhos. Em seguida, *informações topológicas* (e não resultado de cálculo de rotas) são periodicamente divulgadas entre os roteadores para possibilitar o cálculo e a manutenção das rotas aos demais destinos da rede. Esse processo independe de solicitações de usuários.

Tendo em vista o alto dinamismo topológico inerente às MANETs, a adoção da abordagem estado de enlace pura precisa de alguns mecanismos adicionais a fim de

melhor lidar com os agravantes devidos ao uso de enlaces sem fio. Neste sentido, o OLSR utiliza um mecanismo chamado *Multipoint Relay* (MPRs).

Um roteador MPR é um roteador autorizado a repassar adiante mensagens de controle. Assim cada roteador OLSR elege dentre seus vizinhos aqueles roteadores que irão compor seu *conjunto MPR*. Um vizinho não MPR pode processar para si as mensagens de controle que recebe, mas não pode passá-las adiante. De acordo com o OLSR o conjunto MPR deve ser construído de tal forma a se alcançar qualquer vizinho a dois saltos de distância por meio de um nó MPR.

A figura 2.14 (fonte: [94]) compara o processo de divulgação dos enlaces do roteador central nas abordagens tradicional, conhecido por *flood*, com a abordagem MPR usada no OLSR. Na figura, o círculo formado pelos nós mais internos correspondem ao conjunto de vizinhos do roteador central enquanto que os mais externos representam os vizinhos a dois saltos de distância (i.e. não encontram-se na zona de transmissão do roteador central). A quantidade de mensagens de controle transmitidas, expressa pelas setas, reduz drasticamente com o uso dos MPRs.

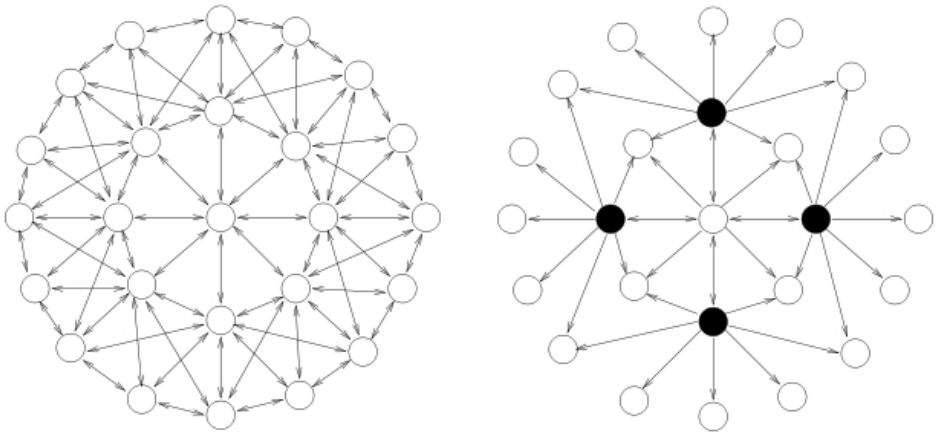

Figura 2.14: Difusão de informações topológicas em roteamento global: inundação tradicional × MPR (OLSR)

Diversos *firmwares* baseados em Linux têm o OLSR como protocolo padrão para implementação de WMNs. Isso tem impulsionado especialmente a construção de

comunidades digitais de livre acesso. Em geral, essas comunidades são caracterizadas pelo acesso a Internet (por exemplo [4; 13; 6]), entretanto as características intrínsecas às WMNs tem possibilitado a implementação redes com propósitos diversos. Um, dentre muitos exemplos que ilustra essa realidade, é o projeto *Hivenetworks* [11]. Esse projeto implementa a noção de *mídias locativas* em certos pontos da cidade de Londres através WMNs baseadas no OLSR. Em outras palavras, é possível que um usuário móvel receba conteúdo de áudio retratando aspectos históricos associados ao local de onde ele está acessando, conforme exemplificado na figura 2.15-a. A figura 2.15-b ilustra a topologia da WMN do projeto *Hivenetworks*.

Figura 2.15: Projeto "*Hivenetworks*": a) Usuário acessando mídias locativas b) Topologia da WMN baseada em OLSR

Outra possibilidade em comunidades digitais é o crescimento da rede através de colaboradores voluntários. A adesão à rede pode ser feita tanto por meio de computadores pessoais equipados com NICs 802.11 e antenas, como por meio APs 802.11 atualizado com um *firmware* disponibilizado pela comunidade (por exemplo [10]). Um exemplo de uso do OLSR nestas condições é a rede *Athens Wireless Metropolitan Network* (AWMN), em Atenas, na Grécia. A figura 2.16, ilustra o *backbone* da rede com cerca de dois mil roteadores [4].

Por fim, o uso do OLSR em WMNs conforme acima exposto tem evidenciado

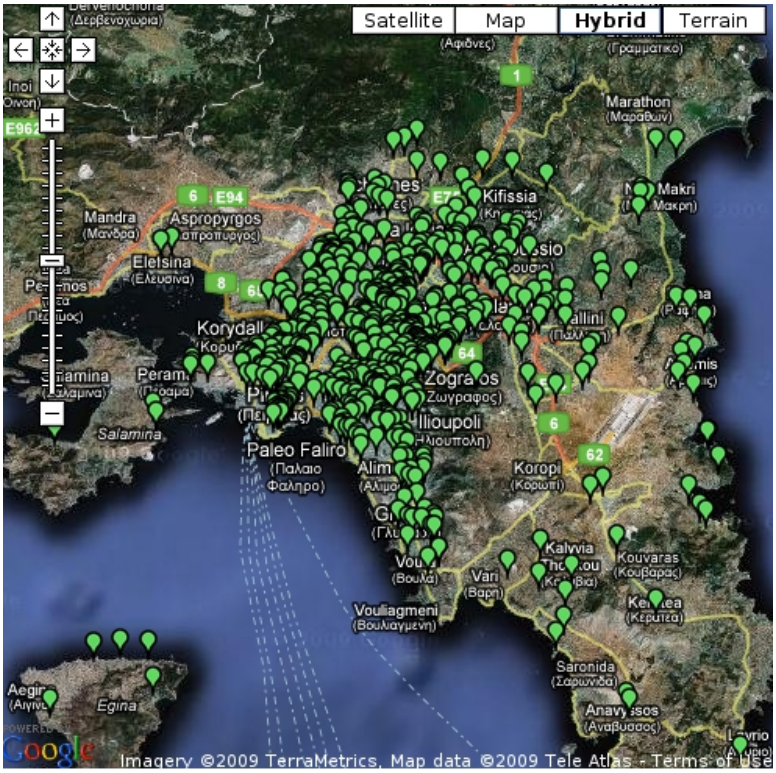

Figura 2.16: Projeto AWMN: Exemplo de crescimento explosivo

novos desafios para protocolo, muitos deles reportado pelas próprias comunidades digitais. No próximo capítulo discutiremos sobre o desafio da manutenção da tabela de roteamento no OLSR.

Capítulo 3

Descrição do problema

Tendo em vista o desafio de se construir e manter rotas ótimas em *backbones* em malha sem fio, o OLSR foi inspirado na forma de operação básica que consolidou o OSPF como um protocolo de roteamento estável nos SAs da Internet.

Nas próximas seções primeiramente apresentaremos esse processo básico, que consiste no levantamento de informações topológicas seguido do construção da tabela de roteamento propriamente dita.

Em seguida, após destacarmos algumas características adicionais das WMNs em comparação aos SAs da Internet, nos concentraremos no *problema dos caminhos mínimos dinâmicos de origem única*, por meio do qual podemos formalizar o problema da manutenção da tabela de roteamento em WMNs. Por fim, discutiremos a solução deste problema por meio do uso de um modelo de computação alternativo ao adotado pelo OLSR.

3.1 O processo de descoberta de topologia no OLSR

O OLSR é um protocolo *orientado a tabelas*. Em outras palavras, ele define conjuntos que registram o conhecimento global do roteador para dar apoio ao fun-

cionamento básico do protocolo. O OLSR define os seguintes conjuntos:

1. *Link Set*: é o conjunto de *links* do roteador. De acordo com o padrão do OLSR um enlace é um par de interfaces OLSR (i.e. um NIC à disposição do OLSR) em roteadores diferentes, por meio do qual uma comunicação poderá ocorrer;

2. *Neighbor Set*: é o conjunto que descreve enlaces entre um roteador e seus vizinhos. Para que um roteador B seja vizinho do roteador A é suficiente que este situe-se na zona de transmissão daquele. Se além disso o roteador B encontrar-se na zona de transmissão de A o enlace entre eles será declarado como simétrico;

3. *2-Hop Neighbor Set*: é o conjunto de enlaces simétricos que conectam os vizinhos de 1 e 2 saltos de distância de um dado roteador A. Neste caso os vizinhos a 1 salto de distância de A devem ser alcançados por meio de enlaces simétricos;

4. *Topology Information Base* ou *Topology Set*: é o conjunto que armazena informações topológicas a partir da recepção de mensagens *broadcast* originadas por outros roteadores da rede;

5. *Multiple Interface Association Set*: é o conjunto que registra as informações divulgadas através de mensagens *Multiple Interface Declaration* (MID). Quando um roteador com múltiplas interfaces OLSR difunde uma mensagem MID, o endereço de cada interface é listado na mensagem. Esses endereços são associados com o endereço da interface OLSR originadora da mensagem (i.e. que está declarada no campo correspondente no cabeçalho da mensagem). Assim uma mesma rota calculada para um roteador OLSR será associada aos endereços que tal roteador declarou em suas mensagens MID;

6. *Multipoint Relay Set* (*MPR set*): define o conjunto de vizinhos que foram selecionados como nós MPR de um dado roteador;

7. *Multipoint Relay Selector Set*: define o conjunto de vizinhos que selecionaram o roteador corrente como um nó MPR;

8. *Duplicate Set*: é o conjunto que registra informações sobre as mensagens de controle mais recentemente recebidas. Por meio dele o OLSR evita a difusão descontrolada de mensagens *broadcast* de controle.

Para manter atualizadas as tabelas a pouco descritas, o OLSR se utiliza de três mensagens de controle, a saber, `HELLO`, `TC` e `MID`. O OLSR também define um pacote básico, conforme ilustrado na figura 3.1.

Um mesmo pacote OLSR pode transportar várias mensagens OLSR diferentes ao mesmo tempo, sendo limitado somente pela unidade máxima de transferência (*Maximum Transfer Unit*, MTU) da tecnologia de enlace utilizada. Abaixo descrevemos brevemente tanto os campos do pacote OLSR (os dois primeiros) como os da mensagem.

- *Tamanho do Pacote*: registra o tamanho em *bytes* do pacote,

- *Número de Sequência do Pacote*: valor incrementado em uma unidade a cada pacote OLSR enviado. Um contador é mantido para cada interface OLSR disponível,

- *Tipo Mensagem*: indica o tipo da mensagem transportada,

- *Tempo Valid.*: esse campo indica o tempo em que o receptor deve considerar válidas as informações transportadas na mensagem caso uma informação mais recente não chegue,

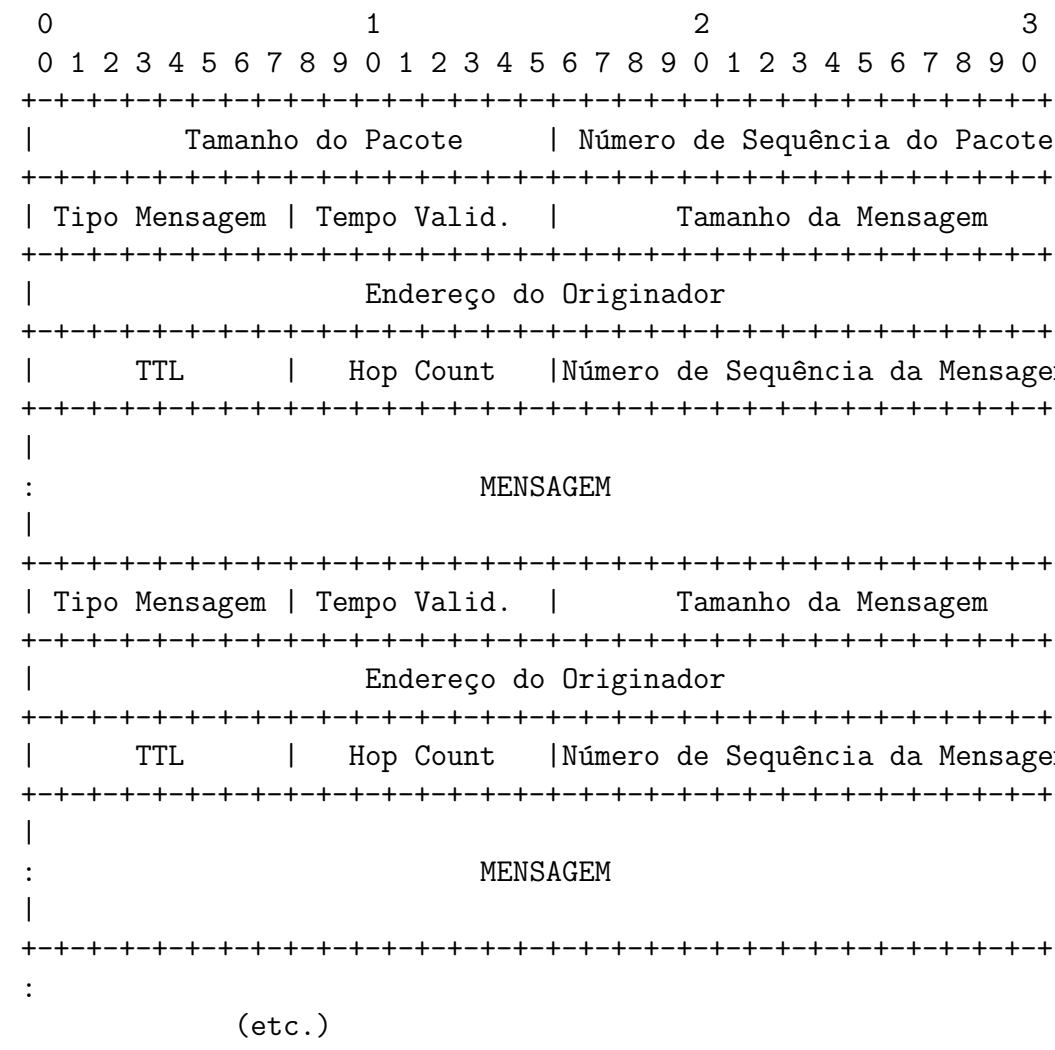

Figura 3.1: Formato básico do pacote de controle OLSR definido na RFC 3626

- *Tamanho da Mensagem*: expressa a quantidade de *bytes* ocupados entre o início do campo *Tipo de Mensagem* e o fim do campo *MENSAGEM*,
- *Endereço do Originador*: endereço da Interface originadora da mensagem. Diferente do campo de endereço do remetente IP, esse campo jamais pode ser modificado, mesmo se a mensagem for retransmitida por outro roteador,
- *TTL*: número máximo de saltos tolerados para se retransmitir a mensagem,
- *Hop count*: incrementado em uma unidade a cada retransmissão,

- *Número de sequência da mensagem*: identificador unívoco da mensagem. Incrementado em uma unidade à cada nova mensagem gerada pelo roteador.

A seguir descreveremos as mensagens do OLSR e suas participações nos processos responsáveis pela instanciação dos conjuntos *neighbor set*, *2-hop neighbor set* e *topology information base*. Esses conjuntos são de fundamental importância para a computação da tabela de roteamento uma vez que o procedimento do cálculo das rotas baseia-se inteiramente nas informações por eles providas.

3.1.1 O processo de descoberta de vizinhos imediatos

No OLSR a mensagem `HELLO` (figura 3.2) é gerada em intervalos regulares, conforme definido no campo *Interv. HELLO*, para ser enviada em *broadcast* no canal sem fio. Ela participa do processo de formação topológica através do processo de *descoberta de vizinhos*. Neste sentido, uma mensagem `HELLO` agrupa os endereços dos vizinhos de acordo com o tipo informado no campo *Código Enlace*. Os tipos de vizinhos possíveis são três, a saber:

- `SYM_NEIGH`: informa que os vizinhos indicados possuem pelo menos um enlace simétrico para o roteador,
- `MPR_NEIGH`: informa que os vizinhos indicados são membros do conjunto MPR,
- `NOT_NEIGH`: informa que os vizinhos indicados (ainda) não possuem enlaces simétricos.

A figura 3.3 ilustra uma sequência de etapas possível durante o processo de descoberta de vizinhos.

Com respeito ao demais campos do cabeçalho da mensagem `HELLO`, o campo *Tamanho da Mensagem de Enlace* lida com os diferentes tamanhos dos grupos de endereços listados. Ele contabiliza o total de *bytes* utilizados desde o campo *Código*

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|          Reservado            | Interv. HELLO |  Willingness  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| Código Enlace |   Reservado   | Tamanho da Mensagem de Enlace |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|            Endereço de Interface de Roteador Vizinho          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|            Endereço de Interface de Roteador Vizinho          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
:                               .   .   .                       :
:                                                               :
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| Código Enlace |   Reservado   | Tamanho da Mensagem de Enlace |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|            Endereço de Interface de Roteador Vizinho          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|            Endereço de Interface de Roteador Vizinho          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
:                                                               :
:                                       :
(etc.)
```

Figura 3.2: Formato da mensagem `HELLO` definida pelo OLSR

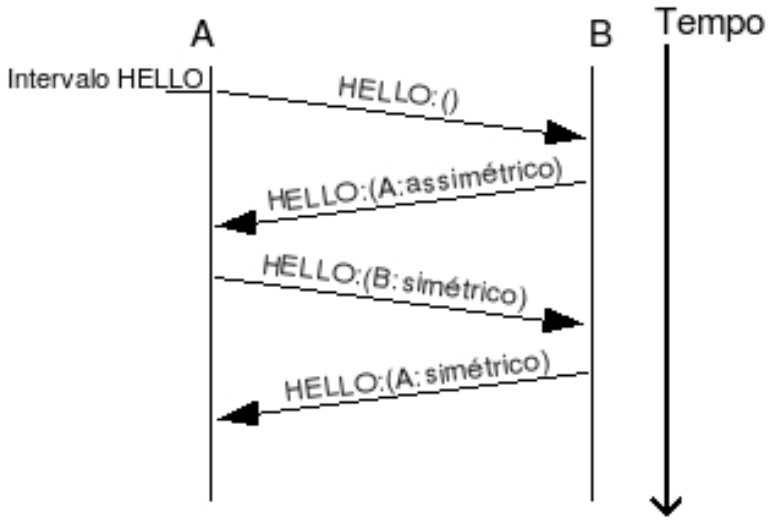

Figura 3.3: Etapas típicas no processo de descoberta de vizinhos do OLSR

Enlace até o último endereço de vizinho declarado no grupo. O campo *willingness* fornece informações aos vizinhos para auxiliar na seleção de nós MPRs. Ele varia de zero a sete. Quando instanciado como zero ("*WILL_NEVER*") o originador da mensagem indica que ele sempre deve ser preterido como nó MPR de seus vizinhos. Ao passo de que o valor sete ("*WILL_ALWAYS*") indica que ele deve sempre ser preferido a outros roteadores vizinhos. O valor *willingness default* estabelecido pela RFC é 3.

Por fim, o campo "*reservado*" não é utilizado, sendo destinado para propósitos futuros.

3.1.2 O processo de descoberta dos vizinhos a dois saltos de distância

A mensagem `HELLO` também é utilizada para a instanciação do conjunto *2-hop neighbor*. De acordo com OLSR, quando um roteador recebe uma mensagem `HELLO` a partir de um vizinho simétrico, ele deve atualizar seu conjunto *2-hop neighbor* com base nos endereços listados nos grupos `SYM_NEIGH` e `MPR_NEIGH`. A única exceção corresponde ao endereço do próprio receptor, a fim de evitar que um roteador seja registrado em seu próprio conjunto *2-hop neighbor*.

3.1.3 Divulgação de mensagens de topologia

No OLSR a mensagem `HELLO` jamais é retransmitida por um roteador receptor. Para que um roteador possa difundir suas informações de estado de enlace na rede através dos nós MPRs, o OLSR define a mensagem `TC`, conforme ilustrada na figura 3.4.

A mensagem `TC` anuncia um *conjunto de endereços de vizinhos* que formam enlaces com o emissor da mensagem. Nesse conjunto devem constar pelo menos

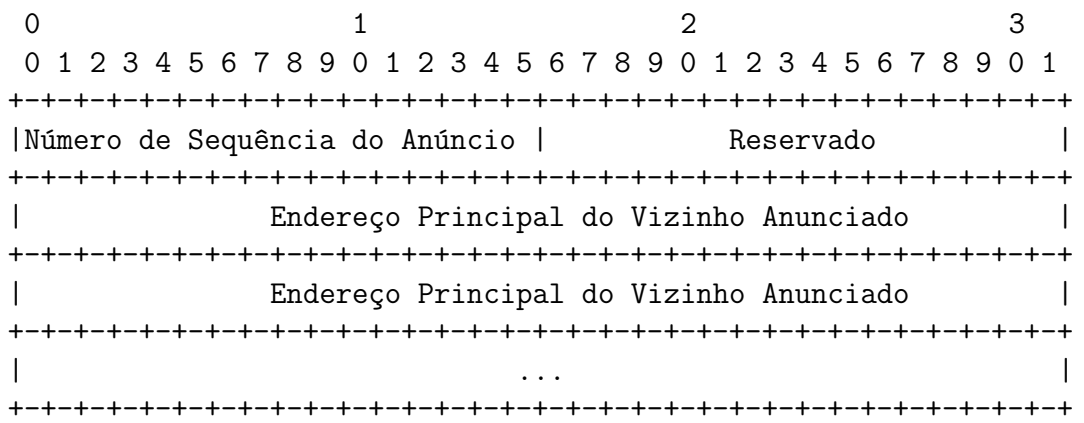

Figura 3.4: Formato da mensagem `TC` definida pelo OLSR

os endereços correspondentes a todos os vizinhos que selecionaram o emissor como nó MPR. Cada um desses endereços é anunciado no campo *Endereço Principal do Vizinho Anunciado*. A mensagem `TC` é enviada em intervalos regulares ou assim que uma modificação for detectada no conjunto *MPR Selector* do emissor. Neste último caso é utilizado o campo *Número de Sequência do Anúncio*. Um número de sequência é atribuído ao conjunto de vizinhos anunciados na mensagem. Esse número permanecerá inalterado a cada mensagem `TC` enviada até que o conjunto *MPR Selector* do emissor seja modificado, i.e., o conjunto de endereços declarados na mensagem seja diferente. Nesse caso o campo *Número de Sequência do Anúncio* será incrementado.

Quanto à formatação dos demais campos da mensagem `TC`, o OLSR recomenda que o campo TTL seja configurado em seu valor máximo (255), para que todos os roteadores sejam alcançados pela mensagem, e que o campo *Tempo de Valid.* seja o triplo do intervalo de envio da mensagem `TC`.

3.1.4 Extensões para métricas dinâmicas

O processo de descoberta e formação topológica para o OLSR não contempla o uso de métricas dinâmicas de roteamento para os enlaces mas somente a métrica

hop count. Em princípio, tal extensão no OLSR ficou a encargo da comunidade de desenvolvedores de protocolos de roteamento.

Conforme comentado na seção 2.2.7, a primeira métrica dinâmica de roteamento desenvolvida para WMNs foi o ETX [42]. Para estimar os pesos dos enlaces e divulgar seus valores, os autores sugeriram o uso de pacotes UDP (*User Datagram Protocol*) extras. No OLSR, a extensão mais aceita na prática foi proposta pelo grupo do projeto OLSR-NG. A solução consiste em adicionar campos extras às mensagens `HELLO` e `TC`, conforme ilustrado nas figuras 3.5 e 3.6, respectivamente. Nelas o campo *Link Quality* declara a qualidade percebida pelo roteador originador da mensagem (cujo endereço está declarado no campo *Endereço do Originador* do pacote OLSR) a partir do vizinho cujo endereço está declarado no campo *Endereço de Interface de Roteador Vizinho* da mensagem OLSR (i.e. valor no sentido do vizinho para o emissor). Já o campo *Neighbor Link Quality*, por sua vez, declara o peso do mesmo enlace no sentido inverso, isto é, do emissor para o vizinho. A mensagem `HELLO` é utilizada para estimar os valores dos enlaces (visto que ela é frequentemente enviada) e a mensagem `TC` é utilizada para difundir os enlaces com seus pesos para os demais roteadores da rede.

Após a grande aceitação das modificações acima descritas, o grupo IETF MANET resolveu criar um padrão à parte que generaliza a definição de pacotes de controle para protocolos de roteamento em MANETs. Por meio desse padrão é possível difundir valores de uma dada métrica de roteamento. O *draft* foi iniciado em 2006 e atingiu sua versão final em fevereiro de 2009, quando tornou-se a RFC 5444 [36]. Esse padrão está sendo considerado, por exemplo, para uso no OLSR2 que, por sua vez, ainda não foi elevado à categoria de RFC.

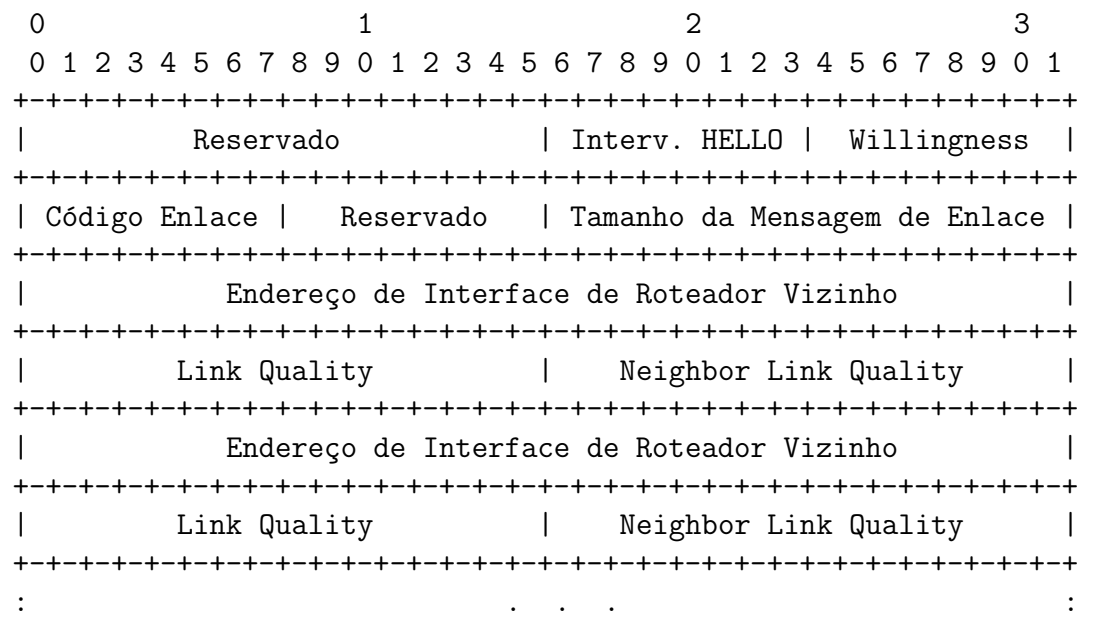

Figura 3.5: Mensagem HELLO: Extensão utilizada para métricas dinâmicas de roteamento

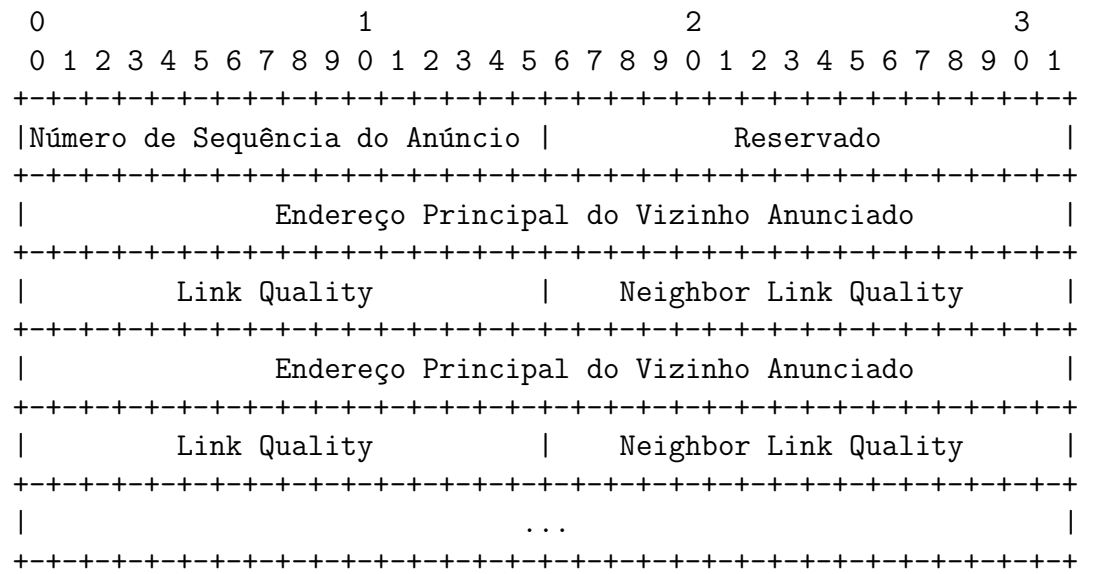

Figura 3.6: Mensagem TC: Extensão utilizada para métricas dinâmicas de roteamento

3.2 O processo de atualização da tabela de roteamento

Após o processo de levantamento da topologia da rede realizada com o auxílio das mensagens `HELLO` e `TC`, o protocolo pode construir a tabela de roteamento. A tabela

de roteamento determina o próximo salto de um pacote em direção ao endereço de destino final nele indicado. Cada destino possível *conhecido* pelo roteador deverá corresponder a um registro na tabela de roteamento. O OLSR define um registro na tabela de roteamento por meio dos seguintes campos,

- `R_dest_addr`: identificação do roteador destino,
- `R_next_addr`: identificação do roteador do próximo salto,
- `R_dist`: número de saltos até o roteador destino,
- `R_iface_addr`: identificação da interface do roteador local.

Em soluções que contemplam o uso de métricas dinâmicas um campo é adicionado para representar o custo de se alcançar o roteador destino a partir do roteador local.

A complexidade computacional dominante no processo de construção e atualização da tabela de roteamento reside no *algoritmo de cálculo dos caminhos de menor custo*, conforme discutiremos na seção 3.4. No OLSR a atualização da tabela de roteamento não deve gerar nenhuma mensagem de controle. Ela deve ocorrer sempre que alguns dos seguintes conjuntos são modificados pelas atividades de manutenção topológica das mensagens `HELLO` e `TC`:

- *Link set*,
- *Neighbor set*,
- *2-hop neighbor set*,
- *topology set*,
- *Multiple Interface Association Information Base*.

A atualização da tabela de roteamento é de fundamental importância para o *backbone* da rede. O padrão do OLSR prescreve que um pacote de dados de entrada deve ser descartado até que na tabela de roteamento haja uma entrada correspondente ao seu destino final.

3.3 Agravantes de roteamento global em WMNs

As etapas a pouco descritas referentes à manutenção da tabela de roteamento do OLSR foram, em sua essência, herdadas do protocolo OSPF-v2. Essa decisão deve-se, em grande parte, ao êxito alcançado pelo OSPF-v2 nos SAs da Internet. Contudo alguns típicos cenários esperados para *backbones* de WMNs (por exemplo comunidades digitais) divergem consideravelmente das características comuns aos SAs da Internet. A seguir comentamos os principais agravantes resultantes dessa diferença.

- *Tecnologia do Enlace*: O propósito de uso dos SAs da Internet, em muitos casos, permite que sua implementação se dê por meio de alguma forma de tecnologia cabeada como fibras óticas, por exemplo. Apesar de tais tecnologias não serem completamente isenta de problemas, elas frequentemente são muito mais estáveis do que enlaces sem fio como os baseados em 802.11. Essa instabilidade relativamente maior em enlaces 802.11 reflete-se no valor da métrica dinâmica de roteamento usada para representá-lo. Em consequência disso a manutenção da tabela de roteamento torna-se uma atividade mais crítica, pois é preciso assegurar a consistência da tabela com as mudanças de peso. De acordo com o padrão do OLSR, a cada alteração de topologia constatada, um roteador deve descartar todas as entradas da tabela de roteamento, atualizar a visão global da rede e recalcular os caminhos mínimos para todos os possíveis destinos conhecidos.

- *Poder computacional dos roteadores* e *dinamismo topológico*: Geralmente os SAs da Internet são operados por entidades administrativas diferentes. A comunicação entre eles ocorre através de um protocolo de roteamento inter-domínio, em sua maioria através do *Border Gateway Protocol*, (BGP) [73]. Isso permite que *a manutenção da topologia dos backbones* (por exemplo quantidade de roteadores no *backbone*) e o poder computacional dos roteadores obedeça a requisitos personalizados de projeto. Por outro lado, em uma WMN de comunidade digital, por exemplo, a topologia pode variar inesperadamente (por exemplo remoção de um AP *mesh*) aumentando ainda mais o desafio de se assegurar rotas ótimas na tabela de roteamento.

  Adicionalmente os recursos computacionais dos APs 802.11 tipicamente utilizados em WMNs são comumente mais limitados em comparação aos encontrados nos roteadores da Internet. Essa questão é ainda mais importante em WMNs, tendo se em vista as características de crescimento não planejado e do dinamismo da topologia. Por estes motivos alguns pesquisadores tem acreditado que a filosofia estado de enlace tenha chegado a seu limite para WMNs e que outras filosofias devem ser pensadas para elas, conforme citação abaixo traduzida de [5]:

  ...devido o constante crescimento das comunidades de redes em malha existentes e a característica inerente aos algoritmos de roteamento "estado de enlace" de recalcular toda a topologia do grafo (uma tarefa particularmente desafiadora para as capacidades computacionais relativamente limitadas de um roteador embarcado), os limites destes algoritmos tem se tornado um desafio. Recalcular toda a topologia do grafo em uma rede em malha com 450 roteadores leva vários segundos em um pequeno computador embarcado".

  Esse ruim desempenho de tempo pode afetar a rede consideravelmente, uma

vez que o cálculo das rotas é precedido pela remoção de todas as entradas da tabela de roteamento, conforme prescreve o padrão do OLSR. Assim, durante o procedimento de recálculo, todo tráfego de entrada é descartado pelo roteador.

O padrão do OLSR não contempla nenhum mecanismo para verificar se as modificações topológicas notificadas a partir das mensagens `HELLO` e `TC` necessariamente afetaram a tabela de roteamento. Além disso, caso uma modificação seja detectada, todas as rotas serão recalculadas, quer tenham sido afetadas pela mudança topológica quer não.

Neste trabalho nós investigamos o uso do modelo BIC como uma alternativa à proposta do OLSR para a questão do recálculo da tabela de roteamento em WMNs 802.11. A alternativa diz respeito tanto ao critério para se recalcular a tabela de roteamento quanto às características do algoritmo de recálculo propriamente dito, conforme melhor detalhado nas subseções 3.4.1 e 3.4.3. Antes disso porém, enunciaremos formalmente a questão do recálculo na seção 3.4.

3.4 O problema do caminho mínimo de origem única dinâmico

Se representarmos as informações topológicas do OLSR por meio de um *grafo dirigido* e ponderado com valores reais positivos, a questão da manutenção das rotas ótimas (i.e. a tabela de roteamento) em WMNs pode ser formalizado por meio do *problema do caminho mínimo de origem única dinâmico* (*Dynamic Single Source Shortest Paths Problem*, DSSSP). Neste problema, dado um grafo $G = (V, A)$ e um nó origem $s \in V$, estamos interessados em computar e manter os caminhos de menor peso da fonte s até cada um dos demais nós $u \in V - \{s\}$ após *atualizações incrementais* sobre A. O diferencial deste problema em relação ao clássico pro-

blema dos caminhos de menor peso de origem única (*Single Source Shortest Paths Problem*, SSSP) é que no primeiro a topologia do grafo pode sofrer mudanças ao longo do tempo o que, por sua vez, *pode* afetar os caminhos mínimos correntemente conhecidos.

As primeiras soluções especificamente projetadas para o DSSSP são conhecidas como *algoritmos incrementais*. Em geral tais soluções apresentavam algum tipo de limitação que inviabilizavam sua utilização prática. Em [49] os autores propõem uma solução cuja complexidade de tempo amortizada é $O(|V| \cdot C)$, onde C representa o valor natural máximo do peso de uma aresta (i.e. $\{e \in A | 1 \leq peso(e) \leq C\}$. Contudo o algoritmo não contempla operações de inserção nem atualização de pesos, antes lida somente com operações de remoção de arestas. Limitação similar é enfrentada pela solução proposta em [33], neste caso operações de inserção não são contempladas. Já as soluções propostas em [63; 45] situam-se dentre as primeiras propostas com suporte para todos os tipos de operações de atualização, contudo o custo de execução referente ao pior caso destes algoritmos é pior do que recalcular todas as rotas da rede do início.

As limitações acima apresentadas contribuíram para que o algoritmo de Dijkstra (comentado na subseção 3.4.2), originalmente projetado para o problema SSSP, fosse utilizado para resolver o DSSSP. Isso possivelmente é um dos fatores que influenciou os projetistas do OLSR a estabelecer a atualização topológica como critério suficiente para se recalcular toda a tabela de roteamento. O fato é que, nesse contexto, o algoritmo de Dijkstra recalcula os caminhos de menor custo para todos os vértices e, apesar de resolver o problema, pode subutilizar muitos recursos computacionais quando inserido no contexto das WMNs.

3.4.1 O modelo de computação incremental limitada

Um dos principais fatores que explica a dificuldade de se construir soluções práticas e eficientes para o DSSSP está no modelo de complexidade de custo inerente aos algoritmos projetados [50]. Neste modelo, o pior caso dos algoritmos é ainda definido em termos de todas as entidades que representam a topologia da rede, isto é, os conjuntos $|V|$ e $|A|$. Diante disso em [86] Ramalingam propôs o modelo de computação incremental de complexidade *restrita* ou modelo de computação incremental limitada (BIC).

Um algoritmo no modelo BIC para o DSSSP tem seu tempo de execução definido em termos da porção do grafo afetada pela alteração topológica e não necessariamente em termos de $|V|$ e $|A|$. Caso a mudança no grafo não leve à mudança nos caminhos mínimos correntemente conhecidos, nenhum recálculo é executado. Alternativamente, o modelo BIC define as seguintes entidades que são utilizadas para definir as medidas de custo dos algoritmos (explicadas no contexto DSSSP):

- δ : conjunto de vértices *afetados* pela alteração topológica. Em WMNs, um roteador é considerado afetado por uma alteração topológica se a rota à ele, até então ótima, aumenta de peso ou se ela é invalidada pela falha de um enlace ou um roteador intermediário;

- $\hat{\delta}$: $|\delta|$ mais o total de arestas que apontam para ao menos um vértice afetado v, i.e. $v \in \delta$. Varia de 1 até $|A| + |V|$.

Na seção 3.4.3 apresentaremos o algoritmo de Ramalingam e Reps (RRs) onde a filosofia intrínseca ao modelo BIC pode ser melhor compreendida.

3.4.2 O algoritmo de Dijkstra

O algoritmo de Dijkstra foi originalmente proposto para resolver o SSSP [43]. Ele é uma generalização do algoritmo da *busca em largura* (*Breadth First Search*, BFS) que calcula as menores distâncias de uma origem s até cada um dos destinos $u \in V - \{s\}$ no grafo $G = (V, A)$.

A suposição básica do BFS é que o grafo não é valorado (i.e. as arestas possuem pesos iguais). Tal suposição é alcançada quando se utiliza a métrica *hop count*. Conforme previamente discutido, à época da evolução e escrita do padrão do OLSR, as MANETs eram o principal foco do padrão. Com o advento das WMNs e das métricas dinâmicas de roteamento, muitas implementações independentes substituíram o BFS pelo algoritmo de Dijkstra a fim de lidar com grafos valorados. Seguindo tal tendência, o grupo IETF MANET, através da linha PMP, tem focalizado seus esforços para o padrão emergente OLSR versão 2 (OLSR2) que, apesar de não recomendar nenhuma métrica dinâmica, reconhece a relevância delas. Prova disso é o fato de que em todos os (até então) nove *drafts* do OLSR2, a definição das mensagens de controle tem sido deixadas a encargo da RFC 5444 [36; 37].

Diferentemente do padrão do OLSR, nos *drafts* do OLSR2 não é estabelecido nenhum algoritmo para o cálculo das rotas, apenas um exemplo baseado no algoritmo de Dijkstra é fornecido. Adicionalmente, os conjuntos do OLSR2 cujas alterações demandam a execução do algoritmo de recálculo são essencialmente os mesmos do OLSR (conferir seção 3.2). Esse procedimento poderia se restringir unicamente aos casos em que a própria tabela de roteamento fosse alterada (não necessariamente uma parte da topologia), o que é possível com o algoritmo a ser apresentado na seção 3.4.3.

O algoritmo de Dijkstra é ilustrado no algoritmo 3.1 A principal suposição do algoritmo de Dijkstra é que não há ciclos com peso negativo, o que não constitui

```
Dijkstra(G = (V, A),s)
01  para cada vértice v ∈ V faça
02    d[v] ⟵ ∞
03    π[v] ⟵ NIL
04  d[s] ⟵ 0;
05  S ⟵ {};
06  Constrói_Fila(Q, V);
07  enquanto Q ≠ ∅ faça
08    u ⟵ Extrái_Mínimo(Q);
09    S ⟵ S ∪ {u};
10    para cada vértice v ∈ Suc_A[u] faça
11      Relaxa(u → v,peso(u → v),Q);
```

Algoritmo 3.1: O Algoritmo de Dijkstra

um problema para o caso das WMNs. Para cada vértice v, o algoritmo mantém duas variáveis, a saber, uma estimativa $d[v]$, que indica o custo do melhor caminho correntemente conhecido de s até v, e o antecessor imediato $\pi[v]$ de v na árvore dos caminhos mínimos. Para a origem essa estimativa é sempre zero (linha 04) enquanto que para qualquer outro vértice v tem-se $d[v] = \infty$ e $\pi[v] = nil$ (linhas 01 a 03). Esses últimos valores serão atualizados durante a execução do algoritmo *à medida em que melhores caminhos são encontrados*, conforme veremos mais adiante. Uma *fila de prioridades* Q é *construída* para os vértices, usando $d[v]$ como chave (linha 06). Em seguida um laço de repetição proporcional à $|V|$ é executado. A cada iteração, o vértice u com melhor estimativa é *extraído* de Q e adicionado ao conjunto S de vértices já processados (linhas 08 e 09, respectivamente).

Por fim, o algoritmo tenta melhorar as estimativa $d[v]$ para cada vértice v que é um sucessor imediato de u (i.e. $Suc_A[u] = \{v \mid \forall v \in V,\, u \rightarrow v \in A\}$) (linha 10). Essa etapa é realizada com o auxílio do procedimento de *relaxamento* de arestas, ilustrado no algoritmo 3.2. Nele é verificado se o caminho partindo da origem s até v por meio do vértice u é melhor do que o atualmente conhecido. Em caso afirmativo, os valores $d[v]$ e $p[v]$ referentes ao vértice v são atualizados e uma operação de diminuição de chave deve ser realizada sobre v na lista de prioridades Q.

Tendo em vista os próximos algoritmos, optamos também por adicionar a informação $\pi[v]$ anexa à cada v na fila Q. Assim, se x é um vértice extraído da fila Q pela operação `Extrai_Mínimo(`Q`)`, então as informações $\pi[x]$ e $d[x]$ são retornadas a um tempo constante pelas operações $pai_Q[x]$ e $chave_Q[x]$, respectivamente. Adicionalmente, a adição do vértice v em Q, com prioridade $d[v]$ e predecessor $\pi[v]$ é realizada pelo comando `Insere_Elemento(`v`,` $\pi[v]$`,` Q`,` $d[v]$`)`.

Relaxa($u \rightarrow v$, $peso_{u \rightarrow v}$, Q)
01 `se` $d[v] > d[u] + peso_{u \rightarrow v}$ `então`
02 $d[v] \longleftarrow d[u] + peso_{u \rightarrow v}$;
03 $\pi[v] \longleftarrow u$;
04 `Diminui_Chave(`v`,`$\pi[v]$`,`Q`,`$d[v]$`)`;

Algoritmo 3.2: O procedimento de relaxamento de uma aresta $u \rightarrow v$

As operações estratégicas para a definição da complexidade de pior caso do algoritmo de Dijkstra são as operações sobre os vértices. A operação de extração (linha 08) é executada $|V|$ vezes. Já o laço mais interno é executado $O(|A|)$ vezes ao todo (linha 10), uma vez que as arestas adjacentes a u não serão mais acessadas depois que ele for adicionado ao conjunto S. Por fim, a definição do pior caso do algoritmo de Dijkstra depende do custo das operações de extração de atualização sobre Q, as quais são dependentes da forma como a fila é implementada. A tabela 3.1 compara complexidade das principais operações de filas de diferentes tipos de implementações.

Implementação de Q	`Extrái_Mínimo`	`Diminui_Chave`	`Insere_Elemento`
Vetor ordenado	$O(\|V\|)$	$O(1)$	$O(\|V\|)$
Heap binário	$O(\log(\|V\|))$	$O(\log(\|V\|))$	$O(\log(\|V\|))$
Heap de Fibonacci	$\widetilde{O}(\log(\|V\|))$	$\widetilde{O}(1)$	$O(1)$

Tabela 3.1: Complexidade de operações básicas de diferentes implementações de filas de prioridades

Como a operação de extração é executada $\Theta(|V|)$ vezes, a utilização de vetores

ordenados resulta na pior alternativa para o algoritmo de Dijkstra, a saber, $O(|V|^2)$.

Para uma implementacão da fila de prioridades por meio de *heap binário*, tanto a operação de extração como a de atualização sobre Q custam $O(\log(|V|))$ *no pior caso*, o que levaria o algoritmo de Dijkstra a assumir uma complexidade de $O((|A|+|V|)\log(|V|)) = O(|A| \cdot \log(|V|))$, no pior caso.

Em um *heap de fibonacci* a complexidade de tempo *amortizada* (e não necessariamente o pior caso) para as operações de extração e atualizacão da fila são $\widetilde{O}(\log(|V|))$ e $\widetilde{O}(1)$, respectivamente. Esse tempo amortizado é alcançado se, por exemplo, a quantidade de execuções da operação de `Diminui_Chave` for assintoticamente maior do que o total de execuções da operação de `Extrái_Mínimo`. Neste caso, podemos afirmar que a complexidade amortizada do algoritmo de Dijkstra com *heaps de fibonacci* é de $\widetilde{O}(E+V \cdot \log(|V|)) = \widetilde{O}(V \cdot \log(|V|))$. Em caso contrário, tal estrutura pode conduzir a uma complexidade pior do que àquela obtida pelo heap binário.

Para um discussão mais aprofundada do algoritmo de Dijkstra incluindo sua correção, consultar [43; 39].

3.4.3 O algoritmo de Ramalingam & Reps

O algoritmo de Ramalingam & Reps (doravante RRs) é conhecido como o primeiro algoritmo no modelo BIC, uma vez que foi proposto pelos mesmo autores por ocasião da formalização do referido modelo [86; 87]. O algoritmo oferece suporte para recalcular os caminhos de menor custo após operações de inserção e de remoção de arestas e, a partir destes, também de diminuição e aumento de peso.

A partir da publicação do algoritmo RRs outras propostas têm surgido a fim de seguir tal tendência. Em [51] foi proposto um algoritmo com complexidade de tempo $\mathrm{O}(\sqrt{|A|} \cdot |\delta| \cdot \log(|V|))$ para as referidas operações, contudo, a rigor teórico, enten-

demos que tal algoritmo não está completamente assentado no modelo BIC, uma vez que sua complexidade apresenta os termos $|A|$ e $|V|$. Por sua vez, na proposta de Katriel & Pascal em [61], um algoritmo no modelo BIC é proposto. Contudo, à época de investigação deste trabalho, tal algoritmo encontrava-se sob apreciação da comunidade científica. Por duas vezes tentamos contatar o *corresponding author* a fim de esclarecer uma dúvida (apêndice A.1) porém não obtivemos resposta. De qualquer forma as melhorias alegadas pelos autores em relação ao RRs são significativas apenas para variações do DSSSP que não são relevantes para o escopo das WMNs (por exemplo ciclos de custo zero).

Com base nas informações acima mencionadas, selecionamos o RRs como representante do modelo BIC a fim de realizarmos uma avaliação comparativa com o algoritmo de Dijkstra no caso médio.

Cabe ressaltar que um dos principais aspectos deste trabalho também é estudar o dinamismo topológico de *backbones* WMN e seu impacto na tabela de roteamento. Em particular, além de demonstrarmos por meio de um estudo de caso uma alternativa competitiva à estabelecida no padrão do IETF, realizaremos também um estudo sobre as características estocásticas de $|\delta|$ em WMNs. Assim, as análises e conclusões dele decorrentes são perfeitamente aplicáveis à qualquer algoritmo no modelo BIC sem perda de generalidade.

Os procedimentos de inserção e remoção de arestas do algoritmo RRs mostrados a seguir tem por base a estrutura $G_d = (V, A')$ denominada a *projeção de* $G = (V, A)$. G_d é um sub*grafo dirigido acíclico* (*Directed Acyclic Graph*, DAG) de G. Independente do tipo da atualização notificada o algoritmo RRs mantém a propriedade de que A' contém todas as arestas que fazem parte de algum caminho mínimo sobre G. Para isso ambos os procedimentos compartilham o algoritmo básico `RecalculaCaminhosMínimo`, ilustrado no algoritmo 3.3, sendo diferenciados apenas na forma como eles inicialmente definem a fila inicial Q de vértices afetados.

```
RecalculaCaminhosMínimos(G, G_d, Q)
01  enquanto Q ≠ ∅ faça
02    x ⟵ Extrai_Minimo(Q);
03    d[x] ⟵ chave_Q[x];
04    π[x] ⟵ pai_Q[x];
05    para cada aresta w → x ∈ A' faça
06      A' ⟵ A' − {w → y};
07    A' ⟵ A' ∪ {π[x] → x};
08    para cada vértice y ∈ Suc_A[x] faça
09        se y ∈ Q então
10          Relaxa(x → y, peso(x → y),Q)
11        senão
12          se d[y] > d[x] + peso(x → y) então
13            π[y] ⟵ x;
14            d[y] ⟵ d[x] + peso(x → y);
15            Insere_Elemento(y, π[y], Q, d[y]);
```

Algoritmo 3.3: Algoritmo RRs: procedimento básico para atualização dos caminhos mínimos

O algoritmo `RecalculaCaminhosMínimos` é uma variação do algoritmo de Dijkstra (3.1) e, na discussão que segue-se, assumimos que os conjuntos básicos $d[]$ e $\pi[]$ foram previamente computados de forma similar à realizada pelo algoritmo de Dijkstra. A cada atualização o algoritmo `RecalculaCaminhosMínimos` assume que os valores $d[x]$ e $\pi[x]$ correspondentes a qualquer vértice x inicialmente presente na fila Q estão obsoletos. Tais valores serão respectivamente substituídos pelos correspondentes $chave_Q[x]$ e $pai_Q[x]$, nos instantes subsequentes à execução da operação `Extrai_Mínimo`(Q) (linhas 02 a 04). O algoritmo não limita-se aos vértices inicialmente passados em Q mas também verifica se seus sucessores foram indiretamente afetados e precisam atualizar as informações de caminhos mínimos (linha 08). Se tal necessidade for confirmada e o vértice em questão já estiver na fila então ele será submetido ao relaxamento (conferir algoritmo 3.2). Do contrário ele será adicionado à fila com a estimativa que indica sua possibilidade de melhoria (linhas 11 a 15).

Quanto à complexidade de tempo de pior caso o procedimento `RRsInsereAresta` é assintoticamente dominado pela rotina `RecalculaCaminhosMínimos`. Neste caso

podemos utilizar o mesmo raciocínio empreendido na subseção 3.4.2 para analisar o algoritmo de Dijkstra. Desta forma a referida complexidade é de $\widetilde{O}(|\hat{\delta}| + |\delta| \log |\delta|)$ utilizando-se *heaps* de Fibonnaci.

Nas próximas subseções verificaremos como os procedimentos de inserção e remoção constroem a fila de prioridades Q de vértices afetados iniciais antes de invocar o procedimento básico.

3.4.3.1 O procedimento de inserção de uma nova aresta $u \rightarrow v$

O procedimento básico do algoritmo RRs para inserção de uma nova aresta $u \rightarrow v$ no grafo G é ilustrado no algoritmo 3.4. Inicialmente o procedimento verifica se é possível melhorar o caminho mínimo de s até v passando pela aresta $u \rightarrow v$ recém inserida. Em caso negativo o algoritmo é imediatamente finalizado (linhas 02 e 03). Por outro lado, se for possível obter a melhoria, o vértice v é *adicionado* a uma fila Q com prioridade correspondente à estimativa $d[u]+peso(u \rightarrow v)$. Em seguida, na linha 06, o procedimento de inserção invocará o algoritmo `RecalculaCaminhosMínimos`, ilustrado no algoritmo 3.3.

```
RRsInsereAresta(V, A, A', u → v)
01  A ⟵ A ∪ {u → v};
02  se d[v] ≤ d[u] + peso(u → v) então
03    finalize_RRsInsereAresta;
04  Q = {};
05  Insere_Elemento(v, u, Q, d[u] + peso(u → v));
06  RecalculaCaminhosMínimos(G, G_d, Q)
```

Algoritmo 3.4: Algoritmo RRs: inserção de uma nova aresta $u \rightarrow v$ em um grafo G

A operação de inserção de uma aresta pode ser ligeiramente modificada para dar origem a operação de diminuição de peso de aresta $u \rightarrow v$. Basicamente basta inserir uma aresta paralela à $u \rightarrow' v$ que tem o novo menor peso. O próprio procedimento se encarregará de substituir as arestas em G_d.

Quanto à complexidade de tempo de pior caso o procedimento `RRsInsereAresta` é assintoticamente dominado pela rotina `RecalculaCaminhosMínimos`. Neste caso podemos utilizar o mesmo raciocínio empreendido na subseção 3.4.2 para analisar o algoritmo de Dijkstra. Desta forma a referida complexidade é de $\widetilde{O}(|\delta| + |\delta| \log |\delta|)$ utilizando-se *heaps* de Fibonnaci.

3.4.3.2 O procedimento de remoção de uma aresta $u \rightarrow v \in A$

O procedimento do algoritmo RRs para remoção de uma aresta $u \rightarrow v \in A$ é ilustrado no algoritmo 3.5. Inicialmente o algoritmo finaliza se nenhum novo caminho mínimo precisar ser recalculado, i.e. a aresta removida não integrava um caminho mínimo até v ou, se integrava, não era o único (linhas 03 a 07). Caso contrário o algoritmo 3.6 será acionado para construir o conjunto $lista\_afetados$ de vértices afetados (linha 08). Nesse procedimento, o primeiro vértice afetado v é adicionado à lista. Após isso, todos os seus descendentes exclusivos em G_d também são incluídos na lista. Nesse processo as arestas utilizadas para integrar os caminhos mínimos entre v e seus descendentes são removidas de A'.

```
RRsRemoveAresta(V, A, A', u → v)
01  Q = {};
02  A ⟵ A − {u → v};
03  se u → v ∉ A' então
04    finalize_RRsRemoveAresta;
05  A' ⟵ A' − {u → v};
06  se Pred_A'(v) ≠ ∅ então
07    finalize_RRsRemoveAresta;
08  lista_afetados ⟵ RRsMarcaAfetados(G_d, u → v);
09  RRsEstimaNovosPais(G, G_d, lista_afetados, Q, u → v);
10  RRsRecalculaCaminhosMínimos(G, G_d, Q);
```

Algoritmo 3.5: Algoritmo RRs: remoção de uma aresta $u \rightarrow v \in A$

A lista de vértices afetados representa, em outras palavras, o conjunto de vértices para os quais não se conhece um antecessor *correto* π'. Neste contexto, o termo "correto" designa um vértice antecessor que pode ser alcançado pela origem s por meio

```
RRsMarcaAfetados(G_d, u → v)
01  lista_auxiliar ⟵ {v};
02  lista_afetados ⟵ {};
03  enquanto lista_auxiliar ≠ ∅ faça
04    v ⟵ retira_último(lista_auxiliar);
05    acrescenta(lista_afetados, v);
06    para cada vértice w ∈ Suc_A'(v) do
07      A' ⟵ A' − {v → w};
08      se Pred_A'(w) = ∅ então
09        acrescenta(lista_auxiliar,w);
10  return lista_afetados;
```

Algoritmo 3.6: Algoritmo RRs: primeira subrotina de remoção: marcação de vértices inicialmente afetados

de um caminho ótimo de custo d'. Desta forma, após a computação da lista de afetados, para cada vértice x nela presente, o procedimento de remoção `RRsRemoveAresta` *procurará* por um novo pai π'_x utilizando o algoritmo 3.7 (linha 09). Os candidatos a tal posição são todos os vértices antecessores imediatos de x em $G = (V, A)$ (i.e. o conjunto $Pred_A(x)$) que não foram afetados pela remoção (i.e. não pertença à lista de vértices afetados). *Dentre* eles, será selecionado aquele por quem x possa ser alcançado a partir de s com a melhor *estimativa* de peso de caminho, isto é, $d'_x = min\{d[y] + peso(y, x), \forall y \in Pred_A[x] \mid y \notin lista\_afetados\}$.

```
RRsEstimaNovosPais(G, G_d, lista_afetados, Q, v)
// Procura melhor pai π[x] não afetado para cada vértice x afetado
01  enquanto lista_afetados ≠ {} faça
02    x ⟵ retira_último(lista_afetados);
03    d'_x ⟵ ∞;
04    π'_x ⟵ NIL;
05    para cada vértice y ∈ Pred_A[x] faça
06       se y ∉ lista_afetados então
07         se d'_x > d[y]+peso(y → x) então
08           d'_x ⟵ d[y]+peso(y → x);
08           π'_x ⟵ π[y];
10     se d'_x < ∞ então
11       Insere_Elemento(x, π'_x, Q, d'_x);
```

Algoritmo 3.7: Algoritmo RRs: segunda subrotina de remoção: estimação inicial de novos antecessores imediatos para os vértices afetados

Entretanto a seleção acima descrita não é suficiente para garantir que os novos caminhos ótimos estejam recalculados aos destinos afetados. Um exemplo simples que ilustra tal situação é o caso em que um vértice afetado w, *após* selecionar seu novo pai π'_w, passa a ter a melhor *estimativa* d'_x para um outro vértice afetado x (i.e. $\exists w \rightarrow x \in A \mid d'_w + peso(w,x) < d'_x$). Contudo, o fato de w ser afetado impede que ele seja selecionado por x. Apesar de este ter sido um exemplo aparentemente simples, a necessidade de se atualizar π'_x pode vir a ser recorrente, cenário esse perfeitamente contemplado pelo algoritmo de Dijkstra por meio do relaxamento. Por este motivo, cada vértice afetado x é armazenado em uma fila Q com prioridade igual a d'_x após ter selecionado seu pai π'_x. Por fim, o procedimento *RRsRemoveAresta*, invoca o algoritmo *RecalculaCaminhosMínimos* (linha 10) para assegurar que a estimativa $d[v]$ de cada vértice afetado corresponda, efetivamente, ao melhor caminho possível.

A figura 3.7 ilustra as etapas do algoritmo `RRsRemoveAresta` para recalcular os caminhos mínimos do grafo $G = (V, A)$ após a remoção de uma aresta.

O diagrama à esquerda na figura 3.7 representa G antes da alteração topológica e com os valores $d[]$ e $\pi[]$ devidamente computados para cada vértice. O diagrama do meio na mesma figura ilustra a situação intermediária de G após a remoção da aresta $s \rightarrow a$ e a execução das sub-rotinas `RRsMarcaAfetados` e `RRsEstimaNovosPais`. Após a execução da primeira sub-rotina, os vértices afetados pela alteração topológica são *marcados* (ilustrado em tons de cinza). A segunda sub-rotina, por sua vez, procura por um pai *correto* (possivelmente temporário) π' para cada vértice marcado como afetado. Para $\pi'[b]$ há somente os vértices candidatos a e o origem s, enquanto que $\pi'[a]$ dispõe dos vértices b e x, este último não é afetado pela mudança topológica e é passível de ser alcançado a partir de s com custo $d[x] = 3$. O fato dos vértices a e b serem afetados (i.e. não serem "corretos") leva o algoritmo a escolher $\pi'[a] = x$ e $\pi'[b] = s$. A esta altura já é possível alcançar os vértices afetados a partir da origem s, contudo não necessariamente por meio dos melhores caminhos. Note

que após b receber s como antecessor correto ele pode ser utilizado como antecessor de a para fornecer um caminho ainda melhor que aquele fornecido por x. Por este motivo, cada vértice afetado v é inserido em uma fila Q com prioridade $d'[v]$ (e antecessor correto $\pi'[v]$) e, por fim, o procedimento `RRsRecalculaCaminhosMinimos` é executado. Em nosso exemplo, o resultado dessa última etapa sobre G é ilustrado no diagrama à direita da figura 3.7.

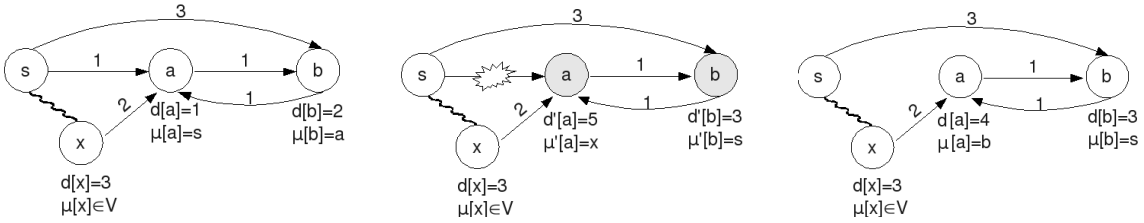

Figura 3.7: Ilustração da sequência de etapas do algoritmo `RRsRemoveAresta`

De forma análoga à relação entre o procedimento de inserção e o de diminuição, o procedimento de remoção acima descrito pode ser adaptado para lidar com o aumento dos pesos de uma aresta. Nesta caso a aresta não será removida do grafo G (linha 02) em vez disso será incrementada e, se for o caso, será removida de G_d e os caminhos recalculados (linhas 03 a 10).

Capítulo 4

Metodologia

No capítulo anterior mencionamos que o problema central de nosso trabalho consiste em investigar o uso da abordagem BIC como uma alternativa à solução prescrita no padrão do OLSR para manutenção do roteamento global em WMNs. Assim sendo, no presente capítulo propomos uma metodologia para *estimar* o caso médio (i.e. a média de execução das operações chaves) dos algoritmos RRs e Dijkstra no contexto das WMNs.

Adicionalmente, dado que em um algoritmo no modelo BIC é possível checar se mudanças no grafo implicaram em mudanças na tabela de roteamento, também propomos uma metodologia para estimar a proporção de tal implicação em WMNs.

4.1 Apresentação geral da metodologia

A figura 4.1 apresenta um esquema geral da nossa metodologia para fundamentar nossa análise comparativa entre a abordagem do padrão OLSR (algoritmo de Dijkstra) e a abordagem BIC (Ramalingam & Reps).

O bloco nomeado "topologia dinâmica" representa a "distribuição de probabilidades" requerida em um estudo de caso médio. Ele é responsável por representar

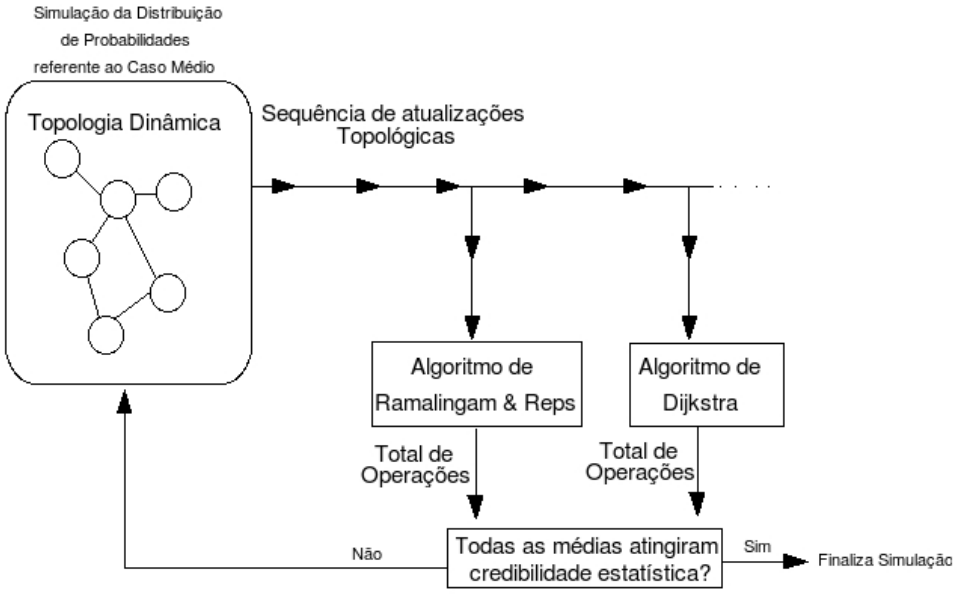

Figura 4.1: Esquema geral da metodologia de avaliação de caso médio proposta

a dinâmica da rede ao longo do tempo. Em nosso caso, o termo "distribuição de probabilidades" representa o resultado da modelagem de diversos componentes heterogêneos, a saber, a modelagem do canal sem fio 802.11 (discutido na seção 2.3), a configuração do protocolo de roteamento e a configuração de transmissão dos NICs 802.11 e suas distâncias (discutidos na seção 4.2).

```
n 5
a 0 1 1667
a 0 2 5000
a 1 3 5000
a 1 2 5000
a 1 3 5000
a 1 2 5000
c 2 2 2
r 0 2 10000
c 0 0 0
```

Figura 4.2: Exemplo de sequência de atualizações topológicas resultado da simulação da distribuição de probabilidades proposta

Quando submetida a um processo de *simulação*, a distribuição de probabilidades

gera uma sequência de atualizações topológicas correspondentes à visão que um roteador em particular tem da WMN simulada. A figura 4.2 ilustra um exemplo de tal sequência. Nela a quantidade de roteadores da rede é indicada na primeira linha (por exemplo `n 5`). As inserções/atualizações são indicadas por um prefixo `a` seguido dos identificadores de origem e destino dos vértices na aresta em questão, com o respectivo peso. (por exemplo a inserção/atualização da aresta $0 \rightarrow 1$ de peso 1667 é indicada por: `a 0 1 1667`). Nos casos de atualizações devido a chegada de novas mensagens OLSR, o prefixo `c` indica a quantidade de enlaces dos conjuntos *Neighbor*, *2-hop Neighbor* e TC, respectivamente, utilizadas pelo algoritmo de melhor caminho para compor o grafo de representação da rede. A remoção de um enlace $0 \rightarrow 2$ de peso 10000, é representada por `r 0 2 10000`.

A ferramenta de avaliação de desempenho que adotamos para simular a distribuição foi a *simulação de horizonte infinito* baseada em controle estatístico. Optamos por tal abordagem por dois motivos básicos. O primeiro é "*esconder a complexidade*" da distribuição de probabilidades resultante. O segundo é prover credibilidade à nossa *estimativa* da "média de todas as entradas de tamanho n" requisitadas pela análise de caso médio. Na seção 4.3, discutiremos os aspectos necessários à simulação para que tal credibilidade possa ser alcançada.

A cada amostra de atualização topológica de entrada, totalizamos as operações dos algoritmos de Dijkstra e Ramalingam & Reps executadas para recalcular as rotas ótimas. As operações assintoticamente mais relevantes por nós contabilizadas foram: as operações sobre vértices (`Insere_Elemento`, `Diminui_Chave`, `Extrai_Mínimo`) e o total de arestas examinadas (o que, em geral, corresponde ao tamanho dos laços dos algoritmos). Optamos por contar as operações para que nossas análises fossem independente de plataforma, especialmente pelo fato de que em WMNs é comum se ter diferentes arquiteturas (por exemplo MIPS, ARM, x86) em uma mesma rede.

A implementação dos algoritmos em linguagem C foi derivada de modificações

no pacote *Dynamic All Pairs Shortest Path Algorithms* (dsp), disponibilizada em [9]. Tais modificações podem ser reproduzidas utilizando os passos e o *patch* que disponibilizamos em [3].

4.2 Configuração geral das simulações

Para simular a topologia da WMN utilizamos o simulador *Network Simulator* 2 (ns-2), versão 2.29 cujo código fonte pode ser obtido gratuitamente em [15].

Em nossas simulações, os nós foram modelados como roteadores 802.11b, equipados com antenas omni-direcionais com ganho de 2 dBi (*decibel isotropic*) e com NICs configuradas em uma potência de transmissão de 21 dBm. Utilizamos o ns-2 modificado para prover as seguintes extensões: modelo de interferência cumulativa, algoritmo *Adaptive Auto Rate Fallback* (AARF) para seleção automática de taxas de transmissão, modelo baseado em BER para recepção de sinal com as seguintes sensibilidades de recepção (em dBm, referentes ao NIC Intersil HFA3861B [59]): -94 para BPSK, -91 para QPSK, -87 para CCK5.5 e -82 para CCK11.

Utilizamos também uma série de correções na camada MAC 802.11 do ns-2 propostas por Schmidt-Eisenlohr et al. em [90]. Para mais detalhes sobre as demais extensões acima mencionadas, consultar [19] e [25].

Os nós foram *geograficamente dispostos* em formato de grade, em uma área de 10000×10000. Foram simuladas as seguintes quantidades de nós com suas respectivas configurações de grade: 5 (5×1), 10 (5×2), 15 (5×3), 20 (5×4), 25 (5×5), 30 (5×6), 35 (5×7), 40 (5×8), 45 (5×9) e 50 (5×10).

A partir de medições realizadas entre dois roteadores em malha verificamos uma taxa de perda de pacotes estabilizada em $< 5\%$, a uma distância de ≈ 25 metros e em um ambiente *indoor* $\leftrightarrow$ *indoor* (detalhes em [71]). Com base nessa informação, *estimamos* o valor da variável *RXThreshold* em $1.84952e^{-12}$ e posicionamos os ro-

teadores na grade a uma distância horizontal e vertical de 25 metros. Desta forma, considerando todos os cenários simulados, verificamos que a quantidade de vizinhos de um roteador de extremidade oscilou entre 2 e 6, enquanto que para roteadores de núcleo essa variação foi de 6 a 12. Isso concorda com os resultados obtidos por Xue & Kumar [97] da seguinte forma. Tais autores verificaram que cada roteador deve estar conectado a $\Theta(\log(|V|))$ outros roteadores da rede a fim de assegurar um compromisso entre o uso do canal (interferência) e a alcançabilidade de todos os nós da rede por meio de múltiplos saltos. Para cada um de nossos cenários, é possível encontrar uma constante c que, multiplicada por $\log(|V|)$ e ajustada para um valor inteiro, resulta no total de roteadores vizinhos previamente mencionados.

Em nossas simulações o modelo de propagação utilizado foi o log-normal *Shadowing* com expoente de degradação $\alpha = 4.0$ e desvio padrão $\sigma = 6.0$. Dada a relevância do modelo de propagação em um estudo de redes 802.11, conforme reiteramos em [85; 75], apresentamos a seguir uma discussão que reflete nossa decisão de modelagem.

4.2.1 Justificativa da escolha do modelo de propagação

A decisão acerca do modelo de propagação é algo muito particular a cada projeto de experimento. Para o nosso caso, verificamos que o modelo *Shadowing* foi o que melhor contribuiu para ampliar a generalidade de nossos resultados. A seguir apresentamos uma discussão sobre tal escolha em relação aos outros modelos apresentados.

Kotz et. al. [65] constatou que os modelos Free-Space e Two-Ray Ground são demasiadamente simplistas para modelar WMNs 802.11 típicas. Tais modelos são incapazes de reproduzir enlaces assimétricos, por exemplo. Além disso eles consideram que a zona de comunicação de um nó é homogeneamente circular e

dependente somente da distância, o que comprovadamente não ocorre.

A idéia geral dos modelos baseado em traçado de raios não consiste em realizar uma aproximação estatística da perda do sinal mas antes computar seu valor exato em cada ponto do espaço, considerando a interação do sinal com cada objeto incluído no modelo. Com base em nossas referências mencionadas nessa seção, entendemos que para nossos propósitos tal característica é um fator limitante, uma vez que os resultados são encerrados a um cenário específico. Por outro lado, em um modelo estocástico como o *Shadowing*, diferentes cenários podem ser contemplados, desde que certas características estatísticas sejam satisfeitas conforme discutiremos a seguir.

Por ser um modelo parametrizável e estocástico *em princípio* qualquer ambiente poderia ser modelado pelo *Shadowing*. O valor de α pode ser obtido do coeficiente linear da reta resultante da aplicação de uma regressão linear sobre medições $(d, PL(d))$. Contudo o modelo só é considerado aceitável se o desvio padrão inerente ao processo satisfazer $\sigma < 8$. Neste sentido, uma quantidade crescente de trabalhos de medição tem verificado tais condições para canais 802.11 em ambientes característicos às WMNs.

Em [28], Akl et al verificou a aplicabilidade do modelo *Shadowing* para APs 802.11a/b/g de diferentes fabricantes em cenários típicos de prédios civis. Em [47], Faria et al verificou a aplicabilidade do referido modelo tanto para a comunicação entre NICs 802.11 no contexto *indoor*↔*indoor*, como no contexto *outdoor*↔*indoor*. Tummala [95] apresenta de forma mais detalhada a aplicabilidade do *Shadowing* para modelar sistemas 802.11. Adicionalmente uma discussão com base em diversos resultados práticos do modelo log-normal *Shadowing* é realizada por Rappaport em [88].

Em nossos estudos iniciais, consideramos também o fenômeno do desvanecimento por múltiplos percursos por meio do modelo de Rice em adição ao modelo *Shadowing*,

de acordo com [70]. Através de um estudo de caso baseado em VoIP verificamos que essa adição resultou em um impacto demasiadamente pessimista nas camadas superiores [85; 75], mesmo se comparado aos resultados que obtivemos em um ambiente propício ao fenômeno da reflexão [71]. Em face disso pesquisamos a literatura especializada e constatamos que esse fenômeno não representa um problema significativo em redes 802.11 uma vez que sua operação está baseada em uma largura de banda relativamente grande (22 Mhz) [91]. Em [27], os autores levantaram a possibilidade de que o *multipath fading* seja a causa do comportamento imprevisível dos canais em uma WMNs baseado em 802.11b. Contudo, em [54], os autores provêem "fortes evidências" para refutar essa hipótese. Eles mostram que "o fenômeno da interferência externa, e não o de múltiplos percursos" é responsável por aquele comportamento.

Por fim, ressaltamos que a questão da propagação está longe de ser encerrada em um modelo específico, ou mesmo em uma classe deles. Mesmo as conclusões obtidas por trabalhos realizados em sistemas sem fio reais, a rigor científico, estão limitadas às condições de propagação de tal sistema, a menos que um modelo de propagação seja derivado a partir dele e o comportamento de propagação de outras redes possam ser aproximadas por aquele modelo. Assim, compete a cada trabalho investigar uma configuração de modelagem considerada razoável (i.e. factível de ocorrer em sistemas sem fio independentes) para as características do sistema que se está querendo avaliar.

4.2.2 Ajuste da configuração do OLSR

Para completar a modelagem de nossa "distribuição de probabilidades" precisamos definir a configuração do OLSR, uma vez que a visão da topologia disponível para os algoritmos é resultado da atividade de conhecimento topológico do OLSR

sobre os canais sem fio, conforme explanado na seção 3.1.

Para o ns-2 utilizamos a implementação do OLSR disponível em [23] com as extensões de métricas dinâmicas propostas em [16]. Verificamos que, por descuido do desenvolvedor, a implementação apresentava uma incompatibilidade com o padrão do OLSR em relação ao processo de atualização da tabela de roteamento. Assim, após o contatarmos, corrigimos tal problema (ver apêndice A.2). Adicionalmente detectamos um problema de gerenciamento de memória na extensão de métricas dinâmica que inviabilizava nossas simulações de horizonte infinito. Desta vez o problema foi corrigido pelo desenvolvedor após nosso contato (ver apêndice A.3).

Por meio de estudos de caso baseado em VoIP, pudemos comprovar certas vulnerabilidades no processo de descoberta de topologia empreendido pelo padrão do OLSR. Nos certificamos por meio de simulações que *instabilidades em enlaces para roteadores MPRs prejudicam a divulgação de informações de topologia*, o que compromete seriamente a consistência das informações disponíveis para o algoritmo de cálculo do melhor caminho, conforme reportamos em [76; 84]. Esses resultados concordam com experimentos realizados em redes reais, conforme reportado pelo grupo OLSR-NG em [22].

A fim de conferir uma maior confiabilidade no processo de construção topológica do OLSR e oferecer informações mais consistentes para o algoritmo de cálculo de rotas, ativamos a difusão tradicional, onde todos os roteadores são nós MPRs, conforme sugerido pela vasta experiência prática dos mantenedores do projeto OLSR-NG [22]. Para controlar uma possível sobrecarga de mensagens de controle, utilizamos o algoritmo *fish-eye* [80]. A idéia básica é utilizar diferentes valores de TTL para que cada roteador envie mensagens mensagem `TC` com mais frequência para seus vizinhos e com menos para destinos remotos da rede. Conforme demonstrado pelos autores do algoritmo, esse mecanismo aumenta a confiabilidade das rotas divulgadas ao mesmo tempo que diminui a sobrecarga da rede.

Para validar a melhoria que esse conjunto de configurações proporciona em comparação à configuração padrão do OLSR, realizamos mais um estudo de caso VoIP. Os resultados indicaram que o primeiro conjunto de configurações conferiu as informações topológicas mais consistentes para apoiar a construção das tabelas de roteamento. Em particular, o número de perdas de pacotes por rotas obsoletas caiu drasticamente, resultando assim em uma melhoria de 84% para a qualidade do serviço [44]. Esta otimização da atividade de difusão topológica do OLSR beneficia nossas conclusões sobre o total de operações dos algoritmos, uma vez que elas serão baseadas em representações (grafos) bem mais consistentes com a realidade da topologia da rede.

Por fim, também avaliamos o comportamento dos algoritmos sob duas diferentes métricas de roteamento. Conforme amplamente discutido ao longo deste trabalho, o comportamento do canal define a topologia da rede o que, consequentemente, afeta o desempenho médio dos algoritmos que iremos avaliar. Contudo, sob a ótica dos algoritmos, esse comportamento é refletido indiretamente por meio das métricas de roteamento. Por esse motivo optamos por variar a métrica de roteamento em nossas simulações. As métricas escolhidas foram a ETX, devido sua relevância prática, e a LD. Esta última escolha deve-se principalmente ao fato que a métrica LD apresenta uma filosofia completamente diferente da ETX, com isso poderemos estimar se tal mudança de configuração altera o crescimento assintótico médio dos algoritmos avaliados.

Todos os detalhes sobre as configurações utilizadas em nosso experimento podem ser verificadas no anexo D.1, onde disponibilizamos nosso *script tcl* para ns-2. Quanto ao *patch* reunindo todas as extensões mencionadas para ns-2.29 achamos mais conveniente disponibilizá-lo em [3].

4.3 Metodologias para avaliação de desempenho

Nesta seção discutimos os principais aspectos relacionados com as metodologias de simulação de horizonte infinito e de cálculo de proporções, empreendidas para gerar os resultados deste trabalho.

4.3.1 Credibilidade estatística em simulações de horizonte infinito

Após a definição da distribuição de probabilidades para amparar nosso estudo de caso médio dos algoritmos, é necessário utilizarmos uma metodologia para cumprir o papel da segunda parte, ou seja, estimarmos a média de todas as amostras de tamanho n da distribuição. Para este propósito adotamos a metodologia de simulação de horizonte infinito cujos princípios básicos descrevemos a seguir.

Na simulação de horizonte infinito estamos interessados que o processo estocástico resultante da distribuição de probabilidades convirja para um estado de equilíbrio, o *estado estacionário*. Teoricamente tal estado ocorre quando a quantidade de amostras tende ao infinito, o que concorda com a exigência teórica do estudo do caso médio de algoritmos. Apesar de, na prática, tal condição ser inviável, existe um ponto em que se pode considerar que o sistema esteja em equilíbrio e os erros possam ser desconsiderados [72].

A priori, não podemos supor que as primeiras amostras geradas com base na simulação da distribuição já pertençam ao estado estacionário. Isso porque a inicialização arbitrária dos parâmetros de configuração de um modelo estocástico pode produzir tendência nos estimadores, o que torna as amostra iniciais não representativas para o cálculo da média. Tais amostras pertencem ao *estado transiente*. A determinação das d^* primeiras amostras do estado transiente constitui um desafio em simulação de processos estocásticos. Sua subestimação pode resultar em tendência

nas médias estimadas o que, em nosso caso, compromete o estudo empírico da classe de complexidade média dos algoritmos para a distribuição considerada. Ao passo que a super-estimação pode eliminar amostras do estado estacionário e, consequentemente, aumentar a variância do estimador.

Outra dificuldade é a natureza das observações da saída de um modelo. Observações coletadas durante simulações estocásticas típicas, normalmente, apresentam alta correlação. Negar a existência de correlação pode resultar em intervalos de confiança muito otimistas. Para um discussão mais profunda dessas questões, consulte [79].

O último ponto a se levar em conta é determinar o fim da simulação assim que a precisão relativa ao tamanho do intervalo de confiança tenha sido atingida (*regra de parada*). Neste ponto há um compromisso entre o nível de precisão e o tempo de execução da simulação uma vez que quanto maior o total de amostras geradas maior será precisão da média [72]. A fim de tornar a demanda computacional das simulações estocásticas mais amena é possível projetar experimentos que não necessitem de muitas observações; isto pode ser alcançado por meio de técnicas de redução da variância que visam aumentar a eficiência estatística de um estudo de simulação. A discussão de tais técnicas foge ao escopo de nosso trabalho, assim procuramos utilizar aquela que fosse recomendada pela literatura especializada, conforme mencionamos a seguir.

Para gerarmos as curvas referentes ao caso médio dos algoritmos considerando os aspectos de credibilidade acima mencionados, utilizamos a ferramenta akaroa-2 [1].

O akaroa-2 corresponde ao bloco que decide sobre o fim da simulação que ilustramos na figura 4.1. Ele provê avançadas heurísticas para detectar o tamanho do estado transiente e utiliza a técnica de análise espectral para reduzir a variância e ajudar a atingir a regra de parada da simulação. Para cada curva média, cada um

de seus pontos foi obtido em um experimento de simulação estocástica de horizonte infinito visando a construção de um intervalo de confiança de 90%. A regra de parada de cada simulação foi baseada no erro relativo menor ou igual a 10%, ou seja, quando a semi-largura do intervalo de confiança era menor ou igual a 10% do valor estimado.

4.3.2 Credibilidade em análise de proporções

Para determinar a quantidade de amostras necessárias em um estudo de proporções, utilizamos a metodologia apresentada por Jain em [60].

Inicialmente realizamos uma quantidade m de execuções piloto para verificar o total x de vezes que ocorre um dado evento. Daí calculamos a proporção preliminar p' com lagura da metade do intervalo de confiança igual a r', conforme ilustrado nas equações 4.1 e 4.2.

$$p' = \frac{x}{m} \tag{4.1}$$

$$r' = z_{1-\frac{\alpha}{2}} \cdot \sqrt{\frac{p' \cdot (1-p')}{m}}, \tag{4.2}$$

onde z extraído da tabela quantil da distribuição normal na interseção da linha $1-\frac{\alpha}{2}$ com a coluna α. O total n de execuções para estimarmos a proporção final p é definida pela equação 4.3. Em nosso caso o intervalo tem nível de confiança de 95% (significância $\alpha = 0.05$).

$$n = z^2_{1-\frac{\alpha}{2}} \cdot \frac{p' \cdot (1-p')}{(r')^2} \tag{4.3}$$

Capítulo 5

Apresentação e análise dos resultados

Este capítulo é dedicado à apresentação e a análise dos resultados obtidos conforme a metodologia proposta no capítulo 4.

5.1 Análise empírica do caso médio

As tabelas 5.1 e 5.2 a seguir apresentam a média de operações de vértices e arestas, respectivamente, para os algoritmos de Dijkstra e Ramalingam & Reps (RRs) cujas complexidades *de pior caso* são $O(|A|\cdot\log(|V|))$ e $O(|\delta|+|\delta|\cdot\log(|\delta|))$, respectivamente. Os resultados foram obtidos sob a métrica ETX. Adicionalmente também avaliamos o algoritmo RRs sob a métrica LD, a qual segue uma abordagem completamente diferente da ETX. Isso fizemos a fim de investigar o quanto a métrica que pesa os enlaces pode afetar o desempenho médio do principal algoritmo investigado neste trabalho.

Nas tabelas a coluna n representa o total de amostras que foram necessárias simular para assegurar a confiabilidade das médias estimadas, conforme já mencio-

namos no capítulo anterior. $\overline{X}_{n-d^*}$ representa a *estimativa* da média de operações dos algoritmos a partir de amostras do estado estacionário. d^* representa a quantidade de amostras (iniciais) descartadas para o cálculo da média, por pertencerem ao estado transiente. H representa a metade do tamanho do intervalo de confiança das médias.

|V|	Algoritmo	$\overline{X}_{n-d^*}$	H	n	d^*
05	Dij_{ETX}	10.8	0.251026	4830	805
	RRs_{ETX}	6.14	0.0214579	2208	368
	RRs_{LD}	6.13	0.0192237	5424	452
10	Dij_{ETX}	24.27	0.412306	6456	1076
	RRs_{ETX}	7.19	0.28054	6594	314
	RRs_{LD}	6.78	0.0232904	3222	358
15	Dij_{ETX}	35.49	0.585612	12834	1426
	RRs_{ETX}	7.41	0.383429	3546	394
	RRs_{LD}	6.91	0.22724	10575	705
20	Dij_{ETX}	48.29	1.31118	8874	986
	RRs_{ETX}	8.83	0.500838	19680	410
	RRs_{LD}	7.06	0.508924	6138	1023
25	Dij_{ETX}	60.33	1.03611	16065	1071
	RRs_{ETX}	9.25	0.276982	128538	386
	RRs_{LD}	7.39	0.403948	2313	257
30	Dij_{ETX}	71.44	1.86504	15135	1009
	RRs_{ETX}	10.88	0.537381	32877	281
	RRs_{LD}	8.81	0.777396	23760	660
35	Dij_{ETX}	86.15	2.10604	5688	948
	RRs_{ETX}	12.19	1.13425	27918	282
	RRs_{LD}	12.42	1.1376	10179	377
40	Dij_{ETX}	91.63	2.1160	11952	1992
	RRs_{ETX}	15.01	1.16292	19071	489
	RRs_{LD}	12.80	1.26878	60147	489
45	Dij_{ETX}	109.01	1.36809	25857	507
	RRs_{ETX}	17.61	1.01431	42405	257
	RRs_{LD}	14.29	1.40865	59907	1051
50	Dij_{ETX}	120.76	2.48851	4782	797
	RRs_{ETX}	18.70	1.82253	35340	620
	RRs_{LD}	14.57	1.42276	44016	262

Tabela 5.1: Estimativa da média de operações de sobre *vértices* para os algoritmos de Dijkstra e Ramalingam e Reps em *backbones* em malha sem fio

|V|	Algoritmo	$\overline{X}_{n-d^*}$	H	n	d^*
05	Dij_{ETX}	12.83	0.225644	3312	552
	RRs_{ETX}	14.89	0.808684	2187	243
	RRs_{LD}	17.48	0.965101	5688	237
10	Dij_{ETX}	62.41	1.28444	2754	459
	RRs_{ETX}	38.88	2.02859	6846	326
	RRs_{LD}	39.51	0.334183	3672	612
15	Dij_{ETX}	138.12	1.66267	3558	593
	RRs_{ETX}	58.84	3.13631	4272	356
	RRs_{LD}	60.05	3.53327	10944	228
20	Dij_{ETX}	226.04	2.01172	8886	1481
	RRs_{ETX}	82.98	4.7966	20856	316
	RRs_{LD}	66.91	6.50783	5952	248
25	Dij_{ETX}	307.82	4.07353	39138	6523
	RRs_{ETX}	95.57	2.33482	127764	308
	RRs_{LD}	68.62	2.66816	1782	297
30	Dij_{ETX}	385.52	0.6219	12524	1009
	RRs_{ETX}	114.59	4.21265	32292	299
	RRs_{LD}	88.66	8.79934	24750	275
35	Dij_{ETX}	466.05	0.3592	29997	948
	RRs_{ETX}	134.612	13.0166	28638	258
	RRs_{LD}	117.15	11.1522	10164	242
40	Dij_{ETX}	532.09	0.5611	32338	1992
	RRs_{ETX}	170.30	16.3866	20304	376
	RRs_{LD}	128.86	1.26878	60147	489
45	Dij_{ETX}	620.15	0.3295	69216	507
	RRs_{ETX}	200.71	11.3913	42687	279
	RRs_{LD}	150.76	15.0366	61110	291
50	Dij_{ETX}	688.10	0.5553	36620	797
	RRs_{ETX}	211.67	20.4578	34992	243
	RRs_{LD}	150.76	15.0366	61110	291

Tabela 5.2: Estimativa da média de operações de processamento de *arestas* para os algoritmos de Dijkstra e Ramalingam e Reps em *backbones* em malha sem fio

5.1.1 Estimativa empírica das relações assintóticas entre os algoritmos no caso médio

As estimativas das curvas referentes ao caso médio dos algoritmos são ilustradas nas figuras 5.1 e 5.2. Na primeira figura o item de desempenho considerado é a soma das operações sobre vértices (i.e. operações sobre fila de prioridades:`Insere_Elemento`, `Diminui_Chave`, `Extrai_Mínimo`), cujas curvas estimadas denotamos por T_{Dij_V}, para o algoritmo de Dijkstra, e T_{RRs_V} para o algoritmo RRs. Já na segunda são consideradas as operações de processamento de arestas, cujas curvas denotamos por T_{Dij_A} e T_{RRs_A}, respectivamente. Os valores foram obtidos das tabelas de resultados 5.1 e 5.2.

Para iniciar nossas análises optamos por apresentar as operações sobre vértices e sobre arestas separadamente. Esse tipo de informação fornece nos uma idéia geral acerca de qual grandeza é assintoticamente dominante na complexidade final dos algoritmos para o contexto de nosso problema. Conforme podemos notar no eixo das ordenadas das figuras 5.1 e 5.2 as operações sobre arestas são mais onerosas para ambos os algoritmos.

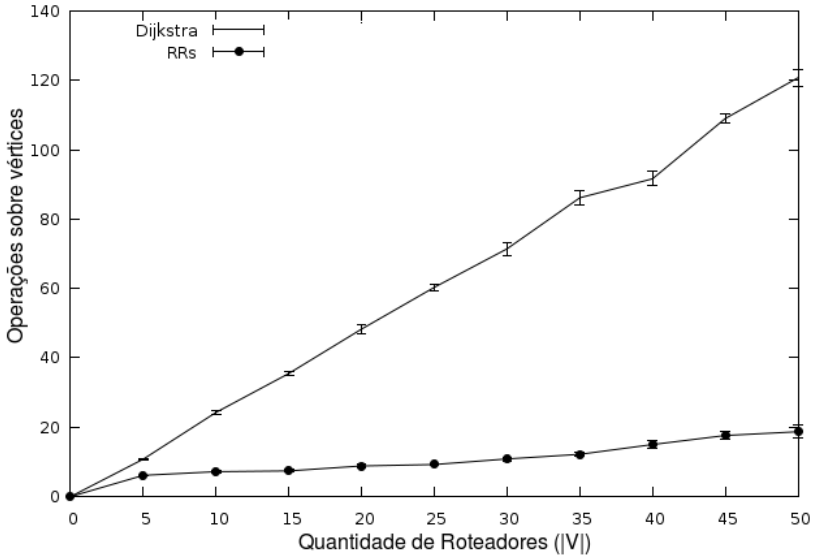

Figura 5.1: Dijkstra×RRs:Média de operações sobre vértices em WMNs

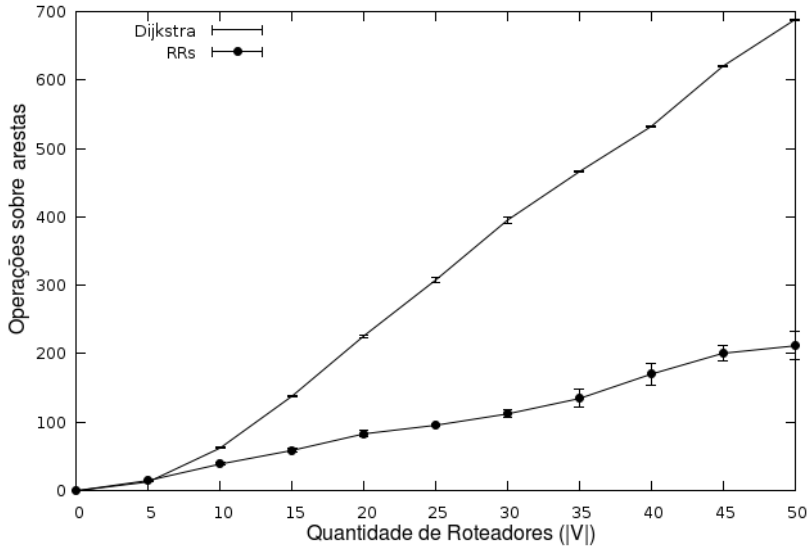

Figura 5.2: Dijkstra×RRs:Média de operações sobre arestas em WMNs

Pelas ilustrações podemos verificar a superioridade do desempenho de algoritmo RRs sobre o de Dijkstra no ambiente WMN proposto. *Apesar de não sabermos a função de custo de cada algoritmo no caso médio*, é razoável inferir que nele a função associada ao algoritmo de Dijkstra domina assintoticamente a associada ao algoritmo de RRs. Por outro lado, também estamos interessados em *estimar empiricamente* a natureza dessa relação a partir dos dados simulados ($O()$ ou somente $o()$).

Uma vez que a quantidade de pontos simulados é finita, seria trivial encontrar uma constante que "comprovasse" uma relação do tipo $T_{Dij} = O(T_{RRs})$, onde $T_{Dij} = T_{Dij_V} + T_{Dij_A}$ e $T_{RRs} = T_{RRs_V} + T_{RRs_A}$. Porém, no intuito de evitar tendência nesse processo de análise empírica, a escolha da constante não pode ser arbitrária mas antes deve guardar uma relação com os valores simulados. A seguir propomos uma abordagem baseada na teoria de classes de complexidade a fim de estimar a natureza da dominação assintótica do algoritmo de Dijkstra sobre o de RRs no caso médio.

Sejam f e g duas relações definidas sobre um domínio finito $\mathcal{X}$. Estamos interessados em investigar evidências acerca do tipo de dominação assintótica do polinômio $P_g(n)$ sobre o $P_f(n)$ ($\forall n \in \mathcal{R}^+$) obtidos de interpolações polinomiais sobre os pontos

$(x, g(x))$ e $(x, f(x))$, respectivamente.

Para *estimarmos* se $P_g(n) = O(P_f(n))$, a partir $\mathcal{X}$, definimos uma constante c a partir da razão $\frac{min(g(\psi))}{f(\psi)}$, sujeito a $g(\psi) > f(\psi)$. Se essa constante satisfizer $c \cdot f(x) \geq g(x)\ \forall x \in \mathcal{X}$, então, a partir do escopo $\mathcal{X}$, temos um *indício favorável* a $P_g(n) = O(P_f(n))$, $\forall n \in \mathcal{R}^+$. Se, por outro lado, frequentemente precisarmos recorrer a novas razões para obtermos constantes maiores a fim de manter a curva $f(x)$ acima de $g(x)$, então, a partir do escopo $\mathcal{X}$, temos um *indício desfavorável* a $P_g(n) = O(P_f(n))$, $\forall n \in \mathcal{R}^+$, o que, dada as condições anteriores, favorece $P_f(n) = o(P_g(n))$ i.e. as curvas polinomiais não são separadas por um fator constante.

A partir do raciocínio acima exposto e com base nos valores das tabelas 5.1 e 5.2, investigamos uma estimativa da relação assintótica entre os polinômios de custos associados aos algoritmos RRs e Dijkstra no caso médio.

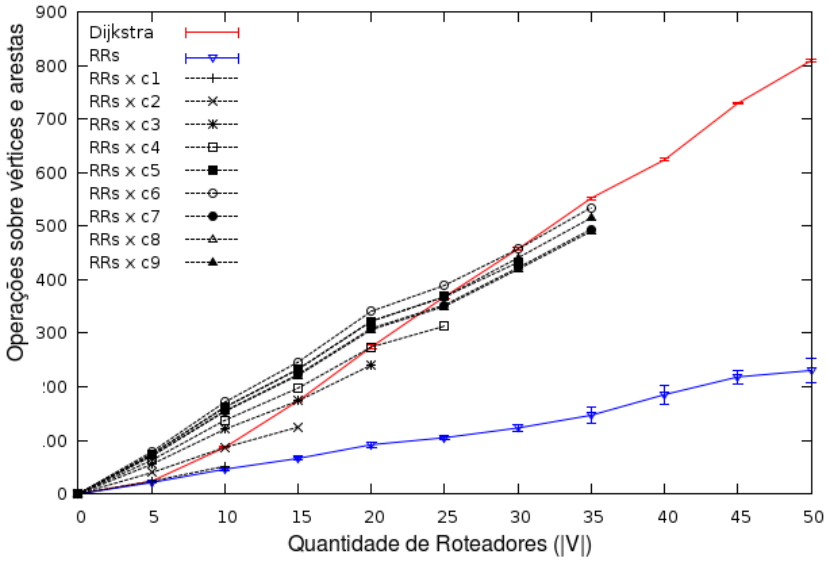

Figura 5.3: Estimativa da natureza da dominação assintótica de T_{Dij} sobre T_{RRs}

Na figura 5.3 a curva de linha cheia que se mantém acima das demais corresponde a T_{Dij} e a que se mantém abaixo corresponde a T_{RRs}. As curvas tracejadas correspondem às tentativas (frustradas) de se estimar a veracidade de $T_{Dij} = O(T_{RRs})$ com base nas razões obtidas a partir dos pontos simulados de T_{Dij} e T_{RRs}. A fre-

quente necessidade de se obter novas constantes para assegurar $T_{Dij} = O(T_{RRs})$ é um indício desfavorável a tal relação assintótica, conforme a abordagem previamente descrita. Como a figura 5.3 já sugere fortemente que T_{Dij} domina assintoticamente T_{RRs}, são prováveis as seguintes relações assintóticas *no caso médio* da distribuição WMN proposta: $T_{Dij} \notin \Theta(T_{RRs})$ e $T_{RRs} = o(T_{Dij})$.

5.1.2 Estimativa empírica do caso médio do algoritmo RRs sob métricas diferentes

Uma vez que a métrica de roteamento define o peso das arestas do grafo *a priori* ela é um componente relevante para o caso médio do algoritmo. Assim, de forma similar a análise a pouco realizada entre T_{RRs} e T_{Dij}, investigamos o impacto da métrica de roteamento na curva média do algoritmo RRs. Por este motivo selecionamos as métricas ETX e LD, que apresenta filosofias diferentes, conforme comentamos na subseção 2.2.7.

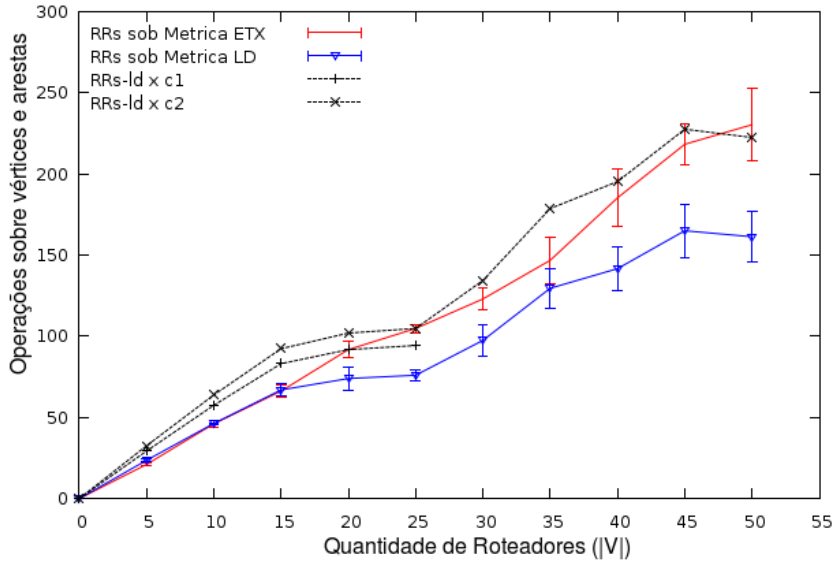

Figura 5.4: Curvas de crescimento do algoritmo RRs sob as métricas ETX e LD

Na figura 5.4 a curva de linha cheia que se mantém superior corresponde à

curva T_{RRs} quando o grafo é ponderado com a métrica ETX (chamada $T_{RRs_{ETX}}$) ao passo que a inferior corresponde à curva $T_{RRs_{LD}}$, isto é, quando o grafo é ponderado com a métrica LD. Por meio de uma análise similar a fornecida na seção anterior, verificamos uma evidência favorável à relação assintótica $T_{RRs_{ETX}} = O(T_{RRs_{LD}})$, conforme ilustrado no gráfico. Nele podemos verificar que na segunda tentativa foi possível obter o indício favorável, mesmo com o decaimento da curva RRs-ld $\times\, c2$ no ponto 50 do eixo da abscissa. Isso porque as curvas das médias $\overline{X}_{n-d^*}$, e as curvas de erro delas decorrentes $\overline{X}_{n-d^*} + H$ e $\overline{X}_{n-d^*} - H$, pertencem à mesma classe Θ (i.e. a dominação sobre uma delas assegura a dominação sobre as demais), conforme passamos passamos explicar a seguir.

Em nossas simulações de horizonte infinito a regra de parada estava condicionada à constante de precisão relativa γ da semi-largura do intervalo de confiança H. Assim, com bastante otimismo, H será no máximo da mesma classe assintótica de $\overline{X}_{n-d^*}$, conforme demonstrado no desenvolvimento matemático em 5.1, 5.2 e 5.3.

$$\begin{aligned} \gamma &\geq \frac{H}{X_{n-d^*}} && (5.1)\\ X_{n-d^*} &\geq \underbrace{\left(\frac{1}{\gamma}\right)}_{cte.} \cdot H \\ X_{n-d^*} &= \Omega(H). \therefore && (5.2)\\ X_{n-d^*} \pm H &\in \Theta(max(X_{n-d^*}, H)), \end{aligned}$$

de 5.2 temos que H é, no máximo, da mesma classe de$X_{n-d^*}, assim$

$$X_{n-d^*} \pm H \in \Theta(X_{n-d^*}) \quad (5.3)$$

Por fim, dado que o gráfico 5.4 por si só sugere $T_{RRs_{ETX}} = \Omega(T_{RRs_{LD}})$ para $c = 1$, temos como consequência final um indício favorável à relação assintótica $T_{RRs_{ETX}} = \Theta(T_{RRs_{LD}})$. Este é resultado, somado ao fato de termos utilizados

métricas com semânticas completamente diferentes, *sugere fortemente* que a *classe de complexidade média* do algoritmo RRs em WMNs, seja ela qual for, permanece a mesma independente da métrica, i.e. as curvas médias resultantes são separadas por fatores constantes.

5.2 Proporção de execuções do pior caso

Conforme discutido nas subseções 3.4.2 e 3.4.3 o pior caso dos algoritmos de Dijkstra e RRs ocorre quando, em uma dada iteração, a condição para a execução da operação `Diminui_Chave` é satisfeita (linha 01 do algoritmo 3.2, **Relaxa**). Utilizamos a metodologia descrita na subseção 4.3.2 para calcular a proporção de execuções da operação `Diminui_Chave` em relação ao total de testes de relaxamento dos algoritmos no contexto da distribuição de WMN descrita no capítulo 4.

A tabela 5.3 apresenta a proporção P de execuções de `Diminui_Chave`, a semi-largura do intervalo de confiança H (com nível de significância $\alpha = 0.05$) e o total n de amostras necessárias para o cáculo de P.

A figura 5.5 fornece-nos uma idéia geral da proporção que cada algoritmo executou a operação `Diminui_Chave`. Podemos verificar que, para ambos os algoritmos, tal operação foi pouca acessada. Em tais condições uma fila de prioridades baseada em *heaps* binários é melhor que a de Fibonacci, uma vez que a operação `Extrai_Mínimo` é executada 100% das iterações (cf. discussão sobre filas de prioridade na subseção 3.4.2).

A maior proporção observada ficou por conta do algoritmo de Dijkstra em que a operação foi executada 17% do total de testes de relaxamento. Em todas as ocasiões consideradas, a proporção referente ao algoritmo RRs em nenhum caso ultrapassou 1% do total de testes. Isso pode ser compreendido porque as possibilidades de tal algoritmo executar `Diminui_Chave` são bastante reduzidas pelo tamanho do conjunto

|V|	Algoritmo	$P(\%)$	H	n
05	Dij	17.0807	0.001649	200003
	RRs	0.3298	0.000251	200135
10	Dij	7.9559	0.001186	200038
	RRs	0.0345	0.000081	200007
15	Dij	5.8060	0.001025	200051
	RRs	0.0435	0.000091	200051
20	Dij	5.6277	0.001009	200213
	RRs	0.0295	0.000075	200248
25	Dij	5.2091	0.000974	200126
	RRs	0.0735	0.000119	200041
30	Dij	5.5157	0.001000	200203
	RRs	0.1200	0.000152	200064
35	Dij	4.8642	0.000943	200099
	RRs	0.1223	0.000153	200098
40	Dij	5.7466	0.001020	200093
	RRs	0.0730	0.000118	200051
45	Dij	6.9683	0.001116	200007
	RRs	0.4107	0.000280	200008
50	Dij	8.0209	0.001190	200145
	RRs	0.2505	0.000219	200006

Tabela 5.3: Resultado das proporções de execução da operação `Diminui_Chave` em relação ao total de testes de relaxamento

de vértices afetados δ. Conforme discutiremos na seção 5.3, esse parâmetro frequentemente é bem menor que $|V|$ ($|\delta| <<< |V|$). Por outro lado, em cumplicidade com o padrão do OLSR, o algoritmo de Dijkstra recalcula os caminhos mínimos para todos os destinos, quer afetados quer não.

5.3 Análise comparativa das abordagens

A série de análises comparativas até então por nós realizada demonstra a superioridade do algoritmo de Ramalingam e Reps sobre o de Dijkstra no contexto de

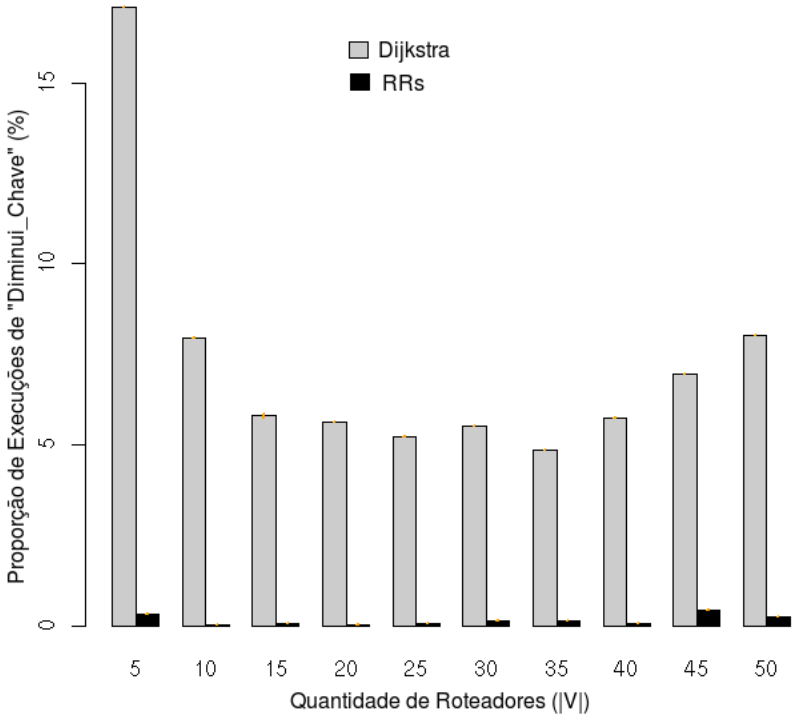

Figura 5.5: Dijkstra×RRs:Proporção de execuções da operação `Diminui_Chave` em relação aos testes de relaxamento

WMNs. Neste seção demonstramos que tal superioridade deve-se essencialmente a dois motivos básicos:

- a suposição implícita no padrão do OLSR de que uma notificação topológica afeta todos (ou a maioria) os destinos listados na tabela de roteamento;
- o modelo de complexidade subjacente ao algoritmo RRs, mais especificamente o caso médio ou a natureza estocástica de $|\delta|$ em WMNs[1].

5.3.1 Proporção de vezes que uma mudança topológica afetou a tabela de roteamento

Um roteador OLSR pode ter seu grafo alterado de duas formas. Na primeira, que denotamos por "tipo A", uma mensagem de atualização de topologia indica a

[1] ou pelo menos no modelo proposto

adição ou a atualização de peso de um enlace. Na segunda, que denotamos por "tipo B", o tempo de vida do enlace pode expirar e, se nenhuma mensagem chegar notificando a existência do enlace, a aresta a ele correspondente no grafo deverá ser removida.

O padrão do OLSR prescreve que a cada ocorrência de uma atualização do tipo A ou do B a tabela de roteamento deve ser recalculada. Na prática, porém, é possível que tais atualizações afetem o grafo sem necessariamente afetar as rotas selecionadas para a tabela de roteamento. Nesse caso a abordagem adotada pelo padrão do OLSR dá margem para que a tabela seja desnecessariamente recalculada.

|V|	Atualização	$P(\%)$	H	n
05	Tipo B	29.03	0.028147	999
	Tipo A	52.68	0.015473	4000
10	Tipo B	12.41	0.020447	999
	Tipo A	31.85	0.014438	4000
15	Tipo B	8.9	0.017649	1000
	Tipo A	26.76	0.013721	3999
20	Tipo B	7.4	0.016225	1000
	Tipo A	22.78	0.013000	3999
25	Tipo B	7.81	0.016637	999
	Tipo A	31.98	0.014456	3999
30	Tipo B	7.81	0.0166370	999
	Tipo A	35.53	0.014834	3999
35	Tipo B	9.81	0.018445	999
	Tipo A	28.38	0.013974	3999
40	Tipo B	8.51	0.017302	999
	Tipo A	32.98	0.014572	3999
45	Tipo B	10.2	0.018758	1000
	Tipo A	40.63	0.015220	4000
50	Tipo B	9.31	0.018018	999
	Tipo A	28.43	0.013981	3999

Tabela 5.4: Resultado das proporções de execução da operação `Diminui_Chave` em relação ao total de testes de relaxamento

A tabela 5.4 apresenta, para cada tipo de atualização topológica (A e B), a proporção P de vezes que a tabela de roteamento foi afetada (com respectivos intervalos de confiança de 95% de semi-largura H e total n de amostras utilizadas no cálculo).

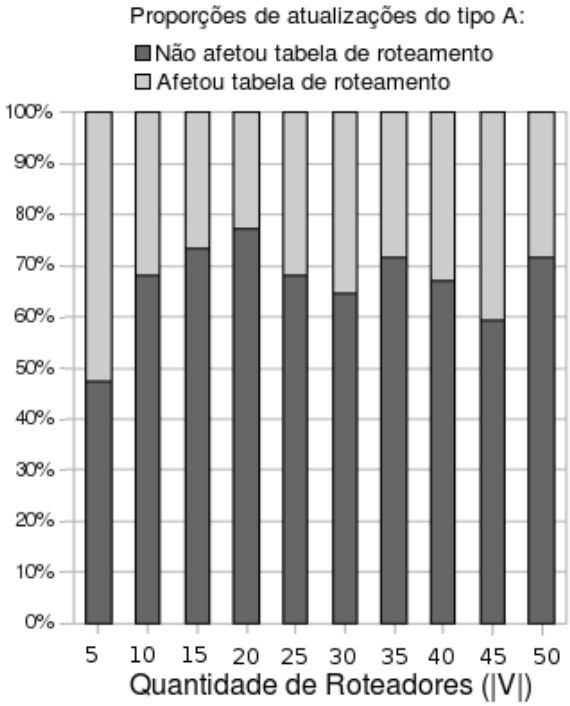

Figura 5.6: Proporção de vezes que a tabela de roteamento foi afetada por atualizações do tipo A

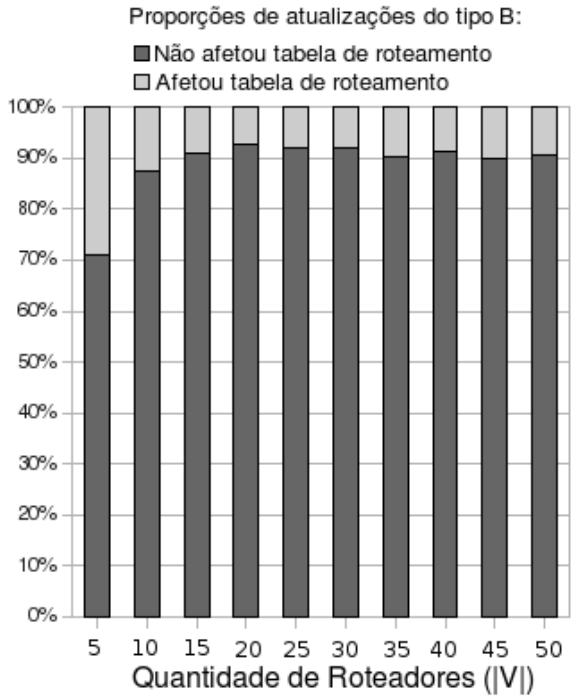

Figura 5.7: Proporção de vezes que a tabela de roteamento foi afetada por atualizações do tipo B

As figuras 5.6 e 5.7 fornecem uma idéia do desperdício de recursos computacionais sofrida pelo padrão do OLSR para lidar com atualizações de topologia dos tipos A e B, respectivamente. Conforme ilustrado, para WMNs com variadas quantidades de roteadores frequentemente a tabela de roteamento é recalculada sem necessidade (i.e. o grafo sofre mudanças que não afetam as rotas ótimas da tabela).

Conforme discutido na subseção 4.2.2, uma "má" configuração do procedimento de descoberta de topologia do OLSR pode levar a tabela de roteamento a se basear em informações inconsistentes ou obsoletas o que, por sua vez, conduz a frequentes impactos na tabela a cada nova notícia de topologia. Uma vez que uma maior con-

sistência entre o grafo e a topologia real da rede pode ser alcançada por meio da configuração do OLSR e do uso de técnicas avançadas como *fish-eye*, acreditamos que o procedimento de recálculo da tabela deve estar estritamente condicionado à invalidação da tabela de roteamento, e não necessariamente à mudança da topologia do grafo. Um procedimento pode ser proposto para este fim sem perda de generalidade quanto ao algoritmo utilizado para o recálculo (e.g. no caso do RRs, é verificado se a modificação afetou Gd).

Submetemos a sugestão sobre o critério de recálculo da tabela na lista de discussão do grupo IETF-MANETs, a qual ainda encontra-se sob debate até o presente momento, conferir anexo B.

5.3.2 Características acerca da natureza estocástica de $|\delta|$

A experiência prática tem demonstrado que o recálculo da tabela de roteamento é um componente vital à escalabilidade daqueles protocolos em *backbones* WMNs. Conforme já mencionado na seção 3.2, experimentos práticos têm demonstrado que supor uma invalidação total da tabela de roteamento à cada atualização topológica pode comprometer a escalabilidade do OLSR.

Na abordagem BIC a cada vez que a tabela de roteamento é afetada, $|\delta|$ assume um valor entre 1 e $|V|-1$. Desta forma a descoberta de evidências acerca da natureza estocástica de $|\delta|$ pode ajudar, por exemplo, a verificar qual a probabilidade de $|\delta|$ assumir valores próximos à $|V|$. Neste sentido apresentamos nas figuras 5.8 a 5.17, as distribuição de frequências relativas (DFR) de $|\delta|$ para cada uma das topologias de grade com $|V|$ roteadores previamente explicadas. Nas figuras o eixo vertical representa a frequência relativa de um dado valor $|\delta|_i^{|V|}$ e o eixo horizontal contém os limites inferior l_j e superior l_{j+1} das classes, tal que $|\delta|_i^{|V|} \in [l_j, l_{j+1})$.

As distribuições de frequências de cada topologia analisada (5.8 a 5.17) sugere

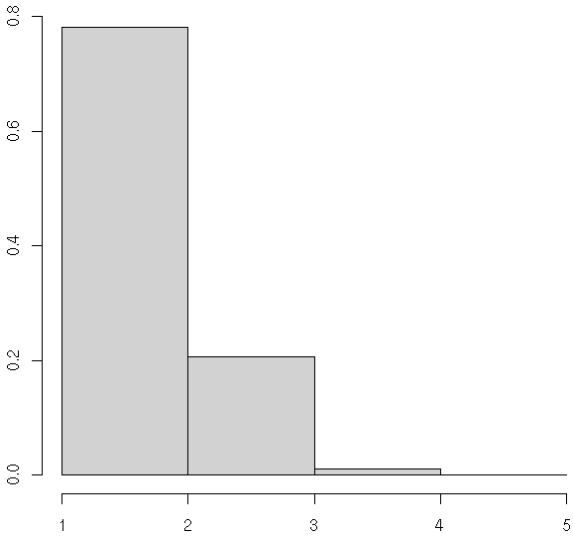

Figura 5.8: DFR de $|\delta|$ ($|V| = 05$)

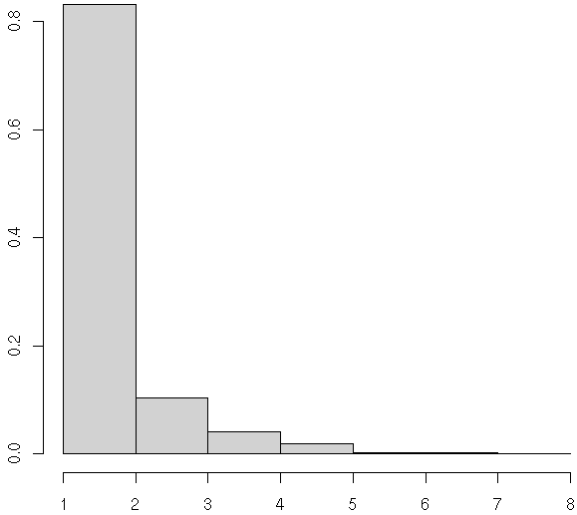

Figura 5.9: DFR de $|\delta|$ ($|V| = 10$)

que maiores valores de $|\delta|$ conduzem às probabilidades mais baixas, enquanto que as probabilidades mais altas estão associadas aos menores valores de $|\delta|$. Esse resultado revela que o superior desempenho do algoritmo RRs frente ao Dijkstra, deve-se ao modelo de complexidade a ele subjacente. Isso torna o parâmetro $|\delta|$ como ponto de referência para classificar propostas para o problema da manutenção de rotas em protocolos estado de enlace. Neste contexto uma solução pode ser considerada escalável se, de alguma forma, a complexidade de tempo da função de custo a ela associada satisfizer $O(f(|\delta|))$.

5.3.2.1 Investigação de auto-similaridade em $|\delta|$

Os resultados obtidos na seção anterior mostram que a probabilidade de $|\delta|$ decresce drasticamente à medida que os valores considerados se aproximam de $|V| - \{s\}$. Verificamos que esse comportamento se manteve para topologias com diferentes quantidades $|V|$ de roteadores [2]. Uma importante análise consiste em fixar um cenário e verificar se tal comportamento se mantém sob variadas escalas de tempo. O grau de auto-similaridade do processo analisado reflete a "habilidade"

[2]tal comportamento também foi verificado sob outras condições, conferir anexo C

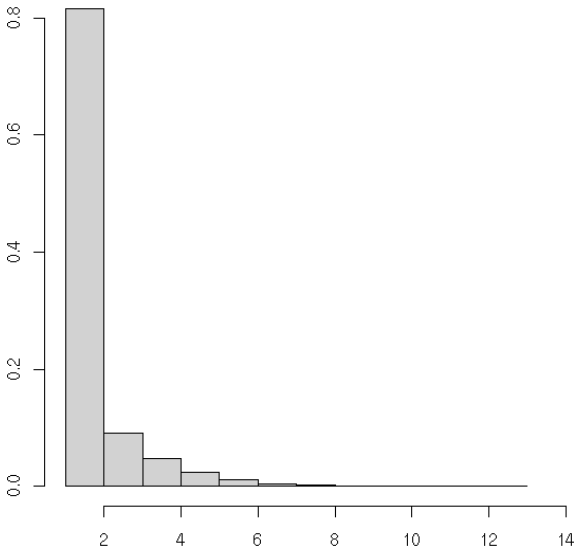

Figura 5.10: DFR de $|\delta|$ ($|V| = 15$)

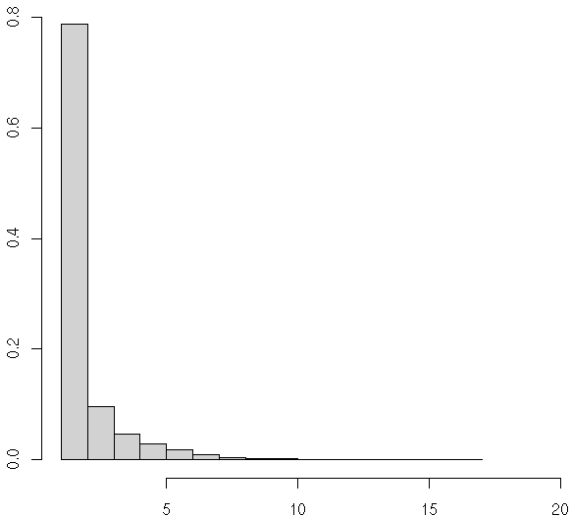

Figura 5.11: DFR de $|\delta|$ ($|V| = 20$)

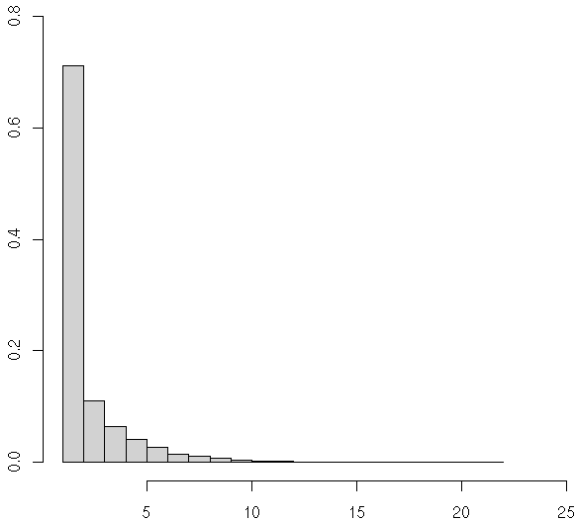

Figura 5.12: DFR de $|\delta|$ ($|V| = 25$)

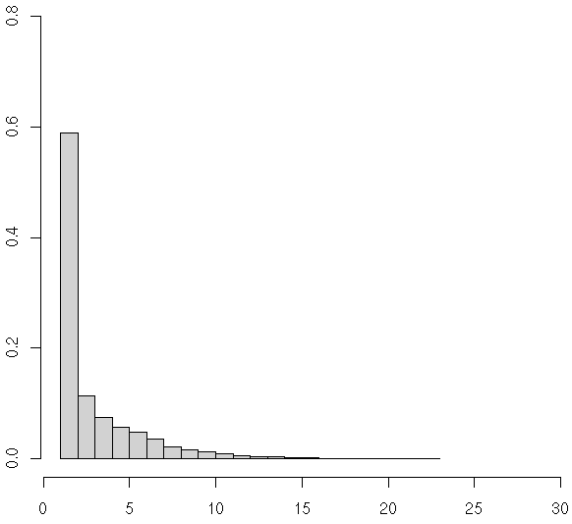

Figura 5.13: DFR de $|\delta|$ ($|V| = 30$)

de um processo preservar uma dada característica estatística sob diferentes escalas de tempo. Formalmente isso pode ser verificado, por exemplo, se as funções de auto-correlação inerentes ao processo em cada escala forem iguais [66]. A seguir investigamos a possibilidade da distribuição de frequência de $|\delta|$ apresentar auto-similaridade.

O ponto inicial para avaliar a distribuição de $|\delta|$ sob diferentes escalas de tempo

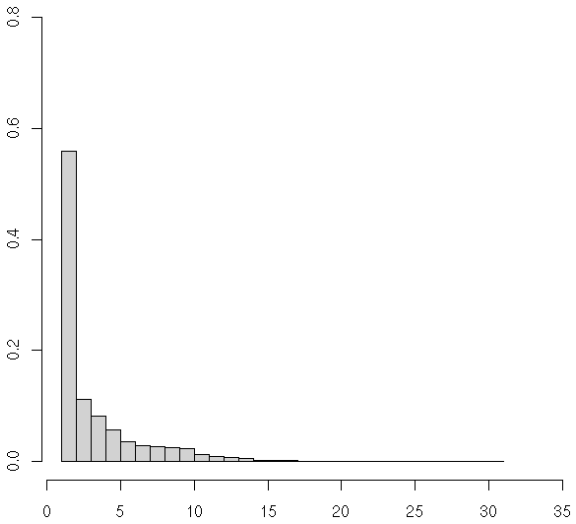

Figura 5.14: DFR de $|\delta|$ ($|V| = 35$)

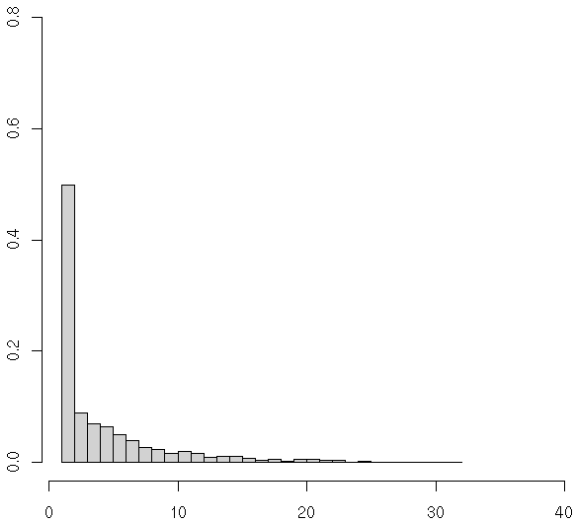

Figura 5.15: DFR de $|\delta|$ ($|V| = 40$)

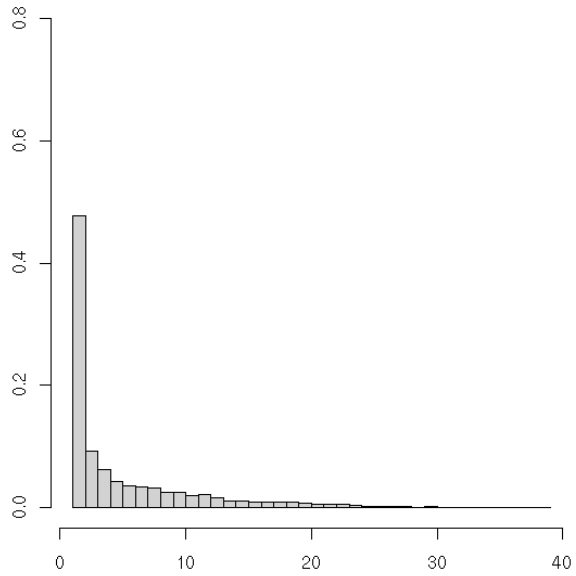

Figura 5.16: DFR de $|\delta|$ ($|V| = 45$)

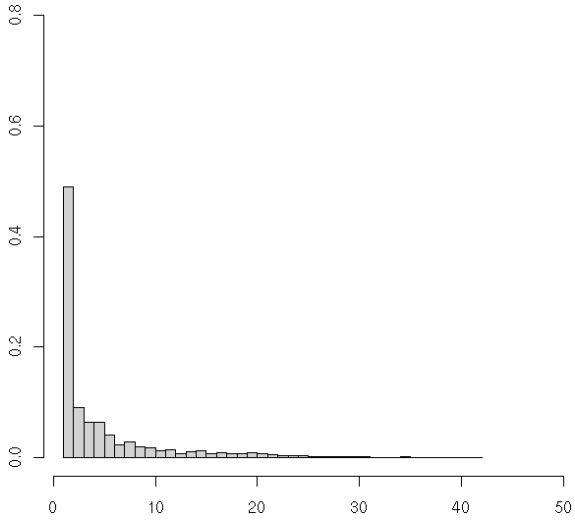

Figura 5.17: DFR de $|\delta|$ ($|V| = 50$)

reside no fato de que *o avanço do tempo é percebido pela chegada de uma mensagem de notificação de topologia.* Desta forma M_i denota a mensagem que chegou no instante i. No intervalo de tempo de chegada entre duas mensagens quaisquer M_i e $M_{i+\Delta}$, a tabela de roteamento pode ser afetada várias vezes, o que nos fornece uma série de "*deltas*" entre as mensagens, conforme explicamos a seguir.

Quando uma mensagem de M_i chega ao roteador, ela porta um conjunto de

notícias sobre a topologia, ou seja, enlaces com seus respectivos pesos (conforme descrito na seção 3.1 e ilustrado nas figuras 3.5 e 3.6). Cada uma dessas informações são usadas para atualizar o grafo do roteador. Esse evento, por sua vez, pode afetar $|\delta|$ caminhos ótimos registrados na tabela de roteamento, i.e. $|\delta|$ roteadores. Assim o processamento todos os enlace notificados por uma única mensagem M_i pode acarretar na geração de vários $|\delta|$'s. Além disso, durante o intervalo de tempo entre as mensagens M_i e $M_{i+\Delta}$, um roteador pode concluir que tempo de vida de um dado enlace expirou (por não chegar atualizações periódicas sobre ele). Tal evento, por sua vez, também pode afetar a tabela de roteamento, gerando assim mais um conjunto de vértices afetados δ. De forma geral, o j-ésimo $|\delta|$ gerado após a chegada da mensagem M_i é denotado por $|\delta|^{(j)}_{M_i}$. Assim, a série temporal de todos os $|\delta|$'s produzidos no intervalo de chegada de duas mensagens quaisquer M_1 e M_n é denotada por $S = \{|\delta|^{(1)}_{M_1}, |\delta|^{(2)}_{M_1}, \ldots, |\delta|^{(j)}_{M_1}, |\delta|^{(1)}_{M_2}, \ldots, |\delta|^{1}_{M_3}, \ldots, |\delta|^{(i)}_{M_n}, \ldots\}$.

Diante da explicação dada, nós investigamos a DF de $|\delta|$ entre intervalos de tempo de chegada de mensagens. Definimos cinco intervalos $\Delta = 50$ (i.e. intervalos: $50 \vdash 100 \vdash 150 \vdash 200 \vdash 250 \vdash 300$). Isso significa que nós avaliamos a DF de $|\delta|$ a cada 50 *unidades* de tempo. Repetimos esse procedimento utilizando três diferentes escalas para as unidades de tempo: $\alpha = \{1, 10, 100\}$. Para a primeira escala, por exemplo, teremos a DF de $|\delta|$ entre os instantes $[50, 100)$, $[100, 150)$ e assim por diante. Ainda a título de ilustração, o histograma obtido no intervalo $[50, 100)$ da escala de 100, contém todos os valores de $|\delta|$ que foram distribuídos ao longo dos intervalos do histograma obtido sob a escala de 1.

Uma última consideração a ser feita diz respeito à quantidade de conjuntos δ gerados após a chegada de uma dada mensagem [3]. Com o auxílio do Akaroa-2, verificamos que as amostras desse processo referentes ao estado transiente estão inteiramente situadas no intervalo de tempo 1 à 50 (i.e. instantes entre a primeira

[3] não confundir com a cardinalidade $|\delta|$, grandeza central de nossa investigação

mensagem e quinquagésima). Apesar do efeito do transiente ser suprimido pelo volume de dados das escalas maiores (por exemplo 100) o mesmo não ocorre para escalas menores (por exemplo 1). Por este motivo desconsideramos a DF de $|\delta|$ no intervalo $[1, 50)$ em todas as escalas.

Tendo por base os valores e notação a pouco explanadas, as figuras 5.18 a 5.32 ilustram a distribuição de frequências de $|\delta|$ para o topologia em grade 5×5 descrita no capítulo de metodologia. Para cada um dos cinco intervalos de tamanho igual a 50 unidades de tempo ($50 \vdash 100 \vdash 150 \vdash 200 \vdash 250 \vdash 300$) são ilustrados os histogramas variando-se somente a unidade de tempo, i.e. a escala de tempo $\alpha = \{1, 10, 100\}$. Conforme pode ser verificado nas figuras, a distribuição apresenta um comportamento quase insensível à variação da escala, sugerindo assim um comportamento auto-similar.

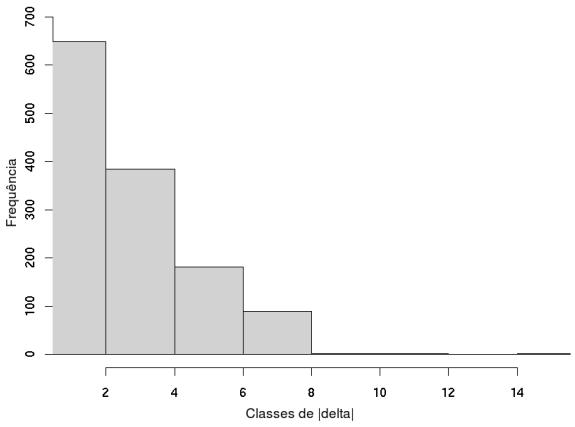

Figura 5.18: $DF_{(25)}$; $[50, 100)$; α=1

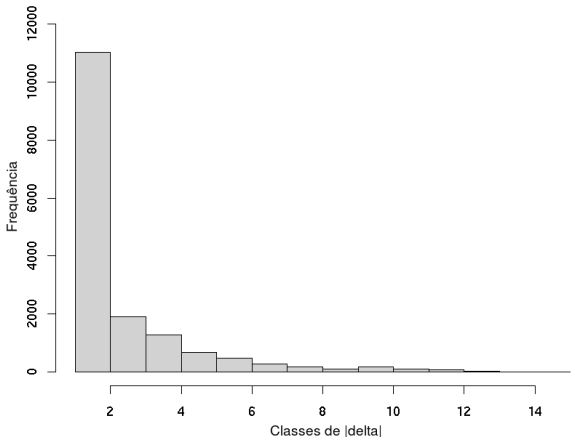

Figura 5.19: $DF_{(25)}$; $[50, 100)$; α=10

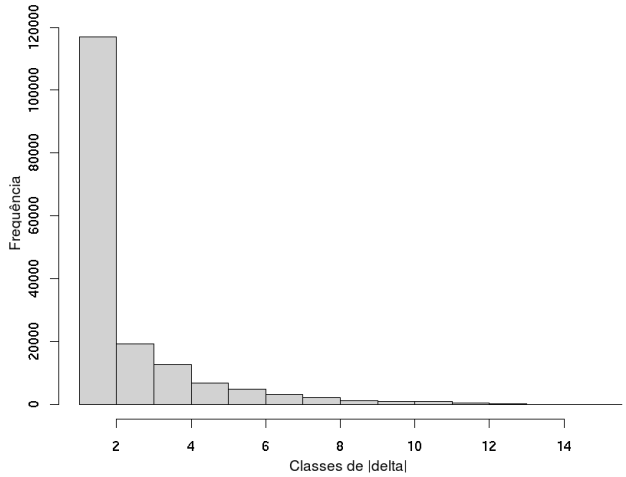

Figura 5.20: $DF_{(25)}$; $[50, 100)$; α=100

Adicionalmente recorremos ao cálculo do parâmetro de Hurst $\mathcal{H}$ para as séries referentes a cada topologia simulada. Esse parâmetro varia no intervalo $(0, 1)$. A condição $0.5 < \mathcal{H} < 1$ indica a presença de auto-similaridade na série. Quanto mais próximo de 1, maior será a persistência da correlação na série.

Algumas técnicas tem sido propostas para a estimação do valor de $\mathcal{H}$. Como tal estudo foge ao escopo de nosso trabalho optamos por selecionar mais de uma técnica para auxiliar nas conclusões, a saber, *Aggregated variance* ($\mathcal{H}_{Ag}$), *Wavelet* ($\mathcal{H}_{Wav}$) e *local Whittle* ($\mathcal{H}_{lW}$), cujos códigos fontes estão disponíveis em [20], e a abordagem não paramétrica ($\mathcal{H}_{np}$) disponibilizada em [69].

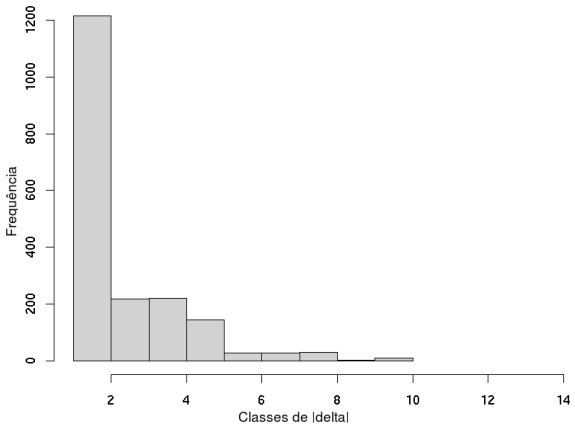

Figura 5.21: $DF_{(25)}$; $[100, 150)$; α=1

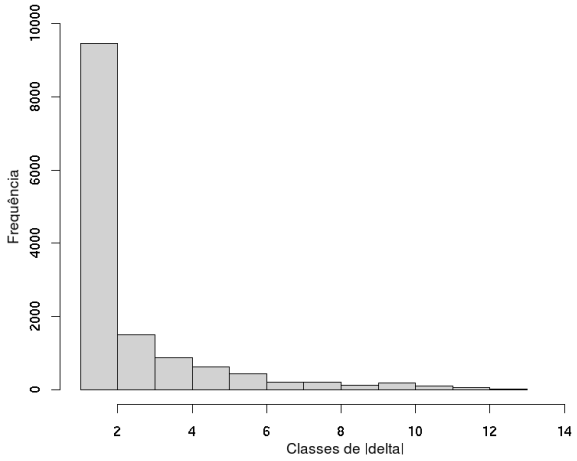

Figura 5.22: $DF_{(25)}$; $[100, 150)$; α=10

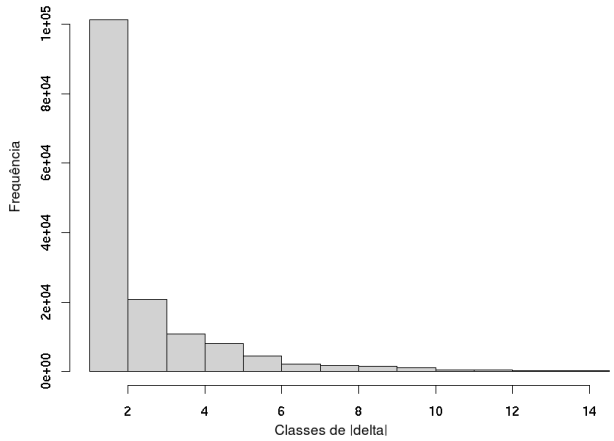

Figura 5.23: $DF_{(25)}$; $[100, 150)$; α=100

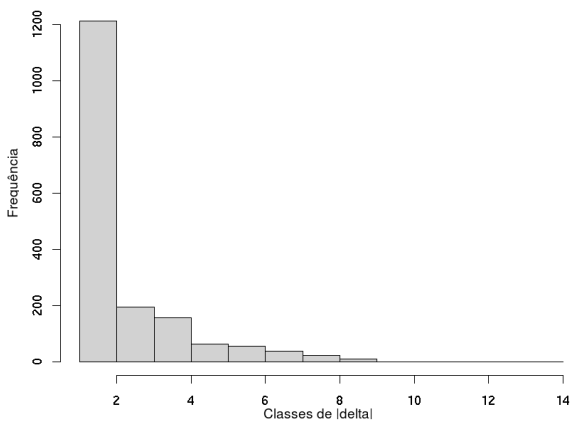

Figura 5.24: $DF_{(25)}$; $[150, 200)$; α=1

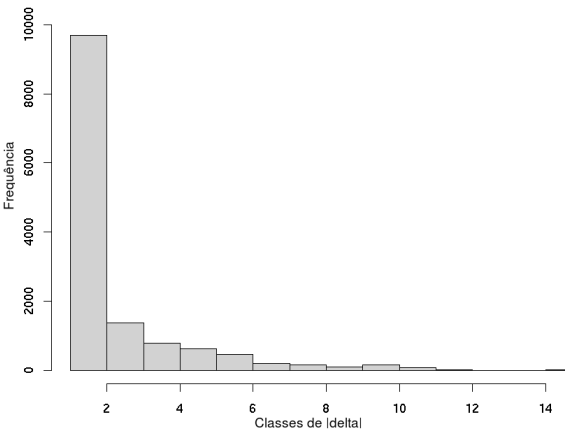

Figura 5.25: $DF_{(25)}$; $[150, 200)$; α=10

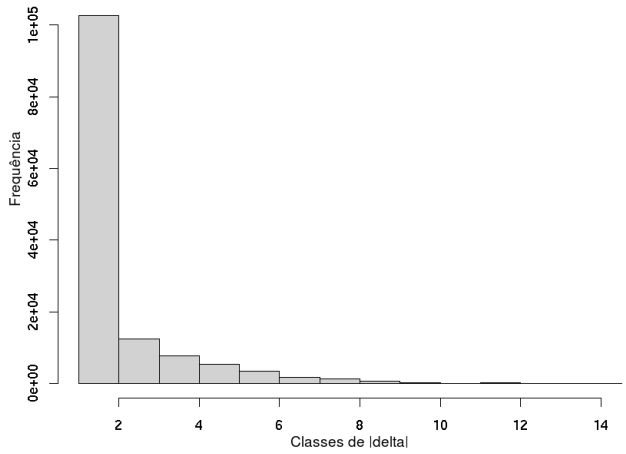

Figura 5.26: $DF_{(25)}$; $[150, 200)$; α=100

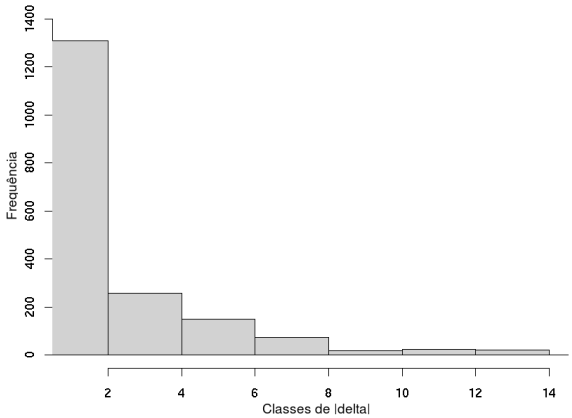

Figura 5.27: $DF_{(25)}$; $[200, 250)$; α=1

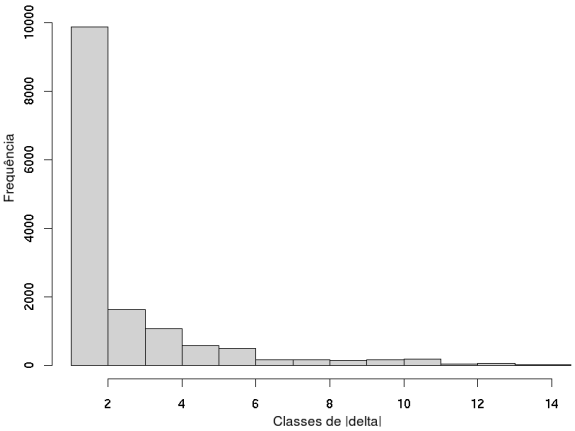

Figura 5.28: $DF_{(25)}$; $[200, 250)$; α=10

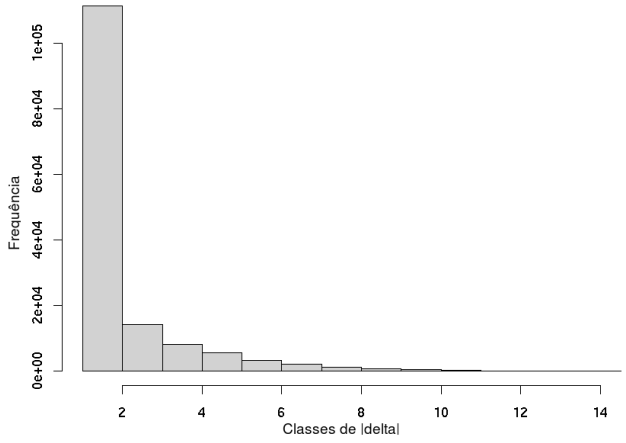

Figura 5.29: $DF_{(25)}$; $[200, 250)$; α=100

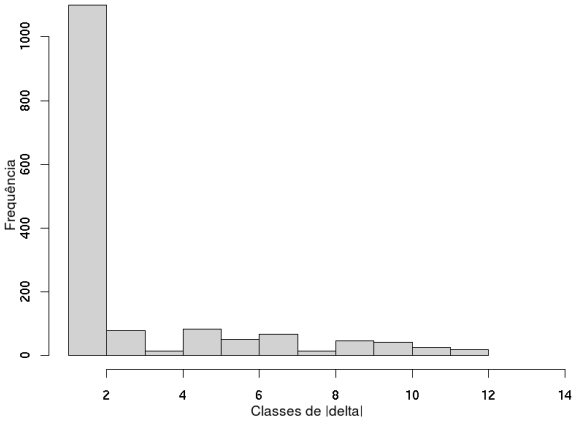

Figura 5.30: $DF_{(25)}$; $[250, 300)$; α=1

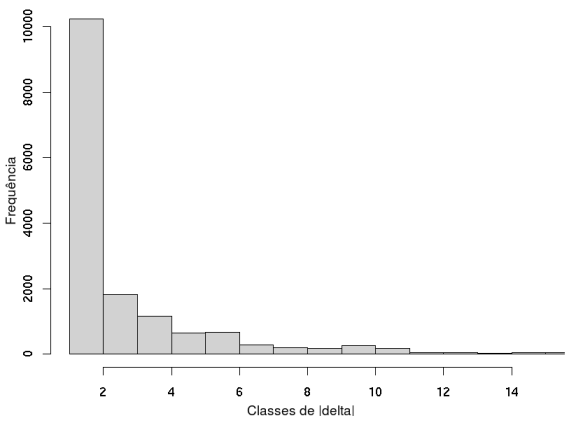

Figura 5.31: $DF_{(25)}$; $[250, 300)$; α=10

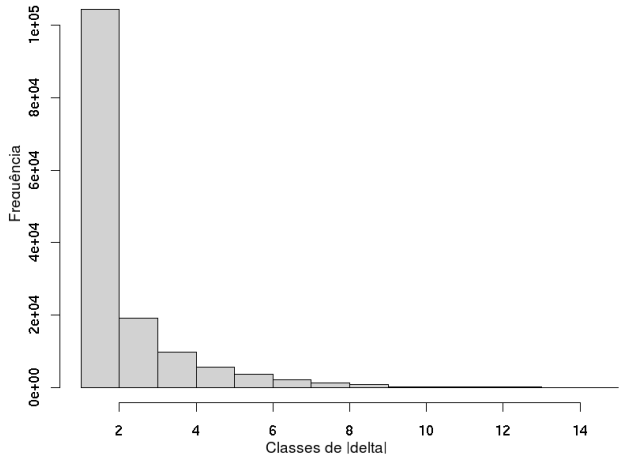

Figura 5.32: $DF_{(25)}$; $[250, 300)$; α=100

Série $S_{\|V\|}$	$\mathcal{H}_{np}$	$\mathcal{H}_{Ag}$	$\mathcal{H}_{Wav}$	$\mathcal{H}_{lW}$	n
S_{05}	0.709	0.777	0.959 ± 0.0313539	0.658	81734
S_{10}	0.756	0.892	0.860 ± 0.1269334	0.619	62139
S_{15}	0.761	0.928	0.651 ± 0.1438138	0.596	200323
S_{20}	0.766	0.910	0.549 ± 0.1382769	0.571	469323
S_{25}	0.810	0.916	0.542 ± 0.1370745	0.591	395791
S_{30}	0.800	0.927	0.512 ± 0.1405020	0.603	87918
S_{35}	0.792	0.930	0.549 ± 0.1430374	0.613	121022
S_{40}	0.775	0.931	0.506 ± 0.1383773	0.601	42844
S_{45}	0.765	0.915	0.524 ± 0.1385613	0.592	72405
S_{50}	0.751	0.927	0.489 ± 0.1347726	0.585	37789

Tabela 5.5: Parâmetro de Hurst ($\mathcal{H}$) para as séries temporais $S_{|V|}$

A tabela 5.5 apresenta as estimativas dos valores de $\mathcal{H}$ para cada uma das técnicas selecionadas, juntamente com o total n de elementos considerados para cada série. Somente na técnica *Wavelet* $\mathcal{H}_{Wav}$, há uma possibilidade de algumas séries não apresentarem auto-similaridade por conta da semi-largura esquerda do intervalo de confiança. Por outro lado, todas as técnicas indicaram a presença de auto-similaridade nas séries, inclusive a *Wavelet*, se para tanto somente o valor $\mathcal{H}$ ou a semi-largura direita do intervalo forem considerados.

Verificamos ainda as mesmas características acima discutidas a partir de séries temporais $|\delta|$ geradas sob variadas circunstâncias, tais como: variação do tráfego de fundo da rede, mudança de modelo de propagação e variação da topologia da WMN, consultar anexo C. A reprodução destas e de outras séries pode ser feita com auxílio dos *patchs* e *scripts* disponíveis no anexo D.1 . A análise das séries em diferentes escalas de tempo pode ser feita com o auxílio do programa em linguagem C disponível no anexo D.2.

Capítulo 6

Conclusões e trabalhos futuros

O presente trabalho apresentou uma avaliação sobre a questão da manutenção de rotas ótimas em WMNs 802.11. Após realizar uma fundamentação acerca dos aspectos que agravam o recálculo das rotas em tais redes (por exemplo variabilidade dos pesos associados aos enlaces, potencial de rápido crescimento da rede, roteadores com recursos computacionais escassos) nós discutimos a estratégia adotada pela RFC 3626 (protocolo de roteamento OLSR) para lidar com esse problema.

Em linhas gerais, tanto no atual padrão do OLSR como no seu potencial futuro substituto, o padrão emergente OLSR2, é estabelecido que o recálculo da tabela de roteamento deve ocorrer *sempre* que uma alteração de topologia é detectada, o que não necessariamente implica na alteração da tabela de roteamento. Adicionalmente, o OLSR estabelece uma variação do algoritmo de Dijkstra para realizar o recálculo das rotas ótimas para todos os destinos, quer tenham sido afetados quer não. Por sua vez, o padrão emergente OLSR2 não prescreve nenhum algoritmo para realizar o (re)cálculo das rotas, limitando-se somente a apresentar a variação do Dijkstra como um exemplo de solução.

Neste trabalho nós investigamos o uso da abordagem BIC como uma alternativa à proposta no padrão do OLSR (e do OLSR2) para o recálculo da tabela de roteamento

em WMNs 802.11. A alternativa diz respeito tanto ao critério para se recalcular a tabela quanto às características do algoritmo de recálculo propriamente dito. A característica chave de um algoritmo no modelo BIC é que seu tempo de execução é dado em termos da porção da tabela de roteamento afetada por eventuais alterações na topologia, i.e. pelo conjunto δ de roteadores cujas rotas ótimas precisam ser recalculadas juntamento com a quantidade de enlaces a eles adjacentes. Além de estudarmos o impacto do dinamismo topológico de *backbones* 802.11 na tabela de roteamento (especialmente, relevantes características estocásticas de $|\delta|$), realizamos um estudo de caso entre a solução proposta no padrão do OLSR e a proposta do algoritmo de Ramalingam e Reps (RRs), baseado no modelo BIC.

Após considerarmos elementos fundamentais de modelagem de canais 802.11, recorremos à metodologia de simulação de horizonte infinito a fim de fornecer uma aproximação empírica para o caso médio dos algoritmos de Dijkstra (representante do padrão do OLSR) e de RRs em WMNs 802.11. Nesta avaliação foram contabilizadas as operações de processamento de arestas e as operações sobre a fila de prioridades, comuns a ambos os algoritmos. Os resultados demonstraram que o algoritmo RRs apresentou um desempenho muito superior ao do Dijkstra.

Estabelecemos constantes a partir dos pontos simulados a fim de proceder com a estimativa da relação assintótica entre as classes de complexidade dos algoritmos. Verificamos que a frequente necessidade de se trocar as constantes a fim de *manter* a curva de tempo do RRs superior à do Dijkstra, é um indício a favor da afirmação de que a curva deste último (T_{Dij}) domina assintoticamente a daquele primeiro (T_{RRs}) por um fator variável, i.e. $T_{RRs} \in o(T_{Dij})$.

Com metodologia *similar* à utilizada para comparar o RRs e o Dijkstra, comparamos o RRs sob as métricas *Expected Transmission Count* (ETX) e *Link Delay* (LD), as quais utilizam diferentes abordagens para abstrair a qualidade de um enlace. Essa comparação teve o objetivo de avaliar quão diferente seria o comportamente

do algoritmo no caso médio sob formas diferentes de se pesar as arestas. Desta vez os resultados sugeriram que a classe Θ dos algoritmo no caso médio é insensível à métrica de roteamento. Isso pode ser intuitivamente entendido pelo fato de que, mesmo sob diferentes óticas, as métricas esmeram-se por acompanhar a variação da qualidade dos enlaces, refletindo assim em um impacto relativamente homogêneo no caso médio dos algoritmos.

A superioridade do algoritmo RRs pode ser explicada por dois motivos básicos. Em primeiro lugar devido ao desperdício de recursos computacionais decorrentes de se recalcular a tabela quando, de fato, ela não foi afetada pela atualização corrente no grafo. Dado que a configuração de descoberta de topologia do padrão do OLSR pode conduzir a grafos inconsistentes em relação à real topologia da rede (por exemplo a falha de um roteador MPR pode levar alguns roteadores a tal estado) [22; 76; 84], a atualização do grafo potencialmente também afeta a tabela de roteamento. Deste ponto de vista é razoável a suposição de invocar o procedimento de recálculo sempre que uma alteração é detectada no grafo somente.

Por outro lado, com o uso de configurações e técnicas não citadas pelo padrão do OLSR (por exemplo *fish-eye*, difusão tradicional) o problema da inconsistência é amenizado (conferir seção 4.2.2). Desta forma concluímos que a modificação da tabela de roteamento deveria ser critério único e suficiente para ocasionar recálculo. No intuito de cooperar com a evolução do futuro padrão OLSR2, sugerimos tal conclusão na lista de discussão do IETF-MANET (vide anexo B).

O segundo aspecto que explica a superioridade do algoritmo RRs diz respeito aos casos em que uma alteração no grafo leva a uma alteração na tabela de roteamento. Enquanto que em um roteador sob o padrão do OLSR a complexidade do recálculo sempre é definida em termos de *todos* os roteadores conhecidos, no algoritmo RRs (e qualquer outro baseado no modelo BIC) ela restringe-se ao conjunto δ de roteadores afetados (juntamento com os enlaces nele incidentes).

Avaliamos o comportamento da distribuição de frequências da cardinalidade do conjunto δ ($1 \leq |\delta| \leq |V| - |\{origem_s\}|$). Verificamos que maiores valores de $|\delta|$ conduzem às probabilidades mais baixas, enquanto que as probabilidades mais altas estão associadas aos menores valores de $|\delta|$. Esse resultado é especialmente importante no quesito da escalabilidade do OLSR, uma vez que a abordagem de recálculo da tabela de roteamento descrita no padrão do OLSR tem mostrado sinais de limitação ante ao potencial de crescimento das WMNs, conforme reportado em [5].

Verificamos ainda que as características estatísticas de $|\delta|$ que analisamos mantiveram-se similares sob diferentes escalas de tempo. Isso sugere uma natureza auto-similar de tal parâmetro o que, juntamente com uma comprovação formal acerca de sua distribuição de probabilidades, pode ser melhor investigado em trabalhos futuros. Isso constituiria um passo fundamental para novas investigações. Particularmente para o caso dos algoritmos no modelo BIC, isso representaria um importante passo para embasar *investigações analíticas* de suas classes de complexidade média.

Adicionalmente, dependendo do quão abrangente forem as conclusões obtidas neste trabalho acerca de $|\delta|$, ele poderia ser utilizado como condição de cumplicidade no padrão emergente OLSR2. Em outras palavras, em eventuais implementações do OLSR2 e, tendo-se em vista a questão da escalabilidade do roteamento em WMNs, a estratégia algoritmica para o recálculo da tabela de roteamento deveria estar sujeita a $O(f(|\delta|))$.

Referências Bibliográficas

[1] **Akaroa Project**. Disponível em:<http://www.cosc.canterbury.ac.nz/research/RG/net_sim/simulation_group/akaroa > Acesso em 23 Fev. 2009.

[2] **Analisador de Espectro Wi-Spy**. Disponível em:<http://www.wispy.com.br/wispy.html> Acesso em 23 Fev. 2009.

[3] **Arquivos de apoio à dissertação "Avaliação de Roteamento Incremental em Redes em Malha sem Fio Baseadas em 802.11"**. Disponível em:<http://grcm.dcc.ufam.edu.br/index.php/Downloads> Acesso em 23 Fev. 2009.

[4] **Athens wireless network**. Disponível em:<http://wind.awmn.net/> Acesso em 23 Fev. 2009.

[5] **BATMAN Overview**. Disponível em:<http://www.open-mesh.net/wiki/BATMANConcept> Acesso em 23 Fev. 2009.

[6] **Berlin Freifunk net**. Disponível em:<http://berlin.freifunk.net/> Acesso em 23 Fev. 2009.

[7] **Chanalyzer 3.2 for Wi-Spy 2.4x**. Disponível em:<http://www.metageek.net/node/658> Acesso em 23 Fev. 2009.

[8] **dd-wrt firmware**. Disponível em:<http://www.dd-wrt.com> Acesso em 23 Fev. 2009.

[9] **Experimental Evaluation of Dynamic All Pairs Shortest Path Algorithms**. Disponível em:<http://www.dis.uniroma1.it/˜demetres/experim/dsp/> Acesso em 23 Fev. 2009.

[10] **Freifunk.net**. Disponível em:<http://www.freifunk.net> Acesso em 23 Fev. 2009.

[11] **Hidden Histories**. Disponível em:<http://www.hivenetworks.net/tiki-view_articles.php> Acesso em 23 Fev. 2009.

[12] **IETF Mobile Ad-hoc Networks**:description of working group. Disponível em:<http://www.ietf.org/html.charters/manet-charter.html> Acesso em 23 Fev. 2009.

[13] **Leipzig Freifunk net**. Disponível em:<http://leipzig.freifunk.net/> Acesso em 23 Fev. 2009.

[14] **Network Stumbler**. Disponível em:<www.netstumbler.com> Acesso em 23 Fev. 2009.

[15] **ns-2 versão 29**. Disponível em:<http://www.isi.edu/nsnam/dist/ns-allinone-2.29.tar.gz> Acesso em 23 Fev. 2009.

[16] **OLSR modules for NS2 (OLSR, OLSR-ETX, OLSR-ML, OLSR-MD)**. Disponível em:<http://www.inf.ufrgs.br/ wlccordeiro/resources/olsr/> Acesso em 23 Fev. 2009.

[17] **OpenWRT Wireless Freedom**. Disponível em:<http://openwrt.org/> Acesso em 23 Fev. 2009.

[18] **RoofNet**. Disponível em:<http://pdos.csail.mit.edu/roofnet/doku.php> Acesso em 23 Fev. 2009.

[19] **Simulate 802.11b Channel Within ns2**. Disponível em:<www.comp.nus.edu.sg/ wuxiucha/research/reactive/publication/ Simulate80211ChannelWithNS2.pdf> Acesso em 23 Fev. 2009.

[20] **Software related to the LRD source project**. Disponível em:<http://www.richardclegg.org/lrdsources/software/> Acesso em 23 Fev. 2009.

[21] **Solution Brief – Wireless Mesh Network** . Disponível em:<www.nortel.com/solutions/wrlsmesh/collateral/nn106481.pdf> Acesso em 23 Fev. 2009.

[22] **The OLSR.ORG story**. Disponível em:<http://www.open-mesh.net/wiki/the-olsr-story> Acesso em 23 Fev. 2009.

[23] **UM-OLSR for ns-2**. Disponível em:<http://masimum.inf.um.es/?Software:UM-OLSR> Acesso em 23 Fev. 2009.

[24] **Wireless Networking in the Developing World**. Disponível em:<http://wndw.net/> Acesso em 23 Fev. 2009.

[25] **Wireless Update Patch for ns-2.29**. Disponível em:<http://www.telematica.polito.it/fiore/index.html> Acesso em 23 Fev. 2009.

[26] Abolhasan, M., Wysocki, T., and Dutkiewicz, E. ***A review of routing protocols for mobile ad hoc networks***. Elsevier Science, *Ad Hoc Networks*, ,n. 1, p. 1-22, Jan. 2004.

[27] Aguayo, D., Bicket, J., Biswas, S., Judd, G., and Morris, R. ***Link-level measurements from an 802.11b mesh network***. *ACM SIGCOMM Computer Communication Review*, Local, vol. 34, n. 4, p. 121-132, 2004.

[28] Akl, R., Tummala, D., and Li, X. ***Indoor propagation Modeling at 2.4 ghz for IEEE 802.11 networks***. In: IASTED International Multiconference on Wireless and Optical Communications, 6th , 2006, Banff. Proc. of Wireless Networks and Emerging Technologies. Banff:IASTED, 2006.

[29] ANATEL. ***Resolução 397 de 06/04/2005***:. Regulamento sobre condições de uso de radiofreqüências da faixa de 2.400 MHz a 2.483,5 MHz por equipamentos utilizando tecnologia de espalhamento espectral ou tecnologia de multiplexação ortogonal por divisão de freqüência, Abr. 2005.

[30] Ben-David, S., Chor, B., and Goldreich, O. ***On the theory of average case complexity***. In:Annual ACM Symposium on Theory of Computing, 21, 1989, Seattle. Proceedings of of the twenty-first annual ACM symposium on Theory of computing. Seattle:ACM, 1989.

[31] Campista, M. E. M., Esposito, P. M., Moraes, I. M., Costa, L. H. M. K., , Duarte, O. C. M. B., Passos, D. G., de Albuquerque, C. V. N., Saade, D. C. M., and Rubinstein, M. G. ***Routing Metrics and Protocols for Wireless Mesh Networks***. *IEEE Network*, Rio de Janeiro, vol. 22, n. 1, p. 6-12, Fev. 2008.

[32] Chakeres, I., and Perkins, C. ***Dynamic MANET On-demand (DYMO) Routing***. *IETF*, MANET Working Group, *draft* 17, Mar. 2009.

[33] Chaudhuri, S., and Zaroliagis, C. D. ***Shortest path queries in digraphs of small treewidth*** . *Springer Berlin*, Lectures Notes in Computer Science, Local, vol. 944, p. 244-255, 1995.

[34] Chen, J., Lee, Y.-Z., Maniezzo, D., and Gerla, M. ***Performance Comparison of AODV and OFLSR in Wireless Mesh Networks***. In:

Fifth Annual Mediterranean Ad Hoc Networking Workshop, 5, 2006, Sicily. Proceedings of Fifth Annual Mediterranean Ad Hoc Networking Workshop. Sicily:IFIP, 2006.

[35] Chung, P., Liew, S., SHA, K. C., and To, W. ***Experimental Study of Hidden-node Problem in IEEE802.11 Wireless Networks***. 2005.

[36] Clausen, T., Dearlove, C., and Adjih, C. ***RFC 5444 - Generalized Mobile Ad Hoc Network (MANET) Packet/Message Format***. *IETF*, Network Working Group, Fev. 2009.

[37] Clausen, T., Dearlove, C., and Jacquet, P. ***The Optimized Link State Routing Protocol version 2 (OLSRv2)***. *IETF*, MANET Working Group, *draft* 8 , Mar. 2009.

[38] Clausen, T., and Jacquet, P. ***RFC 3626 - Optimized Link State Routing Protocol (OLSR)***. *IETF*, Network Working Group, Oct. 2003.

[39] Cormen, T. H., Leiserson, C. E., Rivest, R. L., and Stein, C. ***Introduction to Algorithms***. 2.ed., USA:The MIT Press e McGraw-Hill, 2001.

[40] da Costa Cordeiro, W. L., Aguiar, E. S., Junior, W. A. M., Abelém, A. J. G., and Stanton, M. A. ***Providing Quality of Service for Mesh Networks Using Link Delay Measurements***. In: International Conference on Computer Communications and Networks, 16th, 2007, Honolulu. AProceedings of ICCCN 2007. Hawaii:IEEE Communications Society, 2007.

[41] de Albuquerque, C. V. N., Saade, D. C. M., Passos, D. G., Teixeira, D. V., Leite, J., and andand Luiz Cláudio Schara Magalhães, L. E. N. ***GT-Mesh, Rede Mesh de Acesso Universitário Faixa Larga***

***Sem Fio**: RT3 - Avaliação dos resultados do protótipo*. Niterói: Universidade Federal Fluminense, 2006. Relatório Técnico.

[42] de Couto, D. S. J., Aguayo, D., Bicket, J., and Morris, R. ***A high-throughput path metric for multi-hop wireless routing***. In: International Conference on Mobile Computing and Networking, 9, 2003, San Diego. Proceedings of the 9th annual international conference on Mobile computing and networking. San Diego, CA, USA:ACM, 2003.

[43] Dijkstra, E. W. ***A note on two problems in connection with graphs***. *Numerische Mathematik*, vol. 1, p. 269-271, 1959.

[44] Duhau, D., Queiroz, S., and Mota, E. ***Análise Comparativa de Heurísticas de Seleção de MPR Visando Tráfego de Voz em Redes em Malha sem Fio***. Manaus:UFAM, 2008. Relatório Técnico.

[45] Even, S., and Gazit, H. ***Updating distances in dynamic graphs***. *Methods of Operations Research*, vol. 49, p. 371-387, 1985.

[46] Faccin, S., Wijting, C., Kneckt, J., and Damle, A. ***Mesh Wlan Networks: Concept and System Design***. *IEEE Wireless Communications*, Wireless Mesh Networking, vol. 13, n. 2, p. 10-17, Abr. 2007.

[47] Faria, D. B. ***Modeling Signal Attenuation in IEEE 802.11 Wireless LANs*** . Palo Alto: Stanford University, 2005. Relatório Técnico.

[48] Flickenger, R. et al. ***Redes sem fio no Mundo em Desenvolvimento***. 2.ed. Porto Alegre:Hacker Friendly, 2008.

[49] Franciosa, P. G., Frigioni, D., and Giaccio, R. ***Semi-Dynamic Shortest Paths and Breadth-First Search in Digraphs***. *Springer-*

Verlag, Lectures Notes in Computer Science, London, UK, n. 1200, p. 33–46, Jan. 1997.

[50] Frigioni, D., Ioffreda, M., Nanni, U., and Pasqualone, G. ***Experimental analysis of dynamic algorithms for the single source shortest path problem***. *ACM Journal of Experimental Algorithmics*, New York, vol. 3, 1998.

[51] Frigioni, D., Marchetti-Spaccamelac, A., and Nannic, U. ***Fully Dynamic Algorithms for Maintaining Shortest Paths Trees***. *Journal of Algorithms*, TITULODAEDICAO(sehouver), Local, vol. 34, n. 2, p. 251-281, Fev. 2000.

[52] Furasté, P. A. ***Normas Técnicas para o trabalho científico****: elaboração e formatação*. 14.ed., Porto Alegre:Dáctilo-Plus, 2006.

[53] Ganesan, D., and Estrin, D. ***Complex Behavior at Scale: An Experimental Study of Low-Power Wireless Sensor Networks***. Dartmouth: Dartmouth College Computer Science, 2003. Relatório Técnico.

[54] Gokhale, D., Sen, S., Chebrolu, K., and Raman, B. ***On the Feasibility of the Link Abstraction in (Rural) Mesh Networks***. In: IEEE Conference on Computer Communications, 27.ed., 2008, Phoenix. Proceedings of 27th IEEE INFOCOM. Phoenix:IEEE, 2008.

[55] IEEE802.11. ***802.11 Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specification:***. ago. 1999, <http://www.ieee802.org/11/>, Acesso em 23 Fev. 2009.

[56] IEEE802.11b. ***Wireless LAN Medium Access Control (MAC) and***

Physical (PHY) Layer Specification:. High Speed Physical Layer Extensions in the 2.4 GHz Band, supplement to IEEE 802.11 Standard, Sept.

[57] IEEE802.11G. ***Wireless LAN Medium Access Control (MAC) and Physical (PHY) Layer Specification***:. Further Higher Data Rate Extension in the 2.4GHz Band. 2003.

[58] IEEE802.11S. ***IEEE p802.11s;/d1.08***. draft amendment to standard IEEE 802.11: ESS mesh networking, "work in progress", 2007.

[59] INTERSIL. ***Intersil FA3861B: Direct Sequence Spread Spectrum Baseband Processor***. 2000.

[60] JAIN, R. ***Art of Computer Systems Performance Analysis Techniques For Experimental Design Measurements Simulation And Modeling***. 1 .ed., New York:Wiley Computer Publishing, John Wiley & Sons, Apr. 1991.

[61] KATRIEL, I., AND HENTENRYCK, P. V. ***Bounded Incremental Single-Source Shortest-Paths***. Disponível em:<http://www.cs.brown.edu/people/pvh/bisssp.pdf> Acesso em 23 Fev. 2009.

[62] KIM, S., LEE, S.-J., AND CHOI, S. ***The Impact of IEEE 802.11 MAC Strategies on Multi-Hop Wireless Mesh Networks***. In: IEEE Workshop on Wireless Mesh Networks, 2nd, 2007, Reston VA,. Proc. of 2nd Wireless Mesh Networks . Reston:IEEE, 2006.

[63] KLEIN, P., RAO, S., RAUCH, M., AND SUBRAMANIAN, S. ***Faster shortest-path algorithms for planar graphs***. In: Annual ACM Symposium on

Theory of Computing, 26, 1994, Montreal, Quebec . Proceedings of the twenty-sixth annual ACM symposium on Theory of computing. ACM symposium on Theory of computing. Monteral:ACM, 1994.

[64] Koksal, C. E., and Balakrishnan, H. ***Quality-Aware Routing Metrics For Time-Varying Wireless Mesh Networks***. *IEEE Journal on Selected Areas in Communications*, , vol. 24, p. 1984-1994, Novembro 2006.

[65] Kotz, D., Newport, C., and Elliott, C. ***The mistaken axioms of wireless-network research***. UCLA Computer Science, 2002. Relatório Técnico.

[66] Leland, W., Taqqu, M., , Willinger, W., and Wilson, D. On the self-similar nature of Ethernet traffic (extended version). *IEEE/ACM Trans. Netw.*, vol. 2, n. 1, p. 1-15, 1994.

[67] Mahrenholz, D. ***Providing QoS for Publish/Subscribe Communication in Dynamic Ad-Hoc Networks***. Magdeburg: Universitat Magdeburg, 2006. Tese (Doutorado em Informática), Universitat Magdeburg, 2006.

[68] Malkin, G. ***RFC 2453 - Routing Information Protocol (RIP) Version 2***. *IETF*, Network Working Group, Nov. 1998.

[69] McLeod, A. I., Yu, H., , and Krougly, Z. L. ***Algorithms for Linear Time Series Analysis: With R Package***. *Journal of Statistical Software*, TITULODAEDICAO(sehouver), v. 23, n. 5, Dec. 2007.

[70] Mhatre, V. Enhanced wireless mesh networking for ns-2 simulator. *SIGCOMM Comput. Commun. Rev. 37*, 3 (2007), 69–72.

[71] Moreira, J. L. G., Mota, E., Jr., E. N. S., Carvalho, L., Queiroz, S., and Hoene, C. ***Voice over Wireless Mesh Networks: A Case Study***

in the Brazilian Amazon Region during the Rainy Season. In: International Workshop on Heterogeneous Wireless Networks, 5, 2009, Bradford. Proceedings of 5th International Workshop on Heterogeneous Wireless Network (to appear) . Bradford:IEEE, 2009.

[72] Mota, E. *Performance of Sequential Batching-based Methods of Output Data Analysis in Distributed Steady-state Stochastic Simulation*. Berlin: Technical University of Berlin, 2002. Tese (Doutorado em Ciência da Computação), Technical University of Berlin, 2002.

[73] Moy, J. ***RFC 1771 - A Border Gateway Protocol 4 (BGP-4)***. *IETF*, Network Working Group, Mar. 1995.

[74] Moy, J. ***RFC 2328 - Open Shortest Path First (OSPF) Version 2***. *IETF*, Network Working Group, Abr. 1999.

[75] Nascimento, A., Queiroz, S., Mota, E., Galvão, L., and Nascimento, E. ***Influence of Propagation Modeling on VoIP Quality Performance in Wireless Mesh Network Simulation***. In: IEEE/ACM Modeling, Analysis and Simulation of Computers and Telecommunication Systems, 2008, Baltimore. Proceedings of MASCOTS, Baltimore:IEEE, 2008.

[76] Nascimento, A., Queiroz, S., Mota, E., Galvão, L., and Nascimento, E. ***Influence of Routing Protocol on VoIP Quality Performance in Wireless Mesh Backbone***. In: International Workshop on Future Multimedia Network , 1st, 2008, Cardiff. Proceedings of NGMAST, Next Generation Mobile Applications, Services and Technologies. Cardiff:IEEE, 2008.

[77] Oran, D. ***RFC 1142 - OSI Intermediate System to Intermediate System (IS-IS) Intra-domain Routing Protocol***. *IETF*, Network Working Group, Abr. 1990.

[78] Passos, D., Teixeira, D., Muchaluat-Saade, D., Schara, L., and Albuquerque., C. ***Mesh Network Performance Measurements***. In: International Information and Telecommunication Technologies Symposium, 5, 2006, Cuiabá. Anais do 5th International Information and Telecommunication Technologies Symposium. Cuiabá:, 1999.

[79] Pawlikowski, K. ***Steady-state simulation of queueing processes: survey of problems and solutions***. *ACM Computing Surveys*, vol. 22, n. 2, p., 1990.

[80] Pei, G., Gerla, M., and wei Chen, T. *Fisheye State Routing: A Routing Scheme for Ad Hoc Wireless Networks*. In: IEEE International Conference on Communications, Global Convergence Through Communications, vol. 1, 2000, New Orleans. Proceedings of ICC 2000. New Orleans:IEEE, 2000.

[81] Perkins, C. E., Belding-Royer, E. M., and Das, S. R. ***RFC3561 – Ad hoc On-Demand Distance Vector (AODV) Routing***. *IETF*, Network Working Group, Jul. 2003.

[82] Qiu, L., Zhang, Y., Wang, F., Han, M. K., and Mahajan, R. ***A general model of wireless interference***. In:ACM International Conference on Mobile Computing and Networking ,13th , 2007, Quebec. Proceedings of the 13th annual ACM International Conference on Mobile Computing and Networking. Quebec:ACM, 2007.

[83] Queiroz, S. *Redes Mesh sem fio e os Centros Urbanos*. **Amazonas em tempo**. Manaus, 27 mar. 2008. InterMais: Caderno de Ciência e Tecnologia. Página: 5.

[84] Queiroz, S., Nascimento, A., Mota, E., Carvalho, L. S., and Nascimento, E. ***Comparative analysis of routing protocols for VoIP in a***

Wireless Mesh Backbone: a user perspective. *Int. J. Internet Protocol Technology*, vol. 3, n. 4, p. 216-223, Dec. 2008.

[85] Queiroz, S., Nascimento, A., Mota, E., Oliveira, A., Galvão, L., and Silva, E. ***Impact Evaluation of Radio Propagation Models on Performance Parameters of Application Layer in Wireless Mesh Backbone Simulation***. In:IEEE Symposium on Computers and Communications, 13th, 2008, Marrakech. Proceedings of IEEE Symposium on Computers and Communications. Marrakech:IEEE, 2008.

[86] Ramalingam, G. ***Bounded Incremental Computation***. Madison: UWM, 1996. Tese (Doutorado em Ciência da Computação), University of Wisconsin at Madison, 1996.

[87] Ramalingam, G., and Reps, T. ***An incremental algorithm for a generalization of the shortest-path problem***. *Academic Press, Inc, J. Algorithms, Duluth, vol. 21, n. 2, p. 267-305, MES e 1996.*

[88] Rappaport, T. S. ***Wireless Communication****: Principles and Practice*. 2.ed., Prentice Hall, 2001.

[89] Saade, D. C. M., Gomes, A. G., Carranoa, R. C., Magalhães, L. C. S., Albuquerque, C. V. N., and Tarouco, L. R. ***Multihop MAC: Desvendando o Padrão 802.11s***. In: Gaspary, Luciano Pascoal. **Livro de Minicursos do Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos**. 26.ed., Rio de Janeiro: SBRC, 2008.

[90] Schmidt-Eisenlohr, F., Letamendia-Murua, J., Torrent-Moreno, M., and Hartenstein, H. ***Bug fixes on the IEEE 802.11 DCF module of the network simulator ns-2.28***. Karlsruhe: Fakultät für Informatik, Institut fur Telematik, 2006. Relatório Técnico.

[91] Sridhara, V., and Bohacek, S. ***Realistic propagation simulation of urban mesh networks***. *Computer Networks*, New York, NY, USA, vol. 51, n. 12, p. 3392–3412, Jan. 2007.

[92] Takai, M., Martin, J., and Bagrodia, R. ***Effects of Wireless Physical Layer Modeling in Mobile Ad Hoc Networks***. In: The ACM International Symposium on Mobile Ad Hoc Networking and Computing, 2001, Long Beach. Proceedings of MobiHoc 2001. Long Beach, CA, New York:ACM, 2001.

[93] Tavares, E. ***Um Estudo de Voz sobre IP em Redes em Malha 802.11***. Niterói: UFF, 2008. Dissertação (Mestrado em Engenharia de Telecomunicações), Escola de Engenharia, Universidade Federal Fluminense, 2008.

[94] Tonnesen, A. ***Impementing and extending the Optimized Link State Routing Protocol*** . Oslo: Unik, 2004. Dissertação (Departament of Informatics), University Graduate Center, University of Oslo, 2004.

[95] Tummala, D. ***Indoor Propagation Modeling at 2.4 ghz for ieee 802.11 networks***. Texas:University of North Texas, 2005. Master of Science (Engineering Technology), Department of Engineering Technology, University of North Texas, 2005.

[96] Wormsbecker, I., and Williamson, C. ***On Channel Selection Strategies for Multi-Channel MAC Protocols in Wireless Ad Hoc Networks***. In: Wireless and Mobile Computing, Networking and Communications, 2, 2006, Montreal. Anais do 2nd IEEE International Wireless and Mobile Computing, Networking and Communications. Montreal:IEEE, 2006.

[97] Xue, F., and Kumar, P. ***The number of neighbors needed for con-***

nectivity of wireless networks. *Springer*, Wireless Network, Netherlands, vol. 10, n. 2, p. 169-181, Mar. 2004.

[98] YEE, J., AND PEZESHKI-ESFAHANI, H. ***Understanding Wireless LAN Performance Trade-Offs***. *Communication Systems Design*, Wireless LAN Design, p. 32-35, Nov. 2002.

[99] ZIVIANI, N. ***Projeto de Algoritmos***: *Com implementações em C e Pascal*. 2.ed., São Paulo:Pioneira Thonsom Learning, 2004.

Apêndice A

Discussão com pesquisadores e desenvolvedores

A.1 Questionamento à Irit Katriel sobre algoritmo proposto em [61]

```
from Saulo Jorge Queiroz <ssaulojorge@gmail.com>
to irit@cs.brown.edu
date 11 July 2007 11:59
subject A doubt about "Bounded Incremental Single-Source Shortest-Paths"
mailed-by gmail.com

from Saulo Jorge Queiroz <ssaulojorge@gmail.com>
to irit@daimi.au.dk
date 12 July 2007 11:35
subject Fwd: A doubt about "Bounded Incremental Single-Source Shortest-Paths"
mailed-by gmail.com
```

Hi Irit,

I'm Saulo Jorge from Federal University of Amazonas, Brazil.

Congratulation for your paper "Bounded Incremental Single-Source Shortest-Paths". I read it (from http://www.cs.brown.edu/people/pvh/bisssp.pdf) and have a doubt:

Instead $D(G', y) < D(G, y)$***, in "Definition 5", Wouldn't be "***$D(G', y) > D(G, y)$***"?***

Best regards,
Saulo Jorge bq

A.2 Email para Francisco Ros sobre um-olsr [23]

```
from Saulo Jorge Queiroz <ssaulojorge@gmail.com>
to fjros@um.es,
Weverton Cordeiro <wevertoncordeiro@gmail.com>
date 5 February 2009 12:59
subject um-olsr routing
mailed-by gmail.com
```

Hi Francisco,

I'm an um-olsr user. I noticed that the rtable_computation() procedure is called only by OLSR::nb_loss and OLSR::recv_olsr. According to RFC3626 ...

... the routing table is updated when a change is detected in either:

- the link set,

- the neighbor set,

- the 2-hop neighbor set,

- the topology set,

- the Multiple Interface Association Information Base,

So, shouldn't the rtable_computation be called when an topology tuple or a 2-hop tuple is removed? In others words, after OLSR::rm_nb2hop_tuple, OLSR::rm_topology_tuple procedures ?

best.

– Saulo Jorge bq - "In theory, there is no difference between theory and practice; In practice, there is." – Chuck Reid

A.2.1 Resposta

```
from Francisco J. Ros <fjros@um.es>
to Saulo Jorge Queiroz <ssaulojorge@gmail.com>
cc Weverton Cordeiro <wevertoncordeiro@gmail.com>
date 6 February 2009 08:53
subject Re: um-olsr routing
mailed-by um.es
```

Hi Saulo,

I think you're right. I'll take a deeper look into it by next week and will try to update the implementation.

Thanks for your feedback :-)

Regards, fran

A.3 Email para Weverton Cordeiro sobre etx-um-olsr [16]

```
from Saulo Jorge Queiroz <ssaulojorge@gmail.com>
to weverton.cordeiro@inf.ufrgs.br,
wevertoncordeiro@gmail.com
date 29 November 2007 13:07
subject Sobre consumo de memória
mailed-by gmail.com
```

Oi Weverton,

Estou fazendo simulacoes com o OLSR_ETX de horizonte infinito usando a ferramenta Akaroa (voce já o utilizou?).

Nestas condicoes a simulação não pára enquanto os parametros que estou avaliando nao satisfizer o nivel de confiabilidade exigido, assim o término da simulação já não depende do tempo que configuramos no script tcl.

Notamos que algumas simulaçoes foram "killadas" pelo SO. No /var/log/syslog acusava "out of memory". Pelo top do shell observamos que o consumo de memória do processo ns crescia continuamente no tempo. Assim nossas simulacoes com o OLSR só terminavam com sucesso quando "afrouxávamos" a exigencia do nivel de confiança. Para maiores confiabilidades, onde o processo demora muito mais por exigir inúmeras amostras da simulacao, o consumo de memória tornava-se proibitivo pelo SO.

Notei que com o AODV e com o DSDV o valor do consumo de memória se estabilizava.

Fiz ainda outro teste, removi todos os tráfegos e configurei o olsr para enviar pacotes HELLO e TC a cada 2000 e 5000, respectivamente. Desta vez ainda notei o crescimento contínuo no consumo de memória do processo porém tal crescimento mostrou muito mais lento. Não sou nenhum expert nem em C++ nem em programacao pra ns, parece me que há aí uma relação entre mensagens e memória, nao sei ao certo.

Verifiquei a funcao recv, vi que ela chama a recv_olsr_etx, a qual invoca o Packet::free, que por sua vez invoca o Packet::init. Mas nao consegui enxerga nenhum anormalidade.

Diante disso voce concordar que há alguma anomalia neste sentido? Caso positivo, com sua experiencia, voce suspeita em que regiao de código ou em que classe/arquivo pode estar tal problema? Seria de grande utilidade para me ajudar a detectar se há realmente e qual o problema.

Obrigado mais uma vez,

–

Saulo Jorge bq

A.3.1 Resposta Final

```
from Cordeiro, Weverton <weverton.cordeiro@inf.ufrgs.br>
reply-to weverton.cordeiro@inf.ufrgs.br
to Saulo Jorge Queiroz <ssaulojorge@gmail.com>
date 21 February 2008 17:24
subject Novos patchs para o OLSR
mailed-by gmail.com
```

Olá Saulo, tudo bem?

Queira me desculpar a demora em responder com as modificações ao OLSR. Pensei que nesse período teria alguma folga no mestrado, mas foi justamente nesse período que as coisas complicaram. Além disso, acabei vindo a trabalho em Belém, e portanto fiquei com a missão de conduzir dois projetos em paralelo, fora o OLSR.

Mas bem, finalmente consegui fazer algumas modificações para (tentar) resolver o problema de consumo de memória no OLSR. Em http://www.inf.ufrgs.br/ wlccordeiro/resources/olsr/ Poderás encontrar os novos patchs (olsr-0.2.0-ns-2.3) para o código modificado do OLSR. Acredito que terás de fazer alguma modificação nos teus scripts, uma vez que eu renomeiei o projeto de OLSR_ETX para OLSR, uma vez que o nome ETX já não estava mais adequado. No diretório 'sample' poderás encontrar scripts de exemplo para a simulação.*

Fico no aguardo de um feedback seu, caso ainda estejas continuando a trabalhar com simulação em redes mesh com o OLSR.

Att,
– Weverton Luis da Costa Cordeiro
Msc Student in Computer Science
Federal University of Rio Grande do Sul - UFRGS
http://www.inf.ufrgs.br/ wlccordeiro/

A.3.1.1 Contra resposta final

```
from Saulo Jorge Queiroz <ssaulojorge@gmail.com>
to Weverton Cordeiro <weverton.cordeiro@inf.ufrgs.br>
date 14 March 2008 10:07
subject Re: Novos patchs para o OLSR
mailed-by gmail.com
```

Weverton, A memória 'ficou estável nos testes que fiz. valeu.

Fiz um patch para o ns-2.33-rc2. E também funcionou adequadamente. Porem no novo ns ha um novo mac melhorado, curiosamente somente com esse MAC houve um sensível e continuo crescimento de memória, eu , por hora, usarei o olsr com o ns-2.33 com o mac normal (que nao apresentou problemas).

Valeu!!

Apêndice B

Discussão na lista do grupo IETF-MANET

A discussão que segue-se diz respeito aos critérios necessário para se executar o recálculo da tabela de roteamento. Até o momento de escrita deste apêndice tal sugestão encontrava-se sob debate. Contudo ela pode ser acompanhada a qualquer instante através do *web-link*:

http://www.ietf.org/mail-archive/web/manet/current/msg10971.html.

```
from Saulo Jorge Queiroz <ssaulojorge@gmail.com>
to manet@ietf.org
date 3 April 2009 09:26
subject OLSR2 - Routing Set Updates
mailed-by gmail.com
```

Hello,

I've worked on algorithms for maintaining Routing Set in OLSR. Could anyone clarify if a change of the known symmetric links and/or attached networks necessarily changes the routing set?

– Saulo Jorge bq - "Os maus de coração não amam nunca; apenas se alegram entre comparsas" – Caio Fábio

B.1 Resposta: Jean de Smith

```
from Jens de Smit <jfdsmit@gmail.com>
to Saulo Jorge Queiroz <ssaulojorge@gmail.com>
cc manet@ietf.org
date 3 April 2009 09:35
subject Re: [manet] OLSR2 - Routing Set Updates
mailed-by gmail.com
```

Well.... That actually depends on your algorithm, doesn't it? But, unless your algorithm selects each and every node as router, there's always a topology where the loss of 1 link does not require the routing set to change to keep the network connected.

Regards,

Jen

B.1.1 Resposta à Jean de Smith

```
Saulo Jorge Queiroz <ssaulojorge@gmail.com>
to jfdsmit@gmail.com
cc manet@ietf.org
date 4 April 2009 08:27
subject Re: [manet] OLSR2 - Routing Set Updates
mailed-by gmail.com
```

Up to now, I think it depends on more of topology discovery configuration of OLSR than SSP algorithm. I performed several (time infinite-horizon) simulations and noticed that proportion of times routing table changed could be drammatically reduced by turnning MPR (RFC3626) off and by using fish-eye algorithm (in comparison with RFC3626).

B.2 Resposta do projetista (OLSR2) Christopher Dearlove

```
Dearlove, Christopher (UK) <chris.dearlove@baesystems.com>
to Saulo Jorge Queiroz <ssaulojorge@gmail.com>,
manet@ietf.org
date 3 April 2009 09:40
subject RE: [manet] OLSR2 - Routing Set Updates
```

I think the summary (and we're open to changing the exact wording here if appropriate) of the situation would be

- Any change to anything that goes into the Routing Set (including the cases you note) has the potential to change the Routing Set.

- You ought to keep the Routing Set up to date. (Of course there may be quality of implementation issues - do I allow a slightly longer than necessary delay in which it's out of date, and if so how long?)

- If this means you have to recalculate the Routing Set, so be it. But if (for whatever reason) you can patch the set without recalculating it, so much the better. If you've got any particularly good ways to do so, please share them with us.

- The specification talks in terms of recalculating the Routing Set any time something changes, because that's a convenient way to specify the protocol.. But there's also a blanket "as if"that means you don't have to do it that way (in fact you don't even have to have a Routing Set in exactly that form).

- The Routing Set (and the set of MPRs) also have an additional freedom - while there's a suggested how to do it algorithm, the actual specification is anything that gives minimum path routes given the specified Topology Graph. So there's another permission to do what works for you.

- But don't lose sight of that if you try using outdated routes, for as long as you do, you will risk looping in the network, and that's not a good thing.

– Christopher Dearlove
Technology Leader, Communications Group
Networks, Security and Information Systems Department
BAE Systems Advanced Technology Centre
West Hanningfield Road, Great Baddow, Chelmsford, CM2 8HN, UK
Tel: +44 1245 242194 Fax: +44 1245 242124
BAE Systems (Operations) Limited
Registered Office: Warwick House, PO Box 87,
Farnborough Aerospace Centre, Farnborough, Hants, GU14 6YU, UK
Registered in England & Wales No: 1996687

B.2.1 Resposta à Christopher Dearlove

```
Saulo Jorge Queiroz <ssaulojorge@gmail.com>
to "Dearlove, Christopher (UK)" <chris.dearlove@baesystems.com>
cc manet@ietf.org
date 3 April 2009 10:50
subject Re: [manet] OLSR2 - Routing Set Updates
mailed-by gmail.com
```

The Ramalingan and Reps algorithm (and any one based on Bounded Incremental Computation Model) defines a dag Gd=(V,A') from the graph G=(V,A) in such a way that A' has all edges which belongs to current known shortest paths.

For every new edge information the Topology of graph G is necessarily updated but info about SPTree (pi[v] and d[v] for all v in G) and so, Gd, are updated only if edge inserted / deleted / updated (weight) belongs (belonged) to Gd. Once Gd is affected, the algorithm identify the set of routers for which shortest paths must be recomputed and do it. The running time is given in terms of delta (instead all $|V|$).

I think that routing set could be seen as Gd (i.e. the space complexity is the same as in Dijkstra)

Apêndice C

Resultados adicionais para séries temporais de $|\delta|$

Este apêndice apresenta algumas análises adicionas de $|\delta|$ tendo-se em consideração outras variáveis da rede. Os experimentos adicionais contempla o sudo de tráfego de fundo VoIP (seção C.1) e a variação do modelo de propagação (seção C.2). A configuração geral dos scripts é similar àquela discutida no capítulo 4 a apresentada no anexo D.1.

C.1 Análise de δ em WMN com tráfego de fundo

A séries temporais de $|\delta|$ a análisadas nesta seção, foram obtidas em um backbone WMN com tráfego de fundo VoIP. Variamos a quantidade de roteadores na grade, a saber, 9 (3×3), 16 (4×4) e 25 (5×5). Para o tráfego de fundo VoIP utilizamos 20 fluxos CBRs, correspondente a 10 chamadas bidirecionais, para mais detalhes conferir [84]. Assim foram geradas 3 séries temporais para $|\delta|$, as quais denotamos respectivamente por S_{VoIP9}, S_{VoIP16} e S_{VoIP25}.

C.1.1 Estimativas do parâmetro $\mathcal{H}$ para $S_{VoIP|V|}$

Série $S_{VoIP\|V\|}$	$\mathcal{H}_{np}$	$\mathcal{H}_{Ag}$	$\mathcal{H}_{Wav}$	$\mathcal{H}_{lW}$	n
S_{VoIP9}	0.732	0.890	0.822 ± 0.1289067	0.599	384135
S_{VoIP16}	0.668	0.879	0.785 ± 0.1256895	0.591	159770
S_{VoIP25}	0.739	0.879	0.753 ± 0.1287663	0.569	156560

Tabela C.1: Parâmetro de Hurst ($\mathcal{H}$) para as séries temporais de $|\delta|$ obtidas sob tráfego de fundo VoIP

C.1.2 Histogramas de $|\delta|$ em WMNs com tráfego de Fundo

As figuras C.1 a C.3 ilustram o histograma de $|\delta|$ para as WMNs consideradas na seção anterior. Tais distribuições concordam com as conclusões obtidas no capítulo 5

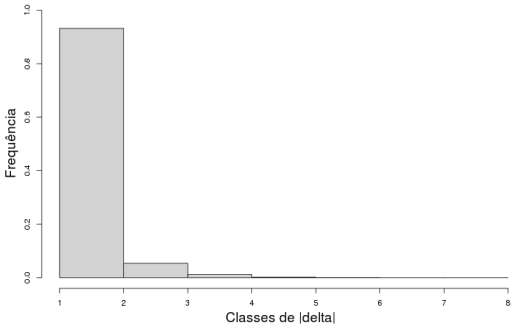

Figura C.1: DFR para WMN$(Voip)_{|V|=9}$

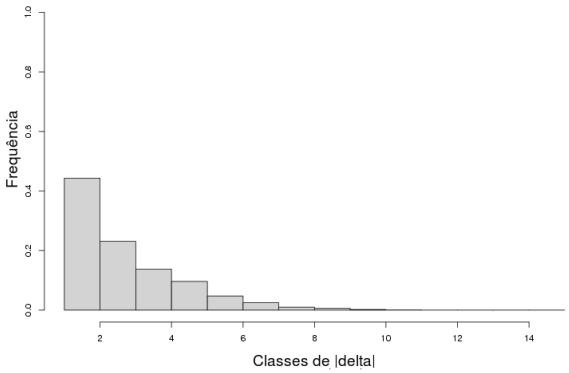

Figura C.2: DFR para WMN$(Voip)_{|V|=16}$

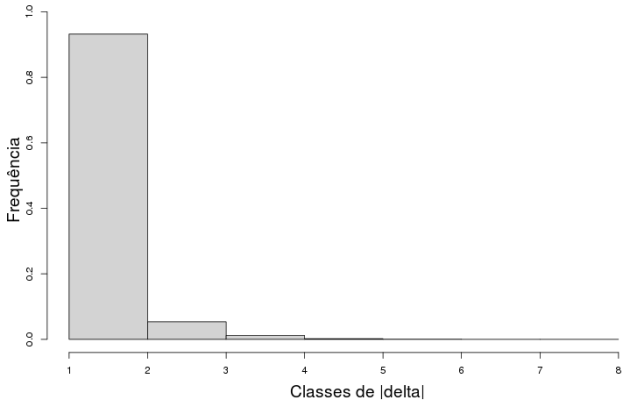

Figura C.3: DFR para WMN$(Voip)_{|V|=25}$

C.2 Análise de δ em WMN com outros modelos de propagação

Nesta seção apresentamos os valores de $\mathcal{H}$ correspondente a séries temporais de $|\delta|$ obtidas pela simulação do script apresentado no anexo D.1 sob os modelos de Rice e Two-Ray Ground. Utilizamos uma topologia linear com cinco roteadores.

C.2.1 Estimativas do parâmetro $\mathcal{H}$ para $S_{|PropModel|}$

Série S_{PModel}	$\mathcal{H}_{np}$	$\mathcal{H}_{Ag}$	$\mathcal{H}_{Wav}$	$\mathcal{H}_{lW}$	n
S_{Ricean}	0.675	0.662	0.809 ± 0.04033203	0.596	60684
$S_{2RayGround}$	0.655	0.649	0.809 ± 0.04033203	0.598	34513

Tabela C.2: Parâmetro de Hurst ($\mathcal{H}$) para as séries temporais de $|\delta|$ obtidas com modelos de propagação de Rice e 2-RayGround

Apêndice D

Script de simulação e software de apoio

D.1 Script de simulação

O *script* a seguir foi testado para executar no ns-2.29 sob as alterações do patch que disponibilizamos em [3].

```
set seed_ [lindex $argv 0]                      ;# semente do Shadowing
set qtdColunas [lindex $argv 1] ;# do grid
set qtdLinhas [lindex $argv 2] ;# do grid
set distancia [lindex $argv 3] ;# do grid

# ======================================================================
# Definicoes das opcoes
# ======================================================================
set val(chan)           Channel/WirelessChannel  ;# channel type
set val(prop)           Propagation/Shadowing    ;# radio-propagation model
set val(netif)          Phy/WirelessPhy          ;# network interface type
set val(mac)            Mac/802_11               ;# MAC type
set val(ifq)            Queue/DropTail/PriQueue  ;# interface queue type
set val(ll)             LL                       ;# link layer type
set val(ant)            Antenna/OmniAntenna      ;# antenna model
set val(ifqlen)         50                       ;# max packet in ifq
set val(nn)             [expr $qtdLinhas*$qtdColunas];# number of mobilenodes
set val(adhocRouting)   OLSR         ;# routing protocol
set val(x)              10000        ;# x coordinate of topology
set val(y)              10000        ;# y coordinate of topology


remove-all-packet-headers             ;# remove todos os cabecalhos do pacote
#add-packet-header RTP
add-packet-header IP
add-packet-header OLSR
add-packet-header LL
add-packet-header Mac


# NIC's Thermal Noise = Frame Receiver Sensitivity - 10
ErrorModel80211 noise1  -104                          ;# 1Mbps datarate thermal noise
ErrorModel80211 noise2  -101                          ;# 2Mbps datarate thermal noise
ErrorModel80211 noise55 -97                           ;# 5.5Mbps datarate thermal noise
ErrorModel80211 noise11 -92                           ;# 11Mbps datarate thermal noise
ErrorModel80211 shortpreamble 1                      ;# toggle 802.11 short preamble on/off
#ErrorModel80211 LoadBerSnrFile ber_snr_intersil.txt    ;# use Intersil BER/SNR empirical curves
ErrorModel80211 LoadBerSnrFile ber_snr_choi.txt       ;# use theoretical curves from del Prado
                                                      ;# Pavon and Choi, "Link AdaptationStrategy
                                                      ;# for IEEE 802.11 WLAN via Received Signal
                                                      ;# Strength Measurement"

##############################################################################################
# Gera uma semente diferente a cada execucao deste script
ns-random 0
##############################################################################################

#=======> Criacao dos roteadores com antenas OMNI
Antenna/OmniAntenna set X_ 0
```

```
Antenna/OmniAntenna set Y_ 0
Antenna/OmniAntenna set Z_ 2.5
Antenna/OmniAntenna set Gt_ 1.0;
Antenna/OmniAntenna set Gr_ 1.0;


# ======================================================================
# Interface wireless
# ======================================================================
Phy/WirelessPhy set CPThresh_   10.0                    ;# capture threshold
Phy/WirelessPhy set CSThresh_    3.9810717e-14           ;# same BPSK noise floor (nf at 1 Mbps -104 dBm ~ 3.9810717e-14 W)
Phy/WirelessPhy set RXThresh_   1.84952e-12             ;#-Pt 0.12589254117942 -fr 2.412e9 -pl 4.0 -std 6.0 -r 0.98 25
Phy/WirelessPhy set Pt_          0.12589254117942       ;# Tx power (Watt ~ 21 dBm ~ 125 mW, antenna gains included)
Phy/WirelessPhy set freq_        2.412e9                ;# channel frequency (Hz).
# canal 1  =   2.412e9
# canal 6  =   2.437e9
# canal 11 =   2.462e9
# canal 13 =   2.472e9                ;# channel frequency (Hz)

Phy/WirelessPhy set L_          1.0

#=======================> outras formas de setar: especificando o tipo dbm
#Phy/WirelessPhy set CSThresh_ [dbm -104]
#Phy/WirelessPhy set RXThresh_ [dbm -94]


Propagation/Shadowing set pathlossExp_  4.0;# path loss exponent
Propagation/Shadowing set std_db_       6.0;# standard deviation
Propagation/Shadowing set dist0_        1.0 ;# reference small distance
Propagation/Shadowing set seed_         $seed_ ;# 1973272912 ;# 0


# ======================================================================
# Parametros da MAC
# ======================================================================
#=======================>
Mac/802_11 set CWMin_                31
Mac/802_11 set CWMax_                1023
Mac/802_11 set SlotTime_             0.000020  ;# 20us
Mac/802_11 set SIFS_                 0.000010  ;# 10us
Mac/802_11 set PreambleLength_       144       ;# 144 bit
Mac/802_11 set ShortPreambleLength_  72        ;# 72 bit
Mac/802_11 set PreambleDataRate_     1.0e6     ;# 1Mbps
Mac/802_11 set PLCPHeaderLength_     48        ;# 48 bits
Mac/802_11 set PLCPDataRate_         1.0e6     ;# 1Mbps
Mac/802_11 set ShortPLCPDataRate_    2.0e6     ;# 2Mbps
Mac/802_11 set RTSThreshold_         2347      ;# bytes
Mac/802_11 set ShortRetryLimit_      7         ;# retransmissions
Mac/802_11 set LongRetryLimit_       4         ;# retransmissions
Mac/802_11 set newchipset_           true     ;# use new chipset, allowing a more recent
                                               ;# packet to be correctly received in place
                                               ;# of the first sensed packet
Mac/802_11 set dataRate_ 11Mb                  ;# unicast rate transmission
Mac/802_11 set basicRate_ 2Mb                  ;# broadcast rate transmission
Mac/802_11 set aarf_ true                     ;# 802.11 Adapt. Auto Rate Fallback

# ======================================================================
# Parametros da camada de Rede: Protocolo de Roteamento OLSR-ETX/ML/MD
# ======================================================================
Agent/OLSR set use_mac_              true
Agent/OLSR set debug_                false
Agent/OLSR set willingness           3
Agent/OLSR set hello_ival_           2
Agent/OLSR set tc_ival_              1

Agent/OLSR set mpr_algorithm_        5
Agent/OLSR set routing_algorithm_    2
# 2 = ETX. Para link delay apenas marque true no item correspondente
Agent/OLSR set link_quality_         2
Agent/OLSR set link_delay_           false
Agent/OLSR set fish_eye_             true


Agent/OLSR set tc_redundancy_        3
Agent/OLSR set c_alpha_              0.6


# ======================================================================
# Criando O Objeto simulador ns_, trace e nam
# ======================================================================
set ns_ [new Simulator]

#=========================> Abrindo arquivo de trace e escrevendo nele o trace
#set tracens_     [open ufam.tr w]
#$ns_ trace-all $tracens_

set tracefd [open "/dev/null" w] ;
#$ns_ use-newtrace
$ns_ trace-all $tracefd
```

```
#==========================> Criacao e configuracao da area
   set topo [new Topography]
   $topo load_flatgrid $val(x) $val(y)
   #The topography is broken up into grids and the default value of grid resolution is 1.
   #A diferent value can be passed as a third parameter to load_flatgrid {} above.

# ======================================================================
# NAM Configurations
# ======================================================================
#==========================> Abre o arquivo out.nam para escrita e rotula-o com nf
set namtracens_ [open "/dev/null" w]
#set namtracens_ [open "nam.nam" w]

#==========================> Escreve dados da simulacao relevantes para o nam no arquivo representado por nf.
#$ns_ namtrace-all-wireless $namtracens_ $val(x) $val(y)


# ======================================================================
# Criando O Objeto GOD-General Operations Director
# ======================================================================
create-god $val(nn)
#set god_ [create-god $val(nn)]


# ======================================================================
# Configuracao e Criacao dos nos (eh necessario primeiro configura-los)
# ======================================================================
#==========================> Configuracao dos nos
$ns_ node-config -adhocRouting $val(adhocRouting)\
-llType $val(ll) \
-macType $val(mac) \
-ifqType $val(ifq) \
-ifqLen $val(ifqlen) \
-antType Antenna/OmniAntenna \
-propType $val(prop) \
-phyType $val(netif) \
-channel [new $val(chan)] \
-topoInstance $topo \
-wiredRouting OFF \
-agentTrace OFF \
-routerTrace OFF \
-lossTrace OFF \
-macTrace OFF \
-toraDebug OFF \
-movementTrace OFF


for {set i 0} { $i < $val(nn)} {incr i} {
   set node_($i) [$ns_ node ]
   $node_($i) random-motion 0 ;# disable random motion
   }


# ======================================================================
# Configurando posicionamento dos nos
# ======================================================================
set indice -1
set dist2 $distancia

for {set i 1} { $i <= $qtdColunas} {incr i} {
  for {set j 1} {$j <= $qtdLinhas} {incr j} {
         incr indice
         #puts "node_($indice) set X_ [expr $distancia * $i]\n"
         #puts "node_($indice) set Y_ $dist2\n"
 $node_($indice) set X_ [expr $distancia * $i]
 $node_($indice) set Y_ $dist2
 $node_($indice) set Z_ 2

         set dist2 [expr $dist2 + $distancia]
       }
      set dist2 $distancia
  }
# ======================================================================

;# pega o id do processo do ns no sistema e depois grava num arquivo, para  continuar simulacao depois, kill -CONT <pid>
set pid_file [open "pid-ns-process" w]
set pid_ [pid]
puts $pid_file "$pid_"
close $pid_file


$ns_ run
```

D.2 Programa para a geração dos histogramas correspondente a diferentes escalas de tempo

O código a seguir recebe um arquivo de texto indicando a sequência o tempo em que uma mensagem chegou ao roteador (indicado pelo inteiro -1) bem como os valores de $|\delta|$ entre tais intervalos (indicados por números naturais não nulos). A exceção é a primeira linha, que deve indicar o total de mensagens que chegaram ao roteador, no formato `t n`, onde `n` é o total mencionado. A execução do programa resulta em arquivos contendo as DFs de $|\delta|$ nos intervalos 50,100,150,200,250,300 para as escalas de tempo 1, 10 e 100.

```
#include <stdio.h>

// Saulo Jorge, 2009
//This program generates



// at least 1200 deltas per interval should be took into account, otherwise simulate more!!!
int main(int argc,char *argv[])
{


  if (argc!=2)
  {            //argv[0]  argv[1]
   printf("Must be:./exec NUMBER_OF_ROUTERS\n");
   return 0;
  }

  int total_of_routers_on_WMN=atoi(argv[1]);

  //is automaticaly read from input file
  long int total_of_message_samples;

  //name of outputs files
  char frequenceFileName[ FILENAME_MAX ];
  const char prefix_of_fileNameFreqs[]="frequencesForScale_";

  //reads and processes infos which depends of total of messages in the input file
  char fileName[18]="afetados-d-dd";
  FILE *deltas_per_message_File=fopen(fileName, "r");
  char any;
  int dont_care;
  fscanf(deltas_per_message_File,"%c %d\n", &any, &total_of_message_samples);
  fclose(deltas_per_message_File);
  //

  //How many scales? Every scale will generate one frequence file (i.e. one graphic)
//   int current_time_scale=0;
   int current_time_scale[4]={ 1,10,100,1000};



   int total_of_scales=4;//atoi(argv[3]);

   //it will variate according to scale.
   //For instance: scale=0.1 -> 5 , scale=1 -> 50, ...
   int time_duration_of_delta_measurements=50;
   long int time_duration_of_delta_measurementsOnScale;
  //Total of x(axis) intervals is constant for all graphics.
  int constant_number_of_intervals=6;

  //files for each segment of each scale
  char fnameintervals[constant_number_of_intervals][FILENAME_MAX];
  FILE *fints[total_of_scales][constant_number_of_intervals];
  //how many delta samples took into account for each interval? i.e. size of scale
  // it message has about 10 to 15 deltas (in steady state)
  int number_of_messages_per_interval;


  //number of elements in delta set (i.e. number of affected routers)
  //if -1 indicates the end of processing of a entire message.
  //if -2 indicates the end of a link expe message processing.
  int delta_value;
  int index_of_time_scale;


  long int current_time_o_clock=0;
  long int time_on_scale=0;
```

```
 long int interval_number;
 int i,j;

  for( i=0; i<total_of_scales; i++ )
     for(j=1; j<=constant_number_of_intervals; j++ )
      {
        sprintf(fnameintervals[j], "%s%d", "frequencesForScale_", current_time_scale[i]);
        sprintf(fnameintervals[j], "%s%d", fnameintervals[j], (j));
        fints[i][j]=fopen(fnameintervals[j],"a");
     }

for (index_of_time_scale=0;index_of_time_scale<total_of_scales;index_of_time_scale++)
{

 //it changes only if a message arrives.
 //i.e. the number of messages processed at the router
 current_time_o_clock=0;



 //scale-esim frequence output file
  sprintf(frequenceFileName, "%s%d", prefix_of_fileNameFreqs, current_time_scale[index_of_time_scale]);
 FILE *frequences_File=fopen(frequenceFileName, "w");
//

 deltas_per_message_File=fopen(fileName, "r");
 //the first line was already pocessed (it is did only one time before scale's loops)
 //jumps the dont care line
 fscanf(deltas_per_message_File,"%c %d\n", &any, &dont_care);


  //larger scales have more samples at the first interval, which neutralizes transient effects
  //but shorter scales (0.1, 1) are negatively affected by transient. So we have remove it.

  //larger scales have more samples at the first interval, which neutralizes transient effects
  //for more details, see first comment at the top of the page.

  // We noticed that the total of deltas generated between the first and the 436-esim message is transient phase.
  // Param    Estimate       Delta  Conf TotalObs TransObs
  //     1     8.33857    0.414915  0.90    11772      436
  // #Results obtained using the Akaroa2(c) automatic parallel simulation manager
  i=0;
  while (i<436)
  {
    fscanf(deltas_per_message_File,"%d\n", &delta_value);
    if (delta_value==-1)
          i++;
  }

 //Remember: Clock changes only if a TC or HELLO message arrives!
 //So a time unit correspond to a message arriving.
 //number_of_messages_per_interval=time_duration_of_delta_measurements * current_time_scale[index_of_time_scale];
   int acabou=0;
   time_duration_of_delta_measurementsOnScale=time_duration_of_delta_measurements*current_time_scale[index_of_time_scale];
 while (!feof(deltas_per_message_File) && (!acabou) )
  {
    fscanf(deltas_per_message_File,"%d\n", &delta_value);
    switch (delta_value)
    {        //-1 denotes the time at a message arrives. So, we must increase clock.
     case -1: current_time_o_clock++;
              break;
     case -2: break; //the time at an link expires. For future proposes.
     default:
       {
       time_on_scale=current_time_o_clock*current_time_scale[index_of_time_scale];

      //Notice that every value of |delta| receive a particular label according to the current time.
      //i.e. the i-esim time interval (time_duration_of_delta_measurements ) in which that |delta| 'happens'.

    interval_number=(long int) current_time_o_clock/(long int)(time_duration_of_delta_measurementsOnScale);

    fprintf(fints[index_of_time_scale][interval_number+1],"%d\n",  delta_value);

    //fprintf(frequences_File,"%d\n", (interval_number*time_duration_of_delta_measurementsOnScale + delta_value));

      if ( current_time_o_clock >
           (long int) (constant_number_of_intervals*time_duration_of_delta_measurementsOnScale)
         )
            {
               acabou=1;
             //clock is already on the last time which interest us
             //Messages arrives after 300 units of time doesnt matter.
              fclose(deltas_per_message_File);
              fclose(frequences_File);
              break;
            }
       }
    }
  }
  if (!frequences_File)
```

```
    fclose(frequences_File);
   if (!deltas_per_message_File)
    fclose(deltas_per_message_File);
  }

   for( i=0; i<total_of_scales; i++ )
      for(j=1; j<=constant_number_of_intervals; j++ )
       {
         fclose(fints[i][j]);
       }

  return 0;
}
```