Universidade Federal Fluminense

LUCIANA ESTEVES NEVES HILARIO

Qualidade de Serviço em Redes Mesh

NITERÓI

2006

LUCIANA ESTEVES NEVES HILARIO

Qualidade de Serviço em Redes Mesh

Dissertação de Mestrado submetida ao Programa de Pós-Graduação em Computação da Universidade Federal Fluminense como requisito parcial para a obtenção do título de Mestre em Computação. Área de concentração: Sistemas Paralelos e Distribuídos.

Orientador:
JULIUS C. B. LEITE

Universidade Federal Fluminense

NITERÓI

2006

Ficha Catalográfica elaborada pela Biblioteca da Escola de Engenharia e Instituto de Computação da UFF

H641 Hilario, Luciana Esteves Neves.
Qualidade de serviço em redes mesh / Luciana Esteves Neves Hilario. - Niterói, RJ : [s.n.], 2006.
146 f.

Orientador: Julius C. B. Leite.
Dissertação (Mestrado em Ciência da Computação) - Universidade Federal Fluminense, 2006.

1. Sistemas paralelos e distribuídos. 2. Redes mesh. 3. Qualidade de serviço. 4. Framework (Programa de computador). I. Título.

CDD 004.3

Qualidade de Serviço em Redes *Mesh*

Luciana Esteves Neves Hilario

Dissertação de Mestrado submetida ao Programa de Pós-Graduação em Computação da Universidade Federal Fluminense como requisito parcial para a obtenção do título de Mestre em Computação.

Aprovada por:

Prof. Dr. Julius C. B. Leite, IC/UFF - Presidente

Prof. Dr. Célio Vinícus Neves de Albuquerque, IC/UFF

Prof. Dr. Carlos Alberto Maziero, PPPGIa-PUCPR

Niterói, 01 de dezembro de 2006.

Ao amor da minha vida, aos meus pais, irmã, avós e toda minha família.

Agradecimentos

Finalmente, o fim de um caminho de superação! Sinceramente, existiram dias que pensei que não fosse conseguir, que iria desistir no meio do caminho. Mas consegui chegar ao fim do mestrado! Não sozinha, mas com a ajuda de muitas pessoas, a quem gostaria de agora agradecer.

Primeiramente, como não podia deixar de ser, quero agradecer a Deus. Meu amigo de todas as horas, que sempre me ouve (mesmo sem precisar) nos momentos de tristeza, angústia e desânimo e me dá forças para continuar o caminho, ele me carrega no colo. Sem Ele, minha vida não teria sentido, não teria alegria, não seria nada. Muito obrigada!

Robson, meu amor, minha vida, muito obrigada por você existir na minha vida, por estar sempre ao meu lado! Esse tempo do mestrado teve um sabor todo especial com você junto de mim! As aulas, trabalhos e horas de estudo no primeiro ano, o nosso casamento, o tempo na Suécia, o retorno ao nosso país e o início de vida na nossa casa, e o último ano, com tristezas e dificuldades. Você foi e é muito mais que um companheiro: é o meu melhor amigo, aquele que está ao meu lado me apoiando, me incentivando, me aconselhando, me animando e, principalmente, me amando. Seu amor é o bem mais precioso que tenho. Agradeço todos os dias a Deus pelo presente maravilhoso que Ele me deu: você. Muito obrigada, minha vida!

Quero agradecer aos meus pais, Angela e Toninho, que não só durante o mestrado, mas durante toda a minha vida, me apoiaram, incentivaram, foram meu suporte nos momentos difíceis e se alegraram comigo nas vitórias. Vocês também são presentes valiosos de Deus para mim, meus primeiros mestres! Junto com eles, minha irmã Fabiana, amiga e "ném" sempre esteve presente, não entendendo muito dos *bits* e *bytes*, mas tendo a paciência de me ajudar com as palavras. Muito obrigada a vocês três que, juntos, sempre formaram o suporte para minha vida!

Graças a Deus, minha família é numerosa, com muitas vozes para me animar e braços para me levantar. Agradeço aos meus avós, Chico e Arine, que formaram nossa família linda, com muito amor, paz e harmonia. Meus tios Teté e Faico e os primos Pic, Rafa e Bruna, que como família, amigos e vizinhos, sempre estiveram presentes nos momentos

bons e também nos difíceis. Tia Leléia, que mesmo sem entender desse "negócio de computador", sempre se preocupou e se interessou pelos nossos estudos. Minha madrinha Maria e meus compadres Sergio e Stella, que de longe, sempre torceram muito pelo meu sucesso. À minha afilhada Arine, agradeço pela compreensão de não ter a dinda presente nos aniversários e férias por causa do mestrado. Quero agradecer ainda a toda a família Neves, pelo apoio e incentivo dos tios e primos. Especialmente aos meus padrinhos José Maria e Walame, que também de longe e sem entender os meus estudos se interessavam pela minha vida. Como comecei dizendo, o suporte familiar que Deus me deu é muito grande e vasto, e agradeço a todos por fazerem parte da minha vida!

Quero agradecer também à família que me acolheu como "filha, irmã e tia", após o casamento: Maria Lúcia, Roberto, Dayvison, Roberta, Fabiano, Ian e Davi. A presença e o carinho de vocês nesses últimos anos foi muito importante, não só durante o mestrado, mas na nossa vida de casados. Vocês também fazem parte do pacote de presentes de Deus.

Durante os dois anos e meio de mestrado, eu tive ajuda de muitas pessoas especiais, tanto no trabalho de pesquisa quanto na vida. O tempo na Suécia me fez crescer como pessoa, como profissional e também aumentou meu círculo de amizade. O professor Gerhard e os amigos Mathias, Christine, Maria, entre outros, nos acolheram num país diferente, fazendo-nos nos sentirmos em casa. Aqui na UFF, os professores Célio, Débora e Schara complementaram a orientação dos trabalhos, enriquecendo o resultado final. Os colegas do GT-Mesh me deram muito suporte técnico e apoio, principalmente Douglas e Diego. Durante os testes, tive ajuda de quem nem fazia parte da pesquisa: meu pai, minha irmã, meu primo Rafa, minha amiga Michelle e meu inseparável amigo de trabalho e vida, meu amor Robson, estiveram comigo na UFF dia e noite, finais de semana e feriados, para me ajudar a concluir essa etapa. Ao meu avô, que nos últimos 7 meses nos acolheu em sua casa com todo carinho e conforto. A todos vocês, e a outros que não citei, mas que me ajudaram direta ou indiretamente para o meu sucesso profissional, meu sincero e eterno muito obrigada!

Aos amigos e companheiros de pesquisa, pelos muitos finais de semana que passamos dentro do laboratório da pós, trabalhando, mas também pela diversão em festas e churrascos: Daniela e Rodrigo, Viviane e Joni, Jacques, Luis, Alessandro Copetti, Renathinha, Glauco, Bruno, Raphael, Alex, Luciana, Stênio, Nilmax, Aline e Alexandre, Kennedy, Thiago, Janine, Idalmis e todos aqueles que, de alguma forma, compartilharam esse tempo de estudo e superação.

Aos amigos da época de faculdade, que foram compreensivos nos momentos de ausência nos eventos por conta do mestrado, e pela presença carinhosa nas vitórias: à amiga-irmã e madrinha Michelle, ao padrinho Alan, a Laura, Heverton, Sílvia e Marquinhos, Rodolfo, Fabrício, Eduardo, Paulinha, Renata, Natali e Marcele.

Aos amigos e funcionários do Instituto, pela força e apoio: Carlinhos, Izabela, Maria e Angela.

Aos professores convidados para a banca, Carlos Maziero e Célio, pelos comentários e contribuições apresentadas.

À Capes e à RNP, pelo apoio financeiro durante a maior parte do mestrado.

Por fim, agradeço muito ao professor, amigo e orientador Julius Leite, pela confiança e o crédito depositados em mim e no meu trabalho. Você foi muito além do que um orientador deve ser: foi conselheiro e amigo, clareando muitas vezes o caminho que estava nebuloso. Você também faz parte do pacote de presentes de Deus na minha vida. Nunca esquecerei das conversas enriquecedoras, das brincadeiras, dos desafios que me apresentou e, ao mesmo tempo, ajudou a superar. Professor, você é um dos amigos que conquistei na vida e que irei levar para sempre. Muito obrigada!

Resumo

O avanço da tecnologia de redes sem fio está possibilitando a proposta de novos modelos de organização e a implementação de novas aplicações. Instituições de ensino e comerciais estão cada vez mais interessadas em adquirir esse tipo de tecnologia, para oferecer serviços de acesso banda larga sem fio para seus alunos, funcionários ou clientes. É neste contexto que surgem as redes *Mesh*, que têm como principal objetivo fornecer acesso banda larga comunitário através de um *backbone* sem fio.

A junção dessas novas redes e da transmissão multimídia apresenta novos desafios a serem superados, como, por exemplo, no fornecimento de Qualidade de Serviço (QoS) aos fluxos contínuos de tempo real. No entanto, dar suporte de QoS em redes sem fio não é uma tarefa trivial. Os problemas intrínsecos dessas redes, como interferência na comunicação causada por ruídos, colisões de pacotes ou quebra de conectividade, gerada possivelmente por causa da mobilidade dos seus nós, devem ser levados em consideração no momento da especificação dos requisitos de QoS.

Nesse sentido, esse trabalho tem como objetivo propor e implementar um *framework* de suporte a QoS para redes *Mesh*, e a sua avaliação em um ambiente real.

Palavras-chave:
(1) Redes *Mesh*;
(2) Qualidade de Serviço;
(3) Transmissão Multimídia;
(4) Sistemas Adaptativos.

Abstract

The advances in wireless networks technology are enabling new organizational models and the implementation of new applications. Educational and commercial institutions are increasingly more interested in acquire this technology, to offer services of wireless broadband access to theirs students, staff or clients. In this context, emerge the Mesh networks, that have as their main purpose to provide broadband communitarian access over a wireless backbone.

The union of these new networks with multimedia transmission presents new challenges to be overcome, like, for example, the provision of Quality of Service (QoS) for real-time continuous flows. However, to have QoS support in wireless networks is not a trivial task. The intrinsic problems of these networks, like the interference in the communication created by noise, packet collisions or connectivity loss, possibly generated due to node mobility, must be considered in the QoS requirements specification.

In this sense, this work main objective is the proposal and implementation of a framework for QoS support in Mesh networks, and its evaluation in a real environment.

Keywords:
(1) Mesh Networks;
(2) Quality of Service;
(3) Multimedia Transmission;
(4) Adaptive Systems.

Sumário

Lista de Abreviaturas

ACK	:	*Acknowledgement*
AODV	:	*Ad hoc On-Demand Distance Vector*
AP	:	*Access Point*
ASAP	:	*Adaptive Reservation and Pre-Allocation*
AWP	:	*Adaptive Wireless Path*
CAN	:	*Community Area Network*
CBQ	:	*Class Based Queueing*
CBR	:	*Continuos Bit Rate*
CSAIL	:	*Computer Science and Artificial Intelligence Laboratory*
DHCP	:	*Dynamic Host Configuration Protocol*
DSR	:	*Dynamic Source Routing*
ETT	:	*Estimated Transmission Time*
ETX	:	*Expected Transmission Count*
FTP	:	*File Transfer Protocol*
HTB	:	*Hierarchical Token Bucket*
IEEE	:	*Institute of Electrical and Electronics Engineers*
IP	:	*Internet Protocol*
ISDN	:	*Integrated Services Digital Network*
LAN	:	*Local Area Network*
LQSR	:	*Link-Quality Source Routing Protocol*
LWAPP	:	*Lightweight Access Point Protocol*
MAC	:	*Medium Access Control*
MACA	:	*Multihop Access Collision Avoidance*
MCL	:	*Mesh Connectivity Layer*
MIT	:	*Massachusetts Institute of Technology*
MR-LQSR	:	*Multi-Radio Link-Quality Source Routing Protocol*
MTU	:	*Maximum Trasmit Unit*
NAP	:	*Network Access Protection*
NOSS	:	*Network Operations Support System*

OLSR	:	*Optimized Link State Routing*
OLSR-ML	:	*OLSR Minimum Loss*
ONMS	:	*Optivity Network Management System*
OSPF	:	*Open Shortest Path First*
PCMCIA	:	*Personal Computer Memory Card International Association*
PID	:	*Proportional Integral Derivative*
QoS	:	*Quality of Service*
RNP	:	*Rede Nacional de Ensino e Pesquisa*
RSVP	:	*Resource Reservation Protocol*
RTP	:	*Real Time Protocol*
RTT	:	*Round Trip Time*
SFQ	:	*Stochastic Fairness Queueing*
SNTP	:	*Simple Network Time Protocol*
TBF	:	*Token Bucket Filter*
TC	:	*Traffic Control*
TCP	:	*Transmission Control Protocol*
TORA	:	*Temporary Ordered Routing Algorithm*
UDP	:	*User Datagram Protocol*
VPN	:	*Virtual Private Network*
WCETT	:	*Weighted Cumulative Expected Transmission Time*
WG	:	*Wireless Gateways*
WLAN	:	*Wireless LAN*

Lista de Figuras

Lista de Tabelas

Capítulo 1

Introdução

1.1 Introdução

Com o avanço das redes *wireless* e o baixo custo desses produtos no mercado, o uso de dispositivos móveis e sem fio que se comunicam através de ondas de rádio está se tornando cada vez mais popular. Dessa forma, tem sido possível que estabelecimentos comerciais como *shoppings*, aeroportos e restaurantes utilizem a tecnologia sem fio para oferecer aos seus clientes acesso à Internet banda larga e a outros serviços. Também as universidades e grandes empresas investem para desenvolver *backbones* de Internet sem fio e que conectem todos os seus usuários sem despender recursos com uma estrutura cabeada.

É nesse contexto que surgem as redes *Mesh*, também conhecidas como redes comunitárias de acesso sem fio. Alguns participantes compõem a estrutura principal da rede, trabalhando apenas como roteadores e comunicando-se via interface sem fio. Outros nós podem se conectar a esses roteadores por cabos e trabalharem apenas como clientes. Uma rede *Mesh* pode ainda permitir a conexão de dispositivos móveis através de uma interface sem fio, o que possibilita o acesso à Internet banda larga.

O artigo [Akyildiz et al. 2005] apresenta uma análise sobre as redes *Mesh* e sobre o que está sendo desenvolvido nesta área. O principal atrativo das redes *Mesh* é o de prover uma rede comunitária de acesso banda larga com infraestrutura sem fio, oferecendo, a baixo custo, acesso à Internet para comunidades que, por exemplo, possuem baixas condições econômicas.

A arquitetura apresentada na figura 1.1 [Akyildiz et al. 2005] mostra roteadores que se comunicam entre si encaminhando as mensagens para as subredes ou até aos *gateways* para a Internet. Juntos eles formam o *backbone* sem fio de uma rede *Mesh*. Outros nós

podem fazer o papel de ponte para subredes *wireless*, conectando-se a pontos de acesso que servirão aos dispositivos móveis como porta de entrada para a rede. A idéia é que os roteadores sem fio possam fornecer acesso para diferentes grupos de dispositivos, como redes de celulares, sensores ou clientes cabeados.

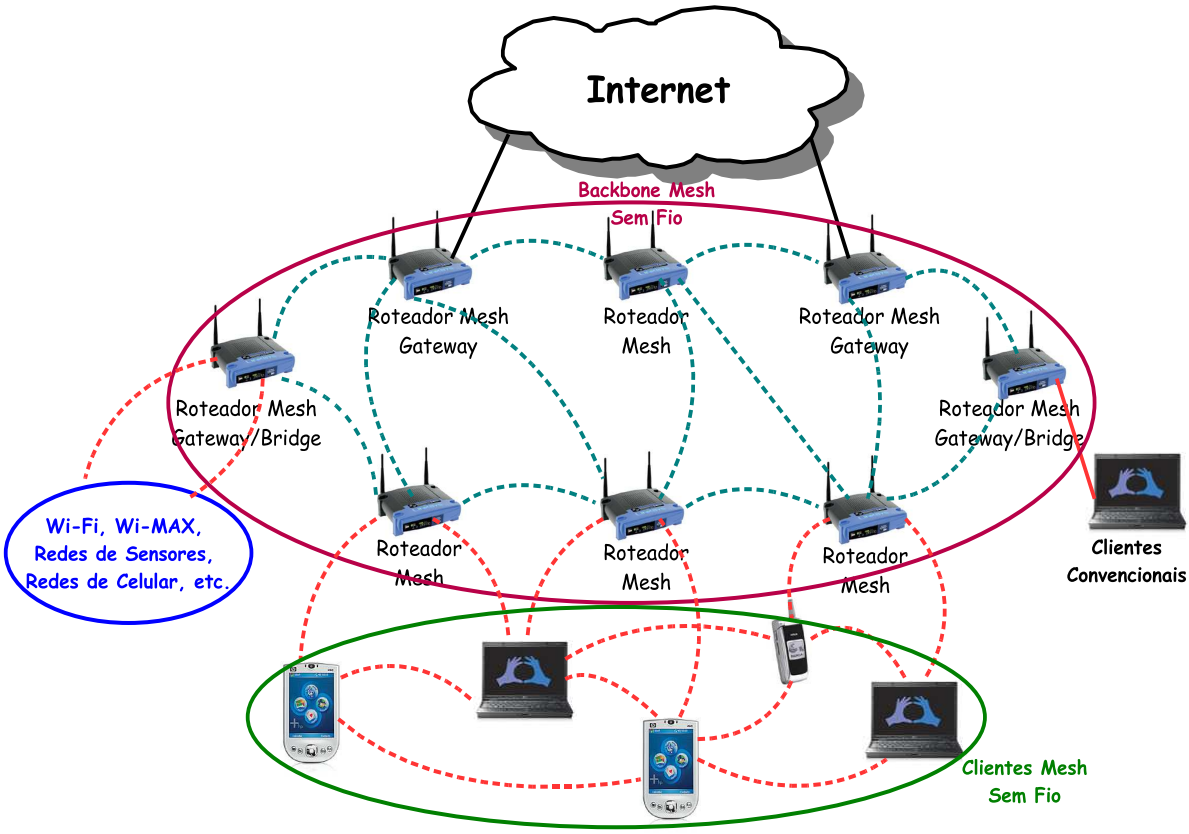


Figura 1.1: Arquitetura de uma rede *Mesh*.

Todos os roteadores de uma rede precisam utilizar algum protocolo de roteamento para montar suas tabelas de rotas e encaminhar os pacotes. Se uma rede *Mesh* não previr que dispositivos móveis possam dela usufruir, os roteadores podem utilizar protocolos para roteamento desenvolvido para redes fixas, como o OSPF. Mas, para que nós móveis possam fazer parte da rede, é conveniente utilizar algum protocolo de roteamento *Ad Hoc*, para que as rotas possam ser alteradas a qualquer momento, por causa da mobilidade ou da ausência de algum nó. Assim, as redes *Ad Hoc* e *Mesh* possuem algumas características em comum, como a mobilidade de alguns participantes.

Dentro da área das tecnologias *wireless*, as redes *Ad Hoc* são aquelas formadas por nós móveis e nós estáticos que cooperam entre si e podem trabalhar tanto como roteadores, encaminhando pacotes, quanto como nós de processamento. É importante ressaltar que seus participantes não necessitam de nenhuma infraestrutura cabeada para se comunicar

e não possuem um sistema central de gerenciamento. A versatilidade das redes *Ad Hoc* permite a sua utilização em diversos ambientes, como campos de batalha, áreas de desastres e áreas de difícil acesso. Além da mobilidade, uma rede *Ad Hoc* prevê que seus nós possam nela entrar ou sair a qualquer instante. Essas características não permitem que os nós de uma *Ad Hoc* executem protocolos de roteamento utilizados em redes fixas.

Os protocolos de roteamento *Ad Hoc* foram criados para fazer a descoberta e a manutenção de rotas numa rede cujos nós podem se movimentar e se ausentar quando quiserem. A literatura apresenta vários tipos de protocolos de roteamento, como AODV [Perkins et al. 2003], DSR [Johnson et al. 2004] e OLSR [Clausen e Jacquet 2003]. Eles são basicamente divididos em dois grupos: reativos e pró-ativos. Os reativos são aqueles que iniciam a fase de descoberta de rota por demanda, ou seja, apenas quando um nó solicita um caminho para um dado destino. Já os algoritmos pró-ativos mantêm suas tabelas de rotas periodicamente atualizadas, independentemente do fato de o nó estar transmitindo dados ou não.

A mobilidade e a entrada e saída de alguns nós podem alterar a topologia da rede. Essas constantes mudanças podem gerar múltiplas rotas entre dois nós. Sendo assim, quanto maior for o número de participantes de uma rede *Ad Hoc*, mais complexa será essa comunicação. Por essa razão, os protocolos de roteamento devem ser capazes de, além de descobrir rotas, monitorá-las para que, a cada quebra de enlace, novos caminhos possam ser descobertos, mantendo a conectividade fim-a-fim entre origem e destino. Além disso, cada rota pode apresentar diferentes capacidades e disponibilidade de recursos e, mesmo que se encontre uma que satisfaça às necessidades da aplicação, essa disponibilidade poderá variar com o tempo.

A comunicação nos ambientes sem fio, através dos seus enlaces de rádio, está sujeita a interferências, colisões e bloqueios por obstáculos. As redes *Ad Hoc* e *Mesh*, além de serem redes sem fio, permitem a mobilidade de alguns de seus nós, o que gera mudança da topologia da rede e conseqüentemente, causa um problema de conectividade inconstante entre seus enlaces. Além disso, muitos nós que compõem a rede apresentam limitações nos recursos disponíveis, como energia e capacidade de transmissão.

As redes sem fio são usualmente redes *best-effort*, ou seja, transmitem os dados sem mecanismos que forneçam garantias de qualidade na transmissão. Porém, à medida que o uso de dispositivos móveis torna-se comum, também aumenta a necessidade das aplicações fornecerem qualidade na transferência de pacotes. Para alguns tipos de dados, como requisições *web*, a falta de garantia de entrega pode não afetar o entendimento final da

mensagem. Porém, para informações que devem ser transmitidas em fluxos contínuos, como, por exemplo, dados multimídia, a perda de pacotes ocasiona uma queda na qualidade com que a mídia será reproduzida no destino. É necessário, então, oferecer serviços na rede que forneçam algum tipo de garantia de qualidade.

Garantir qualidade de serviço em redes *Ad Hoc* e *Mesh* não é uma tarefa trivial. Os mecanismos desenvolvidos devem ter como objetivo superar os problemas causados pelas suas características principais, como instabilidade das conexões. Para isso, os serviços devem ser altamente adaptáveis e responder às mudanças de disponibilidade de recursos ao longo da rota, além de serem capazes de distinguir entre os diferentes requisitos de qualidade, especificados pela aplicação. Os protocolos de roteamento também podem auxiliar, sendo rápidos na descoberta de novos caminhos e trabalhando em conjunto com mecanismos eficientes de sinalização, controle e gerenciamento.

Oferecer qualidade de serviço é uma tarefa essencial, principalmente quando se trata de tráfego em redes sem fio. Para prover QoS é preciso conhecer os problemas do ambiente a serem superados e saber quais são os pontos importantes para as aplicações-alvo. A especificação dos requisitos vai identificar justamente esses pontos, para que sejam observados ao longo da transmissão, a fim de se manter um nível de serviço adequado ao usuário final. Para que esse serviço seja confiável, é essencial que os requisitos de QoS possam ser configurados pelos seus usuários e preservados durante o tempo de execução.

Vários estudos já propuseram mecanismos que oferecem Qualidade de Serviço (QoS) em redes *Ad Hoc* e *Mesh*. As soluções apresentadas são bem variadas: algumas inserem esse serviço já nos protocolos de roteamento, enquanto outras criam uma nova camada que é responsável por garantir QoS na rede.

Este trabalho tem o objetivo de propor um *framework* de suporte à qualidade de serviço para transmissão de fluxos contínuos em redes *Mesh*. Os trabalhos existentes na área de QoS em redes *Ad Hoc* são vastos, porém as propostas para rede *Mesh* ainda não são em grande número. Ao estudar as características das redes *Mesh*, será possível desenvolver um *framework* que seja responsável por identificar os problemas que causam degradação na transmissão. Os dados multimídia são fluxos contínuos que têm restrições de tempo para serem transmitidos e reproduzidos; por essa razão, eles foram escolhidos como alvo para o suporte a QoS.

Além de propor o *framework* de QoS para redes *Mesh*, este trabalho irá apresentar resultados obtidos em ambientes de simulação e outros conseguidos através de um protótipo desenvolvido. Com esses experimentos, será possível comprovar os estudos teóricos e ob-

servar o desempenho na prática do *framework* proposto. A próxima seção irá apresentar a estruturação desta dissertação.

1.2 Roteiro

Esse primeiro capítulo teve como objetivo introduzir o assunto de redes *Mesh* e apresentar os problemas envolvidos em dar suporte a qualidade de serviço para transmissões multimídia.

O segundo capítulo apresenta os trabalhos relacionados na área de redes *Mesh* e *Ad Hoc*, bem como as propostas estudadas sobre fornecimento de qualidade de serviço em redes sem fio.

O terceiro capítulo apresenta a proposta de implementação para o *framework* de suporte a QoS em redes *Mesh*, com as devidas justificativas para a escolha dos mecanismos adotados.

O quarto capítulo expõe um estudo preliminar realizado com o objetivo de, através de um ambiente de simulação, encontrar os melhores valores para os parâmetros dos mecanismos utilizados e comprovar a eficiência do *framework*, antes de implementar e testar um protótipo.

O quinto capítulo descreve as ferramentas utilizadas para a implementação do *framework*, a configuração dos dispositivos usados nos testes, os problemas encontrados e as soluções propostas.

O sexto capítulo apresenta os testes realizados com o protótipo na rede *Mesh* montada na Universidade Federal Fluminense, assim como os resultados observados na prática.

Por último, o capítulo sete apresenta as considerações finais, a experiência adquirida com esse trabalho e propõe algumas tarefas futuras com objetivo de aperfeiçoar o desempenho do *framework* de suporte a QoS em redes *Mesh*.

Capítulo 2

Trabalhos Relacionados

2.1 Introdução

Este capítulo tem como objetivo apresentar os trabalhos atuais na área de redes *Mesh* e fazer um levantamento do que já foi proposto para redes *Ad Hoc*. Na literatura são encontrados muitos trabalhos que desenvolvem mecanismos de fornecimento de QoS. Podem-se separar esses mecanismos basicamente em dois grupos: os sistemas com reserva de recursos e os sem reserva de recursos. Esse assunto ainda apresenta pontos de vista distintos entre pesquisadores da área. Alguns deles defendem a idéia de que os mecanismos com reserva de recursos não se encaixam nas redes móveis e sem fio, por não ser possível fornecer garantias em um ambiente com conectividade inconstante.

Nas seções seguintes deste capítulo serão apresentados alguns trabalhos em desenvolvimento na área de redes *Mesh* e pesquisas que propuseram soluções para o problema de fornecimento de QoS em redes *Ad Hoc*, naquelas duas vertentes: com reserva e sem reserva de recursos. Como as redes *Mesh* também se comunicam através de enlaces de rádio e podem utilizar protocolos de roteamento *Ad Hoc*, com base nos trabalhos de QoS estudados, será apresentada no próximo capítulo uma proposta para o *framework* de suporte à qualidade de serviço em redes *Mesh*. É importante observar que os trabalhos estudados que se baseiam em reserva de recurso não apresentam soluções na camada MAC. O objetivo do trabalho a ser proposto é gerar uma solução que possa utilizar componentes *off-the-shelf*.

2.2 Redes *Mesh*

As redes *Mesh* estão sendo amplamente estudadas, e diferentes soluções para fornecimento de QoS são propostas por grupos de pesquisa ou empresas. Nesta seção serão apresentadas

sete soluções para redes *Mesh*, entre projetos acadêmicos e produtos comerciais.

A primeira é um trabalho de pesquisa que está sendo desenvolvido na Universidade de Thessaly, Grécia. O segundo trabalho, desenvolvido pela Microsoft, não tem caráter comercial, e é disponibilizado no seu *website* para uso acadêmico. A rede *Mesh* denominada *RoofNet* é uma iniciativa do Laboratório de Ciência da Computação e Inteligência Artificial (CSAIL) do Instituto de Tecnologia de Massachusetts (MIT), e seu objetivo é estudar as questões envolvidas em redes sem fio de grande escala. A rede *MeshNet* é uma rede sem fio desenvolvida pela Universidade da Califórnia e instalada no *campus* de Santa Bárbara, EUA. As outras duas soluções, implementadas pela Nortel e pela CISCO, estão sendo comercializadas para empresas e cidades que desejem instalar uma rede *Mesh* para acesso banda larga. Por último, será apresentado o projeto *ReMesh*, desenvolvido pela Universidade Federal Fluminense e financiado pela RNP. A arquitetura de fornecimento de QoS proposta nesta dissertação será implantada na rede *Mesh* do projeto *ReMesh*.

2.2.1 Projeto Acadêmico *Mesh*

Em [Tsarmpopoulos et al. 2005] é apresentado um modelo de rede *Mesh* que está sendo desenvolvido na Universidade de Thessaly, Grécia. O objetivo desse projeto é implantar uma rede *Mesh* de baixo custo na cidade de Volos, Grécia, utilizando roteadores 802.11b/g *off-the-shelf* que executem uma distribuição Linux. A rede de roteadores irá operar em modo *Ad Hoc*, o que resultará numa maior flexibilidade e reduzirá o gerenciamento. Um ponto importante desse projeto é permitir que estudantes, funcionários e professores tenham acesso, de suas casas aos servidores da universidade e à Internet, de uma maneira rápida e mais barata do que com uma conexão telefônica convencional.

A arquitetura projetada pelo grupo da Universidade de Thessaly inclui vários elementos estacionários e nós móveis (figura 2.1). Esses elementos fixos seriam os roteadores sem fio, instalados no alto de prédios e telhados para alcançar uma melhor conectividade. Sua principal função é a de permitir que um ou mais dispositivos clientes se conectem localmente para ter acesso ao resto da rede. Como esta *Mesh* prevê que nós móveis poderão se conectar à rede, os roteadores utilizam o protocolo de roteamento OLSR.

Os dispositivos clientes que quiserem se conectar à rede *Mesh* através de um roteador sem fio terão um endereçamento estático. Cada roteador poderá suportar até 29 dispositivos conectados a ele pela porta LAN. Cada roteador é registrado num servidor central e recebe um intervalo de 32 endereços IP válidos. Destes 32, 29 serão utilizados para dispositivos que desejem fazer parte da rede *Mesh*, que receberão um endereço por DHCP

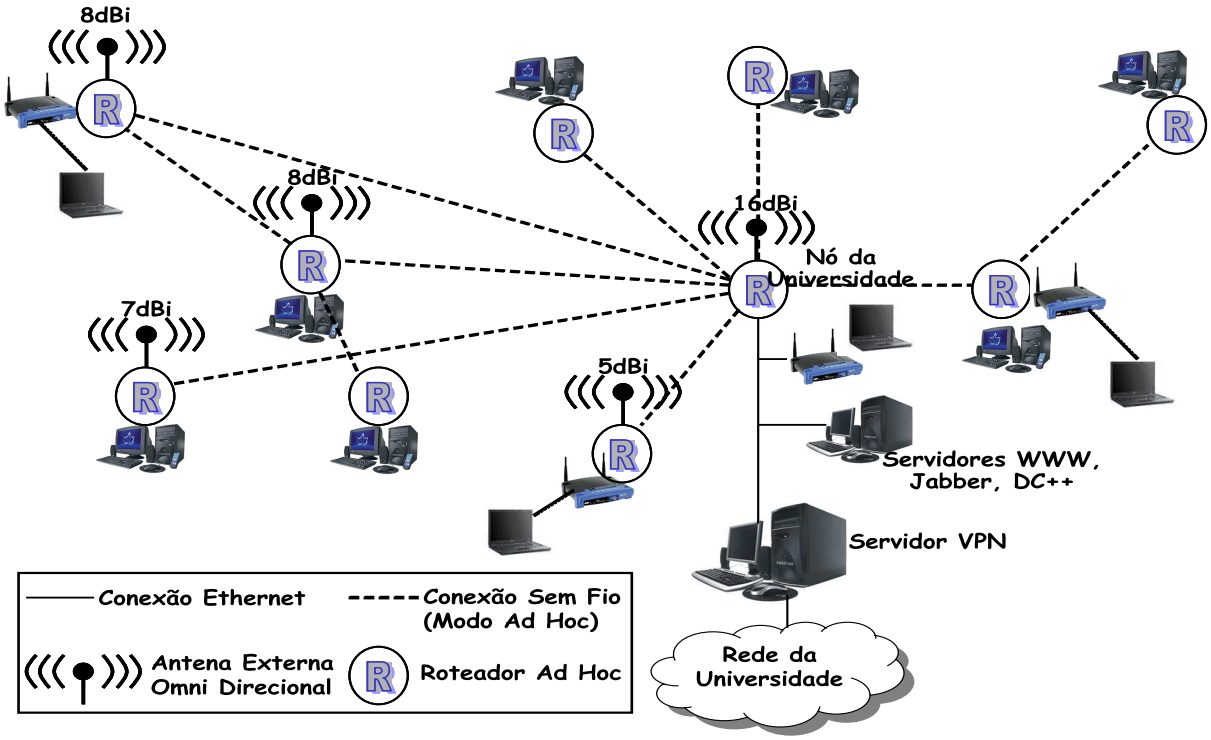


Figura 2.1: Arquitetura *Mesh* da Universidade de Thessaly.

local pelo roteador.

Numa rede comunitária, é importante garantir que seus participantes não serão capazes de explorar e utilizar as informações que forem roteadas através dos seus roteadores privados. Assim, enquanto estudantes, funcionários e professores poderão usar serviços e aplicações fornecidos pelos departamentos da Universidade, usuários que não forem afiliados à instituição não terão o acesso liberado. Desta forma, o controle de acesso à rede e a segurança são implementados através de uma *Virtual Private Network* [1].

O servidor VPN está instalado num computador que dá acesso à rede da Universidade. Ele é equipado com duas interfaces *Ethernet*. A primeira é conectada com a LAN da Universidade, que acessa os servidores do Departamento de Computação e a Internet pública. A segunda interface é conectada na interface LAN de um roteador *Ad Hoc*, que irá conectar o servidor VPN com a rede comunitária. Assim, todos os dispositivos que estiverem conectados à *Mesh* precisarão passar pelo servidor VPN para utilizar os serviços fornecidos pela Universidade.

Esse projeto permitiu um avanço significativo nas pesquisas relacionadas à área de re-

[1] Uma VPN é uma rede de comunicação privada, utilizada para que entidades, como empresas ou universidades, possam se comunicar através de uma rede pública utilizando protocolos de segurança e mecanismos de autenticação.

des *Mesh*, demonstrando a viabilidade de implantação desse tipo de rede. Diversos fatores importantes devem ser ressaltados, como a escolha da plataforma e do *hardware* utilizado, que garantiu a flexibilidade necessária à realização dos experimentos. O desempenho da rede também se apresentou satisfatório, considerando as limitações impostas pela tecnologia. Além disso, o custo do conjunto *hardware/software* é relativamente baixo.

De acordo com as informações em [Tsarmpopoulos et al. 2005], esse projeto inicialmente não oferece nenhum mecanismo para prover qualidade de serviço para transmissão de fluxos contínuos. A transferência dos dados é feita somente por melhor esforço. Mais detalhes sobre a implementação dessa rede *Mesh* e os resultados de desempenho poderão ser encontrados no artigo citado e também no *site* [VMesh 2005].

2.2.2 Microsoft Mesh Toolkit

A Microsoft disponibiliza no seu *site* [Microsoft 2005] uma implementação de rede *Mesh* com distribuição gratuita, para ser utilizada principalmente por instituições acadêmicas. A solução *Mesh* presente neste *kit* foi implementada através de uma camada chamada *Mesh Connectivity Layer* ou simplesmente MCL. Estão presentes nesse *kit* alguns outros *softwares* como o *Venice*, que faz a instalação da MCL, e algumas ferramentas de medição de desempenho.

A MCL é um *driver* que implementa um adaptador virtual de rede. A rede *Mesh* que estiver disponível para conexão aparecerá para o usuário como um enlace adicional de rede, porém virtual. O MCL é implementado entre as camadas de enlace e de rede. Para as camadas mais altas, ele é visto como mais um enlace *Ethernet*, embora seja virtual. Para as camadas mais baixas, o MCL aparece como apenas um outro protocolo que está sendo executado acima do nível físico.

O protocolo de roteamento utilizado foi desenvolvido pela Microsoft e é chamado de *Multi-Radio Link-Quality Source Routing Protocol* ou MR-LQSR [Draves et al. 2004c]. Ele é uma combinação do LQSR [Draves et al. 2004a] com uma nova métrica chamada *Weighted Cumulative Expected Transmission Time* (*WCETT*). O LQSR é uma versão modificada do protocolo *Dynamic Source Routing* [Johnson et al. 2004]. O DSR foi estendido para que pudesse suportar várias métricas de qualidade de enlace e múltiplos rádios.

As métricas utilizadas para medição da qualidade do enlace foram três, e o estudo sobre elas pode ser encontrado em [Draves et al. 2004b]. A primeira, chamada de *Expected Transmission Count* (ETX), é obtida através da medição da taxa de perda de pacotes *broadcast*

entre pares de nós vizinhos e apresentou melhor desempenho quando todos os nós da rede eram estacionários. A segunda métrica é a *Per-hop Round Trip Time* (RTT), que mede o atraso da transmissão ida e volta de pacotes *probe unicast* entre nós vizinhos. Por último, a métrica chamada de *Per-hop Packet Pair Delay* (PktPair) é baseada no atraso entre um par de pacotes *probes back-to-back* para um nó vizinho.

O mecanismo para fornecimento de qualidade de serviço foi inserido no protocolo de roteamento, como já mencionado. O protocolo LQSR é uma extensão do protocolo DSR, que objetiva aprimorá-lo, inserindo métricas de qualidade dos enlaces. O LQSR executa as mesmas etapas de descobrimento e manutenção de rota. As modificações para dar suporte às métricas de qualidade foram feitas tanto na etapa de descobrimento quanto na de manutenção da rota, além da adição de novos mecanismos de manutenção de métricas. A mensagem de requisição de rota, além de carregar a rota que deve ser seguida, também possui os valores das métricas do enlace que o pacote percorreu. Esses valores também são adicionados na mensagem de resposta da rota. Assim, quando todas as mensagens de descobrimento de rota chegarem de volta à origem, será possível escolher a que apresentar a melhor qualidade.

A Microsoft utiliza múltiplos rádios [Draves et al. 2004c] em cada nó para aumentar a capacidade de transmissão da rede *Mesh*. Cada nó recebe dois ou mais rádios que irão trabalhar em canais com freqüências diferentes. Isto possibilita que os nós recebam e enviem pacotes simultaneamente. Em um sistema de apenas um rádio, cada nó precisa sincronizar suas transmissões e recepções de pacote.

Com esse trabalho, a Microsoft pretende mostrar que as redes *Mesh* são viáveis, apesar das tecnologias existentes atualmente ainda serem inadequadas. A médio prazo, será possível ter redes desse tipo operando em situações reais, considerando o trabalho que vem sendo desenvolvido pelas pesquisas, em paralelo ao desenvolvimento da indústria de *hardware* e dos organismos de padronização.

2.2.3 *Roofnet*

A *Roofnet* [Aguayo et al. 2004] consiste numa rede *Mesh* de 38 nós espalhados numa área de aproximadamente seis quilômetros quadrados, em Cambridge, Massachusetts. Cada nó consiste num *kit* fornecido pela equipe de desenvolvimento do projeto, do Instituto de Tecnologia de Massachusetts, MIT. Este *kit* é composto de um computador pessoal com uma placa de rede sem fio 802.11 conectada a uma antena omni-direcional, em que todos os nós operam no mesmo canal.

As placas de rede foram configuradas para transmitir a 1, 2, 5,5 ou 11 Mbits/s, com um seletor automático de *bit-rate*. Nos testes realizados, porém, esse seletor foi desabilitado. Os nós da rede operam no modo *Ad Hoc* para que seja possível a implantação de uma rede multi-saltos. Os usuários são voluntários para o projeto que foram escolhidos aleatoriamente, sem nenhum plano especial para manter uma conectividade básica entre os nós. Eles ligam os seus PCs aos nós da rede *Mesh* por meio de cabos *Ethernet*.

Cada nó da rede *Roofnet* executa o sistema operacional Linux e possui um servidor de DHCP, para o endereçamento das máquinas dos usuários, e um servidor *web*, para que possa haver o monitoramento do *status* da rede por parte dos desenvolvedores e dos usuários. Ao conectar o cabo do PC ou do *Laptop* na interface *Ethernet* do nó da rede *Mesh*, este configura o computador do usuário automaticamente, fornecendo um endereço para a máquina via DHCP e trabalhando como um roteador da rede.

Para que a rede seja completamente autoconfigurável, é necessário que cada nó roteador solucione vários problemas, como alocar um endereço novo para cada usuário conectado, encontrar um *gateway* entre a rede *Roofnet* e a Internet e escolher uma rota com multi-saltos ótima até o *gateway*.

Na rede *Roofnet*, o fornecimento de qualidade para suas transmissões é tratado pelo protocolo de roteamento. O protocolo utilizado é o *Srcr* [Bicket et al. 2005]. Ele é uma extensão do protocolo de roteamento DSR [Johnson et al. 2004], tendo como diferença a métrica utilizada para encontrar o caminho ótimo entre dois nós. Em vez de utilizar o número de saltos, como o DSR, o *Srcr* utiliza a métrica denominada *Estimated Transmission Time* (ETT). O caminho escolhido deve ser aquele que possua a vazão mais alta entre os pares de nós que formam a rota, ou seja, aquele que apresente o menor ETT. Dessa forma, a rota encontrada oferece qualidade dos enlaces para as aplicações.

No artigo [Bicket et al. 2005], os pesquisadores do MIT apresentam com detalhes as principais características da *Roofnet*, como o *hardware* utilizado, os *softwares* e toda a configuração dos nós roteadores, o protocolo de roteamento e a métrica ETT utilizada como parâmetro para escolha da melhor rota; expõem ainda como é feito o mecanismo de seleção de *bit-rate* no 802.11b. Também nesse artigo é possível encontrar os resultados sobre o desempenho da rede *Roofnet* e os experimentos que foram realizados. Em linhas gerais, o desempenho da rede alcançou o esperado pela equipe de pesquisadores, apresentando uma vazão média de 627kbits/seg, numa taxa de transmissão de 5.5 Mbits/s.

2.2.4 *MeshNet*

Desenvolvido na Universidade da Califórnia, o projeto Santa Bárbara *Mesh* é uma rede *wireless* instalada no *campus* da Universidade de Santa Bárbara, com 25 nós equipados com interfaces 802.11a/b/g [MeshNet 2005]. Os nós são distribuídos em cinco andares dentro do edifício da Faculdade de Engenharia.

A rede está sendo utilizada para o desenvolvimento de protocolos e sistemas para operações robustas em redes *Mesh*. O foco principal são especialmente as pesquisas sobre protocolos de roteamento escaláveis, gerenciamento eficiente de redes, transmissão multimídia e soluções de QoS para redes sem fio multi-saltos. Em seu site [MeshNet 2005], é possível visualizar a rede em tempo real, com informações sobre sua localização e a qualidade dos enlaces.

A rede possui ainda um *gateway* para o escoamento do tráfego da rede *Mesh* para Internet. O *gateway* é um computador com processador Intel Celeron com sistema operacional Linux. Ele é equipado com uma interface de rede PCMCIA *wireless* 802.11b e uma interface *Ethernet*. A interface sem fio é utilizada para conectar o *gateway* à rede *Mesh*, enquanto a interface cabeada é utilizada para prover acesso à Internet e também para permitir o gerenciamento dos dados.

Sendo uma rede desenvolvida principalmente para pesquisas, alguns módulos específicos para monitoramento e gerenciamento de redes *Mesh* já foram desenvolvidos, como o *MeshViz*, que consiste em uma ferramenta para visualização em tempo real das métricas da rede. Atualmente, nenhum mecanismo de suporte à qualidade de serviço foi implantado na rede. Mais informações sobre a *Meshnet* e as ferramentas desenvolvidas pela equipe podem ser encontradas no site [MeshNet 2005].

2.2.5 Solução *Mesh* da Nortel

A Nortel desenvolveu uma solução para redes *Mesh* cuja arquitetura é apresentada na figura 2.2 [Nortel 2005]. Ela é composta por pontos de acesso (AP) que, juntos, formam uma rede comunitária (CAN), trabalhando como portas de entrada da rede para os dispositivos móveis. Os AP utilizam dois tipos de enlaces: o 802.11a para trânsito, ou seja, comunicação entre AP, e 802.11b/g como enlace de acesso, utilizado para comunicação entre dispositivos móveis e AP.

As CAN estão conectadas a roteadores (NAP) que dão acesso à rede principal, combi-

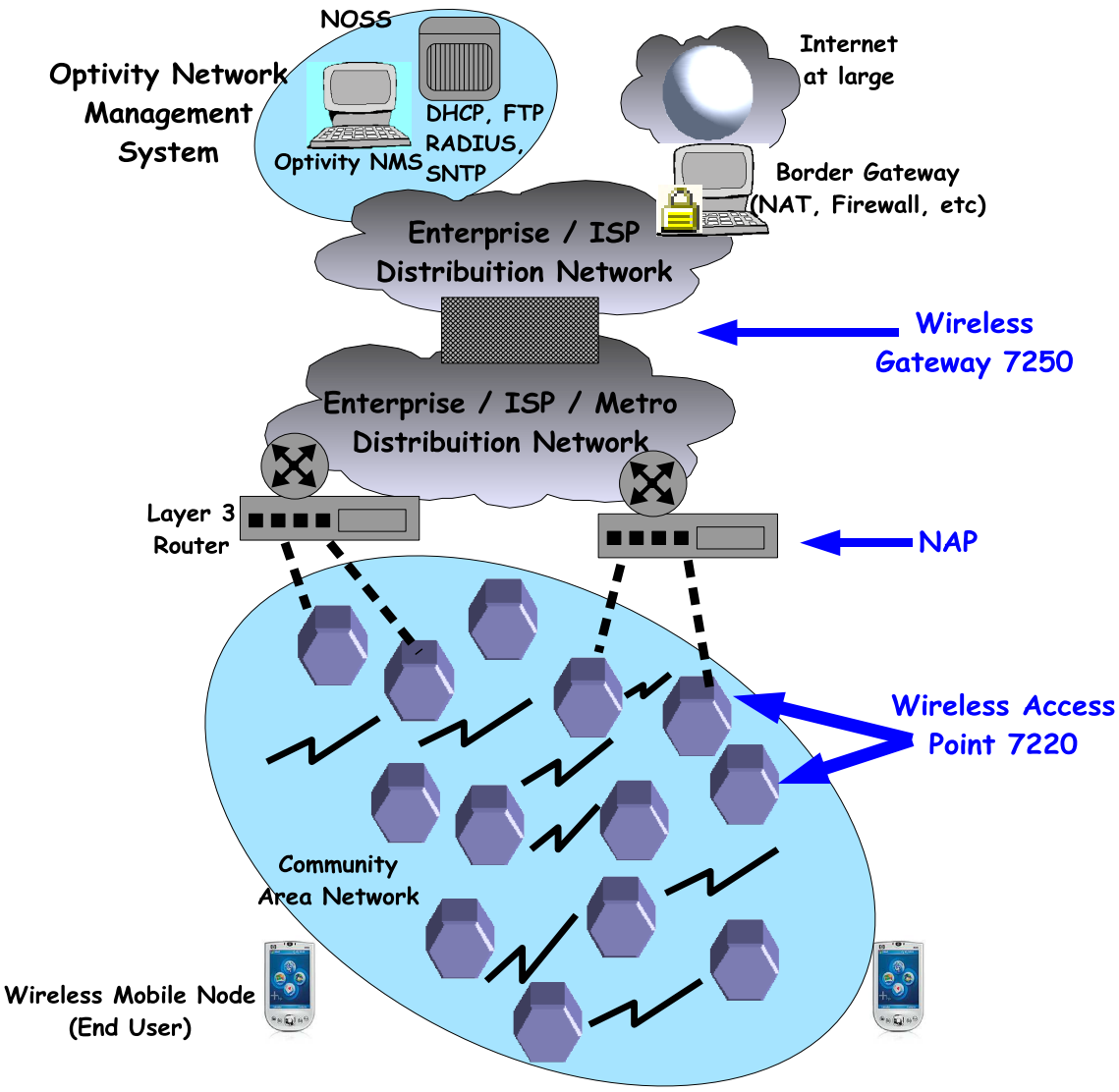


Figura 2.2: Arquitetura para redes *Mesh* da Nortel.

nando funções de *switch* e de um AP sem fio. Fazendo a conexão entre a rede distribuída e o *backbone* estão os *Wireless Gateways* (WG), que têm a responsabilidade de fornecer segurança e dados para o enlace de trânsito.

Conectado ao *backbone*, o *Network Operations Support System* (NOSS) fornece facilidades centralizadas de monitoração e gerenciamento das operações da rede. O NOSS possui o *Optivity Network Management System* (ONMS), que é responsável pelo gerenciamento de falhas, desempenho e configuração. Além do ONMS, o NOSS também possui servidores FTP, DHCP e SNTP.

O endereçamento dos nós móveis é feito de forma dinâmica: através da comunicação entre o AP e WG, ele recebe um número IP fornecido pelo servidor de DHCP único. A

comunicação entre o AP e o WG é feita por IP Móvel. O WG faz o papel do *Home Agent*, enquanto o AP é o *Foreign Agent*, que cria o túnel entre eles.

Esta solução está sendo utilizada em diferentes cenários. Em Ottawa (Canadá), Taipé (China) e na Filadélfia (EUA), a solução da Nortel foi implantada para fornecer acesso banda larga para uma região da cidade. No Arkansas, EUA, a rede *Mesh* foi instalada numa universidade. Essa solução também pode ser utilizada em portos, armazéns ou lugares públicos como aeroportos.

No material disponibilizado pela Nortel, não foi encontrada nenhuma descrição ou menção a mecanismos de suporte à qualidade de serviço.

2.2.6 Solução *Mesh* da CISCO

A CISCO desenvolveu uma solução para redes *Mesh* [Cisco Systems 2005] que tem seu foco na implantação de redes *WiFi* seguras em pólos empresariais, ou ainda em espaços metropolitanos, dando cobertura *wireless* a uma cidade, por exemplo. Essa solução *Mesh* oferece um protocolo inteligente de roteamento sem fio, um sistema de gerenciamento *Mesh* e uma arquitetura WLAN unificada. Sua arquitetura é apresentada na figura 2.3.

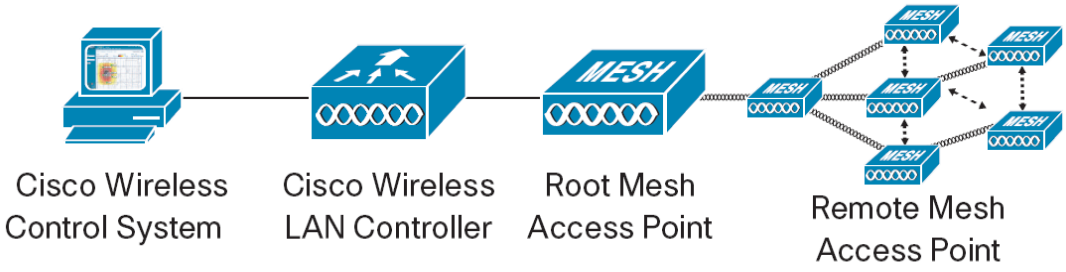


Figura 2.3: Solução para redes *Mesh* da CISCO.

O *Cisco Wireless Control System* permite o gerenciamento de todo o sistema para projetar, controlar e monitorar uma rede *wireless* num ambiente aberto, de um ponto centralizado, simplificando as operações e reduzindo o custo de implantação.

O *Cisco Wireless LAN Controller* faz a ligação entre os pontos de acesso e a parte cabeada da rede, além de centralizar algumas funções do padrão 802.11. Baseado num protocolo chamado *Lightweight Access Point Protocol* (LWAPP) [Calhoun et al. 2005] e operando como parte da arquitetura, o *Controller* provê um gerenciamento da configuração dos dispositivos da rede, das políticas de segurança e das freqüências de rádio, além de permitir mobilidade

nas camadas 2 ou 3 da rede. Através da tecnologia IEEE 802.11e embutida no *Controller*, fornece QoS via camada MAC por meio de reserva de banda.

Com suporte simultâneo para os padrões IEEE 802.11a e 802.11b/g, o *Wireless Mesh Access Point* utiliza um protocolo de roteamento proprietário da CISCO, chamado *Adaptive Wireless Path Protocol* (AWP), para formar uma rede *Mesh* dinâmica entre os pontos de acesso remotos, fornecendo acesso sem fio seguro para qualquer cliente *WiFi*. O protocolo AWP é baseado numa tecnologia de roteamento que responde dinamicamente às variações das condições de uso da rede, propiciando a seleção de rotas ótimas para o caso de falhas nos enlaces ou mudanças no ambiente. O *Root Mesh Access Point* trabalha como *gateway* de ligação entre os AP remotos e a rede. Tipicamente, são instalados no alto de telhados ou prédios e utilizam o padrão IEEE 802.11a para comunicação. Já os *Remote Mesh Access Points* fornecem o acesso para clientes via padrão IEEE 802.11b/g, porém se conectam ao *Root* via 802.11a. Adicionalmente, eles também possuem portas *Ethernet* para conexão de dispositivos periféricos.

2.2.7 Projeto *ReMesh*

As redes *Mesh* podem ser aplicadas em regiões com baixas condições econômicas, em que uma estrutura cabeada seria muito cara para ser instalada. Atualmente, embora possuam uma tecnologia avançada, os dispositivos sem fio, como roteadores, podem ser obtidos a baixo custo. Um dos pontos atrativos das redes *Mesh* é exatamente este: implantar uma rede comunitária sem fio de acesso banda larga a baixo custo. Os cenários de aplicação de uma *Mesh* poderiam ser, por exemplo, um bairro, cidade ou universidade, onde exista apenas um ponto central de acesso à Internet, que é distribuído numa determinada área com cobertura sem fio.

O projeto *ReMesh* [ReMesh 2005] está sendo desenvolvido através de uma parceria entre o Instituto de Computação e o Departamento de Telecomunicações da Universidade Federal Fluminense, UFF, e está sendo financiado pela RNP. Seu objetivo é implantar uma rede sem fio de acesso comunitário em um dos *campi* da UFF, com o intuito de fornecer, a baixos custos, acesso banda larga para alunos, funcionários e professores que residem ao redor do *campus*.

Após o estudo das várias soluções de redes *Mesh*, de redes *Ad Hoc* e de técnicas de suporte à qualidade de serviço, foram determinados os principais pontos a serem utilizados, como o protocolo de roteamento. De acordo com a análise dos projetos estudados, escolheu-se o protocolo de roteamento OLSR, por se apresentar o mais estável para ser

executado nesse tipo de rede. Também foram definidos o servidor de autenticação (*Wi-Fi Dog*), modelos de roteadores sem fio (*Linksys WRT 54G*) e outras configurações.

Inicialmente, o projeto instalou uma rede *Mesh* dentro do prédio onde ficam as duas unidades acadêmicas, para executar testes num ambiente mais controlado. Ao mesmo tempo, convidou alunos dos cursos das unidades envolvidas, que moram nas vizinhanças do *campus*, a se candidatarem a participar do projeto como voluntários, instalando um nó da rede *Mesh* no edifício onde residem. No escopo do projeto, foi prevista a instalação de apenas 7 nós do *backbone* da rede *Mesh*, para avaliação do protótipo, sendo que um desses nós é o *gateway* para Internet. Este *gateway* foi instalado no topo de um dos prédios da Escola de Engenharia. Os outros 6 nós foram instalados no topo dos edifícios de 6 alunos selecionados. Após escolhidos esses pontos, iniciou-se a fase de instalação dos roteadores nas residências dos alunos. Tanto a rede *indoor* como a rede que está sendo montada ao redor do *campus* podem ter a sua topologia visualizada em tempo real através do *site* do projeto *ReMesh* [ReMesh 2005].

Os computadores dos usuários vão acessar a rede *Mesh* via cabo, ligados aos roteadores pela porta LAN. O endereçamento dessas máquinas será feito via DHCP, com um intervalo de 32 endereços IP válidos para cada roteador, dos quais 29 serão distribuídos para as máquinas que se conectarem a ele.

Os dispositivos móveis que eventualmente participarão da rede *Mesh* poderão também ser *Pockets PC* e *laptops* com interface sem fio, ajustados para se conectarem a uma rede no modo *Ad Hoc*. Eles poderão se conectar à rede *Mesh* para ter acesso ao *backbone* de duas formas. Caso desejem se movimentar sem restrições, eles podem ter um endereçamento estático, precisando requisitar um IP válido na rede e executar o protocolo de roteamento *Ad Hoc* escolhido, OLSR, para montar as rotas entre os nós e para o *gateway* da Internet. Ou ainda, eles poderão obter um endereço via DHCP pelo roteador ao qual estiverem mais próximos, aí autenticando-se. No entanto, se eles se moverem para fora do alcance do roteador que lhes deu o endereço e entrarem no alcance de outro roteador, perderão o endereço antigo, ganharão um novo e precisarão novamente se autenticar na rede.

O protocolo de roteamento OLSR utilizado no projeto *ReMesh* não é o original. Denominado OLSR-ML, ele foi gerado a partir de algumas alterações no OLSR especificado em [Clausen e Jacquet 2003]. Essas modificações foram feitas para que o protocolo se tornasse mais estável. Ao longo dos testes, foi observado que o modo com que o OLSR calculava as métricas para escolha da rota gerava uma grande instabilidade na rede. A equipe do ReMesh então propôs uma modificação do cálculo das métricas, o que gerou um

novo protocolo.

O OLSR original utiliza a métrica ETX para calcular a melhor rota entre dois nós. A métrica ETX é calculada usando as taxas de recepção de pacotes nos dois sentidos do enlace. A taxa de recepção é a probabilidade de um pacote chegar no próximo nó do caminho com sucesso. Desta forma, o ETX é calculado com a seguinte fórmula:

$$ETX = 1 / (df * dr)$$

em que *df* e *dr* são a taxa de recepção de ida e de volta, respectivamente. O ETX de um caminho com múltiplos saltos é então medido com a soma dos ETX de cada salto. Assim, quanto maior o ETX do caminho, pior a qualidade do enlace.

Um ETX próximo ou igual a 1 representa um enlace perfeito. Desta forma, o OLSR seleciona como melhor rota entre dois nós aquela que tiver o menor ETX. No entanto, a equipe do projeto *ReMesh* observou que o uso dessa métrica pode resultar em instabilidade nas tabelas de rotas e em altas taxas de perda de pacotes. Isso acontece porque o OLSR-ETX acaba selecionando caminhos mais curtos com taxas de perda mais altas, em vez de caminhos mais longos com menores taxas de perda.

A proposta do OLSR-ML é que a métrica de qualidade para a escolha de uma rota seja a probabilidade de transmissão com sucesso entre dois nós, obtida com a expressão abaixo:

$$P = (df * dr)$$

em que *df* e *dr* têm a mesma definição da equação anterior. Para um caminho com múltiplos saltos, a probabilidade de transmissão com sucesso em todo o caminho será o produto das probabilidades em cada salto. Como pode ser observado, a probabilidade P é o valor inverso de ETX. Assim, a melhor rota entre origem e destino é aquela com a maior probabilidade de transmissão com sucesso, ou seja, com a menor probabilidade de perda de pacotes. Os testes com essa nova abordagem apresentaram uma estabilidade maior na tabela de rotas e uma perda de pacotes menor.

A solução de suporte à qualidade de serviço na transferência de fluxos contínuos que é proposta nesta dissertação será implantada na rede *Mesh* do projeto *ReMesh*. Inicialmente, o cronograma do projeto prevê que os testes do *framework* de QoS sejam feitos apenas na rede interna, implantada no *campus* da UFF. Caso haja tempo hábil, os testes serão realizados também na rede *outdoor*. Um dos objetivos do projeto *ReMesh* é propor e implementar um sistema de fornecimento de QoS, ao invés de apenas simular os cenários.

2.3 Soluções de QoS Com Reserva de Recursos

Alguns trabalhos na área de QoS em redes *Ad Hoc* defendem a idéia de que a reserva de recursos, apesar de ser uma técnica utilizada em redes fixas, pode ser adaptada e produzir resultados satisfatórios em ambientes sem fio. Nesta seção serão apresentados dois desses trabalhos. Primeiramente, o INSIGNIA [Lee et al. 2000], que é um *framework* de suporte a QoS em redes *Ad Hoc* com mecanismo de sinalização *in-band*. Em seguida, será apresentado o ASAP [Xue et al. 2003], um trabalho baseado no INSIGNIA, que propõe mudanças neste *framework* com o intuito de superar algumas limitações.

2.3.1 INSIGNIA

O INSIGNIA [Lee et al. 2000] foi desenvolvido por um grupo de pesquisadores da Universidade de Columbia, Nova York, EUA, para ser um *framework* de fornecimento de qualidade de serviço e dar suporte à adaptabilidade em redes móveis *Ad Hoc*. De acordo com as características de uma rede *Ad Hoc*, esta só consegue fornecer uma transmissão ponto-a-ponto por "melhor esforço" (*best effort*). Para fornecer QoS, tanto para transmissão de fluxo contínuo como para micro-fluxo, é necessária mais uma camada que monitore as condições da rede e permita que a aplicação se adapte às constantes mudanças do ambiente.

Entende-se por micro-fluxo aqueles fluxos de vida curta, como, por exemplo, interações *web* do tipo cliente/servidor, que compreendem uma quantidade limitada de pacotes de dados. O INSIGNIA é capaz de prover QoS para fluxos de tempo real através de um mecanismo de sinalização *in-band*, da reserva de recursos e de um gerenciamento de recursos *soft-state*.

Os autores desse trabalho, ao desenvolverem o INSIGNIA, apontaram algumas questões que deveriam ser essenciais para o seu funcionamento. Como um *framework* de fornecimento de QoS, seu principal objetivo é o de dar suporte a serviços adaptáveis. Para isso, esses serviços devem fornecer garantias mínimas de largura de banda para fluxos de tempo real, como áudio e vídeo, permitindo que níveis elevados de serviço sejam fornecidos quando os recursos ficarem disponíveis.

Outro ponto importante é a separação entre protocolo de roteamento, sinalização e encaminhamento de mensagens. Os desenvolvedores do INSIGNIA defendem a idéia de que os mecanismos de roteamento, sinalização e transferência das mensagens devem ser modelados independentemente. Eles argumentam que esses componentes apresentam algoritmos e tempos de execução muito diferentes. O INSIGNIA foi projetado para que

uma variedade de protocolos de roteamento fosse utilizada no *framework*.

Outra questão se refere ao mecanismo de sinalização, tão importante no suporte a serviços adaptáveis. O sistema de sinalização do INSIGNIA é chamado de *in-band* porque as informações de controle são transmitidas juntamente com os dados. Esse sistema permite que a reserva para um fluxo seja restaurada no momento da mudança da topologia. Além disso, o mecanismo *in-band* evita que mensagens de controle sejam enviadas separadamente dos dados, o que as torna sujeitas à quebra de rotas e provoca um atraso no envio dos dados.

O último ponto defendido pelos desenvolvedores do INSIGNIA é o gerenciamento de recursos *soft-state*. Essa técnica de gerenciamento se mostra mais apropriada para redes *Ad Hoc*, pois é mais flexível. O termo *soft-state* significa que cada nó irá armazenar o *status* da reserva para cada fluxo de tempo real, porém apenas por um determinado intervalo de tempo. Para essa reserva se manter válida, é necessário atualizá-la.

O nó origem, ao enviar pacotes de dados, utiliza um caminho disponível na tabela de rotas. Ao passar pela primeira vez por uma rota, se um pacote de um fluxo enviado chegar num roteador onde não exista nenhuma reserva para esse fluxo, então o controle de admissão e a reserva de recursos tentarão estabelecer uma reserva *soft-state* para este fluxo. Assim, os pacotes subseqüentes que chegarem nesse roteador irão usar a reserva estabelecida e atualizá-la por mais um intervalo de tempo. Se esse intervalo de tempo terminar antes que algum pacote passe pelo roteador, essa reserva será descartada.

O INSIGNIA foi projetado e desenvolvido para redes *Ad Hoc*. Porém, sua implementação completa ficou apenas no nível da simulação. Foi desenvolvido um protótipo para ser executado no sistema operacional *FreeBSD*, que, no entanto, não incluiu todas as características do INSIGNIA e estacionou na versão alfa. O teste desse protótipo foi realizado numa rede *Ad Hoc* formada por *laptops*.

2.3.1.1 Arquitetura do *Framework*

A arquitetura do *framework* INSIGNIA está representada na figura 2.4, que foi retirada do artigo [Lee et al. 2000]. O módulo mais importante é o INSIGNIA *signaling*, que contém o mecanismo de sinalização. Nessa seção, as funções de cada módulo serão brevemente explicadas.

No módulo do protocolo de roteamento (*Routing Protocol*), qualquer protocolo pode ser utilizado e interagir com o *framework* INSIGNIA. Para a realização dos testes de simulação,

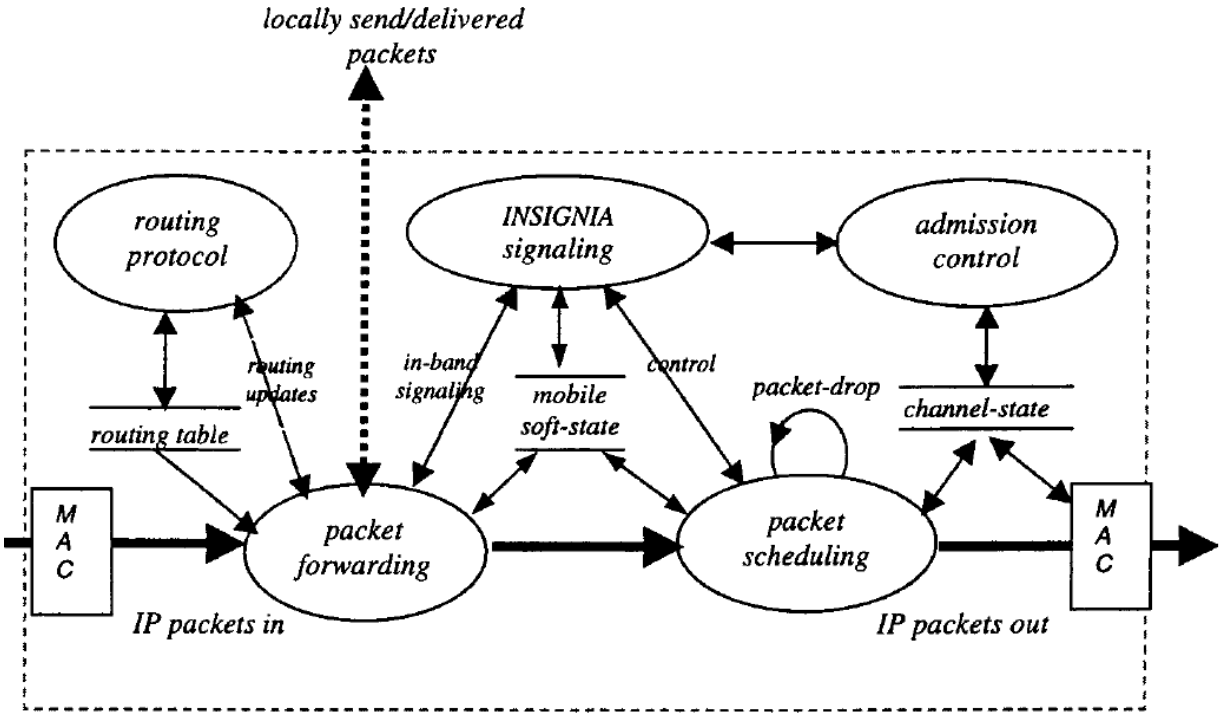


Figura 2.4: Arquitetura do *framework* INSIGNIA.

o protocolo usado foi o TORA. Já na implementação piloto para o sistema operacional *FreeBSD*, o protocolo utilizado foi o DSR. Os autores, porém, já realizaram um trabalho de análise do desempenho do INSIGNIA utilizando os protocolos AODV, DSR e TORA [Lee et al. 2001].

O módulo denominado *Packet Forwarding* é responsável pela transmissão de pacotes, classificando os que chegam ao nó e encaminhando-os para os módulos apropriados. As mensagens de sinalização vão para o módulo INSIGNIA *signaling*, enquanto os pacotes de dados podem ter dois destinos: serem entregues localmente, caso o nó corrente seja o destino, ou serem transferidos para o módulo de escalonamento de pacotes (*Packet Scheduling*), para serem encaminhados ao próximo nó.

O *Packet Scheduling* faz o escalonamento de pacotes de acordo com as condições do canal. Uma variedade de modelos pode ser usada. Para testes, foi utilizado o escalonamento *Weighted Round Robin*.

O módulo de controle de admissão (*Admission Control*) é responsável pela alocação de banda para os fluxos, seguindo a especificação requisitada de larguras de banda máxima e mínima. A admissão do fluxo com reserva é feita apenas se houver banda disponível. Essa disponibilidade é obtida pela razão entre a capacidade do nó, representada por um valor

estático, e a banda utilizada, verificada pela tabela de reservas. Essas reservas são sempre atualizadas quando um pacote é transferido através do roteador. Novas requisições não causam impacto nas reservas já existentes. O *framework* INSIGNIA foi projetado para ser transparente a qualquer protocolo MAC e é posicionado para operar sobre múltiplas tecnologias na camada IP.

Por último, o módulo de sinalização INSIGNIA (*INSIGNIA Signaling*) é responsável por estabelecer, restaurar, adaptar e terminar serviços adaptáveis entre origem e destino. Os algoritmos de restauração de fluxo respondem às mudanças de rotas, enquanto os algoritmos de adaptação respondem às mudanças na disponibilidade da largura de banda. A sinalização é chamada de *in-band*, pois transmite as informações de controle de QoS no cabeçalho do pacote IP.

Para entender melhor o módulo de sinalização INSIGNIA, é necessário compreender a diferença entre mecanismos *in-band* e *out-of-band*. O termo *out-of-band* se refere ao fato de que as informações de controle de uma mensagem de sinalização são transmitidas em pacotes de controle separados dos dados e podem utilizar canais distintos entre a origem e o destino. Isso significa que as mensagens da sinalização *out-of-band* necessitam manter as informações sobre rotas e também responder às mudanças de topologia.

Já a sinalização *in-band* não sofre esses problemas, pois as informações de controle são transmitidas junto com o pacote de dados. Os sistemas de sinalização *in-band* são capazes de operar com a mesma velocidade da transmissão dos dados, podendo assim responder mais rapidamente às mudanças nos ambientes *Ad Hoc*. A próxima seção irá detalhar o módulo de sinalização INSIGNIA, pois esse é o principal mecanismo do *framework* desenvolvido.

2.3.1.2 INSIGNIA *Signaling*

No *framework* INSIGNIA, as informações da sinalização *in-band* são transmitidas no campo de opções do cabeçalho IP. Esse campo no protocolo IP é opcional e de tamanho variável. No *framework* INSIGNIA ele tem 20 *bits* e possui o seguinte formato, apresentado em [Lee et al. 1999]:

O campo *reservation mode* indica se um pacote pertence a um fluxo que está requisitando uma reserva de largura de banda (preenchido com REQ), ou se essa reserva já foi efetuada (RES). Se um nó receber um pacote com o valor do *reservation mode* igual a RES e esse fluxo não estiver na tabela de reservas, então o controle de admissão será acionado para

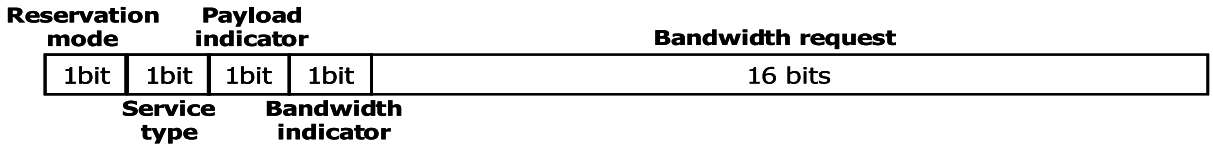


Figura 2.5: Formato do campo opções no cabeçalho IP.

tentar fazer uma reserva para o fluxo, pois essa situação indica que ele está passando por uma nova rota.

O campo *service type* pode ser preenchido como serviço adaptável (AS) ou melhor esforço (BE - *best-effort*). Porém, a interpretação desse campo é dependente do valor do campo *reservation mode*. Se a combinação desses campos for REQ/AS, por exemplo, isso significa que o pacote está requisitando uma reserva para um fluxo de tempo real. Já se os valores forem iguais a REQ/BE, isso quer dizer que o pacote não necessita da reserva de recursos, pois fará sua transmissão por melhor esforço.

O campo *payload indicator* indica o nível de QoS que o fluxo está requisitando. Os níveis podem ser o básico (*base layer* - BL) ou o fluxo melhorado (*enhanced layer* - EL). Uma reserva de banda máxima é requisitada para suportar fluxos básicos e melhorados, enquanto a reserva mínima dá suporte apenas para os fluxos básicos.

O *bandwidth indicator* pode receber os valores MAX ou MIN, que representam o potencial de disponibilidade de um recurso para um fluxo ao longo de um caminho entre a origem e o destino. Esse campo tem um papel importante nas fases de configuração do fluxo e de adaptação. Na fase de configuração, ele representa a disponibilidade dos recursos ao longo de uma rota. No processo de adaptação, o nó destino pode monitorar esse campo para iniciar uma operação de melhora de qualidade, através de um relatório de QoS, e assim restaurar uma reserva que tinha sido previamente degradada para *best-effort*.

No campo *bandwidth request* são indicados os valores máximo e mínimo de largura de banda para fluxos tempo real. O valor MIN suporta serviços básicos (*base layer*), enquanto o valor MAX suporta tanto o básico quanto o serviço melhorado (*enhanced layer*).

2.3.1.3 Funções INSIGNIA

O INSIGNIA pode executar sobre um fluxo cinco operações: (i) estabelecimento, quando um nó deseja enviar dados; (ii) restauração, no caso de quebra de rota e encaminhamento de pacotes por uma nova rota; (iii) gerenciamento *soft-state* das reservas dos recursos; (iv)

adaptação dos fluxos para o requisito de QoS possível com os recursos disponíveis; (v) envio periódico de relatórios de QoS, que são responsáveis por manter a origem informada sobre a situação da transmissão. Através dos campos descritos na seção anterior, os nós poderão executar essas operações e dar suporte a QoS para os fluxos de tempo real.

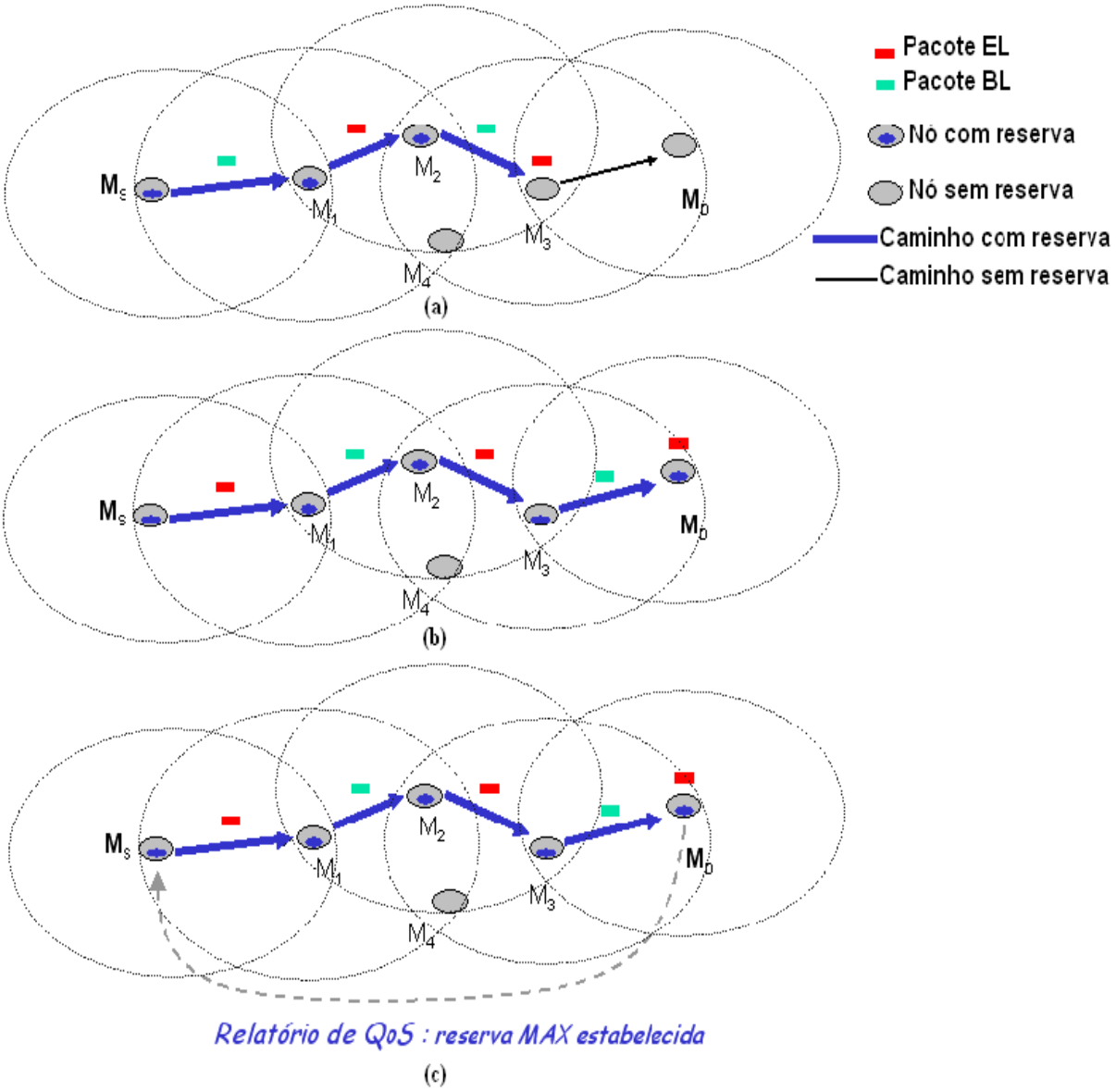


Figura 2.6: (a) e (b) Estabelecimento do fluxo; (c) Relatório de QoS.

O processo de estabelecimento do fluxo é demonstrado nas figuras 2.6(a) e 2.6(b). O nó origem envia mensagens de requisição de reserva até receber um relatório de QoS que indica o *status* da reserva. Os nós intermediários, ao receberem um pacote de requisição, executam os módulos de controle de admissão e reserva de recursos. Se apenas a requisição mínima puder ser atendida, o campo *bandwidth indicator* é preenchido com MIN. Caso esse nó receba outros pacotes de requisição de reserva para o mesmo fluxo ele apenas atualiza

o *status* da reserva, previamente efetuada.

Os relatórios de QoS são usados para informar à origem sobre o estado dos fluxos de tempo real, tanto no estabelecimento destes como no processo de adaptação. Tais relatórios são enviados periodicamente, como mostra a figura 2.6(c), em um intervalo de tempo que é sensível à Qualidade de Serviço requisitada pela aplicação. Eles não precisam passar obrigatoriamente pelo caminho reverso até a origem.

Esses relatórios também fazem parte do processo de adaptação dos fluxos. Eles são enviados à origem para informá-la de que o mecanismo de adaptação deve ser acionado, por causa de uma degradação ou melhora nas condições da rede. Mais detalhes sobre o processo de adaptação serão descritos mais adiante.

O gerenciamento de recursos chamado de *soft-state* é considerado pelos autores do INSIGNIA como o mais apropriado para ambientes dinâmicos como as redes *Ad Hoc*. Em um fluxo de tempo real, o nó origem envia os pacotes de dados continuamente por uma rota fornecida pelo protocolo de roteamento. Após o estabelecimento da sessão, o fluxo possui, em cada nó do caminho, uma reserva efetuada para transmitir seus dados.

Se um pacote chegar em um roteador em que não exista nenhuma reserva de recurso, isto significa que a rota anterior foi perdida e que uma nova rota foi fornecida. Assim, será necessário executar o controle de admissão e o mecanismo de reserva para estabelecer uma reserva nesse novo caminho. A recepção de pacotes subseqüentes do mesmo fluxo no roteador será usada para atualizar a reserva. Cada reserva é válida por um determinado intervalo de tempo. Se até o fim desse intervalo nenhum pacote for recebido, a reserva é então removida automaticamente, e os recursos, liberados.

Este tipo de reserva traz benefícios para ambientes dinâmicos como redes *Ad Hoc*. À medida que ocorrem as mudanças de rotas, novas reservas vão sendo feitas automaticamente, e os recursos alocados no antigo caminho, que não pode mais ser alcançado, são liberados quando o intervalo de tempo da reserva termina.

A mobilidade das redes *Ad Hoc* faz com que rotas sejam quebradas e os fluxos tenham que passar por outros caminhos. O processo de restauração tem como objetivo reestabelecer uma reserva no novo caminho o mais rápido possível. Esse processo é semelhante ao do estabelecimento do fluxo, pois envolve o controle de admissão e reserva de recursos do nó da nova rota. O *framework* INSIGNIA especifica dois tipos de restauração: a imediata, que é aquela em que o novo caminho possui a mesma disponibilidade de recursos da antiga rota, e a restauração degradada, em que o fluxo sofre uma queda na qualidade por falta

de recursos disponíveis.

A aplicação que estiver utilizando os serviços do INSIGNIA para obter qualidade nos fluxos de tempo real deve especificar a política de adaptação a ser utilizada. O INSIGNIA apenas determina quais são as ações que o processo de adaptação deve executar. Porém, quando e com que freqüência elas serão executadas depende da política de adaptação determinada pela aplicação.

Uma adaptação do INSIGNIA é chamada de *scale-down*. Ela é induzida pela rede devido à falta de recursos disponíveis. O processo avisa à origem que passe a mandar os pacotes EL por melhor esforço, e não por reserva. Caso a degradação da rede persista, o nó destino requisita à origem que desista de enviar os pacotes EL, mesmo que por melhor esforço. Essa fase é chamada de *drop*.

Quando o INSIGNIA percebe que os recursos ao longo do caminho voltaram a ficar disponíveis, é acionada a adaptação *scale-up*. Nela, o nó destino informa à origem que pode voltar a enviar pacotes EL e BL com requisição de reserva. É importante observar que, numa degradação, a preferência é dada aos pacotes BL que são encaminhados com o modo de reserva mínima, enquanto os pacotes EL são transferidos por melhor esforço.

A decisão de "desistir" ou "melhorar" é de inteira responsabilidade da política de adaptação especificada pela aplicação e residente no nó destino. Por exemplo, um fluxo de vídeo pode ser sensível a atrasos e a constantes mudanças da banda. Assim, uma vez que os pacotes EL são descartados, é necessário que os recursos apresentem uma disponibilidade estável para que o processo de melhora aconteça.

A situação chamada de "reserva parcial" acontece quando um fluxo que conseguiu uma reserva máxima ao longo do caminho sofre um *scale down* e passa a enviar apenas pacotes BL por reserva, enquanto envia os EL por melhor esforço. Essa situação pode ser vista como um desperdício de recursos, já que os pacotes BL necessitam apenas da reserva mínima. O tratamento a essas "reservas parciais" ainda não tem uma solução única. Por enquanto, o processo de adaptação permite que os nós apaguem essas reservas ou as mantenham nas tabelas de reservas.

2.3.1.4 Resultados

O *framework* INSIGNIA foi testado principalmente em ambiente de simulação, com o cenário montado no ns-2 [NS-2 2000]. Os testes tiveram como objetivo avaliar o desempenho do sistema de sinalização em dar suporte para a adaptação dos fluxos em uma rede *Ad Hoc*,

sob condições adversas, como grande volume de tráfego, mobilidade e interferências no canal. Os pontos analisados foram: capacidade de restauração do fluxo, suporte à adaptação, performance do mecanismo *soft-state* de gerenciamento de recursos e comportamento do sistema sob diferentes níveis de mobilidade dos nós.

Foram utilizados no cenário de testes 19 nós, que transmitiam dados num raio de 50m e tinham seus enlaces randomicamente removidos e recuperados. Para caracterizar de forma correta uma rede *Ad Hoc*, cuja conectividade entre os nós não é constante, 85% dos participantes mantiveram uma conexão ativa durante o teste. Como citado anteriormente, o protocolo de roteamento utilizado nas simulações foi o TORA.

Como foi visto, o INSIGNIA considera dois tipos de restauração: a imediata, em que o novo caminho possui a mesma disponibilidade de recursos que o anterior, e a degradada, gerada quando a nova rota não possui recursos suficientes. Os resultados da análise de desempenho da restauração do fluxo mostram que, à medida que a mobilidade dos nós aumenta, intensificando as mudanças na topologia da rede, maior é o número de restaurações imediatas e degradadas. Porém, as restaurações degradadas acontecem em maior número do que as imediatas quando a velocidade dos nós é maior do que 45km/h. Isso acontece pois com o aumento da mobilidade dos nós, a topologia é alterada rapidamente, diminuindo a conectividade da rede, causando uma queda do número de rotas disponíveis e, conseqüentemente, aumentando a concorrência pelos recursos.

De acordo com os testes, o *framework* INSIGNIA apresentou os resultados esperados na restauração dos fluxos contínuos, mantendo o suporte a QoS principalmente para os pacotes de nível básico, que têm maior prioridade. Foi observado que, mesmo com o aumento da mobilidade dos nós, a restauração de fluxos *best-effort* para fluxos com reserva mínima (nível de QoS básico) está acima dos 60%.

Para a avaliação de desenvolvimento da capacidade de adaptação do sistema, foram realizados testes em que dois fluxos foram arbitrariamente escolhidos para serem monitorados, medindo o *throughput* no nó destino ao longo da simulação. Pôde-se observar que, em diferentes graus de mobilidade (velocidade dos nós de lenta a rápida, variando entre 3km/h e 72km/h), apesar de ser usada a mesma política de adaptação, as reações às mudanças de topologia foram diferentes. Assim, quanto maior for a velocidade dos nós, menor a capacidade dos nós de melhorarem a qualidade do fluxo.

Os testes com o mecanismo de gerenciamento de recursos *soft-state* foram realizados com o intuito de avaliar o impacto causado pelo tempo de reserva na utilização dos recursos da rede. Os resultados mostraram que, quanto menor for o tempo de validade de

uma reserva, melhor será o desempenho da rede. Ou seja, mais pacotes reservados serão entregues ao destino, e menos pacotes *best-effort* irão trafegar pela rede.

A razão para esse resultado é que um intervalo de tempo grande de validade da reserva faz com que os recursos fiquem presos por mais tempo, causando muitas vezes uma subutilização da largura de banda. Os testes foram realizados variando o tempo de reserva entre 0,01 e 30 segundos. O tempo observado como o de melhor utilização da rede foi o de 2s, e o pior foi o de 30s. Contudo, se o tempo de reserva for menor que 2s, ocorre o que foi chamado de "falsa restauração", pois a reserva é removida prematuramente devido à curta duração do intervalo.

Também foi observado nos testes que o tempo válido para uma reserva pode variar de acordo com o tipo de fluxo. O valor que para um grupo de dados pode ser satisfatório, para outro pode causar excessivas "falsas restaurações". O mais apropriado é que o intervalo *soft-state* seja baseado na medição dos intervalos entre os pacotes e no *jitter*. Esse modelo foi implementado, e seus testes mostraram que essa é a maneira mais adequada de eliminar as "falsas restaurações" e evitar que recursos fiquem presos.

Por último, o comportamento do sistema sob diferentes graus de mobilidade foi avaliado num cenário em que a velocidade dos nós variou entre 0 e 72 km/h. O resultado do teste com os nós estáticos se aproximou do de uma rede fixa, em que os fluxos recebem garantias estáveis de QoS. O *framework* INSIGNIA apresentou melhores resultados nas velocidades baixas e moderadas (entre 3,6 e 18km/h), e obteve até 86% de entrega de pacotes com reserva de recursos. A natureza do sistema de sinalização *in-band* permite que, mesmo em altas velocidades, o INSIGNIA responda rapidamente às mudanças da rede. Na velocidade mais alta, 72km/h, também se observou um aumento na entrega de pacotes fora da ordem, mas que não passou dos 25%.

2.3.1.5 Avaliação

O trabalho apresentado possui idéias interessantes de serem implementadas nas redes *Ad Hoc*, solucionando problemas de forma simples. A reserva de recursos *soft-state* apresenta a flexibilidade necessária para tornar possível a implementação do mecanismo de reserva em redes *Ad Hoc*, acompanhando as freqüentes mudanças de topologia e de disponibilidade dos recursos. Em um ambiente dinâmico, a técnica de reserva de recursos torna-se mais viável quando a garantia do recurso é válida apenas por um intervalo pequeno de tempo, pois é difícil prever o comportamento da rede ao longo de uma transmissão.

Outro ponto importante é a abordagem de sinalização *in-band*. Esta técnica reduz o *overhead* de pacotes na rede, além de agilizar a percepção das mudanças do ambiente, principalmente por parte do nó origem. Quanto mais rápido forem executados mecanismos para adaptação da aplicação às alterações da rede, melhor será a qualidade da transmissão dos dados e menor será a percepção do usuário de que os recursos se tornaram escassos.

Porém, se por um lado a reserva de recursos garante largura de banda por um intervalo de tempo, por outro, caso a mobilidade da rede seja intensa, o intervalo de tempo da reserva não será suficiente para garantir banda no caminho, pois a rota será alterada constantemente. Como foi mostrado nos resultados dos testes no INSIGNIA, é necessário encontrar um intervalo de reserva apropriado para cada aplicação, para cada tipo de fluxo.

Uma desvantagem do *framework* INSIGNIA é que ele utiliza a largura de banda como único requisito de QoS a ser suportado pela aplicação, enquanto outros fatores como perda de pacotes e *jitter* também são importantes na transmissão de fluxos de tempo real. Esses requisitos poderiam ser medidos e serem utilizados no método de admissão e monitoramento do fluxo.

Um ponto importante que o INSIGNIA deixa a desejar é sobre os níveis de QoS que ele suporta. Os fluxos alvo deste trabalho, como áudio e vídeo, quando transmitidos em redes *Ad Hoc*, devem ter seus requisitos de QoS especificados na forma de intervalos, e não pontuais. No caso do *framework* em questão, ele apenas suporta dois níveis de QoS: o básico, que especifica uma largura de banda mínima, e o melhorado, que requisita uma largura de banda máxima. Para se adequar à flexibilidade das redes *wireless* seria mais interessante que a largura de banda fosse apresentada em intervalo, fazendo com que cada nó reservasse o recurso dentro de uma faixa [MIN, MAX].

O INSIGNIA é um *framework* que se encontra entre as camadas de rede e de transporte. Assim, pode-se perceber que a reserva de banda implementada não é diretamente na camada de enlace MAC, responsável pelo controle de acesso ao meio. Como visto, a largura de banda disponível que será reservada é representada por um valor estático. A banda utilizada é medida na tabela de reserva dos fluxos. O que o *framework* faz é garantir que os pacotes dos fluxos reservados terão maior prioridade de entrega ao passarem pelos nós do caminho. Na verdade, a transmissão feita é uma espécie de melhor esforço com filas de prioridade.

2.3.2 ASAP

Este trabalho [Xue et al. 2003] apresenta o *framework* chamado ASAP (*Adaptive Reservation and Pre-Allocation*), que tem como objetivo fornecer Qualidade de Serviço para redes *Ad Hoc*, apoiando aplicações que lidem com transmissões multimídia. Os autores do ASAP basearam sua arquitetura no INSIGNIA, com o intuito de superar limitações existentes nesse trabalho. Eles argumentam que o INSIGNIA possui alguns pontos negativos e que são questões essenciais para um *framework* de suporte a QoS em redes *Ad Hoc*. Por exemplo, um protocolo de QoS deve ser capaz de fornecer qualquer nível de QoS dentro de um intervalo [Min, Max] de largura de banda e não apenas dois níveis como faz o INSIGNIA. Também é observado que num mecanismo de reserva de recursos, o excesso dessas reservas deve ser evitado ou minimizado, para não causar um desperdício dos recursos. Outro ponto muito importante é a redução do *overhead*, causado pelas mensagens de sinalização e pelo seu processamento.

As aplicações alvo para os testes são principalmente as de tempo real com transmissão de dados multimídia. Todos os testes do ASAP foram feitos em ambiente de simulação (ns-2). Foram criados diferentes cenários para avaliar diferentes características do protocolo.

2.3.2.1 Framework ASAP

O protocolo ASAP utiliza o mecanismo de reserva de recurso. Esta reserva é feita em duas fases, para evitar que sejam realizadas reservas em excesso ou que reservas que não estejam sendo mais usadas fiquem prendendo recursos ociosos. A primeira fase a ser executada é a fase *soft*. Quando uma requisição de reserva chega em um nó, este executa a fase *soft* da reserva, verificando se há banda disponível e efetuando a reserva para uso futuro. Esta banda reservada não será utilizada de imediato, podendo ser usada temporariamente por fluxos *best-effort*, enquanto a reserva não é confirmada. Após a fase *soft*, a reserva *hard* faz o caminho inverso da mensagem *soft*, limpando as reservas temporárias e admitindo a reserva de recurso para o fluxo de tempo real. Depois da fase *hard*, nenhum outro fluxo poderá utilizar a banda da reserva confirmada.

Para entender melhor o mecanismo é necessário compreender o sistema de mensagens de sinalização. As mensagens das fases *soft* e *hard* possuem formatos semelhantes. Uma mensagem da fase *soft*, SR, possui os seguintes campos: *MinBW*, *MaxBW*, *SoftBW* e *HardBW*. Os dois primeiros campos indicam os valores mínimo e máximo da largura de banda a ser reservada. Os dois últimos mostram qual o valor da largura de banda que já foi

reservada no caminho em cada fase. A mensagem HR, que é enviada na fase *hard*, contém 3 campos: *SetBW*, *SoftBW* e *HardBW*. Os dois últimos possuem a mesma definição dos campos equivalentes na mensagem SR. Já o campo *SetBW* define a quantidade de banda que todos os nós intermediários devem reservar igualmente.

De forma semelhante ao INSIGNIA, o ASAP também possui algumas funções que devem ser executadas para o processo de fornecimento de QoS. Para iniciar uma transmissão é preciso estabelecer a reserva para o fluxo. Como já foi mencionado, a primeira fase a ser executada na formação de um fluxo é a reserva *soft*. O nó origem então envia uma mensagem SR para o destino. Todas as mensagens SR são transferidas *in-band*, ou seja, juntamente com o fluxo de dados. Da mesma forma que no INSIGNIA, as informações das mensagens *soft* são carregadas no campo opções do cabeçalho IP. Quando o primeiro nó intermediário recebe a mensagem SR ele cria uma entrada na tabela de fluxos, fazendo a reserva *soft* dentro do intervalo Min-Max de banda e de acordo com a largura de banda disponível. Na mensagem SR, ele preenche o campo *SoftBW* com o valor da banda reservada e o campo *HardBW* recebe o valor 0. Os outros nós repetem o mesmo procedimento, mas apenas atualizam o valor do campo *SoftBW* se a banda reservada for menor que o valor original do campo.

Quando o nó destino recebe a mensagem SR ele verifica a largura de banda reservada, presente no campo *SoftBW*, e então responde à origem enviando uma mensagem HR com o valor de *SetBW* igual ao campo *SoftBW* da SR. A mensagem HR faz o caminho reverso ao da mensagem SR, permitindo que cada nó intermediário possa trocar a reserva temporária *soft* pela definitiva *hard* com o valor do campo *SetBW*. Assim que a origem receber a mensagem HR ela pode iniciar o fluxo multimídia com a velocidade correspondente à largura de banda reservada. As mensagens HR são enviadas *out-of-band*.

Depois do fluxo ser estabelecido, mensagens SR são inseridas periodicamente no fluxo, com o objetivo de coletar informações de QoS ao longo do caminho. Deste modo, o destino pode manter a última situação de QoS armazenada e enviar ao emissor essas informações de qualidade usando mensagens HR, para que aplicações de tempo real possam se adaptar aos requisitos de QoS.

Ao receber essas mensagens SR de monitoração, cada nó intermediário irá verificar se a requisição máxima de banda para o fluxo corrente já foi totalmente atendida na reserva. Caso não tenha sido reservada, e houver largura de banda disponível, o nó tentará alocar mais recursos para aumentar a reserva efetuada. Se cada nó intermediário conseguir efetuar uma reserva extra para o fluxo, quando a mensagem SR chegar ao destino este

perceberá que banda reservada aumentou. A mensagem HR que ele enviará de volta à origem fará com que cada nó do caminho confirme a reserva extra que tinha efetuado na fase *soft*. As adaptações na origem serão realizadas tanto no caso de uma diminuição quanto no de um aumento da largura de banda disponível no caminho. A velocidade de transmissão do fluxo de tempo real será adaptada de acordo com a largura de banda informada pela mensagem HR.

O mecanismo de reparo do caminho do ASAP é simples e também semelhante ao INSIGNIA. Quando um nó recebe uma mensagem SR que não pertence a nenhum dos fluxos da sua tabela de reservas, e os campos *SoftBW* e *HardBW* não forem iguais a 0, isso significa que o fluxo está fora do caminho original. O nó então inicia o processo de conserto do caminho. Ele irá criar uma nova entrada na tabela de fluxos e fazer uma reserva *soft* de acordo com o intervalo [MinBW, MaxBW] da mensagem SR. Diferentemente do processo de estabelecimento do fluxo, o nó trocará imediatamente a reserva *soft* pela reserva *hard* com o valor indicado pelo campo *HardBW* da SR. A mensagem SR é então modificada e enviada para o próximo nó. Quando o nó destino receber a mensagem SR, o caminho quebrado já estará restabelecido.

A liberação do fluxo pode ser feita de duas formas: ou de forma implícita, por *time-out*, ou explicitamente enviando uma mensagem HR com o campo *SetBW* com o valor 0.

2.3.2.2 Resultados

O ASAP foi avaliado em quatro aspectos. Primeiramente, os testes analisaram o desempenho do *framework* na adaptação de um fluxo para fornecer QoS de acordo com a qualidade do enlace. Os resultados apresentaram apenas uma pequena porcentagem do tráfego perdendo as garantias de QoS. O tempo que o nó leva para reagir às alterações da banda está relacionado com o intervalo em que as mensagens SR são enviadas. Este intervalo ajuda a determinar a velocidade com que o ASAP irá se adaptar às mudanças da rede. Um intervalo pequeno pode acelerar o processo de adaptação, como também pode gerar mensagens excessivas e alto processamento nos nós. O tempo entre mensagens SR deve ser determinado de forma flexível, de acordo com as condições da rede.

O segundo aspecto avaliado foi o desempenho do *framework* em prover qualidade em diferentes circunstâncias de mobilidade. O testes foram realizados com 20 nós móveis, com velocidades variando entre 10km/h e 100km/h. O total de pacotes que tiveram violação de QoS inclui pacotes tratados como *best-effort* e pacotes que são abandonados por falta de recursos disponíveis. A porcentagem deste último tipo, no pior caso, não chega a 2%.

Os resultados mostram que, conforme a mobilidade dos nós de uma rede *Ad Hoc* aumenta, cresce também a porcentagem de pacotes que são tratados como *best-effort*. Em altas velocidades, as constantes mudanças de rotas fazem com que o trabalho de restauração dos níveis de QoS seja um esforço contínuo. Mesmo assim, a porcentagem de pacotes que tiveram QoS degradado fica em torno de 10% na velocidade máxima, segundo os autores.

A eficiência com que o *framework* gerencia os recursos da rede é medida pelo excesso de reservas, isto é, a razão entre a capacidade de transferência total reservada em toda a rede e o total de tráfego QoS processado. No melhor caso, o excesso de reservas será igual a 1, indicando que todas as reservas estão sendo utilizadas. Foram realizados testes com e sem o mecanismo de reserva *Soft/Hard*. Na velocidade de 10km/h, por exemplo, as reservas em excesso tiveram uma queda de 28% com a utilização do ASAP, em relação aos testes realizados sem o mecanismo proposto. Os resultados mostram que, independentemente do nível de mobilidade dos nós, a quantidade de reservas excedentes é maior quando não é usado o mecanismo de reserva *Soft/Hard*. A redução média que o ASAP conseguiu foi de 20% a 30%.

Já o último aspecto da avaliação mediu o *overhead* gerado pela sinalização e está intimamente ligado ao valor do intervalo de envio das mensagens SR. Quanto menor for este intervalo, maior o *overhead*, porém menor será a degradação de QoS. O ajuste de forma adaptável deste intervalo entre as mensagens SR pode ajudar a minimizar o *overhead*.

2.3.2.3 Avaliação

Um ponto negativo da proposta ASAP é que a fase *Hard* da reserva de recursos pode causar uma sobrecarga na rede, pois o destino envia periodicamente pacotes para a origem pelo mesmo caminho do fluxo de tempo real. Assim, os pacotes podem colidir ou até mesmo serem descartados pelos nós intermediários por falta de espaço nos *buffers*.

Um aspecto muito importante para o entendimento completo do protocolo ASAP e que não foi apresentado no artigo é o mecanismo de determinação de banda disponível no nó. Para redes sem fio, esse não é um mecanismo trivial, pois deve-se levar em conta a transmissão e recebimento dos nós vizinhos, que também estão interferindo no canal.

2.4 Soluções de QoS Sem Reserva de Recursos

Alguns trabalhos para fornecimento de QoS em redes *Ad Hoc* defendem que o mecanismo de reserva de recursos não é adequado para este tipo de rede. O argumento utilizado é que os ambiente móveis e sem fio possuem características, como conectividade sujeita a interferências e a própria mobilidade, que impedem o fornecimento de garantias de disponibilidade dos recursos. Os trabalhos apresentados argumentam que apenas com mecanismos de adaptação às mudanças da rede é possível oferecer os níveis de QoS requeridos pelas aplicações. O artigo [Sanzgiri e Belding-Royer 2004] propõe uma alteração do protocolo de roteamento AODV, inserindo requisitos de QoS na descoberta de rotas. O trabalho apresentado na seção 2.4.2 [Ruiz e Garcia 2002] desenvolve um componente com uma lógica de adaptação que, de acordo com alguns requisitos de QoS, como largura de banda e perda de pacotes, determina como os recursos do sistema serão utilizados pela aplicação.

2.4.1 AODV Adaptável

Esse trabalho tem como objetivo apresentar uma modificação do protocolo de roteamento AODV para que este tenha conhecimento dos requisitos de QoS ao longo do processo de estabelecimento de uma rota [Sanzgiri e Belding-Royer 2004]. Além disso, visa também tirar vantagem de uma importante característica das redes *Ad Hoc*: a mobilidade dos seus nós. Dois protocolos foram desenvolvidos e testados: o Q-AODV (*QoS-Aware AODV*), que é uma modificação direta do AODV, inserindo requisitos de QoS determinados pela aplicação no processo de descobrimento de rota; e o MQ-AODV (*Mobility-Enhanced QoS-Aware AODV*) que é uma extensão do Q-AODV, onde, além do uso dos requisitos de QoS, também é proposta uma mudança de localização para o nó com o objetivo de fornecer uma melhor qualidade de serviço.

Os autores ressaltam que os protocolos propostos não apresentam o mesmo desempenho nos diferentes cenários que uma rede *Ad Hoc* pode ser implantada. De acordo com as características do protocolo, ele obtém uma performance melhor em redes com baixa ou média mobilidade, onde os seus nós representam transeuntes ou estejam até mesmo parados. Até por isso, o cenário idealizado para a aplicação destes protocolos seria o *campus* de uma universidade, onde estudantes e professores estariam de posse de seus dispositivos móveis, formando entre eles uma rede *Ad Hoc*.

Os protocolos propostos iriam auxiliar no fornecimento de QoS para aplicações que

lidam com dados multimídia. Por exemplo, caso um professor necessitasse participar de uma conferência, o protocolo de roteamento iria fornecer a melhor rota que não estivesse congestionada, ou então iria sugerir um deslocamento para uma região em que o meio não estivesse tão congestionado. Já em uma rede *Ad Hoc* com alta mobilidade, como, por exemplo, carros se movimentando numa auto-estrada, a idéia de sugerir uma mudança de localização não seria bem aplicada.

Os protocolos propostos não foram implementados num ambiente real e sim desenvolvidos e testados no simulador *ns-2*. A aplicação utilizada para testá-los foi modelada como uma aplicação multimídia simples. Ela é uma aplicação cliente-servidor, onde o servidor envia *streams* de áudio e vídeo requisitados pelo cliente e ambos os nós estão estáticos. No caso do protocolo MQ-AODV, o cliente e o servidor só se moveriam se precisarem se aproximar de um nó para obter uma melhor qualidade na transmissão.

2.4.1.1 Q-AODV

Esta modificação do AODV tem como objetivo alterar o protocolo de roteamento para que ele forneça à aplicação uma rota que tenha maior probabilidade de atender aos requisitos de QoS por ela especificados. O requisito de QoS escolhido é a largura de banda, pois este é um dos aspectos mais importantes para uma transmissão adequada de dados multimídia. A mensagem de requisição de rota (RREQ) é alterada para receber três parâmetros, que representam a largura de banda tolerável: banda mínima, banda máxima e banda no gargalo (*bottleneck*). Os dois primeiros valores são preenchidos na origem, de acordo com a especificação definida pela aplicação. O parâmetro que representa o gargalo vai sendo preenchido ao longo do descobrimento de rota, com a menor largura de banda encontrada.

A fase de descobrimento de rota sofreu algumas modificações. O nó origem irá enviar por *broadcast* a mensagem RREQ com os novos campos devidamente preenchidos. Cada nó que recebe essa mensagem verifica se ele é o nó destino desejado. No algoritmo AODV tradicional, quando um nó intermediário recebe uma mensagem RREQ, ele apenas a encaminha adiante e não faz nenhum processamento. No Q-AODV, o nó intermediário, após verificar que não é o destino da mensagem, avalia se a sua largura de banda disponível está dentro do limite definido na mensagem. Caso afirmativo, o nó verifica se o campo *bottleneck* está vazio ou se possui um valor maior do que a sua banda disponível. Se sim, ele copia para este campo o seu valor corrente de largura de banda. Se não, ele apenas encaminha a mensagem RREQ para os outros nós. Caso o seu valor de banda disponível não esteja entre os valores de largura mínima e máxima, definidos no RREQ,

o nó descarta a mensagem. Desta forma, se a rede *Ad Hoc* utilizada não possuir em sua topologia uma rota que atenda aos requisitos de largura de banda determinados pela aplicação, o protocolo de roteamento não irá fornecer nenhum caminho entre os nós origem e destino.

Quando a requisição de rota alcança o nó destino, este responde enviando a rota encontrada na mensagem RREP (*Route Reply*). A única alteração feita nesta mensagem foi a inclusão de um campo que especifica o valor da largura de banda disponível no caminho, e que é preenchido com o valor do campo *bottleneck* da mensagem RREQ.

No protocolo Q-AODV, diferentemente de algumas versões do AODV, a mensagem RREQ não pode ser respondida por nenhum nó intermediário. A requisição de rota deve percorrer todo o caminho da origem até o destino, para poder estabelecer qual é a largura de banda disponível em toda a rota. Outra diferença importante é que no Q-AODV o nó destino pode e deve responder a mais de uma requisição de rota. Isso porque não significa que a primeira mensagem de RREQ que chegue no destino seja a que tenha melhor banda disponível. Cabe então a origem, após receber por um intervalo de tempo as mensagens RREP, escolher qual dentre as que chegaram a que apresenta uma rota com melhor requisito de QoS.

2.4.1.2 MQ-AODV

O outro protocolo proposto, o MQ-AODV, é uma extensão do Q-AODV, que apresenta novos aspectos para aprimorar a aquisição de rota com fornecimento de QoS. Para o MQ-AODV ser executado, todos os nós devem possuir algum sistema de localização geográfica por coordenadas, como por exemplo, um GPS. Os autores supõem que todos os nós possuem um raio de transmissão padrão e uniforme, mas são capazes de aumentar o seu poder de transmissão para alcançar raios maiores, se for necessário. Também não é considerada a presença de obstáculos.

O MQ-AODV mantém uma interação com o usuário. Para iniciar o protocolo é preciso que o usuário informe se deseja se locomover para obter a qualidade de serviço desejada, caso seja necessário. Ele também deve especificar a área em que deseja se mover. Caso contrário será assumida uma área padrão. No trabalho realizado, foi assumido que o usuário está preparado para se mover para qualquer direção numa distância máxima especificada, ou seja, uma área circular de raio D.

Os autores definiram alguns aspectos para conduzir a maneira com que os nós da rede

devem se movimentar. Primeiramente, é suposto que qualquer nó se movimentará apenas por uma zona de área circular de raio *D*. O segundo aspecto é o grupo de vizinhos em potencial. Esse grupo define os nós próximos ao nó origem que potencialmente podem ser escolhidos como o primeiro nó da rota para o destino. Porém, o grupo de vizinhos em potencial não abrange apenas os nós que estão dentro do raio *rt* de alcance de transmissão do nó origem, mas também aqueles em que o emissor precisa se mover em direção a eles para alcançá-los. O grupo então é definido através de uma área circular em torno do nó origem de raio *(D + rt)*, ou seja, os nós que estão ao seu alcance mais os que estão na zona de movimentação, além do alcance.

Com esses parâmetros definidos, o MQ-AODV pode iniciar a fase de descobrimento de rota. A mensagem RREQ será enviada em *broadcast* com a maior capacidade de transmissão, para que alcance os nós do grupo de vizinhos em potencial. A mensagem RREQ recebe os mesmos parâmetros inseridos no Q-AODV, banda máxima, mínima e banda no gargalo. Porém, ela também deve armazenar as coordenadas de localização no nó. O encaminhamento das requisições RREQ acontece da mesma maneira que no Q-AODV, exceto que os vizinhos em potencial armazenam a localização do nó origem.

Ao chegar no destino, a mensagem RREP é enviada para a origem da mesma forma que no Q-AODV. Porém, cada nó intermediário quando recebe uma RREP, verifica se ele é o último nó antes de chegar a origem. Caso ele seja, isso significa que ele pertence ao grupo de vizinhos em potencial. Por isso precisa verificar qual a distância que ele está do nó origem através da localização armazenada quando a mensagem RREQ passou por ele. Se a distância for maior do que o alcance regular de transmissão, ele deve encaminhar a mensagem RREP usando a capacidade maior para transmissão, possibilitando que a mensagem alcance o nó origem. Também é incluída na mensagem RREP a localização do último nó.

Ao receber a mensagem RREP o nó origem verifica se a localização do primeiro nó da rota está dentro do seu alcance regular ou não. Caso positivo, o nó pode utilizar a rota imediatamente. Se não estiver no alcance, o usuário é avisado para que possa se mover em direção ao seu nó vizinho, fazendo que este entre em seu alcance de transmissão. Como já mencionado, nessa implementação a origem pode receber mais de uma mensagem RREP. Por isso, o usuário pode esperar por um certo tempo até que várias mensagens RREP cheguem e ele possa decidir qual a melhor rota a ser escolhida, para depois então se mover, caso necessário.

2.4.1.3 Limitações

Em relação aos dois protocolos desenvolvidos, os autores apontaram algumas limitações que propuseram solucionar em trabalhos futuros. Um problema a ser resolvido é que se múltiplas sessões tentarem executar o descobrimento de rota simultaneamente todas elas serão aceitas, mesmo que os recursos disponíveis sejam insuficientes. Isso acontece porque o protocolo não apresenta nenhuma política de (pré) reserva de recursos, apenas de estimação de largura de banda.

Já para o MQ-AODV, as limitações são outras. No sistema de movimentação para aprimoramento de QoS, se o nó origem estiver muito perto de uma região altamente congestionada, ele deverá se mover uma distância muito grande para sair do congestionamento, sendo que muitas vezes isso não será possível. Um outro problema é que se muitos usuários executarem o descobrimento de rota ao mesmo tempo eles provavelmente irão se locomover na mesma direção. O local para onde esses nós estão se movendo pode ser incapaz de suportar a requisição de recursos de todos os usuários e tornar-se congestionado.

Um benefício gerado pelo MQ-AODV é o de possibilitar que um grupo de nós se conecte a partes da rede que antes estavam desconectadas. Pelo fato dos nós terem que se mover para melhorar a largura de banda disponível da rota, muitas vezes eles se aproximam de grupos que não se comunicavam, formando assim pontes entre esses grupos e os seus nós vizinhos. Por outro lado, a movimentação dos nós também pode gerar desconexões, separando grupos de nós que anteriormente se comunicavam.

É necessário avaliar também como os protocolos reagem em ambientes com obstáculos. Neste caso, muitos vizinhos em potencial podem se tornar inatingíveis, não sendo possível que o usuário caminhe em sua direção.

2.4.1.4 Resultados

O cenário montado para os testes inclui 50 nós randomicamente colocados numa área de 1000m x 1000m. O tempo de cada execução foi de 500 segundos de tempo de simulação. Foi usado o modelo de propagação *Two Ray Ground*, com alcance de transmissão de 250m, podendo aumentar esse alcance para 550m. Desses 50 nós, 20 deles são estáticos e 30 se movem de acordo com o modelo *Random-WayPoint*, com pausa de 20 segundos. Desses nós móveis, 10 se movem com velocidade até 5m/s, 10 com velocidade entre 5m/s e 10m/s, e os outros 10 nós entre 10m/s e 20m/s. Isso foi feito para simular um cenário de uma

universidade, onde se encontram pessoas paradas, outras caminhando, e algumas andando de bicicleta ou dirigindo um automóvel. Dos 20 nós estacionários, dois foram escolhidos para serem o servidor e o cliente. Eles só se moveram caso fosse necessário para melhorar a qualidade de serviço fornecida para transmissão da mídia.

A avaliação do desempenho dos dois protocolos propostos foi feita com base em algumas métricas definidas pelos autores. A fração de pacotes entregues representa a razão entre os pacotes de dados enviados pelo servidor e os pacotes recebidos pelo cliente. Quanto maior for este valor, mais efetivo o protocolo é em reduzir o congestionamento na rede. O atraso médio dos pacotes fim-a-fim foi medido pelos pacotes recebidos pelo cliente, para avaliar o congestionamento da rede e se os dados foram entregues a tempo de serem reproduzidos. Também foi avaliada a fração de pacotes recebidos com um atraso fim-a-fim inaceitável, para verificar se os recursos da rede foram usados de forma eficiente. Por último, a carga do roteamento mediu o *overhead* que o protocolo causa com o envio de mensagens de controle na rede.

Todas as métricas foram medidas em comparação com o número de sessões de dados criadas. Os testes realizados com os protocolos propostos Q-AODV e MQ-AODV obtiveram melhores resultados do que o AODV puro. A fração de pacotes entregues, por exemplo, fica em torno de 90% no Q-AODV e 95% no MQ-AODV, enquanto no AODV ela cai para 50% quando o número de sessões cresce. O atraso médio dos pacotes nos protocolos propostos fica entre 0 e 10ms, enquanto no AODV ele cresce exponencialmente conforme aumenta o número de sessões. Isto se reflete no resultado da frações de pacotes entregues fora do tempo aceitável, sendo igualmente exponencial no AODV e constante no Q-AODV e MQ-AODV.

É necessário ressaltar que, mesmo os resultados sendo comparados entre os protocolos AODV, Q-AODV e MQ-AODV, o AODV não realiza um controle de admissão de sessão de dados, ao contrário dos outros dois protocolos. A comparação entre os resultados do desempenho dos três protocolos é feita para ressaltar a importância de um controle de admissão na descoberta de rotas.

Os testes provaram que os protocolos de roteamento podem ser aperfeiçoados para também fornecer QoS para a aplicação, através do descobrimento de rotas que atendam aos requisitos. Ainda mostrou que a inserção da mobilidade como fator de aumento de QoS, no caso do MQ-AODV, trouxe benefícios para a aplicação.

2.4.1.5 Avaliação

O protocolo Q-AODV apresenta uma abordagem interessante de descobrimento de rota, que avalia os requisitos de QoS encontrados no caminho. Porém, restringir esses requisitos a apenas largura de banda disponível reduz as possibilidades de rotas. Esse conceito poderia ser ampliado para a fase de manutenção de rota, onde outros requisitos como perda de pacotes e *jitter*, poderiam também ser monitorados. Além disso, o fato do protocolo fornecer uma rota apenas quando for encontrada uma que atenda aos requisitos de QoS especificados restringe as possibilidades do usuário. Ele deve ter o poder de escolha, e poder resolver se não transfere o tráfego, se o envia por melhor esforço, ou reduz a qualidade da transmissão, sabendo que o caminho disponível não dispõe de recursos suficientes para o atendimento de qualidade.

O protocolo MQ-AODV, através dos resultados, apresenta uma melhora na qualidade da transmissão. Porém, um protocolo de roteamento deve funcionar de forma transparente para o usuário. O MQ-AODV, para fornecer QoS para a aplicação, depende da vontade do usuário se locomover ou não. Seria mais interessante criar uma interface com o usuário em que lhe é apresentada informação sobre a qualidade do sinal na área em que ele está e numa área em torno dele, para que, independentemente do protocolo de roteamento, possa escolher uma rota com QoS, ou se mover para uma área menos congestionada. Contudo, este sistema também necessitaria que todos os nós da rede possuíssem um GPS, o que não é comum nos dispositivos móveis atualmente no mercado. Apenas a visualização da qualidade do sinal na posição onde ele se encontra, como num celular, já auxiliaria na obtenção de QoS para o fluxo.

2.4.2 Lógica de Adaptação

O objetivo desse trabalho [Ruiz e Garcia 2002] é o de apresentar uma arquitetura de adaptação para redes sem fio que permita que as aplicações minimizem o impacto para o usuário causado pelas mudanças e condições adversas de um ambiente *wireless*. Essa diminuição de efeitos é feita através de técnicas de ajuste aplicadas nas camadas mais altas, utilizando informações da rede. Com esse trabalho, os autores pretendem mostrar que é possível alcançar um nível de QoS satisfatório em redes sem fio fazendo com que as aplicações se adaptem às mudanças ocorridas na rede, ao invés de confiar em reservas de largura de banda que são difíceis de serem garantidas.

Esse trabalho tem como foco os ambientes sem fio e com mobilidade dos nós. A

arquitetura proposta não foi testada em ambiente de simulação, tendo sido desenvolvido um protótipo com base em algumas implementações existentes [Kassler et al. 2000] [Systems]. O artigo não fornece detalhes dessa implementação, como os métodos utilizados para medição de perda de pacotes, *jitter* e largura de banda estimada. A descrição desses mecanismo iria enriquecer o conteúdo do trabalho.

2.4.2.1 Técnicas e Algoritmos

A arquitetura do sistema proposto constitui-se basicamente de dois elementos que trabalham em conjunto: o mecanismo de sinalização de QoS e o componente chamado *Adaptation Logic*. O segundo componente é o mais importante, sendo aquele que apresenta o diferencial do trabalho.

Este componente foi criado tendo a responsabilidade de decidir, de acordo com um grupo de parâmetros, qual é a melhor forma para se adaptar às condições da rede num dado momento. Assim, o *Adaptation Logic* possui uma lógica que utiliza três aspectos da transmissão de dados para gerar suas saídas. São esses: largura de banda fim-a-fim, *jitter* e porcentagem de pacotes perdidos num dado período. A perda de pacotes é o principal responsável pelos efeitos que mais são percebidos pelo usuário. Os autores ainda explicam porque não consideram o problema do atraso fim-a-fim como um parâmetro importante na definição de um sistema deste tipo, apesar de ser um aspecto que também gera interferências na reprodução da mídia. Usualmente, os problemas de atraso de pacotes podem ser evitados com técnicas de gerenciamento de *buffer* apropriadas, sem necessitar reduzir o uso da banda.

A lógica para adaptação desenvolvida no componente *Adaptation Logic* é acionada quando a qualidade de reprodução da mídia é degradada. Essa degradação é percebida quando a porcentagem de pacotes perdidos exceder a 5% ou quando três relatórios consecutivos de QoS forem perdidos. Em contrapartida, a qualidade é aprimorada quando quatro relatórios de QoS consecutivos recebidos indicarem 0% de perda de pacotes. Esses parâmetros foram determinados pela equipe de desenvolvimento de acordo com as experiências adquiridas em outros trabalhos, mas podem ser ajustados, dependendo do ambiente utilizado. A solução mais apropriada seria que esses valores fossem definidos de forma dinâmica, como, por exemplo, por algum mecanismo de aprendizagem inserido no *framework*. A implementação de alguma técnica de inteligência para aprendizagem, como Lógica *Fuzzy*, no componente *Adaptation Logic* é proposta para o futuro.

A segunda parte do projeto, que também é muito importante para todo o sistema e

que trabalha em conjunto com o *Adaptation Logic*, é o mecanismo de sinalização de QoS. O mecanismo implementado é simples e permite que a origem tenha conhecimento das condições da transmissão. A qualidade de serviço é medida de acordo com a porcentagem de pacotes perdidos e a largura de banda estimada fim-a-fim. Esses relatórios são enviados periodicamente para o nó emissor. Se a origem não receber um desses relatórios dentro do intervalo de tempo determinado, é então detectada uma perda dessa mensagem, causada provavelmente por uma quebra de rota ou interferência no canal. Caso contrário, ao receber um relatório, o emissor encaminha-o para o componente *Adaptation Logic*, que então executa a lógica já descrita anteriormente.

Como já explicado, o componente *Adaptation Logic* gera dados de saída a serem informados para a aplicação. Essas informações podem ser a taxa de amostragem da mídia mais adequada ou o *codec* apropriado para as condições da rede. A aplicação ISABEL [Systems], utilizada para videoconferência e que manipula diferentes fluxos RTP, foi estendida para dinamicamente adaptar seu comportamento à disponibilidade dos recursos. As principais capacidades de adaptação implementadas foram:

◇ **Escolha de *Codec*:** a aplicação pode escolher os *encoders* a utilizar, como H263 ou MJPEG para vídeo e GSM, G722 ou G711 para áudio;

◇ **Taxa de amostragem:** o *Adaptation Logic* pode variar a taxa de amostragem (quadros por segundo) que a aplicação pode transmitir, permitindo a economia de largura de banda;

◇ **Tamanho:** em ambientes em que a banda é limitada, o usuário pode preferir assistir a vídeos numa visualização menor do que com uma qualidade ruim;

◇ **Bufferização:** *buffers* que se adaptam de forma inteligente e dinâmica ajudam a oferecer uma melhor qualidade em condições adversas da rede, amenizando os efeitos do *jitter* e do atraso.

2.4.2.2 Resultados

Os ambientes utilizados para teste foram dois. No primeiro, foi montada uma rede com computadores *desktop* interligados, com funcionalidades de roteamento, e um *laptop* representando o nó móvel. A estrutura desta rede possui enlaces na sua maioria *Ethernet* (10Mbps) e alguns ISDN PPP (64kbps). O objetivo desta diferença dos enlaces será explicado posteriormente. Todo o sistema executa em Linux, utiliza pilha Ipv4/Ipv6 e

Hierarchical Mobile IP[2] em seus testes.

O segundo cenário apresenta uma rede *Ad Hoc* com apenas três nós. Esses nós são computadores *desktop* com a mesma formatação apresentada no cenário anterior. Em ambos os cenários, foram executados testes utilizando o componente de adaptação e testes sem o *Adaptation Logic*.

Os testes no primeiro cenário tiveram 100 segundos de duração. Ao longo do teste foram realizados dois *handovers*: o primeiro foi do ambiente com a banda mais alta (*Ethernet* 10Mpbs) para a banda mais escassa (64kbps), e o segundo fez a passagem de volta para o ambiente *Ethernet*. Foram executados testes com e sem o mecanismo de adaptação proposto. Os resultados da transmissão de áudio sem o mecanismo de adaptação mostraram a perda total dos pacotes enviados no momento do primeiro *handover* e uma grande variação do *jitter*. Já nos testes realizados utilizando o componente *Adaptation Logic*, após a queda da banda disponível, o sistema informou à aplicação para alterar o *codec* de áudio a ser utilizado, minimizando o consumo de banda para menos de 20Kbps, mantendo o *jitter* constante em 60ms.

No sistema *Ad Hoc*, comparando os resultados dos testes com e sem o componente *Adaptation Logic*, a porcentagem de perda de pacotes quando foi utilizado o mecanismo de adaptação foi 66% menor do que na ausência do componente. Os pacotes perdidos nestes casos são devidos a perdas que não excederam o valor limite de 5% para que o mecanismo de adaptação fosse acionado. Mesmo assim, essas pequenas perdas ainda estão bem abaixo do valor de 20% de perda tolerável para que a baixa qualidade do áudio comece a ser percebida pelo usuário.

2.4.2.3 Avaliação

Os resultados deste trabalho mostram que os autores conseguiram alcançar seu objetivo, ou seja, fornecer QoS para sistemas sem fio que lidem com dados multimídia apenas utilizando adaptabilidade, ao invés de reserva de recursos. Porém, alguns pontos não foram claramente explicados. Além disso, os cenários dos testes apresentaram redes muito pequenas (3 nós na rede *Ad Hoc*), o que não permite que seja induzido que esses resultados

[2] *Hierarchical Mobile IP* significa que os *foreign agents* (FA) podem ser construídos de forma hierárquica entre o *home agent* (HA) e o nó móvel. Esses *foreign agents* formam uma árvore e o FA de nível mais baixo se comunica com os nós móveis. A hierarquia foi criada para permitir que o IP Móvel tenha um desempenho melhor em situações de *handovers* constantes. A idéia é que o nó móvel não tenha que sempre se registrar no *home agente*, mas enquanto está se movendo juntamente com o FA, o tunelamento possa ser alterado internamente, sem precisar avisar ao HA. Assim, as conexões podem ser mais rápidas entre os FA. Mais detalhes em [Dynamics 2001].

possam ser válidos para redes maiores.

A técnica de estimação de banda utilizada não foi descrita, como também o mecanismo de cálculo da porcentagem de perda de pacotes e a medição do atraso e do *jitter*. Os mecanismos de bufferização apontados como importantes para minimizar os efeitos do *jitter* e do atraso também não foram mencionados.

A contribuição interessante deste trabalho se encontra no mecanismo de sinalização de QoS implementado. É importante para a origem saber das condições da rede e de como está a qualidade do fluxo fim-a-fim. A forma de envio periódico de informações sobre a qualidade da transmissão também é interessante, se apresentando como uma solução mais simples do que, por exemplo, priorizar nos nós intermediários o envio desses relatórios. Porém, podem existir outras formas de manter a origem informada sobre as condições da transmissão ao invés de aumentar o *overhead* da rede. Como foi visto em outros trabalhos, o envio de mensagens *in-band* minimiza a sobrecarga na rede. No caso de uma rede *Ad Hoc*, por exemplo, essas informações poderiam ser inseridas nas mensagens de estabelecimento e manutenção de rota, mensagens estas que são inevitáveis.

Outra característica importante da arquitetura proposta é a de trabalhar com dispositivos *of-the-shelf*. O mecanismo de adaptação irá utilizar o que o sistema em que ele está executando pode oferecer, como os *codecs* disponíveis e a taxa de amostragem da mídia suportada pelo dispositivo.

Os testes realizados para avaliar o desempenho da solução foram pouco representativos, pois não apresentaram diferentes situações em redes sem fio. Nos testes das redes *Ad Hoc*, por exemplo, não foi considerada a mobilidade dos nós, que gera mudança na topologia e pode causar perda de pacotes. Por isso, os resultados da avaliação podem não representar o desempenho verdadeiro do sistema desenvolvido em situações reais.

2.5 Comparação Entre as Soluções

Cada *framework* e protocolo apresentado nesta seção possui características específicas que agregam à solução vantagens e desvantagens. Nesta seção será feita uma comparação entre as soluções com e sem reserva de recursos, apontando seus pontos positivos e negativos.

Um ponto importante para ser levado em consideração é que, na maioria dos trabalhos apresentados, a qualidade de serviço fornecida ao fluxo está diretamente ligada com a disponibilidade de largura de banda em cada nó. Isso mostra que o requisito largura de

banda para fornecimento de QoS é essencial quando se trata de transferência multimídia em redes sem fio, sendo um requisito importante e indispensável.

Os mecanismos de reserva de recursos podem fornecer garantias mais precisas de qualidade para as aplicações, quando aplicados de forma flexível à dinâmica das redes *Ad Hoc*. No entanto, os trabalhos apresentados que utilizam reserva não garantem a banda na camada MAC. No nível do acesso ao meio, as redes *Ad Hoc* continuam fornecendo serviço *best-effort*. Na realidade, o que as soluções propostas fazem é garantir que os fluxos que realizaram as reservas tenham prioridade na transmissão. Estas soluções são mais simples, permitindo que dispositivos disponíveis no mercado sejam utilizados sem realizar alterações nas suas configurações físicas.

Os mecanismos de reserva podem gerar uma sub-utilização dos recursos da rede. Como citado nos trabalhos INSIGNIA e ASAP, se o tempo de validade da reserva não for adequado para o tipo de fluxo que está sendo transportado, o excesso dessas reservas pode fazer com que os recursos fiquem escassos, degradando a qualidade de outros fluxos. Neste caso, outras sessões seriam impossibilitadas de enviar seus dados com as garantias de reserva, já que não poderiam estar utilizando os recursos, que estariam reservados, porém ociosos. Além disso, é necessário que haja em cada nó da rede processamento extra para controlar a tabela de reservas. Quanto maior for o número de fluxos trafegando na rede, maior será o processamento.

As soluções que trabalham sem a reserva de recursos, para fornecer métricas mais precisas para as técnicas de adaptação, utilizam mecanismos de estimação que geram processamentos mais pesados do que o controle das tabelas de reserva. Processos de estimação de largura de banda e medição de atrasos e perda de pacotes geram um *overhead* de execução, podendo aumentar o gasto de energia nos dispositivos móveis ou portáteis.

Por outro lado, os sistemas sem reserva não prendem os recursos e são mais flexíveis às mudanças dos ambientes *wireless*. Apesar de não fornecerem garantias de qualidade aos fluxos, através dos processos de estimação de banda e medições dos requisitos de QoS, a utilização dos recursos e a situação da transmissão é apresentada com mais realismo. Porém, se essas medições não forem realizadas no momento em que o fluxo está sendo transmitido, pode ser que as condições favoráveis apresentadas pela estimação de banda, por exemplo, não mais existam.

A tabela 1 contém um resumo dos principais mecanismos utilizados por cada trabalho apresentado nesta seção. A tabela 2 apresenta uma comparação entre as técnicas de fornecimento de QoS com e sem reserva de recurso, apontando suas vantagens e desvantagens.

Soluções	Características
INSIGNIA	Mecanismo com reserva de recurso Sistema de sinalização *in-band* Gerenciamento de recursos *soft-state*
ASAP	Mecanismo com reserva de recurso Sistema de sinalização *in-band* Gerenciamento de recursos *Soft/Hard*
AODV Adaptável	Mecanismo de determinação de banda Adaptação do nível do roteamento Mudança de posição para melhor QoS
Lógica de Adaptação	Estimação de banda Medição de perda de pacotes Adaptação no nível da aplicação

Tabela 2.1: Principais características dos trabalhos apresentados.

Mecanismo	Vantagens	Desvantagens
Com Reserva de Recursos	Garantias mais precisas de qualidade para aplicações	Pode gerar uma sub-utilização dos recursos da rede
Sem Reserva de Recursos	Não prende os recursos; é mais flexível; estimação de banda e medições de QoS apresentam a situação da rede com mais realismo	O processo de estimação de banda gera maior processamento e mais tráfego (transmissões), podendo aumentar o gasto de energia dos dispositivos

Tabela 2.2: Comparação entre as técnicas estudadas.

2.6 Análise dos Trabalhos Estudados

Fazendo uma análise dos resultados dos trabalhos estudados, pode-se chegar a algumas conclusões sobre qual seria a solução adequada para fornecer QoS em transmissões de fluxo contínuo. As análises que serão apresentadas nessa seção têm o objetivo de serem demonstrativas, não podendo ser levadas em consideração como resultados definitivos, pois os cenários e os testes realizados nos trabalhos não possuem as mesmas condições. Porém, é importante observar o resultado dos principais requisitos de QoS, como perda de pacotes, atraso fim-a-fim e *jitter*. Esses pontos são essenciais para fluxos multimídia (vídeo e áudio), onde não é preciso apenas que o pacote seja entregue no destino, mas que chegue a tempo de ser exibido.

Com base nesses aspectos, observou-se que, por exemplo, no INSIGNIA, a porcentagem de perda de pacotes foi reduzida (em torno de 0% a 1%) quando a taxa de transmissão era menor que 10 pacotes por segundo e a velocidade dos nós era baixa ou moderada (va-

riando entre 3km/h e 18km/h). O atraso fim-a-fim dos pacotes foi igualmente diminuto, apresentando seu valor máximo de 40 ms numa velocidade alta, com os nós a 72 km/h. Os gráficos abaixo foram tirados do artigo [Lee et al. 2000] que apresenta o *framework* INSIGNIA.

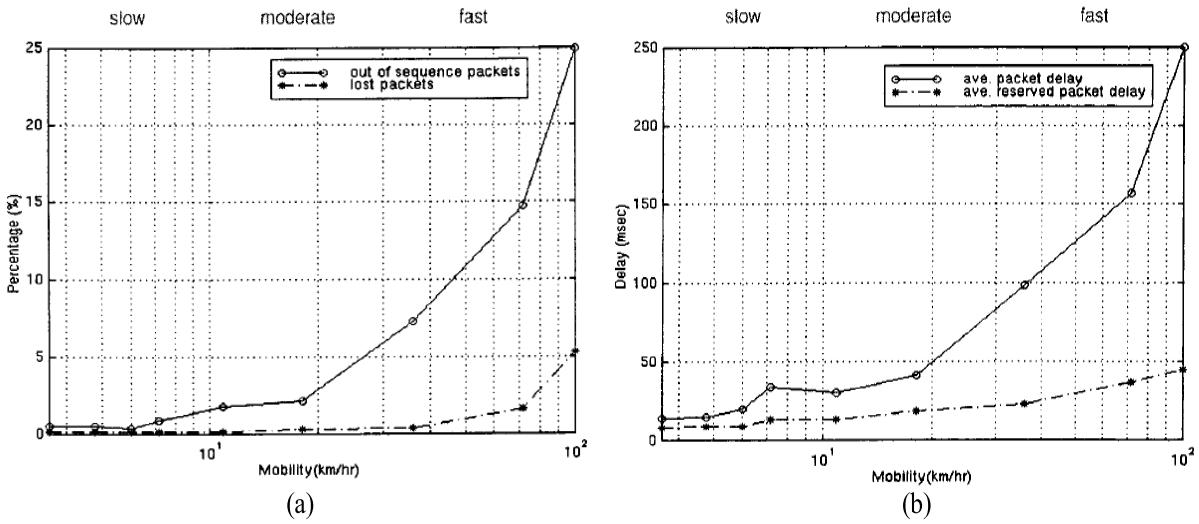
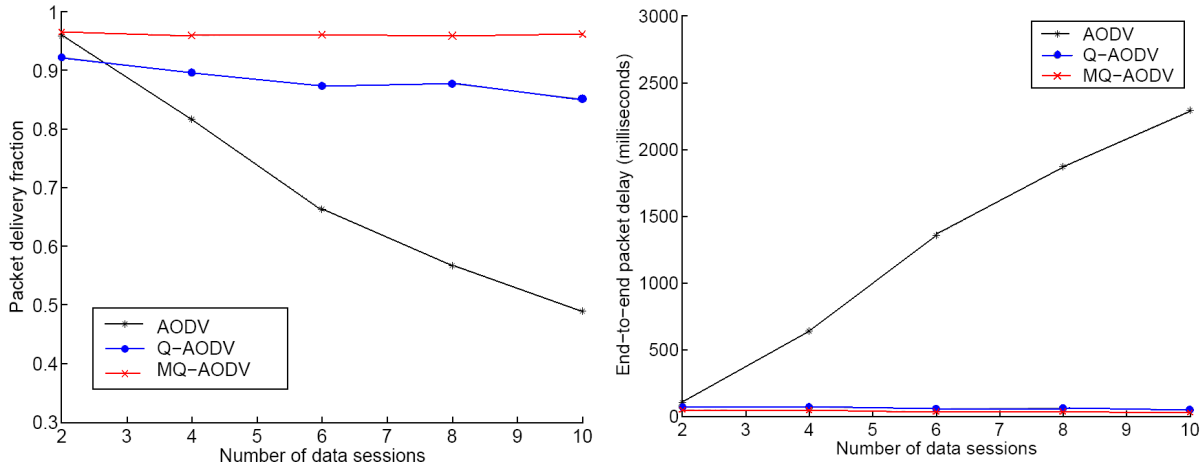


Figura 2.7: Gráficos INSIGNIA - (a) Porcentagem de perda de pacotes; (b) Atraso fim-a-fim.

Nas soluções sem reserva de recurso, como no AODV Adaptável, a porcentagem de perda de pacotes variou entre 5% e 10%, sendo que a velocidade de todos os nós não era a mesma (entre 5m/s e 20 m/s). Já o atraso de pacotes sofreu uma leve variação, apresentando seu resultado em torno de 60ms. Os gráficos da figura 2.8 foram extraídos do artigo [Sanzgiri e Belding-Royer 2004], que propõe uma alteração no protocolo de roteamento AODV, gerando dois novos protocolos (Q-AODV e MQ-AODV). O gráfico 2.8(a) apresenta a fração de pacotes entregues, que é a razão entre os pacotes que foram enviados pela origem e os recebidos pelo destino. Esses valores estão variando entre 90% e 95% nos protocolos Q-AODV e MQ-AODV. O segundo gráfico apresenta o atraso fim-a-fim. Infelizmente, nenhum dos trabalhos apresentou resultados em relação ao *jitter*.

Os cenários dos testes das soluções exemplificas são diferentes. O cenário do INSIGNIA consistiu em 19 nós com velocidades variando entre 0 e 100km/h, utilizando 802.11 a 2Mbps com 50m de alcance e executando o protocolo TORA para roteamento. Já o AODV Adaptável possui 50 nós espalhados aleatoriamente numa área de 1000m X 1000m, utilizando também o 802.11 mas com um alcance de 250m.

Mesmo com cenários distintos, pode-se perceber que os trabalhos que desenvolveram



(a) (b)

Figura 2.8: Gráficos AODV Adaptável - (a) Fração de pacotes entregues; (b) Atraso fim-a-fim.

soluções com reserva de recursos apresentaram melhores resultados do que as soluções que optaram por não fazer reserva. Apesar dos trabalhos que realizaram reserva de largura de banda não a fizeram no nível de acesso ao meio, esta solução oferece prioridade de transmissão para o fluxo multimídia, o que garante um melhor desempenho na tarefa de suportar QoS.

2.7 Conclusão

Os trabalhos apresentados são apenas alguns exemplos de soluções de fornecimento de Qualidade de Serviço para fluxos contínuos em redes *Ad Hoc*. Outras pesquisas na área apresentaram soluções interessantes. O projeto SWAN [Ahn et al. 2002] propõe um mecanismo de *feedback* para fornecer serviços flexíveis para fluxos de tempo real em redes *Ad Hoc*. Ele utiliza controle da taxa para tráfegos *best-effort* UDP e TCP e controle de admissão no emissor para tráfegos de tempo real UDP. As informações de *feedback* para a origem não dependem de um sistema de sinalização, mas são geradas pela análise das condições da rede. Através da medição de atrasos na camada MAC, o emissor configura automaticamente o mecanismo de controle de taxa de transmissão. A configuração do controle de admissão é feita também pela origem através de uma avaliação da largura de banda ainda disponível para fluxos de tempo real, de acordo com as informações geradas pela medição da taxa destes fluxos que trafegam pelos seus vizinhos.

Outras soluções também utilizam mecanismos de reserva de recurso para garantir

QoS para fluxos contínuos em redes *Ad Hoc*. Em [Mirhakkak et al. 2000] é apresentado um mecanismo em que as reservas são representadas através de limites, e as aplicações devem se adaptar ao nível de QoS disponível pela rede. Os autores propõem um novo protocolo, chamado *dynamic RSVP*, uma extensão do *Resource Reservation Setup Protocol* [Clausen e Jacquet 1997].

O problema de fornecer Qualidade de Serviço para redes *Ad Hoc* é uma fonte de estudos inesgotável e escolhida por muito autores. Os artigos [Guimarães et al. 2004], [Zhu et al. 2004] e [Mohapatra et al. 2003] apresentam uma análise sobre os trabalhos existentes nesta área. Em todos eles, os mecanismos de adaptação de requisitos no fluxo, reserva de recursos e controle de admissão são citados como as técnicas mais utilizadas e que apresentam melhores resultados.

Ainda existem outros tipos de soluções para o suporte a fluxos de tempo real em redes *Ad Hoc*. Alguns trabalhos na área propõem que mecanismos de fornecimento de QoS sejam inseridos na camada MAC para minimizar os problemas nos enlace sem fio, fornecendo qualidade para os fluxos de tempo real. O esquema de contenção *Black Burst* descrito em [Sobrinho e Krishnakumar 1999] e o sistema MACA/PR (*Multihop Access Collision Avoidance with Piggyback Reservation*) [Lin e Gerla 1997] apresentam soluções que tem como objetivo evitar a colisão de pacotes. O primeiro trabalho também resolve o problema de *starvation* de pacotes, escalonando os fluxos de uma mesma classe de serviço de maneira distribuída. Já o MACA/PR garante a reserva de banda para tráfegos de tempo real ao longo do caminho. O padrão IEEE 802.11e [Xiao 2004] também fornece qualidade de serviço e suporte para fluxos multimídia na camada MAC. Porém, este tipo de solução não é o foco do estudo deste trabalho. Por esta razão é que não tratamos dos detalhes dessas soluções.

Os protocolos de roteamento também podem auxiliar no suporte a QoS, reagindo rapidamente às mudanças de topologia. Porém, se estes também estiverem cientes dos requisitos de QoS especificados pela aplicação, a rota descoberta poderá ser o caminho mais apropriado, com os recursos com maior disponibilidade. O protocolo de roteamento OLSR [Clausen e Jacquet 2003] [Jacquet et al. 2001] faz parte do grupo de protocolos pró-ativos, que mantêm suas tabelas de rotas sempre atualizadas e fornecem aos nós uma visão da topologia da rede. Uma de suas características é a de fornecer rotas ótimas em relação ao número de saltos, podendo auxiliar o trabalho de camadas superiores no fornecimento de QoS. Além disso, o OLSR possui um parâmetro que apresenta a disposição de um nó em encaminhar e transmitir tráfego de outros nós. Este parâmetro pode ser definido manualmente, tendo seu valor inalterado por todo o tempo em que o nó estiver

ativo, ou cada nó poderá ajustá-lo dinamicamente de acordo com suas condições. Um possível algoritmo para este ajuste seria de acordo com a energia disponível no nó. Caso o dispositivo esteja conectado a uma fonte inesgotável de alimentação de energia, ele poderá se dispor a transmitir qualquer fluxo para qualquer nó, pois isso não afetará o seu consumo. Porém, se a fonte for limitada e a energia do nó estiver muito baixa, ele poderá informar que não deseja encaminhar nenhum pacote, para prolongar seu tempo de vida.

Essas características do OLSR permitem que se estabeleça uma diferenciação entre os nós no momento da escolha de uma rota ótima, e por conseqüência, na qualidade dos enlaces em que o tráfego irá ser transmitido. Isso poderá ser importante para o *framework* de suporte a QoS que será proposto no próximo capítulo.

Um ponto importante nos trabalhos estudados é a comunicação e a troca de informações, principalmente entre os nós origem e destino, sobre as condições da transmissão e o andamento dos requisitos de QoS. Os mecanismos de sinalização e monitoramento do fluxo permitem que o emissor dos dados possa controlar o envio destes.

Através dos exemplos dos trabalhos descritos, pode-se perceber que os dois grupos de soluções apresentam vantagens e desvantagens. Fornecer QoS em redes *Ad Hoc* e *Mesh* não é uma tarefa trivial e, independentemente da solução, há sempre um preço a ser pago. Contudo, é sempre importante ressaltar que a escolha da técnica a ser implantada também depende da aplicação para qual esta solução está sendo proposta. De acordo com os requisitos especificados é que se pode medir o preço que é necessário pagar para garantir QoS.

É possível observar também que a habilidade de adaptação às constantes alterações dos ambientes sem fio é o mecanismo essencial para soluções em redes *Ad Hoc* e *Mesh*. Esta é a chave para que os sistemas se tornem flexíveis e possam se adequar aos requisitos de QoS especificados pela aplicação.

Capítulo 3

Proposta para o Framework de Suporte a QoS

3.1 Introdução

Como foi visto no capítulo anterior, existem muitos pesquisadores que trabalham em redes sem fio e procuram soluções para os diferentes problemas gerados pela própria natureza da comunicação a rádio. As redes sem fio podem apresentar instabilidade nos seus enlaces, provocadas por interferências, obstáculos, ou até mesmo pela possibilidade de movimentação dos nós da rede. No caso de uma rede *Mesh*, o seu *backbone* é formado por roteadores *wireless*, que se mantêm fixos, conectados a uma fonte de alimentação. Assim, a mobilidade dos participantes principais da rede *Mesh* não existe, o que minimiza a quebra de enlaces por conta da movimentação dos nós. Porém, o uso dessa malha por diversos dispositivos móveis pode causar congestionamento e colisões, gerando altas taxas de perda de pacote e *jitter*.

Além da necessidade de ultrapassar esses problemas intrínsecos, é preciso também analisar como as características de uma rede sem fio podem afetar o desempenho de certas aplicações. O caso das transmissões de fluxos contínuos, como áudio e vídeo, é o mais citado na literatura. As desconexões, as colisões, os congestionamentos e a má qualidade dos enlaces podem prejudicar a transmissão de dados multimídia e afetar a qualidade da informação que chega ao receptor. No entanto, também nesse tipo de transmissão encontramos diferenças de requisitos entre as aplicações. Uma reunião em videoconferência pode tolerar uma perda maior dos pacotes de vídeo do que, por exemplo, uma aplicação de telemedicina, na qual mais de uma equipe médica pode estar participando de uma cirurgia a distância.

Dessa forma, pode-se perceber que o mais recomendável é procurar uma solução que se comunique com a aplicação para tomar conhecimento dos requisitos especificados. Em particular, no caso de se fornecer QoS a fluxos contínuos em uma rede sem fio, não é possível obter resultados satisfatórios sem que a camada de serviços troque informações sobre os requisitos de qualidade com a aplicação. Essa interação faz com que a qualidade a ser fornecida para a transmissão se aproxime da desejada pela aplicação, sem que ela mesma tome conhecimento dos procedimentos executados num nível mais baixo.

Com essa idéia em mente, este capítulo tem por objetivo apresentar uma proposta para uma camada de suporte à Qualidade de Serviço para transmissão de fluxos contínuos em redes *Mesh*. Primeiramente, com base na análise dos trabalhos relacionados descritos no capítulo anterior, serão feitas algumas considerações que servirão de apoio para a proposta deste *framework*. Após a descrição do ambiente *Mesh* em que esta proposta será implantada, a estrutura do *framework* será apresentada, e serão apontadas justificativas para os mecanismos escolhidos.

3.2 Considerações

A seção 2.6 apresentou uma análise comparativa dos trabalhos estudados. De acordo com os resultados desses artigos, é possível fazer algumas considerações sobre qual seria a melhor solução para dar suporte à Qualidade de Serviço em redes *Ad Hoc*.

Um ponto comum aos trabalhos que desenvolvem soluções para transmissão em redes sem fio é a preocupação com a largura de banda disponível no canal. Como já foi visto, os participantes de uma rede sem fio disputam entre si o acesso ao meio. Assim, dependendo da distância entre eles, a transmissão de um nó pode atrapalhar a transmissão/recepção de outro. Por isso, a largura de banda disponível para transmissão é um requisito tão importante, já que pode ter uma variação grande ao longo do tempo. Se for possível conhecer a capacidade do canal no momento do envio dos dados, poderia-se evitar o congestionamento do meio e também as altas taxas de perda de pacotes.

Os trabalhos apresentados no capítulo anterior mostram que a banda pode ser utilizada como requisito de QoS de duas formas: na forma de medições ou estimativas dos recursos disponíveis que mostrem à aplicação como está a disputa pelo meio, ou na forma de reserva de recursos, que garantem às camadas superiores uma fatia da banda disponível. O trabalho denominado INSIGNIA se enquadra no segundo caso. Porém, ele não implementa a reserva de recursos na camada MAC, mas cria filas de prioridades nos nós

roteadores, dando alta preferência aos fluxos multimídia. Esse tipo de reserva é chamada de *best effort* ("melhor esforço").

Apesar da largura de banda ser um requisito importante de QoS, outras características da transmissão podem ser muito úteis para apresentar as condições da rede. Em fluxos de tempo real, a perda dos pacotes e a variação do tempo em que eles chegam ao cliente podem prejudicar a qualidade de reprodução da mídia. Os valores aceitáveis para perda de pacotes e *jitter* são dependentes da aplicação e de outros fatores, como tipo de mídia e codificação. Se a aplicação é de videoconferência ou vídeo sobre demanda, esses requisitos são menos exigentes, pois a mensagem pode ser compreendida mesmo que haja algum atraso ou perda de pacotes maior. Porém, no caso das aplicações de telemedicina, em que muitas vezes se estão transmitindo imagens de cirurgias ou exames, uma pequena perda de pacotes ou atraso pode fazer uma grande diferença.

Com isso, pode-se perceber que não apenas a largura de banda disponível na rede traduz as condições do meio, mas outros requisitos, como perda de pacotes e *jitter*, também representam a qualidade de transmissão do fluxo. O monitoramento desses requisitos pode detectar, por exemplo, congestionamentos e colisões, e possibilitar o acionamento de algum mecanismo de reparo ou adaptação da transmissão.

Um ponto que deve ser levado em consideração sobre os trabalhos estudados é que esses foram desenvolvidos para redes *Ad Hoc*. Ainda que o *framework* proposto seja implementado em uma rede *Mesh*, que utilizará um protocolo *Ad Hoc* para roteamento das mensagens, o objetivo e o comportamento desses dois grupos de redes são diferentes.

Uma rede *Ad Hoc* tem como uma de suas características principais o fato de que seus nós trabalham de forma cooperativa no roteamento das mensagens. Dessa forma, fica mais fácil desenvolver uma solução que utilize essa cooperação dos nós para que, além de rotear as mensagens, eles trabalhem em conjunto para fornecer qualidade na transmissão dos dados.

Já uma rede *Mesh* tem como objetivo fornecer um *backbone* sem fio de acesso comunitário banda larga, interligando diferentes dispositivos. Assim, os nós móveis que se conectarem à rede *Mesh* para usufruir dessa infra-estrutura podem não estar interessados em cooperar com outros nós para encaminhar pacotes e dar suporte à qualidade de serviço. Além do mais, muitos dispositivos móveis possuem recursos limitados, como capacidade de processamento e capacidade energética, não tendo disponibilidade para reservá-los para outros nós, ao preço de degradarem a qualidade de seus fluxos.

Para propor um *framework* de suporte a QoS, é preciso também levar em consideração a estrutura da rede *Mesh* em que será instalado. O projeto *ReMesh* está desenvolvendo uma rede na qual os roteadores serão alterados de forma a satisfazer as suas necessidades, como, por exemplo, a mudança do sistema operacional para um que apresente mais recursos e outras configurações. Essa característica é diferente da rede *Mesh* proposta pela Microsoft, por exemplo, que pressupõe que a camada *Mesh* da rede, a MCL, será instalada no dispositivo do cliente, sem alteração das configurações do *hardware*. Por essa razão, a MCL apenas faz a estimação de métricas de qualidade do enlace, em vez de fazer reserva de recursos, por exemplo.

A próxima seção apresentará a arquitetura da rede *Mesh* sobre a qual o *framework* será implantado. Com base nas considerações apresentadas, a seção 3.4 detalhará a proposta para o *framework* de suporte a QoS em redes *Mesh*.

3.3 Ambiente *ReMesh*

Esse projeto instalou inicialmente uma rede *Mesh* dentro de um dos prédios do *campus*, criando uma rede interna para testes. Em uma segunda fase, instalou roteadores no topo dos edifícios onde residem alguns alunos e funcionários, nas redondezas da Universidade, além de um roteador que trabalha como *gateway* no topo de um dos prédios do *campus*. O *framework* de QoS foi instalado e testado apenas na rede interna, pois o acesso aos roteadores é mais fácil, e esse é um ambiente um pouco mais controlado.

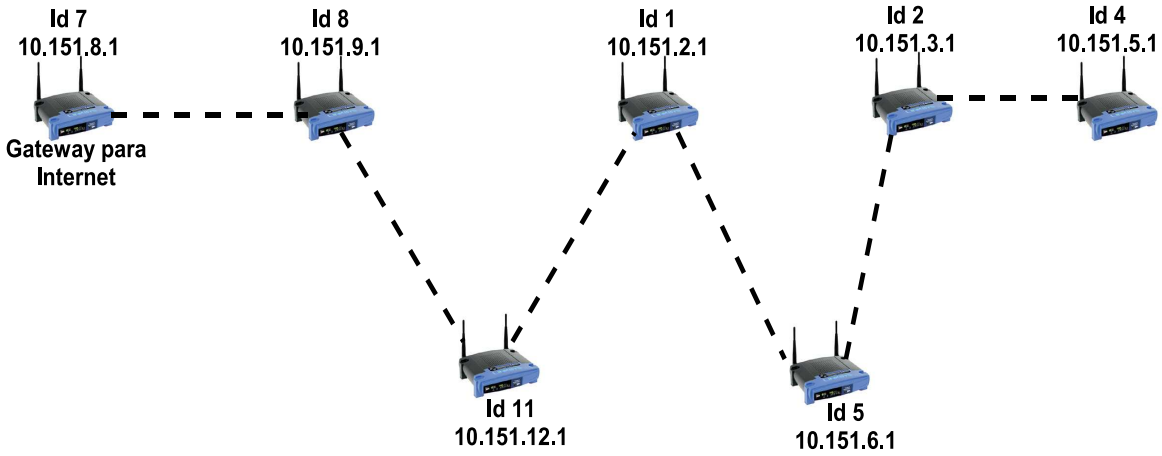


Figura 3.1: Topologia da rede *Mesh* interna.

A topologia da rede interna pode ser vista na figura 3.1. A rede consta de 7 roteadores, dos quais 6 se encontram espalhados no terceiro andar de um dos prédios da Escola de

Engenharia da UFF; o roteador com identificador 4 (verificar na figura 3.1) se encontra no quarto andar do mesmo prédio. Os endereços IP que aparecem na figura são os endereços da interface sem fio de cada nó. O roteador de identificador 7 tem a interface de rede local ligada à rede interna da UFF, sendo assim, o *gateway* para a Internet.

3.4 Proposta para o *Framework*

Com base nos argumentos apresentados e nas avaliações dos trabalhos estudados, a proposta para o *framework* de suporte a QoS é que ele seja desenvolvido de forma híbrida: serão utilizados os mecanismos de reserva de recursos e a medição de requisitos. Os pontos positivos de cada mecanismo serão explorados para que sejam melhor aproveitados de acordo com as características da estrutura da rede *Mesh* implementada para esse projeto. O *framework* de suporte a QoS pode ser então estruturado da seguinte forma:

1. Os **requisitos da aplicação** permitirão que haja uma participação do usuário no processo de suporte a QoS, especificando os valores desejáveis para os requisitos de qualidade;

2. Para auxiliar na reserva de banda, no processo de monitoramento e no acionamento da adaptação, o sistema de **sinalização *in-band*** enviará, junto com os pacotes de dados, informações de controle do *framework*;

3. O processo de **reserva de recursos** fornecerá garantias de largura de banda do tipo *best-effort* para a transferência de fluxos multimídia;

4. O **monitoramento do fluxo** permitirá que os requisitos de QoS sejam avaliados durante a transmissão, enviando os resultados para que o sistema de adaptação faça as comparações necessárias;

5. O **sistema de adaptação** irá comparar os resultados obtidos pelo monitoramento com os valores especificados pela aplicação e será acionado quando as condições da rede estiverem atrapalhando o desempenho da transmissão dos dados.

Para garantir qualidade aos fluxos multimídia no *backbone* da rede *Mesh*, foi implementado um mecanismo de reserva de recursos em cada roteador. Os roteadores *wireless* têm uma grande disponibilidade do recurso energia, pois estão ligados a uma fonte de alimentação permanente. Já o recurso largura de banda tem que ser compartilhado entre

os diversos fluxos que passam por ele. Como observado nos trabalhos estudados, que utilizavam a solução de reserva de banda, o atraso e a perda de pacotes apresentam valores reduzidos em velocidades baixas, o que pode ser vantajoso para os roteadores que se mantêm fixos. É importante ressaltar que, de forma semelhante ao feito no trabalho INSIGNIA, a reserva de banda não será feita no nível da camada MAC, mas será implementada através de filas de prioridades nos roteadores.

Além da reserva de recursos nos roteadores sem fio, todo nó móvel cliente que estiver recebendo os pacotes multimídia realizará periodicamente medições de perda de pacotes e de *jitter* da transmissão. Esses dois requisitos de QoS trabalharão como parâmetros para o acionamento de um mecanismo de adaptação. Essas medições servirão como monitoramento das condições da rede.

Para a implantação de um sistema de suporte à qualidade de serviço em redes sem fio, é essencial que haja algum mecanismo de adaptação do fluxo às condições do meio. O mecanismo de reserva permitirá que os dados multimídia tenham prioridade de banda nos roteadores, e o processo de monitoramento dos requisitos avaliará o andamento da transmissão, que reflete as condições do ambiente. Caso esse monitoramento detecte que a qualidade da transmissão está abaixo da desejada pela aplicação, o *framework* deve acionar o processo de adaptação. Da mesma forma, se o monitoramento perceber que um fluxo antes degradado apresenta uma melhora de qualidade, a adaptação é acionada novamente.

Para auxiliar os três mecanismos citados anteriormente, o *framework* possui um sistema de sinalização *in-band* e necessita que a aplicação especifique os valores desejáveis dos requisitos de qualidade. A sinalização *in-band* permite que os nós do caminho possam fazer a priorização dos fluxos de vídeo e possibilita também a troca de informação entre o cliente e o servidor. A especificação de requisitos auxilia os nós das extremidades a avaliar a transmissão e a permitir que o *framework* forneça a qualidade que a aplicação deseja.

Em resumo, o *framework* contém uma combinação de mecanismos para dar suporte a QoS: (i) especificação de requisitos da aplicação; (ii) sinalização *in-band*; (iii) reserva de recursos; (iv) monitoramento da qualidade do fluxo; e (v) sistema de adaptação do fluxo às condições da rede. As seções seguintes descreverão com mais detalhes cada mecanismo do sistema, explicando os algoritmos propostos, bem como a comunicação entre o *framework* e a aplicação.

3.4.1 Requisitos da Aplicação

Como já foi mencionado na seção 3.2, os valores aceitáveis para os requisitos de QoS utilizados diferem de acordo com a aplicação que está usufruindo dos serviços oferecidos. No caso do *framework* que está sendo proposto, os requisitos de QoS a serem utilizados já estão definidos, porém os valores deverão ser determinados pela aplicação.

Os requisitos que deverão ser especificados pela aplicação são:

- Limite mínimo e máximo suportáveis de ***jitter***, em milisegundos;
- Limite mínimo e máximo suportáveis de **perda de pacotes**, em porcentagem;
- **Taxa de envio de pacotes** mínima e máxima aceitáveis, em *frames* por segundo;

Esses requisitos auxiliarão os mecanismos de monitoramento da rede e de adaptação. Esta seção explicará como cada requisito será usado em cada um desses mecanismos.

O processo de monitoramento da rede irá utilizar os limites mínimo e máximo de *jitter* e de perda de pacotes como parâmetros para o acionamento da adaptação, para avaliar se a qualidade da transmissão está deteriorada, ou não. Os limites máximos dos dois parâmetros definem quando a qualidade da transmissão está ruim e quando o sistema de adaptação deve ser acionado. Quando a monitoração registra que esses parâmetros apresentam valores abaixo dos limites mínimos, então percebe-se que a qualidade da transmissão melhorou, e o processo de adaptação pode ser revertido. Mais detalhes sobre o monitoramento e o sistema de adaptação serão apresentados nas seções seguintes.

O sistema de adaptação também precisa de informações fornecidas pela aplicação. A adaptação que o *framework* realiza no fluxo é a alteração da taxa de transmissão dos dados no emissor. Essa taxa é diminuída ou aumentada de acordo com as condições da rede, avaliadas pelo processo de monitoramento. Por essa razão é que se faz necessário especificar as taxas de envio mínima e máxima suportadas pela aplicação. Mais detalhes sobre o processo de adaptação serão apresentados na seção 3.4.5.

3.4.2 Sinalização *In-band*

O sistema de sinalização *in-band* envia informações de controle do *framework* junto com os pacotes de dados. Essas informações serão importantes para os três mecanismos utilizados: reserva de recursos, monitoramento do fluxo e sistema de adaptação. As principais informações de controle que formam o cabeçalho de uma mensagem do *framework* são:

- Nível de adaptação;
- Tipo de dados do pacote;
- Informação quanto ao pedido, ou não, de suporte à QoS ao fluxo;
- Número de identificação do pacote.

O nível de adaptação determina se este é um fluxo que já perdeu qualidade com uma deterioração da rede e precisou ser adaptado. O significado desse nível será melhor explicado na seção que apresenta o sistema de adaptação. O tipo de dados informa se o presente pacote é de um fluxo de vídeo, se é um pacote de controle do *framework* ou de um outro fluxo não prioritário. Também é preciso saber se o pacote pertence a um fluxo que requisitou suporte ou não.

Os três primeiros itens auxiliam o processo de reserva de largura de banda. Como será explicado na próxima seção, cada roteador possui diferentes classes, cada qual com uma fatia de banda reservada para ela e uma política de tratamento de dados. Cada classe representa um tipo de fluxo, o nível de adaptação em que esse fluxo se encontra e se ele possui prioridade ou não. Ao passar por cada roteador, o sistema de reserva classifica os pacotes e os envia para sua classe correspondente. Cada classe encaminhará os pacotes adiante de acordo com a largura de banda reservada para ela e com a sua prioridade.

O número de identificação do pacote permitirá que o processo de monitoramento dos fluxos faça a contagem da perda de pacotes e perceba quando algum pacote chegou ao cliente fora de ordem.

3.4.3 Reserva de Recursos

O sistema de reserva de recursos tem o objetivo de permitir que o fluxo multimídia encontre um caminho com condições favoráveis no *backbone*, para que sua transmissão não tenha problemas de desempenho. O mecanismo aqui proposto escolheu o recurso largura de banda para que seja reservado aos fluxos que irão trafegar pela rede *Mesh*. Porém, é muito importante ressaltar que, diferentemente de outras propostas [Xiao 2004], o mecanismo de reserva de banda não atuará na camada MAC. A proposta escolhida será detalhada nesta seção.

O sistema de reserva de recursos trabalhará da seguinte maneira: serão criadas diferentes classes para cada tipo de tráfego e será disponibilizada para cada classe uma parte

da capacidade nominal de largura de banda do roteador. Quando um pacote chegar ao nó sem fio, ele será encaminhado para a classe em que se encaixa. Não é possível saber quantos fluxos estão passando por cada classe, mas apenas a quantidade de pacotes.

Caso a quantidade de dados que está trafegando em uma classe exceda a largura de banda reservada para ela, tal classe poderá pegar banda emprestada de outras que não estão usando toda a fatia reservada para si. Esse limite de banda que cada classe pode pegar emprestado também deve ser determinado. Uma classe também pode ter o seu valor de banda adicional igual a zero. Isso significa que tal classe não poderá pegar banda emprestada com outras, caso necessite. Essa característica é indicada ao registrar que o valor de banda reservada é igual ao seu limite, como mostrado na figura 3.2, na classe Outros.

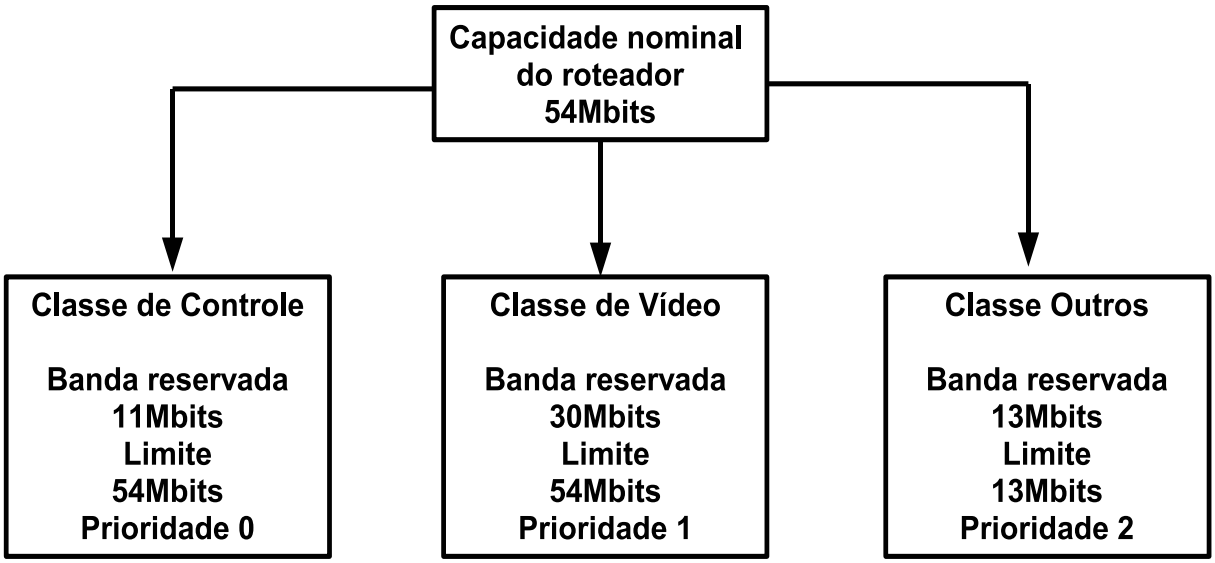


Figura 3.2: Estrutura de classes do mecanismo de reserva de recursos.

Cada tipo de tráfego também terá uma prioridade nos roteadores. É importante ressaltar mais uma vez que a reserva não está sendo feito da camada MAC, mas sim na forma de priorização de tráfego. Por isso, cada tipo de dados possuirá uma prioridade de transmissão e de reserva de banda. A prioridade é dada às classes, e aquelas que tiverem prioridades mais altas poderão pegar banda emprestada primeiro, caso necessitem. As classes que tiverem prioridade mais baixa, caso precisem pegar banda emprestada, só conseguirão fazê-lo se sobrar recurso disponível.

Para efeito de teste, as classes que foram definidas para o *framework* são: fluxo de controle, que consiste nos pacotes de controle que gerenciarão a transmissão dos dados; fluxo de vídeo, que são os dados propriamente ditos; e outros tipos de fluxo não prioritários.

A classe de controle terá a prioridade mais alta, seguida pelos pacotes de vídeo e depois pelos outros fluxos. Esta última não terá direito de pegar banda emprestada das outras classes, podendo apenas utilizar a fatia que lhe foi reservada. A figura 3.2 apresenta um esquema de como é feita essa divisão da largura de banda em cada roteador.

Para viabilizar a implementação do sistema de reserva de recursos, será utilizada uma ferramenta de controle de tráfego e gerenciamento de banda disponível no *firmware* usado nos roteadores. Essa ferramenta possibilita a criação de classes para diferentes tipos de tráfegos e a alocação de uma fatia de largura de banda para cada classe. Também é possível determinar diferentes políticas de envio dos pacotes para cada classe. Mais detalhes sobre essa ferramenta e o *firmware* utilizados serão apresentados no capítulo 5.

É importante ressaltar novamente que a reserva de largura de banda em cada roteador será estática. Cada classe terá uma fatia de banda reservada, que não será alterada ao longo da transmissão. Esse formato de sistema foi proposto para simplificar o controle de admissão e poder utilizar as ferramentas disponíveis nos roteadores, sem alterar suas configurações ou inserir algum novo programa de monitoramento de banda. Como já foi dito, no capítulo 5, no qual se descrevem os detalhes da implementação, serão apresentados o sistema operacional escolhido para executar nos roteadores sem fio e as ferramentas deste sistema utilizadas para a implantação do mecanismo de reserva de recursos.

3.4.4 Monitoramento do Fluxo

O mecanismo de monitoração da transmissão será responsável por avaliar as condições da rede através de medições dos requisitos de QoS no cliente. Os valores obtidos por essa monitoração serão passados para o sistema de adaptação, que poderá detectar problemas no ambiente ou perceber quando eles já foram sanados. Esses problemas podem ser, por exemplo, um congestionamento nos enlaces, uma instabilidade das conexões por causa da mobilidade do nó cliente ou uma alta disputa pelo meio e espera nas filas dos roteadores, o que causa colisões e descartes de pacotes.

Durante a transmissão da mídia, o cliente irá periodicamente monitorar o fluxo para analisar as taxas de perda de pacote e *jitter*. Essa periodicidade pode ser por intervalo de tempo predeterminado ou por número definido de pacotes recebidos. Independentemente de como será determinado esse intervalo, a cada tempo de monitoração o nó móvel medirá qual foi a porcentagem de perda e a variação do atraso que os pacotes recebidos entre a última monitoração e o presente momento sofreram para chegar até o cliente móvel. A partir desses dados, o sistema de adaptação no cliente analisará quais são as condições da

transmissão para poder tomar alguma providência, caso seja necessário.

A medição da perda de pacotes no intervalo corrente será feita com base no identificador seqüencial de cada pacote. O *jitter* da transmissão, que é a variação do atraso entre os pacotes, será medido por meio da observação da variação do tempo de chegada entre os pacotes no cliente.

O processo de análise das condições do fluxo, após medir a perda de pacotes e o *jitter* correntes, aplica pesos à medição corrente e às medições passadas, fazendo uma amortização exponencial. Dessa forma, as equações que refletem as variáveis perda de pacotes e *jitter* são:

$$P_i = P_{i-1}\alpha + P_c(1-\alpha)$$

$$J_i = J_{i-1}\alpha + J_c(1-\alpha)$$

em que P_c e J_c são as medições correntes de perda e *jitter*, respectivamente, e P_i e J_i são os valores de perda e *jitter* que serão considerados na monitoração i. Os valores iniciais P_0 e J_0 são iguais a 0. O fator α representa o peso que será dado às medições passadas ou à corrente.

Após o cálculo dos requisitos, estes serão enviados para o sistema de adaptação a fim de que sejam comparados com os valores limite fornecidos pela aplicação, verificando, assim, se houve piora da qualidade ou se o fluxo degradado apresentou melhora. Essa comparação e toda a adaptação proposta serão apresentadas na próxima seção.

3.4.5 Sistema de Adaptação

O sistema de adaptação é a parte essencial do *framework*. Esse mecanismo é que vai possibilitar fornecer qualidade para a transmissão nos momentos de instabilidade e congestionamento dos enlaces. Para ser melhor compreendido, ele pode ser dividido em três partes: a avaliação da qualidade da transmissão, os níveis de adaptação e o mecanismo de adaptação em si.

O sistema de adaptação inicia sua atuação ao receber do mecanismo de monitoramento os valores de perda de pacotes e de *jitter* medidos. Com esses valores, será possível fazer uma comparação entre as condições atuais da rede e os valores aceitáveis definidos pela aplicação. Essa comparação está representada no pseudo-código a seguir:

Se ((P_i > PERDA_MÁXIMA) OU (J_i > JITTER_MÁXIMO)) então
    Degrada o fluxo
senão
    Se (Nível_de_adaptação =! 0) então
        Se ((P_i < PERDA_MÍNIMA) E (J_i< JITTER_MÍNIMO)) então
            Melhora o fluxo

No código acima, o nível de adaptação 0 representa a melhor condição de operação - ou seja, não houve necessidade, até o momento, de degradações de qualidade. A primeira parte do código mostra que, se a perda de pacotes ou o *jitter* forem maiores que o valor máximo determinado pela aplicação, a rede não está conseguindo fornecer a qualidade desejada à transmissão. Assim, o nível de adaptação do fluxo aumentará um degrau em sua escala, ou seja, a taxa de transmissão do emissor será reduzida, respeitando o valor mínimo estipulado pela aplicação. É importante ressaltar que, se apenas um dos parâmetros apresentar um valor acima do desejado, o processo de adaptação é acionado.

A segunda parte, depois do *senão*, mostra que um fluxo degradado será melhorado, ou seja, terá a taxa de envio de pacotes aumentada, se a perda de pacotes e o *jitter* tiverem seus valores abaixo de seus valores mínimos. Nessa parte pode-se perceber que, diferentemente da degradação, o fluxo só tem a qualidade melhorada se os dois parâmetros apresentarem valores abaixo do mínimo. Esse método de comparação com valores máximo e mínimo para degradação e melhora do fluxo representa um efeito de histerese e ajuda que o sistema só volte para o nível de qualidade superior quando o fluxo estiver mais estável.

Quando os requisitos medidos apresentam valores acima dos limites máximos, a taxa de envio de pacotes deve ser imediatamente diminuída. Porém, após o fluxo apresentar degradação, quando esses requisitos apresentarem valores abaixo dos limites mínimos, a taxa de envio não pode subir imediatamente. É preciso que antes seja verificado o histórico de alterações do nível de adaptação, para que não haja muitas mudanças sucessivas de taxa de transmissão.

Dependendo do ambiente em que esteja instalada, uma rede sem fio pode apresentar muita instabilidade nas conexões e na disponibilidade de banda. Se for um ambiente em que muitos dispositivos sem fio estejam presentes, enviando e recebendo dados constantemente, a rede *Mesh* poderá apresentar congestionamento nos enlaces, o que gera atrasos e perdas provavelmente fora dos limites aceitáveis da aplicação.

Por outro lado, o cenário também pode ser o oposto: um ambiente estável, em que as transmissões encontrem obstáculos apenas ocasionalmente ou o meio seja levemente perturbado. Dessa forma, é preciso manter um mecanismo que identifique se a degradação ocorrida no fluxo é passageira, resultante de uma rápida interferência no meio, ou se ela está sendo causada por, por exemplo, um congestionamento prolongado na rede.

Se a degradação for momentânea, após um curto período de tempo será possível que o nível de adaptação da transmissão seja alterado, aumentando a taxa de envio dentro do intervalo requerido pela aplicação. Porém, se a má qualidade do fluxo for conseqüência de um grande congestionamento do meio, pelo menos por um determinado período de tempo não será possível que essa transmissão volte à taxa máxima de envio de pacotes, evitando que a instabilidade do meio continue a atrapalhar a qualidade da transmissão. Uma questão importante a ser apontada é que, independentemente do problema que esteja ocorrendo no ambiente sem fio, a injeção de mais dados na rede só irá causar maiores perdas e atrasos nos pacotes.

Ainda assim, é preciso pensar que, se a taxa de transmissão for mantida no nível mais baixo, permitindo que os valores de perda de pacotes e *jitter* se estabilizem, os recursos disponíveis na rede poderão estar sendo sub-utilizados, já que será possível existir um nível de adaptação intermediário em que se obtenha a mesma qualidade na transmissão. Dessa forma, o mecanismo que identifica se a degradação é passageira ou prolongada também permite que, no segundo caso, seja encontrado um nível intermediário de adaptação que obtenha os valores aceitáveis para os requisitos de QoS, ao mesmo tempo em que utiliza os recursos disponíveis de forma satisfatória.

Para este trabalho, é chamado de "nível ótimo" aquele nível que apresentar condições favoráveis à transmissão dos dados, fornecendo a qualidade desejada pela aplicação de forma estável, porém não sub-utilizando os recursos do ambiente. No início de todas as transmissões, o "nível ótimo" será o nível 0, que representa a maior taxa de envio.

O mecanismo de adaptação, ao avaliar que os parâmetros de *jitter* e perda de pacotes apresentaram valores abaixo do limite mínimo, percebe que pode aumentar a taxa de envio dos pacotes, alterando o nível de adaptação. Porém, antes de enviar um relatório de QoS ao servidor fazendo esta requisição, o *framework* analisa se o próximo nível é considerado o "nível ótimo" para o fluxo. Se for, então o *framework* verifica quantas vezes o fluxo já esteve neste "nível ótimo" e não conseguiu manter-se nele, por falta de qualidade. Se esse valor exceder uma quantidade T, o fluxo então não muda de nível de adaptação, pois há uma probabilidade de não conseguir novamente se manter com essa taxa de envio. O

framework impede que o fluxo volte para esse nível apenas durante um determinado tempo, chamado, neste trabalho, de tempo de manutenção. Passado esse tempo de manutenção, ele pode tentar novamente obter uma qualidade satisfatória nesse nível de adaptação, que anteriormente apresentava uma qualidade ruim. A escolha desse valor T será apresentada no capítulo que relata os testes e resultados.

Os níveis de adaptação representam em que taxa o servidor deve enviar os pacotes para o cliente. Para efeito de implementação e avaliação da proposta, foram definidos quatro níveis de adaptação para o *framework*, variando de 0 a 3. Um fluxo sempre começa sua transmissão no nível 0, que permite a taxa de transmissão máxima. As taxas de envio dos outros níveis serão definidas de forma justa, fazendo uma distribuição proporcional dentro da faixa de envio mínimo e máximo. Podem-se tomar como exemplo os valores 25fps para taxa máxima e 10fps para taxa mínima. Assim, o nível de adaptação 0 corresponde à taxa de envio de 25fps; o nível 1 terá uma taxa de 20fps; o nível 2, uma taxa de 15fps; e o nível 3, uma taxa de 10fps, que é a taxa mínima.

Uma vez decidido que a taxa de transmissão corrente não equivale à permitida pelas condições da rede, um relatório de QoS é enviado do cliente ao servidor para avisar que uma providência deve ser tomada. Assim, é o nó cliente quem decide se a taxa corrente de envio do nó servidor deve ser aumentada ou diminuída. O fato de o mecanismo de decisão do sistema de adaptação estar no cliente apresenta duas vantagens: primeiramente, o sistema torna-se escalável com o aumento do número de nós móveis na rede, já que o servidor não será responsável por monitorar e avaliar a qualidade de cada fluxo; em segundo lugar, ocorre uma diminuição do tráfego de controle na rede, uma vez que o cliente apenas se comunica com o servidor para requisitar uma mudança do nível de adaptação.

Ao receber um relatório de QoS, o nó servidor toma conhecimento de que precisa acionar o processo de adaptação. Nesse relatório, o nó cliente determina para qual nível de adaptação o nó servidor deve ir. Esse servidor tem o conhecimento de qual é a taxa de transmissão que corresponde a cada nível de adaptação. Assim, ele passa a enviar os pacotes de vídeo a uma taxa referente ao novo nível de adaptação, além de alterar também o valor do campo correspondente ao nível no cabeçalho das mensagens.

3.5 Conclusão

A proposta para o *framework* de suporte a QoS em redes *Mesh* apresentada neste capítulo é uma solução simples que juntou características positivas dos trabalhos estudados das classes com e sem reserva de recursos. A escolha de implantar um mecanismo de reserva no *backbone* da rede *Mesh* tem como objetivo dar maior prioridade e largura de banda aos fluxos multimídia em detrimento de outros tráfegos. Porém, também se mostrou necessário um mecanismo de monitoração da qualidade da transmissão fim-a-fim para que o mecanismo de adaptação possa ser acionado, caso os requisitos de QoS apresentem resultados fora do desejável.

Os requisitos escolhidos para monitoração foram perda de pacotes e *jitter*, pois são eles que causam mais impacto na qualidade da transmissão de dados multimídia, como áudio e vídeo. A partir dessa monitoração é que o processo de adaptação pode entrar em ação para adequar a transferência dos pacotes às condições da rede. A adaptação consiste em diminuir ou aumentar a taxa de transmissão. Dessa forma, caso a rede esteja congestionada, a taxa de injeção de tráfego na rede irá diminuir, e a qualidade da transmissão irá melhorar. Caso a rede volte a apresentar melhores condições, o envio dos pacotes poderá ser normalizado, utilizando os recursos agora disponíveis.

Vale ressaltar que o mecanismo de adaptação proposto envolve apenas as extremidades da transmissão. A partir da monitoração da qualidade do fluxo, o nó cliente decide se deve adaptá-lo ou não. Caso positivo, o nó servidor participa dessa adaptação, diminuindo ou aumentando a taxa de transmissão. Os roteadores do caminho não alteram dinamicamente suas reservas por causa do processo de adaptação na presente implementação. Os pacotes enviados apenas apresentam níveis diferentes de adaptação e são então encaminhados para as diferentes classes nos roteadores.

O próximo capítulo apresentará os testes realizados em um ambiente de simulação e os resultados obtidos nesta avaliação preliminar.

Capítulo 4

Avaliação Preliminar da Proposta

4.1 Introdução

Este capítulo apresentará um estudo preliminar sobre a proposta do *framework* de suporte a QoS para fluxos de tempo real em redes *Mesh*. O objetivo deste estudo foi avaliar, via simulação, se a proposta sugerida realmente resultava em uma melhora de qualidade da transmissão de fluxos multimídia, antes de implementar o protótipo. Além disso, o estudo com simulação também facilita a calibragem dos parâmetros da proposta, como os limites mínimo e máximo de *jitter* e perda de pacotes, entre outras variáveis dos algoritmos.

A próxima seção explicará o que foi usado para realizar as simulações e como foi implementado o *framework*. Para a avaliação comparativa, também foram executados testes com o *framework* INSIGNIA, proposto em [Lee et al. 2000]. O site [INSIGNIA 2001] disponibiliza o código desse *framework* para ser executado no simulador de rede *ns-2*. Além da comparação entre as duas propostas, também foram executados testes sem política nenhuma de adaptação, para se ter uma noção da melhora de qualidade proporcionada pelas propostas deste trabalho.

4.2 Simulação

Esta seção irá explicar o processo de escolha do ambiente de simulação e do protocolo de roteamento utilizado. A implementação do *framework* será descrita na subseção 4.2.3. É importante ressaltar que os testes no simulador têm o objetivo de apresentar um estudo preliminar. O produto final deste trabalho é o protótipo implementado para a rede *Mesh* do projeto *ReMesh*.

4.2.1 *Network Simulator - ns-2*

O simulador de rede escolhido foi o *ns-2* [NS-2 2000]. Ele é muito utilizado em pesquisas na área de redes, tanto cabeadas quanto sem fio. A comunidade de usuários pelo mundo é extensa, e a documentação apresenta um conteúdo satisfatório para os usuários. Desde 1989, os desenvolvedores do *ns-2* já lançaram diversas versões do simulador.

A escolha de qual versão utilizar foi influenciada pelo estudo do *framework* INSIGNIA. O código disponibilizado pelos seus desenvolvedores foi implementado para a versão *ns-allinone-2.1b3* do *network simulator*. Atualmente, o *ns-2* está na sua versão 2.30. Como a implementação do INSIGNIA altera o código do próprio simulador, e a proposta do nosso *framework* para redes *Mesh* também iria fazer algumas alterações, ficou decidido que se trabalharia com a versão *ns-allinone-2.1b3* para que a comparação entre os dois trabalhos não fosse prejudicada. Na subseção que explica a implementação do *framework* para *Mesh* serão citadas quais partes do código do *ns-2* foram alteradas.

4.2.2 Protocolo de Roteamento

Como foi citado na seção 2.2.7, o projeto *ReMesh* escolheu o protocolo *Ad Hoc* OLSR para fazer o roteamento das mensagens na rede *Mesh*. Porém, o código do *framework* INSIGNIA utiliza o protocolo DSR para fazer seus testes e gerar resultados. Para fazer uma comparação adequada entre o INSIGNIA e o *framework* para redes *Mesh* aqui proposto, também foi usado o protocolo de roteamento DSR nas simulações realizadas.

Os protocolos de roteamento OLSR e DSR apresentam algumas diferenças. A principal delas é que o DSR faz parte do grupo dos protocolos reativos, que disparam a busca por uma rota apenas quando um nó deseja transmitir dados. Já o OLSR faz parte do grupo dos protocolos pró-ativos, pois mantém sempre sua tabela de rotas atualizada. Mesmo com essa diferença, mais à frente será possível perceber que os resultados da simulação e do protótipo tiveram comportamentos semelhantes.

4.2.3 Implementação do *Framework*

Para facilitar a implementação do *framework* no ambiente de simulação *ns-2*, o código do INSIGNIA foi estudado e utilizado como modelo para a codificação. Mesmo que as duas soluções utilizem reserva de recursos, o modo como esse mecanismo foi proposto por cada trabalho é diferente. No entanto, o código no INSIGNIA já estava inserido no *ns-2* de

maneira que atuasse como uma camada de tratamento de pacotes nos nós ao longo da transmissão.

Cada nó, antes de encaminhar adiante um pacote que chegou até ele ou que foi nele gerado, deve passá-lo para a camada do *framework* de suporte a QoS. A partir das informações contidas no cabeçalho IP do pacote, o nó poderá verificar se ele deve se comportar como origem, destino ou como um nó intermediário. Cada um desses comportamentos terá funções no processo de suporte a QoS. As seções seguintes irão detalhar como foi a implementação das funções em cada nó.

4.2.3.1 Nó Origem

Uma função do nó origem na simulação é a de inserir o cabeçalho do *framework* em todos os pacotes que forem enviados. Além dos campos descritos na seção 3.4.2, também é inserido no cabeçalho o instante em que o pacote saiu da origem. Na simulação, o nó origem faz o papel do servidor.

A transmissão do vídeo foi simulada como um fluxo constante, ou *constant bit rate*, usando um tráfego CBR. O tamanho do pacote foi fixado em 2048 *bytes*, que é o tamanho médio de um quadro de vídeo. Inicialmente, esse fluxo é enviado com a taxa máxima definida pela aplicação. Ao receber um relatório de QoS enviado pelo nó destino, o nó origem então altera essa taxa para a correspondente ao novo nível de adaptação especificado na mensagem.

Para que fosse possível fazer a alteração da taxa de envio ao receber um relatório de QoS indicando um novo nível de adaptação, foi alterada a classe do *ns-2* que simula o tráfego CBR. Assim, a camada do *framework* no nó origem recebe um objeto que representa o gerador de tráfego CBR na rede, podendo manipular a taxa de transmissão.

4.2.3.2 Nó Intermediário

Os nós intermediários farão o papel dos roteadores sem fio da rede *Mesh*. Assim, eles serão responsáveis por, além de encaminhar pacotes, fazer a reserva de banda para os fluxos. Como foi explicado na seção 3.4.3, a reserva é feita para as diferentes classes de fluxos definidas pelo *framework*. Para efeito de testes, a distribuição da capacidade nominal da largura de banda de cada nó foi realizada de acordo com a figura 4.1.

A classe destinada aos pacotes de vídeo foi dividida em quatro subclasses. Cada subclasse corresponde a um nível de adaptação. A classe de nível 0 possui uma fatia maior

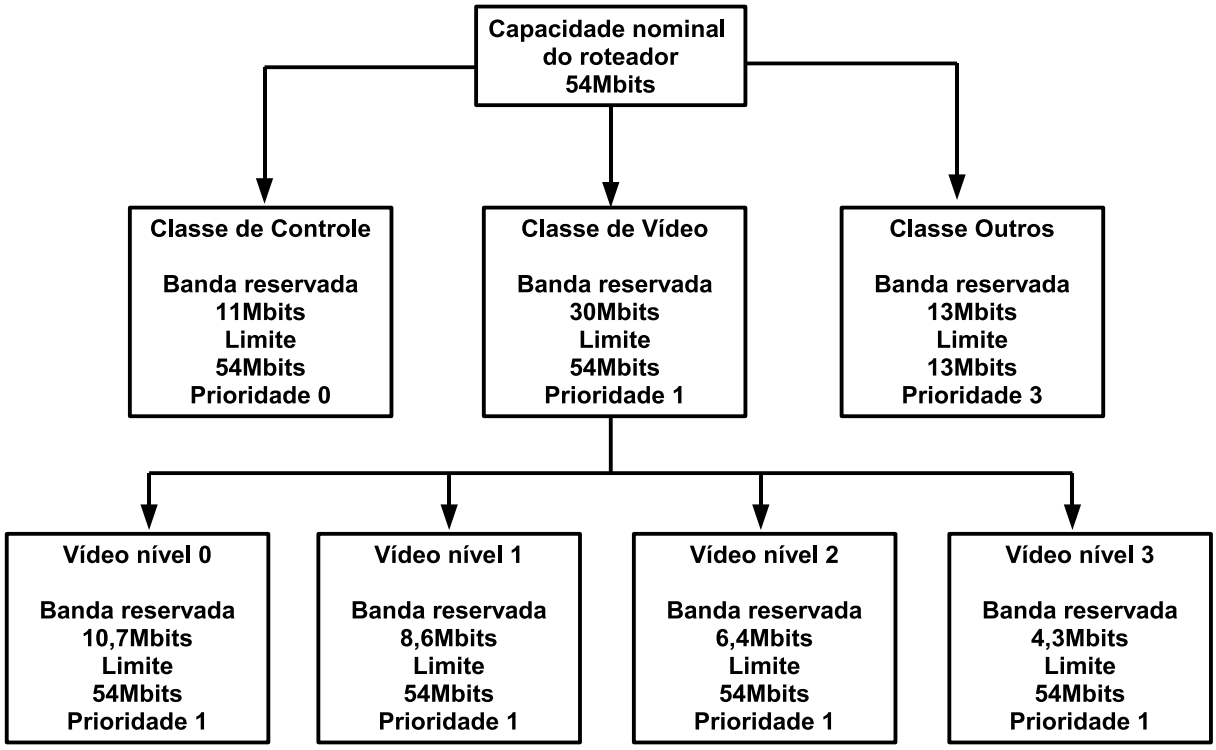


Figura 4.1: Estrutura de classes e subclasses do mecanismo de reserva de recursos.

de banda reservada pois apresenta a taxa máxima de transmissão. Da mesma forma que é feita a distribuição proporcional de taxa de envio entre os níveis de adaptação, de acordo com os limites mínimo e máximo de transmissão, também a banda reservada para a classe de vídeo é distribuída proporcionalmente para suas subclasses.

Assim, para simular a ferramenta de controle de tráfego utilizada para fazer o gerenciamento de largura de banda, além de terem sido criadas as classes e subclasses, alguns campos foram inseridos no cabeçalho do *framework* para auxiliar a simulação. Os três novos campos foram: largura de banda mínima, de acordo com a taxa de envio mínima e o tamanho dos pacotes; largura de banda máxima, de acordo com a taxa de envio máxima e o tamanho dos pacotes; e largura de banda reservada ao longo do caminho.

Assim que o nó intermediário receber um pacote com requisição de reserva, ele irá identificar a qual classe o pacote pertence e qual a disponibilidade de banda que essa classe possui. De acordo com os campos de limites mínimo e máximo e largura de banda reservada, o *framework* irá então gerenciar a fatia de banda destinada para cada classe. Caso uma classe de vídeo já esteja utilizando toda a banda reservada para ela, o *framework* poderá pegar banda emprestada de outra classe. Assim, para esse experimento, um fluxo com QoS nunca terá sua reserva negada pelos roteadores do caminho. Uma solução

também possível é a de realizar um controle de admissão para todos os fluxos, mesmo para aqueles com suporte a QoS. Embora, essa solução não tenha sido implementada, ela será mencionada como trabalho futuro.

4.2.3.3 Nó Destino

As funções do nó destino são fundamentais para o fornecimento de QoS à transmissão. Como já foi explicado no capítulo 3, o nó destino deve fazer periodicamente a monitoração dos requisitos de QoS, perda de pacotes e *jitter*, para conhecer as condições da rede. Na simulação, o nó destino faz o papel do cliente.

Essa monitoração é feita através de um mecanismo de janela, que define a quantidade de pacotes que se deve esperar que cheguem ao destino. Essa janela também é um parâmetro investigado na simulação, a fim de se determinar qual seria o melhor tamanho. As informações que serão usadas para as medições de perda de pacotes e *jitter* são o número identificador do pacote, o tempo de saída do pacote da origem e o tempo de chegada ao destino. Essas informações são armazenadas em um *buffer*. Após as medições, esse *buffer* é apagado e novamente preenchido com as informações dos novos pacotes que chegarem ao destino.

Da mesma forma que descrito na seção 3.4.4, os valores de perda de pacotes e *jitter* ainda sofrem uma amortização, sendo atribuído um peso α às medições passadas e um peso (1 - α) à atual. São esses valores amortizados de perda de pacotes e de *jitter* que serão usados na comparação com os limites máximo e mínimo, para verificação da qualidade da transmissão.

Para avisar que o nível de adaptação deve sofrer uma alteração, o nó destino envia um relatório de QoS para a origem, informando o novo nível que o fluxo deve apresentar. Essa mensagem de QoS também terá uma reserva de banda para ela e prioridade ao longo do caminho. Ao receber um pacote com o novo nível de adaptação, o destino fica sabendo que seu relatório de QoS foi recebido pela origem e que as providências de alteração da taxa de envio já foram executadas.

4.3 Avaliação do *Framework*

A avaliação do desempenho do *framework* será apresentada de forma comparativa, entre os seus resultados, os do INSIGNIA e os resultados dos testes sem nenhum mecanismo de

reserva. A próxima seção descreverá os cenários criados para a execução dos testes. Depois, serão descritas as métricas investigadas para avaliação de desempenho. Por último, serão apresentados os gráficos extraídos dos testes.

4.3.1 Cenários

Os cenários criados para a realização dos testes tiveram como objetivo representar situações reais em um ambiente universitário. Para efeito de teste, a topologia montada para os roteadores foi a mostrada na figura 4.2. Ela apresenta os roteadores sem fio dispostos de forma linear. As distâncias entre eles foram definidas para que não houvesse nenhum espaço sem cobertura na área de testes.

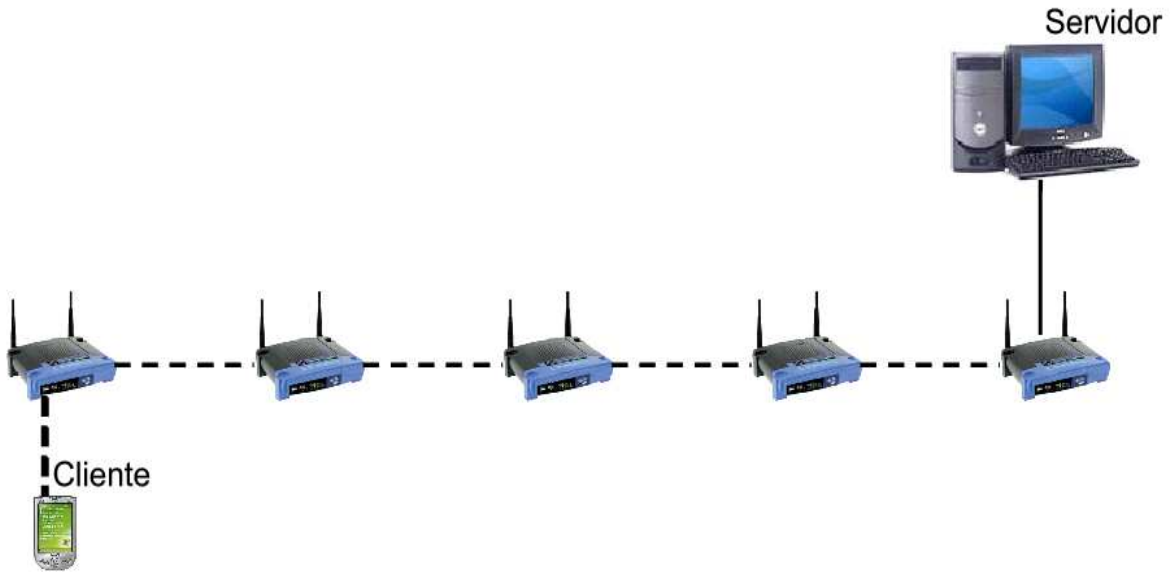


Figura 4.2: Topologia da rede *Mesh* utilizada nas simulações.

Os 5 nós que representam os roteadores sem fio utilizam o modelo 802.11 na camada MAC, com um alcance de 250m e uma taxa efetiva de 54Mbits/s. Como já foi descrito, o protocolo de roteamento utilizado foi o DSR, por já fazer parte do código do *framework* INSIGNIA. A área definida para todos os cenários é de 1250m x 250m. Todas as simulações têm a duração de 500s.

A aplicação escolhida foi uma cliente/servidor, em que um nó móvel cliente faz uma requisição de um vídeo a um servidor, parado e conectado em uma das extremidades da

rede *Mesh*. Foram feitos testes com diferentes valores de taxa de envio mínima e máxima, variando entre 25fps e 5fps. Foram definidos 4 cenários para os testes:

- **Cenário 1**: o nó cliente se mantém parado na extremidade oposta ao servidor e requisita um fluxo de vídeo, sem haver nenhuma interferência no ambiente ou tráfego extra na rede.
- **Cenário 2**: o nó cliente se movimenta em direção ao servidor com velocidade de uma pessoa caminhando (entre 3km/h e 5km/h) e requisita um fluxo de vídeo, sem haver nenhuma interferência no ambiente ou tráfego extra na rede.
- **Cenário 3**: o nó cliente se mantém parado na extremidade oposta ao servidor e requisita um fluxo de vídeo. Porém, é inserido um tráfego CBR de 1 pacote por segundo em um dos roteadores, para simular o tráfego gerado pelos computadores que estiverem conectados aos roteadores pela porta LAN, com objetivo de causar perturbação e tráfego extra na rede.
- **Cenário 4**: o nó cliente se movimenta em direção ao servidor e com velocidade de uma pessoa caminhando (entre 3km/h e 5km/h) e requisita um fluxo de vídeo. Porém, é inserido um tráfego CBR de 1 pacote por segundo em um dos roteadores, para simular o tráfego gerado pelos computadores que estiverem conectados aos roteadores pela porta LAN, com objetivo de causar perturbação e tráfego extra na rede.

4.3.2 Resultados

Para a análise dos resultados, foram utilizadas as métricas de perda de pacotes, *jitter*, atraso médio e vazão (em fps). Antes de iniciar os testes comparativos, foram executados testes para descobrir quais seriam os melhores valores para os parâmetros do *framework*. Esses parâmetros são:

- Tamanho da janela de monitoração;
- Valores limites para perda de pacotes e *jitter*; e,
- Parâmetros para o algoritmo que procura a melhor taxa de envio de acordo com as condições da rede.

A monitoração é feita de acordo com a quantidade de pacotes que chega ao destino. A subseção 4.3.2.1 apresenta os testes realizados com o objetivo de descobrir o melhor tamanho para a janela de monitoração. Os valores limite de perda de pacotes e *jitter* devem ser especificados pela aplicação, porém alguns testes foram realizados para se ter um indicação de quais são os valores que, de acordo com o cenário de uma rede *Mesh*, apresentam melhores resultados de qualidade.

O algoritmo proposto para encontrar uma taxa de transmissão adequada à qualidade da rede possui dois parâmetros: quantidade limite de vezes que um fluxo já esteve em um determinado nível de adaptação e não conseguiu se manter nele e o tempo em que o *framework* não passará o fluxo para esse dado nível, posto que ele não apresenta condições favoráveis de qualidade. Os resultados encontrados são mostrados na subseção 4.3.2.3.

Não foram feitos testes separadamente para descobrir o melhor valor para o fator α de amortização. Porém, ao longo dos testes iniciais, foi percebido que é preciso dar maior peso para a situação corrente da rede. Por essa razão, foi atribuído o valor 0,4 ao fator α, dando um peso de 0,6 para as medições correntes.

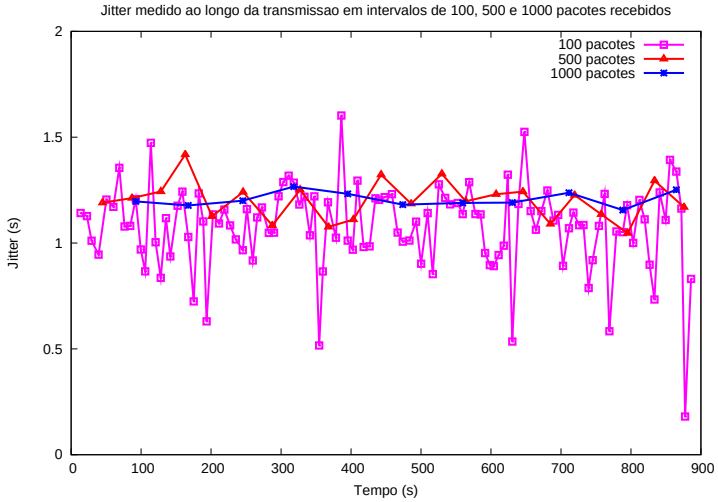
4.3.2.1 Janela de Monitoração

Os testes para identificar o tamanho da janela de monitoração foram executados apenas no cenário 1, em que o nó cliente está parado na extremidade oposta da rede ao servidor. A taxa de envio de pacotes também foi única, de 30fps. É possível perceber que, mesmo com o cenário mais simples, no qual não existe nenhuma interferência, a taxa de 30fps, com pacotes de tamanho 2048 *bytes*, já causa grande perturbação no meio e gera perda de pacotes e *jitter*.

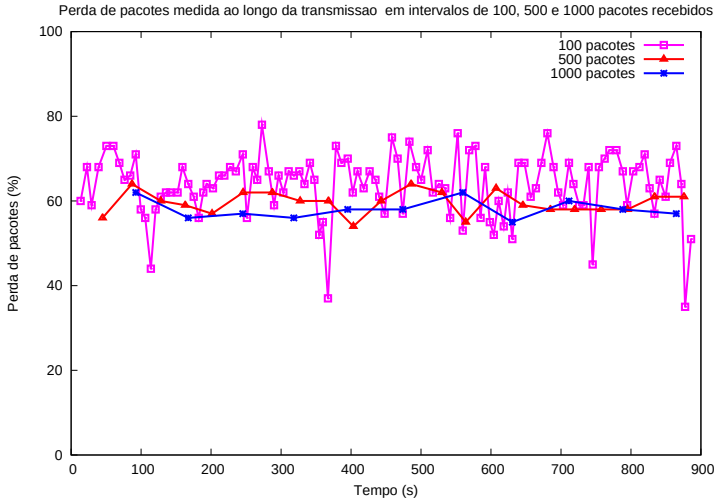
O código do *framework* foi modificado para que ele não acionasse a adaptação em caso de queda da qualidade. Isso foi feito para se perceber como o *framework* se comportaria caso não tivesse esse mecanismo, e para observar a evolução das monitorações. O mecanismo de reserva de recursos nos nós intermediários continuou em funcionamento.

Foram escolhidos três valores para testar o tamanho da janela que armazenará os pacotes para fazer a monitoração: 100, 500 e 1000. Os resultados dos testes são mostrados nos gráficos da figura 4.3.

A execução com janela de tamanho 100 pacotes apresentou uma variação maior tanto de perda de pacotes como de *jitter*. Os testes com o tamanho de janela igual a 500 e a 1000 tiveram uma variação menor nos dois requisitos, já que o tempo entre monitorações



(a)



(b)

Figura 4.3: Resultados dos testes de monitoração - (a) *Jitter* em segundos; (b) Perda de pacotes em porcentagem.

é maior.

Um tamanho menor de janela permite que qualquer alteração das condições da rede, tanto a degradação quanto a melhora, seja percebida mais rapidamente. Porém, em um ambiente que apresente condições estáveis de transmissão, essa quantidade maior de monitorações pode vir a ser desnecessária. Nesses casos, um tamanho maior de janela pode atender melhor às necessidades de avaliação periódica da rede.

Na situação oposta, em que o ambiente esteja muito congestionado e instável, um tamanho de janela igual a 1000, por exemplo, pode fazer com que o *framework* demore a perceber a degradação da rede. Assim, é preciso pensar no pior caso, em que as con-

dições do ambiente estejam desfavoráveis à aplicação e, por isso, seja preciso realizar monitorações mais freqüentes.

Por estas razões, mesmo que a variação da perda de pacotes e do *jitter* seja maior, a melhor solução é escolher um tamanho de janela menor. Dessa forma, o *framework* consegue perceber mais rapidamente uma degradação do fluxo. A solução mais adequada, porém mais complexa, seria a de determinar dinamicamente o tamanho da janela. Nesse protótipo foi escolhida uma solução mais simples: fixar o tamanho da janela em 100 pacotes.

Com esses resultados, é possível então executar os próximos testes para conseguir uma previsão dos limites mínimo e máximo de perda de pacotes e *jitter* para a execução do *framework*. Após a avaliação dos limites dos requisitos de QoS e dos parâmetros do algoritmo de busca pela melhor taxa, serão apresentados os resultados de alguns testes que confirmam a escolha do tamanho igual a 100 para a janela de monitoração.

4.3.2.2 Limites de Requisitos de QoS

Para os testes de calibragem de *jitter* e de perda de pacotes, a amortização de valores e o efeito de histerese foram inseridos no mecanismo de adaptação; ficou de fora apenas o algoritmo que procura a melhor taxa de transmissão para o fluxo. Esses testes também foram feitos apenas no cenário 1 e com taxa de transmissão mínima e máxima de 10fps e 30fps, respectivamente. Foram usados dois limites para cada parâmetro, como mostram as tabelas abaixo.

Jitter	**Mínimo**	**Máximo**
Limite 1	0,15s	0,6s
Limite 2	0,25s	0,6s

Tabela 4.1: Limites para *Jitter*

Perda de pacotes	**Mínima**	**Máxima**
Limite 1	0,75%	2,75%
Limite 2	2,00%	5,00%

Tabela 4.2: Limites para Perda de pacotes

Foram executados quatro testes, fazendo uma combinação entre os limites estabelecidos (tabela 4.3). Como já dito anteriormente, esses valores-limite dependem muito do tipo de aplicação que irá utilizar os serviços do *framework*. A aplicação escolhida foi uma videoconferência, que não possui restrições tão rigorosas quanto à qualidade da mídia como, por exemplo, uma aplicação de telemedicina. Esses valores foram então escolhidos após uma pesquisa sobre esse tipo de aplicação.

Os gráficos dos testes são apresentados nas figuras 4.4 e 4.5. Cada gráfico possui dois eixos y. Os que medem *jitter* ao longo da transmissão têm o eixo y esquerdo marcando o *jitter* em segundos, e o direito marcando a taxa de transmissão corrente em fps. Os

Teste	Combinação de valores
Teste 1	Limite de *jitter* 1 com Limite de perda de pacotes 1
Teste 2	Limite de *jitter* 1 com Limite de perda de pacotes 2
Teste 3	Limite de *jitter* 2 com Limite de perda de pacotes 1
Teste 4	Limite de *jitter* 2 com Limite de perda de pacotes 2

Tabela 4.3: Combinação de limites para os testes

que medem perda de pacotes ao longo da transmissão têm o eixo y esquerdo marcando a porcentagem de perda e o direito, também a taxa de transmissão. Em todos eles são mostrados os limites mínimo e máximo do parâmetro medido.

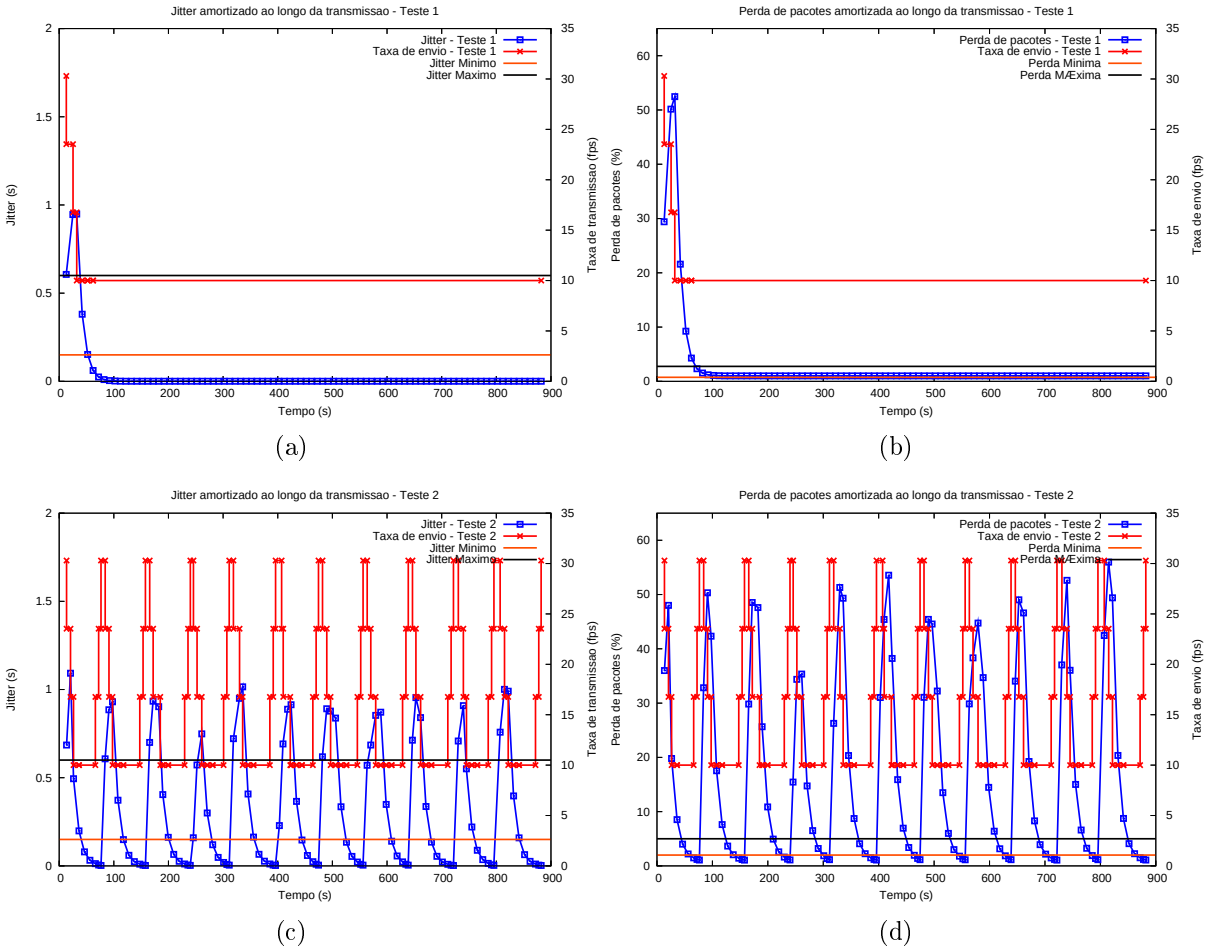


Figura 4.4: Resultados dos testes 1 e 2 para escolha de limites - (a) e (b) *Jitter* e Perda de pacotes no teste 1; (c) (d) *Jitter* e Perda de pacotes no teste 2.

Analisando os quatro testes, tem-se a impressão de que os resultados dos testes 1 e 3 são iguais, e que os resultados dos testes 2 e 4 também são iguais. Mas eles possuem pequenas diferenças nos momentos em que ocorrem as adaptações. Essa semelhança acontece porque nesses testes os valores-limite de perda de pacotes são os mesmos. Isso

leva à conclusão de que o parâmetro perda de pacotes exerce uma influência maior na tomada de decisão para acionar a adaptação. Porém, o parâmetro *jitter* gera uma diferença de poucos segundos entre os testes no momento dessas decisões. Nos testes com o protótipo no ambiente real, também será verificada a atuação do *jitter* no processo de adaptação.

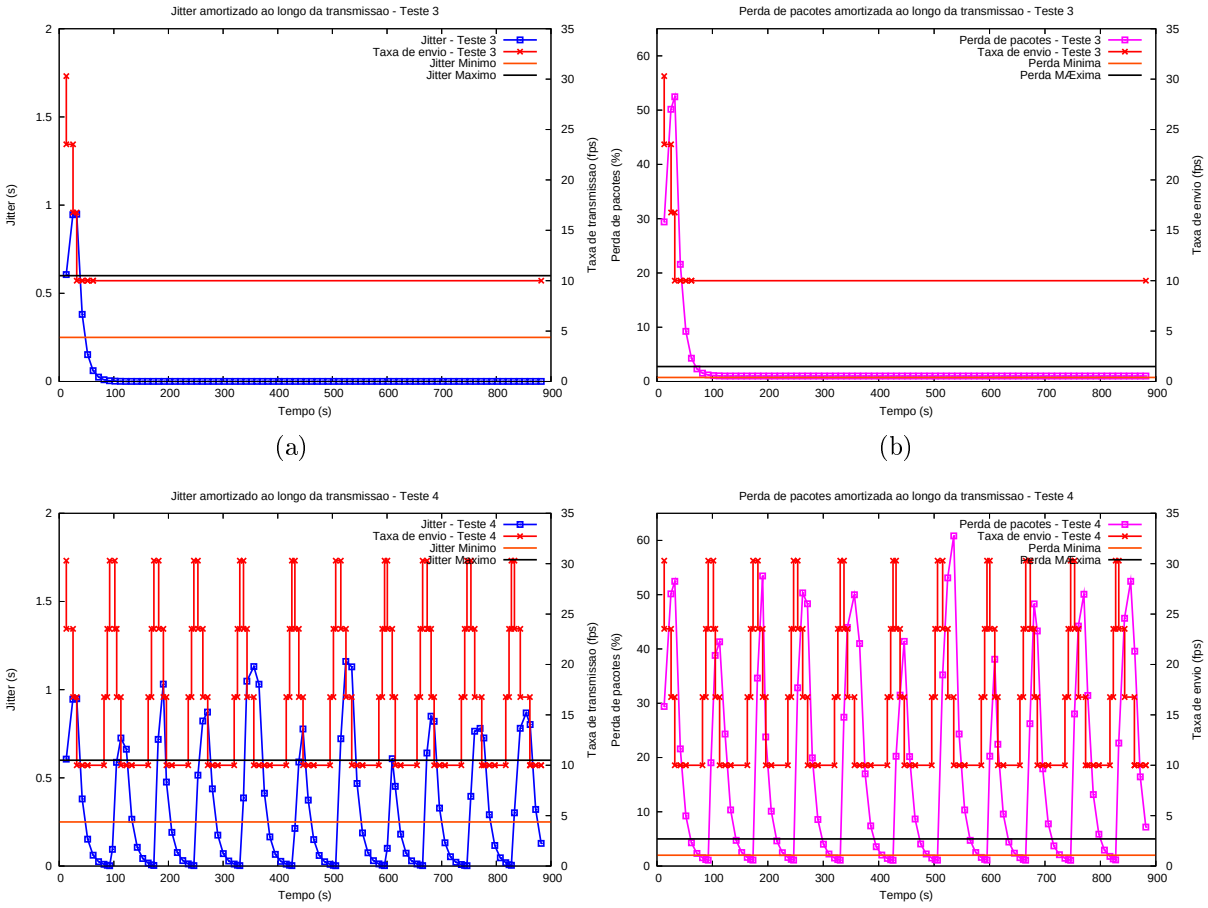


Figura 4.5: Resultados dos testes 3 e 4 para escolha de limites - (a) e (b) *Jitter* e Perda de pacotes no teste 3; (c) e (d) *Jitter* e Perda de pacotes no teste 4.

Os testes 1 e 3, apesar de terem mantido os valores de *jitter* e perda de pacotes perto de 0, não permitiram que o *framework* averiguasse se os outros níveis de adaptação também tinham condições de manter a qualidade do fluxo. O valor mínimo de perda de pacotes nesses testes é muito baixo (0,75%), o que dificulta o processo de aumento da taxa de transmissão. Com isso, os testes subutilizaram os recursos disponíveis da rede, pois também seria possível conseguir um nível de qualidade desejável com uma taxa de transmissão maior, como pode ser observado nos testes 2 e 4. No entanto, nesses últimos, a taxa de envio sofreu muitas variações, pois a taxa máxima de 30fps gera um volume de tráfego que não atende aos requisitos de qualidade definidos. Esse grande número de variações acontece pois, nesses testes, o algoritmo utilizado para encontrar a melhor

taxa de envio não foi inserido. A próxima seção irá apresentar os testes feitos com esse algoritmo.

Com esses resultados, foram escolhidos os limites 0,25s e 0,6s de *jitter* mínimo e máximo, respectivamente, e os valores 2,0% e 5,0% para limite mínimo e máximo de perda de pacotes, de acordo com o teste 4. Com a inserção do algoritmo de estabilização da taxa de envio, o *framework* irá apresentar um maior equilíbrio na sua função.

4.3.2.3 Busca pela Melhor Taxa de Envio

O *framework* utiliza um algoritmo que procura encontrar o nível de adaptação que apresente o melhor desempenho de acordo com as condições do ambiente, chamado nesse trabalho de "nível ótimo". Ao identificar que as condições da rede estão aptas a aumentar a taxa de envio, o sistema de adaptação primeiro verifica se o próximo nível é o ótimo. Caso seja, o *framework* então confere quantas vezes aquele fluxo já esteve no nível ótimo e apresentou qualidade ruim, tendo que voltar ao nível corrente. Se essa quantidade for maior que um valor T, o *framework* então não faz o aumento da taxa, pois a probabilidade de que o fluxo apresente uma qualidade ruim e volte novamente ao nível corrente é grande. Esse valor T é que será testado nessa seção.

As condições desses testes foram as mesmas que as dos testes anteriores: apenas o cenário 1 foi testado com taxa de envio mínima de 10fps e máxima de 30fps. Os limites mínimo e máximo de *jitter* e de perda de pacotes utilizados foram os definidos na seção anterior.

Os testes 1 e 2 avaliaram dois valores para o parâmetro T, que determina quando um nível pode ser considerado ruim, pois não conseguiu manter um fluxo por um número limite de vezes. Os valores-limite testados foram T igual a 2 no primeiro teste e T igual a 3 no segundo. Os resultados são apresentados na figura 4.6.

Os resultados mostram que, no teste 1, a taxa de envio se estabilizou mais rapidamente do que no teste 2, o que gerou também uma estabilidade na perda e no *jitter*. É possível perceber então que, neste cenário, o melhor nível de adaptação para o fluxo é o 1, que representa um taxa de envio de 23fps. Por essa razão, foi escolhido o valor 2 para o parâmetro T.

Tanto no teste 1 como no teste 2, após o tempo de manutenção no nível estável, o *framework* tenta novamente subir a taxa de envio para verificar se as condições da rede melhoraram. Como novamente o fluxo não conseguiu se manter no melhor nível, após

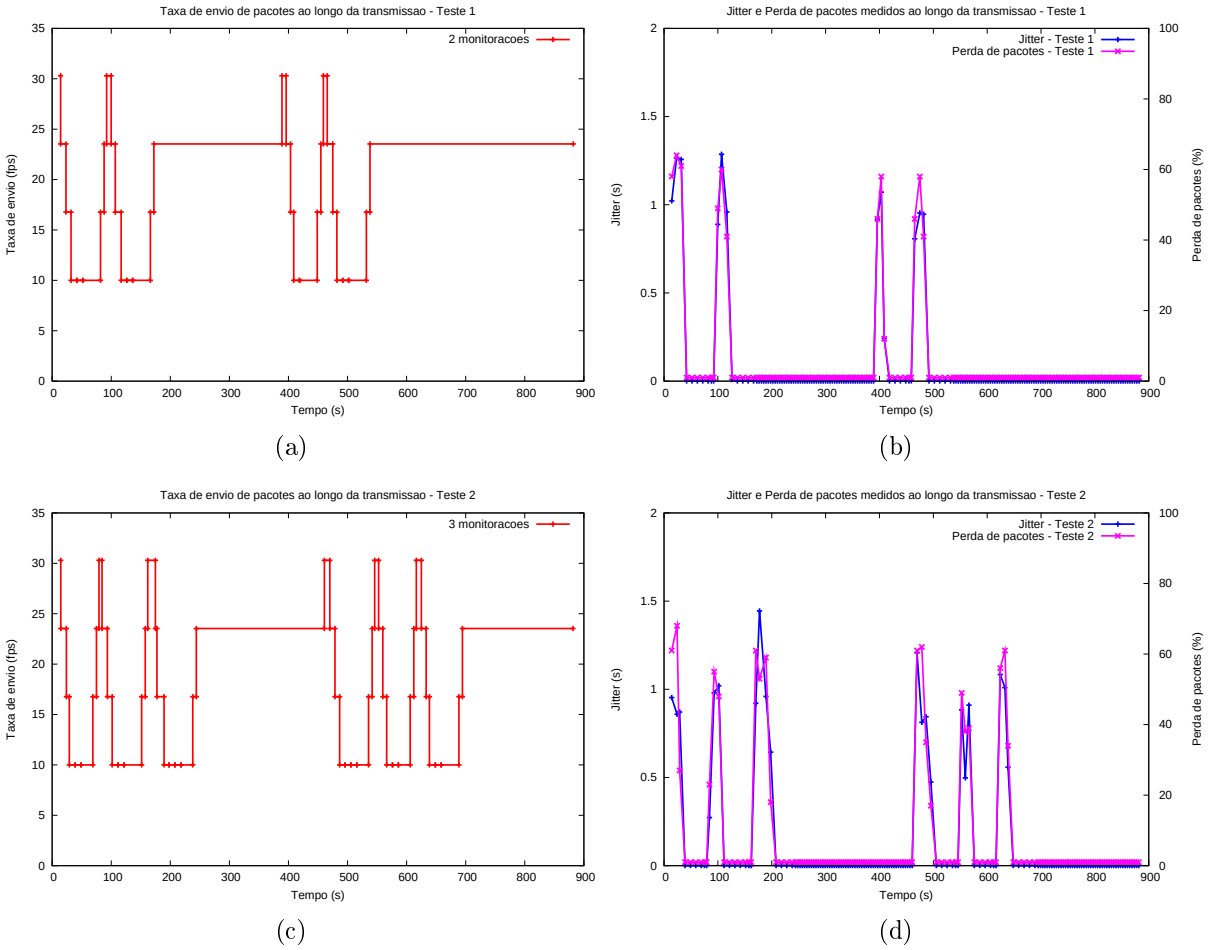


Figura 4.6: Resultados dos testes de procura pela melhor taxa de envio - (a) e (c) Evolução da taxa de envio nos testes 1 e 2; (b) e (d) Evolução do *jitter* e da perda de pacotes nos testes 1 e 2.

dois ou três momentos de queda, o *framework* mantém o fluxo um nível abaixo, só que agora o tempo de manutenção dobra. Esse tempo no melhor nível é inicialmente de 50 monitorações, o que nos testes realizados representa cerca de 3 minutos.

Com os três testes avaliados até aqui, os parâmetros definidos para o *framework* são:

- o tamanho da **janela de monitoração** será de 100 pacotes;
- o ***jitter* mínimo** terá o valor de 0,25 segundos e o ***jitter* máximo**, de 0,6 segundos;
- a **perda de pacotes mínima** terá o valor de 2,0%, e a **perda de pacotes máxima** de 5,0%; e,
- o **algoritmo para encontrar a melhor taxa de envio** terá o valor de T igual a 2. O tempo de manutenção será igual a 50 monitorações.

Com a definição dos valores-limite dos requisitos de QoS e do parâmetro T do algoritmo, foram realizados testes com três diferentes tamanhos de janela de monitoração: 50, 100 e 500 pacotes. Os resultados desses testes são mostrados na figura 4.7 através de gráficos das mudanças de taxa de envio em cada experimento.

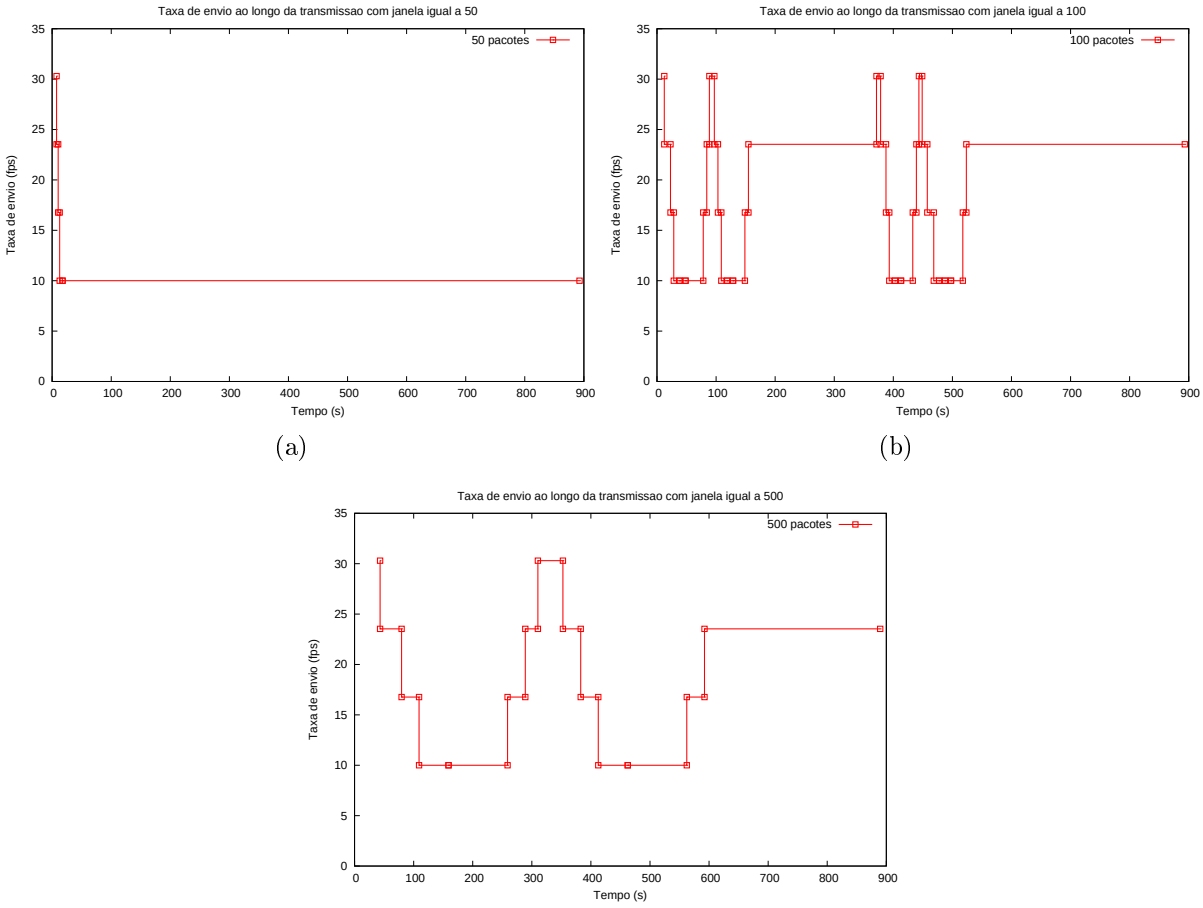


Figura 4.7: Taxa de envio ao longo da transmissão com diferentes tamanhos de janela - (a) 50 pacotes; (b) 100 pacotes; (c) 500 pacotes.

Na avaliação desses resultados, observou-se que o uso de uma janela de monitoração muito pequena, como a de tamanho 50, que gera intervalos muito pequenos entre as monitorações, impede a detecção das más condições da rede, levando o sistema a operar à taxa mais baixa (figura 4.7(a)). Isso causa então uma sub-utilização dos recursos do ambiente.

Já nos testes com tamanho de janela igual a 500 (figura 4.7(c)), há uma demora para captar alterações na qualidade do ambiente e assim tomar uma providência. Porém, observando a variação da taxa de envio dos testes com o tamanho da janela igual a 100 (figura 4.7(b)), pode-se perceber que, nesse caso, o fluxo conseguiu encontrar a melhor

taxa de envio mais rapidamente. Esse resultado justifica a escolha do tamanho de janela de monitoração igual a 100. Novamente, a solução aqui assumida é a mais simples e, provavelmente, alguma técnica adaptativa para a seleção do tamanho da janela, embora mais complexa, leve a resultados mais precisos.

As próximas seções irão apresentar resultados de testes feitos com os valores definidos nas seções 4.3.2.1, 4.3.2.2 e 4.3.2.3.

4.3.2.4 Comparação de Resultados

Esta seção apresenta uma comparação dos resultados obtidos nos testes com o *framework* de suporte a QoS em redes *Mesh*, com o INSIGNIA e sem nenhum mecanismo de qualidade. Os testes foram realizados nos quatro cenários, com taxas máximas de envio de 10fps, 15fps, 20fps, 25fps e 30fps. Os resultados apresentados são de *jitter*, perda de pacotes e atraso médio fim-a-fim, calculados com todos os pacotes enviados durante a simulação. Cada ponto em cada curva representa a média de 10 experimentos simulados com *seeds* diferentes. Em todos os testes, o *framework* apresentou melhores resultados do que as outras duas execuções.

Comparando os resultados do cenário 1 (figura 4.8) com o cenário 2 (figura 4.9), pode-se perceber que a nossa proposta do *framework* no segundo cenário apresentou uma pequena melhora nos resultados dos testes na taxa de 30fps. Essa melhora também pode ser notada nos testes sem nenhum mecanismo de QoS inserido. Isso acontece porque nesse cenário, apesar do nó móvel estar em movimento, ele se move em direção ao nó destino, o que diminui, assim, o número de saltos no caminho. Já nos testes com o INSIGNIA, os resultados de *jitter* e atraso fim-a-fim foram piores no segundo cenário, enquanto a porcentagem de perda de pacotes se manteve praticamente a mesma. No INSIGNIA, a movimentação do nó cliente juntamente com a injeção de tráfego extra provocada pelos relatórios periódicos de QoS fez com que a situação da qualidade do fluxo de vídeo fosse piorada.

Um resultado interessante é que o INSIGNIA apresentou resultados piores do que a execução dos cenários sem nenhum mecanismo de suporte a QoS. A causa desse resultado está na própria proposta do INSIGNIA. O mecanismo de envio periódico de relatórios de QoS cria um tráfego extra na rede que atrapalha a transmissão do fluxo contínuo. Em uma rede *Mesh*, todo o tráfego entre os nós sempre passa pelo mesmo caminho. Como a proposta de rede que está sendo trabalhada não utiliza multi-rádios em cada nó *Mesh*, a transmissão e recepção de pacotes em um nó sempre interfere no nó vizinho. Assim,

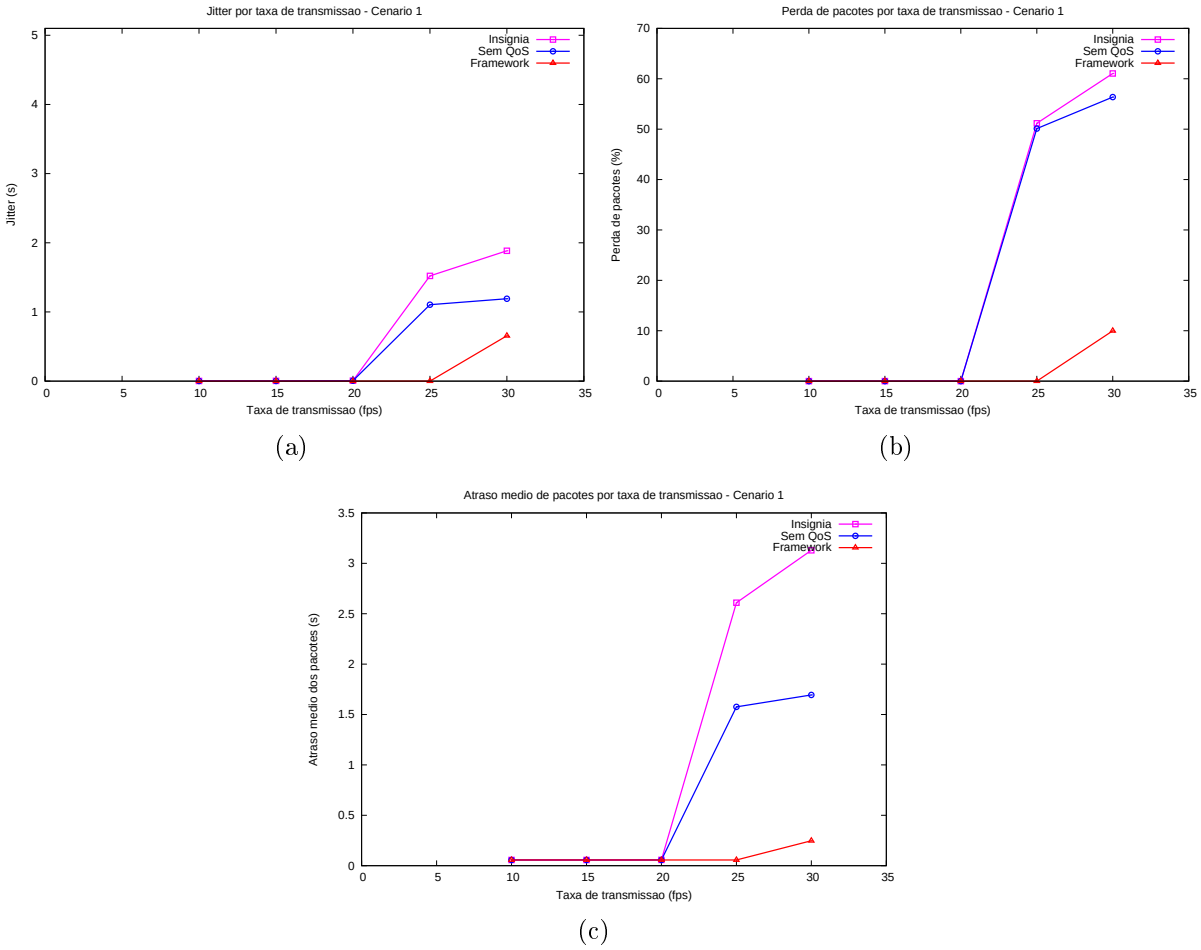


Figura 4.8: Comparação de resultados do cenário 1 - (a) *Jitter*; (b) Perda de pacotes; (c) Atraso médio fim-a-fim.

quanto mais tráfego extra for inserido na rede, maior será a interferência na recepção e transmissão dos nós.

O INSIGNIA foi projetado para redes *Ad Hoc*, na qual normalmente mais nós apresentam mobilidade; por causa dessa característica, a topologia da rede muda constantemente. Essa mudança faz com que novos caminhos entre nós origem e destino sejam encontrados, possibilitando, assim, que o tráfego extra seja dispersado.

Por essa razão é que o *framework* de suporte a QoS envia uma mensagem de QoS do destino para a origem apenas quando os limites de *jitter* e perda de pacotes apresentam valores acima do desejável. Isso permite que o caminho entre os nós fique livre para a transmissão dos pacotes do fluxo contínuo, sem que os nós intermediários tenham que disputar mais do que o necessário o acesso ao meio.

Nos cenários 3 (figura 4.10) e 4 (figura 4.11), o tráfego CBR extra foi inserido em um

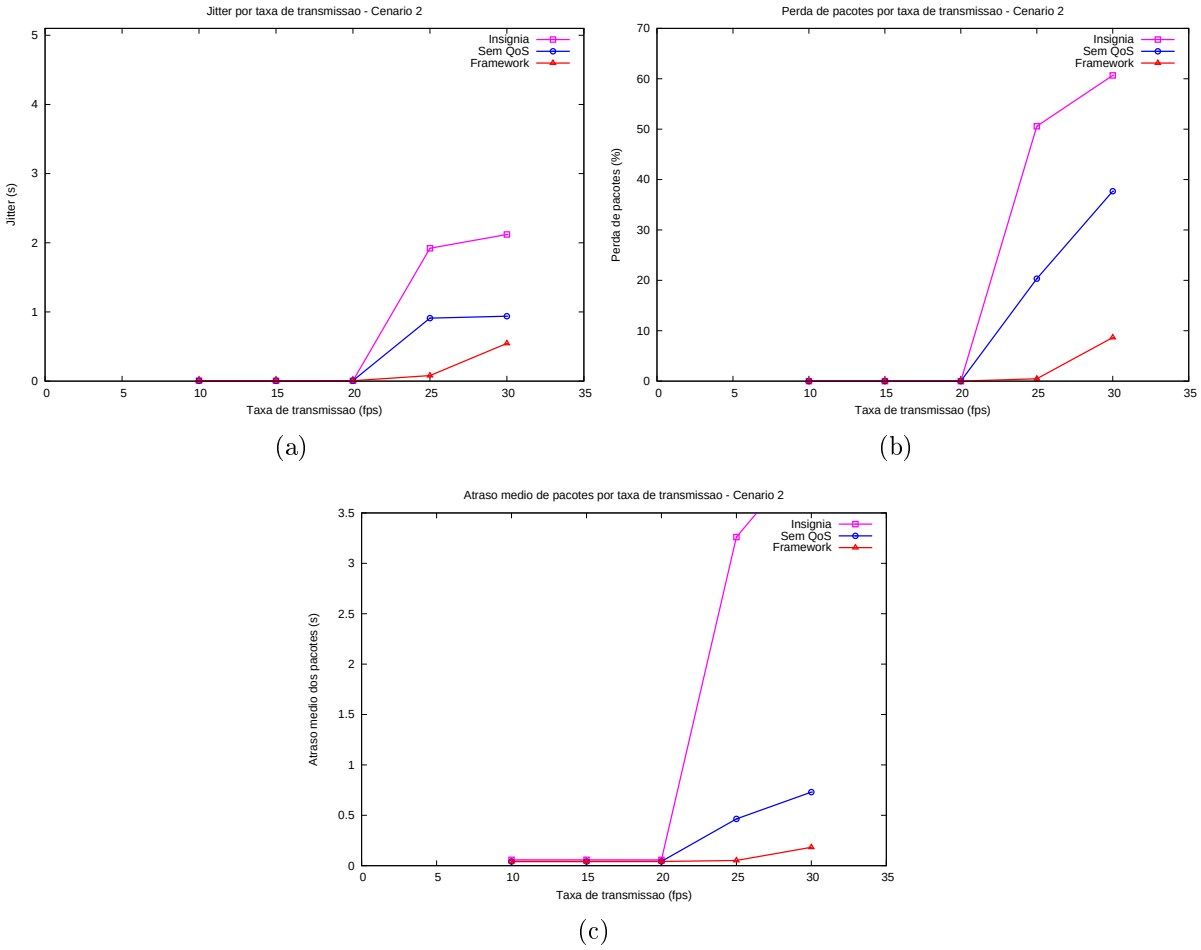


Figura 4.9: Comparação de resultados do cenário 2 - (a) *Jitter*; (b) Perda de pacotes; (c) Atraso médio fim-a-fim.

dos roteadores para simular o tráfego gerado por computadores que estão utilizando a rede *Mesh* através da porta LAN. Nesses cenários, o *framework* apresentou uma leve piora, em comparação com os cenários sem essa interferência. Ainda assim, ele apresentou resultados melhores do que o INSIGNIA.

Os resultados dessas comparações foram muito satisfatórios. Com eles foi possível perceber que a proposta do INSIGNIA de enviar periodicamente relatórios de QoS do destino para a origem não se aplica a redes *Mesh*, pois causa congestionamente nos enlaces. A proposta do nosso *framework* de monitorar a qualidade da transmissão apenas no nó cliente se mostrou eficiente e escalável. Dessa forma, o nó origem não precisa monitorar todos os fluxos existentes e é informado pelo destino apenas quando precisa acionar o sistema, de adaptação. Os testes do protótipo que serão apresentados no capítulo 6 ajudarão essa validação e permitirão a análise dos resultados visuais da ação do *framework*.

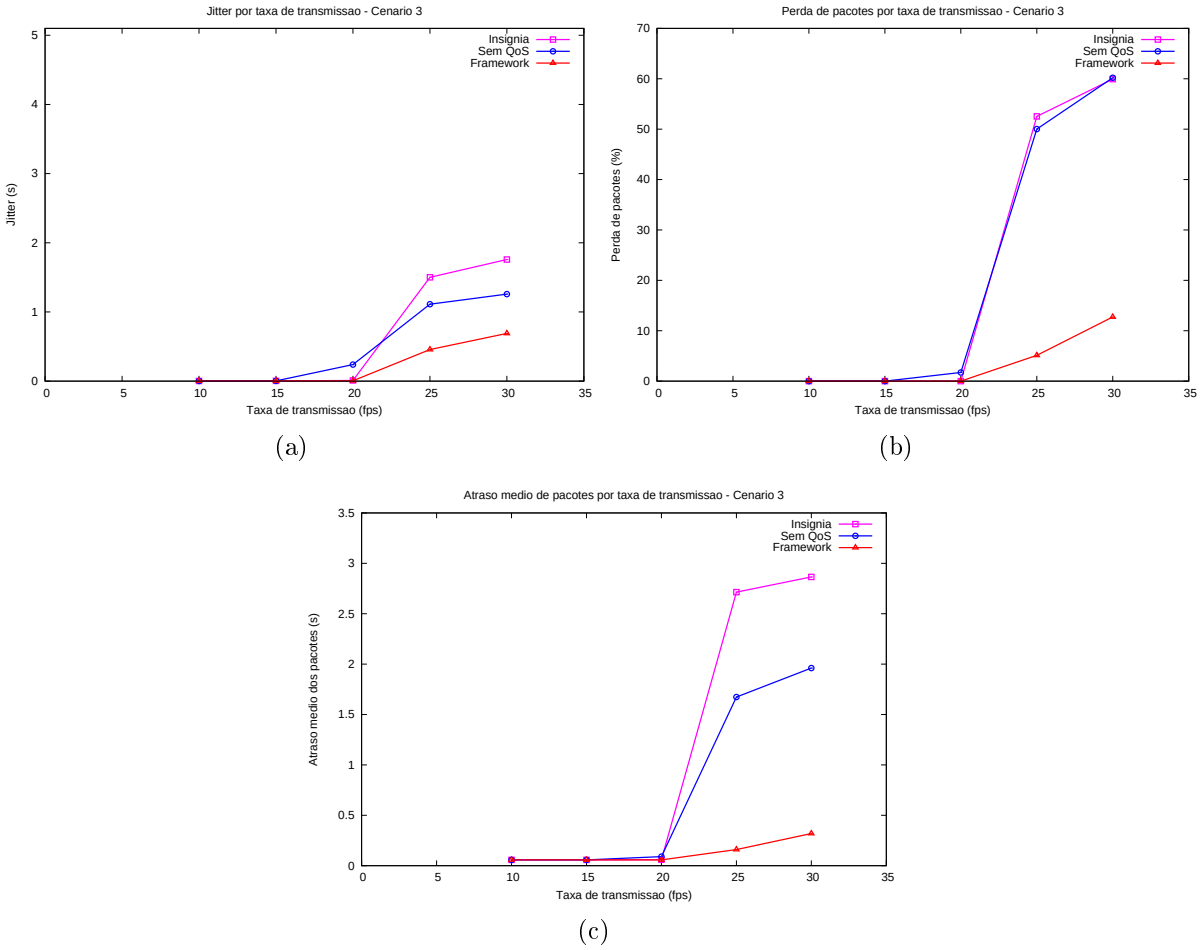


Figura 4.10: Comparação de resultados do cenário 3 - (a) *Jitter*; (b) Perda de pacotes; (c) Atraso médio fim-a-fim.

4.3.2.5 Resultados do *Framework* ao Longo da Simulação

Esta seção irá apresentar os resultados de *jitter* e de perda de pacotes ao longo da simulação, a forma como esses resultados influenciaram o acionamento do processo de adaptação, diminuindo ou aumentando a taxa de envio e, consequentemente, a maneira como essa mudança de taxa afetou a vazão no nó cliente. A figura 4.12 apresenta os resultados do cenário 2, com taxa de envio mínima igual a 10fps e máxima igual a 25fps.

Os gráficos 4.12(a) e 4.12(b) apresentam os limites máximo e mínimo de *jitter* e perda de pacotes definidos nos testes. Quando um dos dois requisitos fica acima do limite máximo, o *framework* aciona a adaptação, o que diminui a taxa de envio. Quando os valores de *jitter* e de perda de pacotes ficam abaixo do limite mínimo, a taxa de envio pode então ser aumentada.

Nesse cenário 2, o fluxo sofreu uma adaptação no início da simulação, pois o nó cliente

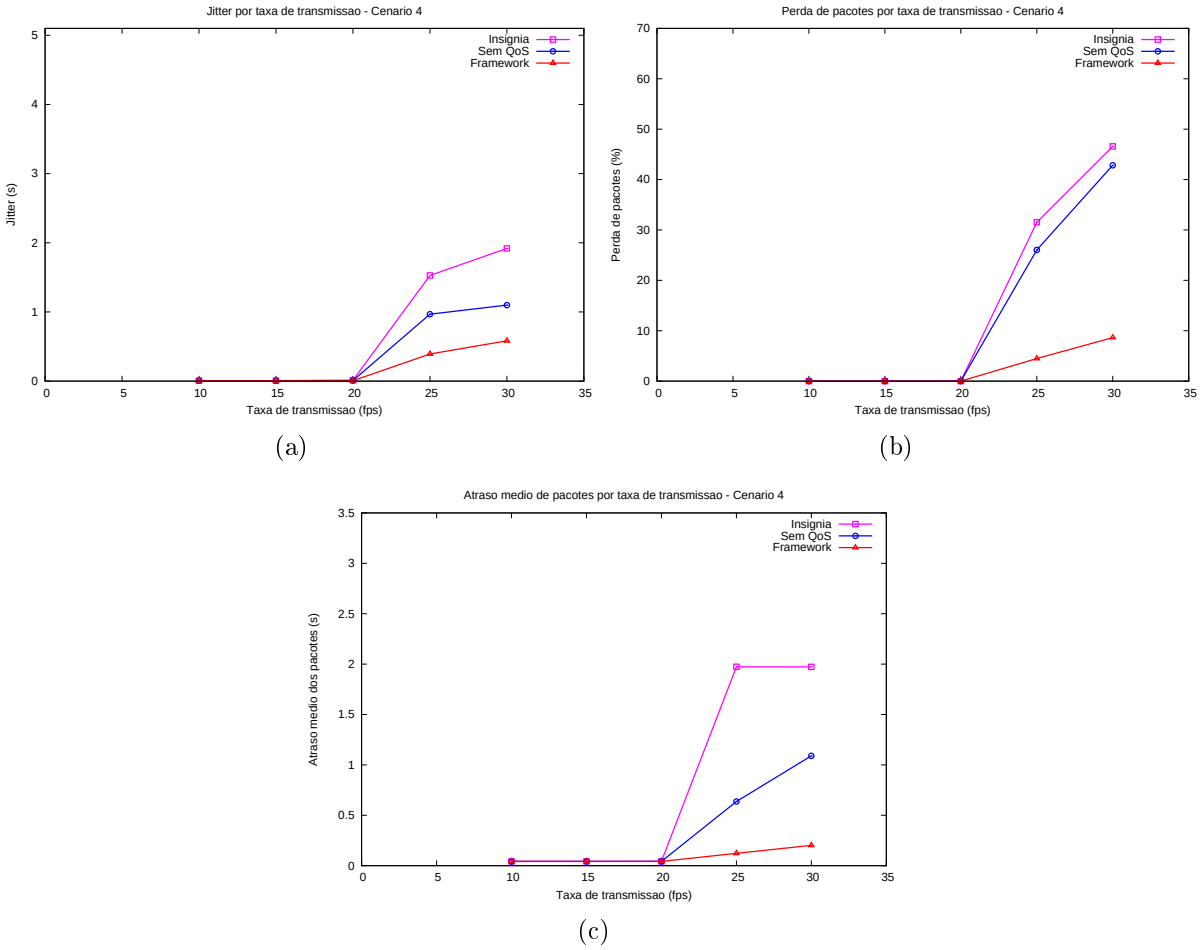


Figura 4.11: Comparação de resultados do cenário 4 - (a) *Jitter*; (b) Perda de pacotes; (c) Atraso médio fim-a-fim.

estava se movendo e era preciso encontrar uma nova rota. Como a velocidade era baixa, a transmissão conseguiu se estabilizar novamente. A vazão do fluxo acompanhou a taxa de envio.

Os gráficos dos cenários 3 e 4 apresentaram resultados interessantes no que diz respeito à procura pelo "nível ótimo". Neles é possível observar que, após a injeção de um tráfego extra na rede, a taxa de envio que apresenta melhor desempenho não é mais a de 25fps, como nos testes anteriores, mas sim a de 20fps.

A figura 4.13 apresenta os resultados do *jitter*, da perda de pacotes e da vazão no cenário 3. Esse cenário possui características semelhantes às do cenário 1, pois em ambos o nó cliente se encontra parado na extremidade oposta ao servidor na rede. Contudo, a cena 3 apresenta um tráfego extra nos roteadores.

Nesses gráficos, é possível observar o algoritmo que procura a melhor taxa de envio

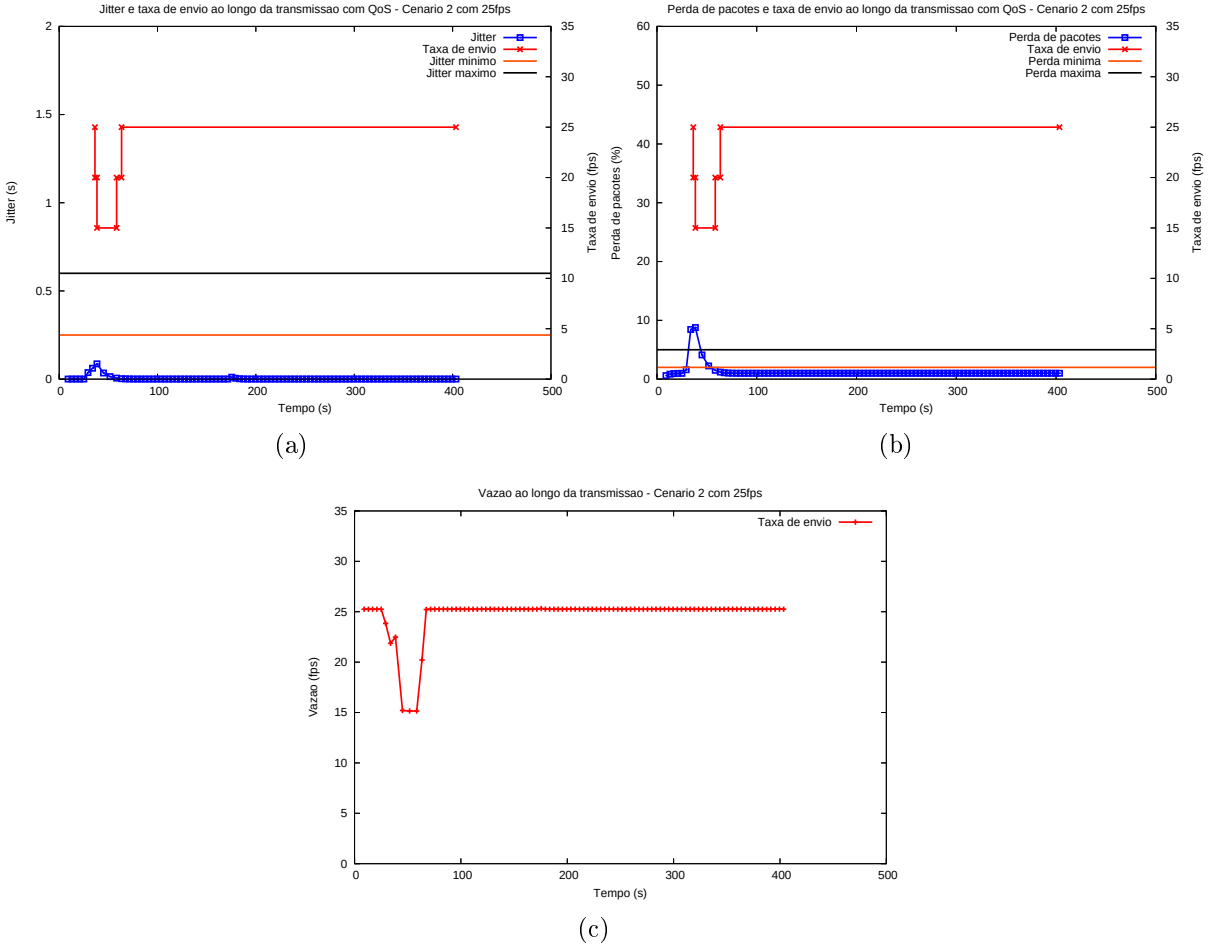


Figura 4.12: Resultados do cenário 2 com 25fps de taxa de envio máxima - (a) *Jitter* e mudança de taxa de envio; (b) Perda de pacotes e mudança de taxa de envio; (c) Vazão ao longo da simulação.

agindo. Em torno de 150s, o *framework* poderia ter aumentado a taxa de envio dos pacotes, já que tanto o *jitter* como a perda de pacotes se encontram abaixo do limite mínimo. Mas, como já tinha estado duas vezes no nível acima (25fps) e não tinha conseguido manter qualidade na transmissão, o *framework* então optou por manter a taxa um nível abaixo.

A mesma situação acontece no cenário 4 (figura 4.14), que pode ser comparado com os resultados do cenário 2. Novamente, por causa do tráfego extra nos roteadores, a taxa de envio com qualidade se manteve um nível abaixo do teste anterior.

Como foi possível perceber, a evolução do *framework* durante as simulações seguiu a proposta descrita. No capítulo 6, será feita uma comparação entre os resultados das simulações e dos testes realizados com o protótipo.

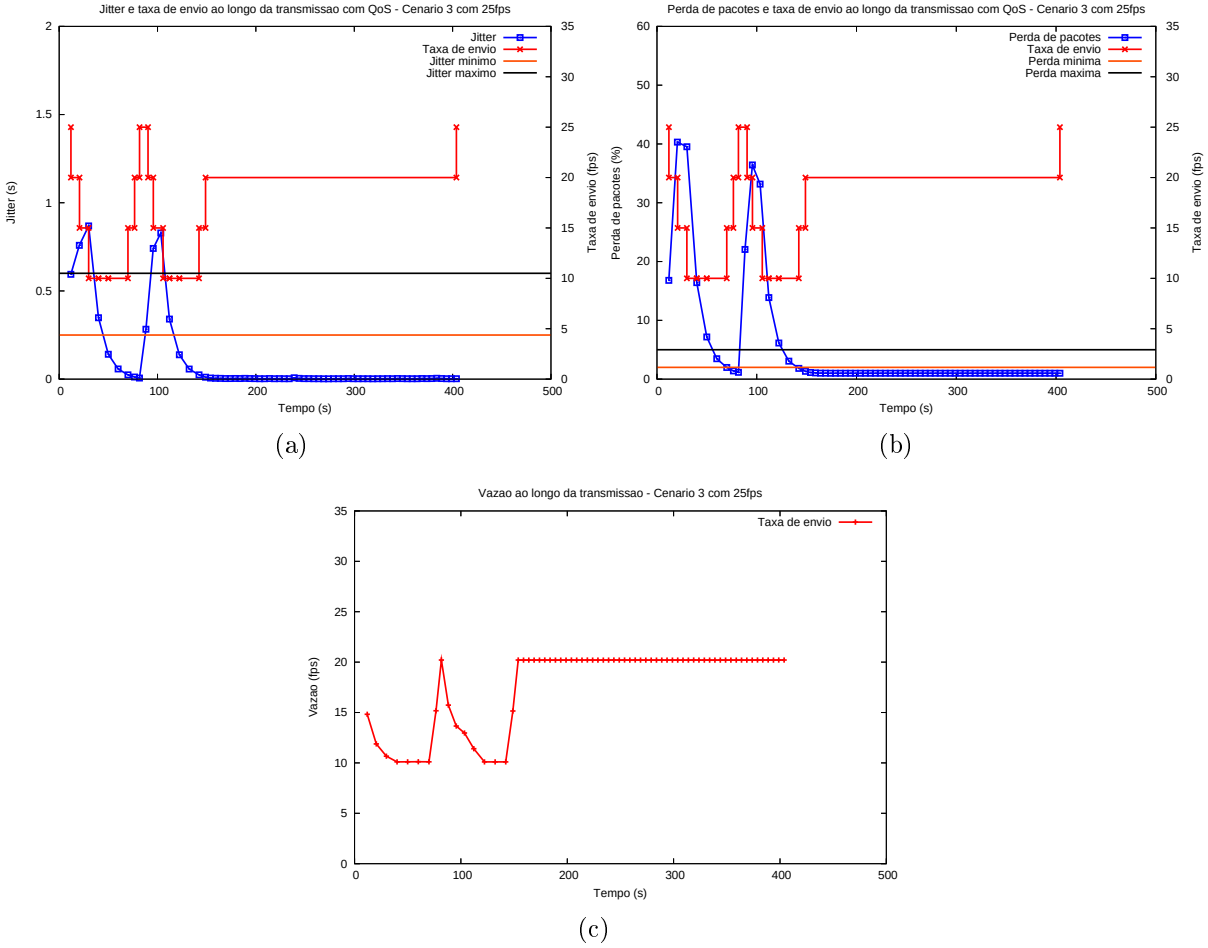


Figura 4.13: Resultados do cenário 3 com 25fps de taxa de envio máxima - (a) *Jitter* e mudança de taxa de envio; (b) Perda de pacotes e mudança de taxa de envio; (c) Vazão ao longo da simulação.

4.4 Conclusão

O estudo preliminar apresentado neste capítulo serviu como base para encontrar defeitos na proposta desenvolvida e valores adequados para os parâmetros do *framework*. Além dos testes descritos aqui, antes da conclusão da proposta para o *framework*, foram feitos rápidos testes para testar cada mecanismo em separado.

As comparações com o INSIGNIA e com testes em que não se tinha nenhum mecanismo de QoS envolvido deram a indicação de que a proposta para o *framework* de suporte a QoS em redes *Mesh* apresenta idéias interessantes, e que combinam com as características desse tipo de rede.

Após a conclusão desse estudo preliminar, foi possível então passar para a fase de implementação do protótipo. O próximo capítulo irá descrever esta fase, bem como as

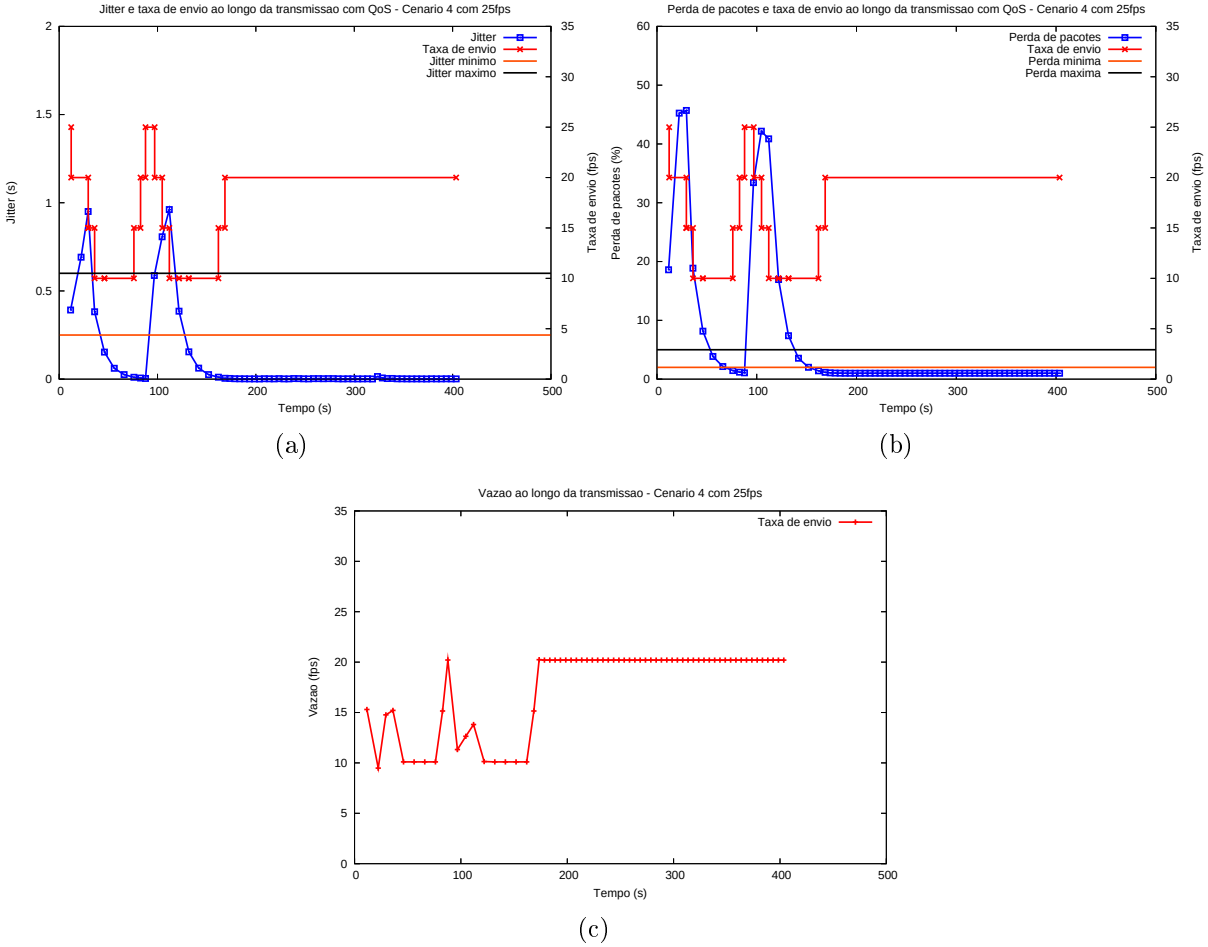


Figura 4.14: Resultados do cenário 4 com 25fps de taxa de envio máxima - (a) *Jitter* e mudança de taxa de envio; (b) Perda de pacotes e mudança de taxa de envio; (c) Vazão ao longo da simulação.

ferramentas e os dispositivos utilizados.

Capítulo 5

Implementação do Protótipo

5.1 Introdução

Atualmente, é possível encontrar na literatura muitas pesquisas, na área de redes sem fio, que implementam e testam suas propostas apenas em ambientes de simulação. O objetivo deste trabalho é implementar um protótipo da proposta apresentada no capítulo 3. Essa implementação permitirá a comparação entre os resultados em um ambiente real e aqueles da simulação. Conseqüentemente, também será possível descobrir algumas limitações da tecnologia hoje existente.

Este capítulo é dividido da seguinte forma: a seção 5.2 descreve a configuração dos dispositivos utilizados para os testes; a seção 5.3 apresentará as ferramentas utilizadas para a implementação do protótipo, tanto nos nós cliente e servidor como nos roteadores sem fio; a seção 5.4 relata os problemas encontrados durante a implementação e as soluções adotadas, bem como os passos da implementação em si.

5.2 Configuração

Para formar o *backbone* da rede *Mesh* foram utilizados roteadores modelo WRT54G e WRT54GS da *Linksys*, com padrão 802.11g. Cada roteador teve o seu *firmware* original substituído pelo sistema operacional OpenWRT, como será detalhado na próxima seção.

O dispositivo móvel utilizado como cliente foi um laptop *Latitude* 120L da *Dell*, com processador *Pentium* M 740 (1,73 GHz) e sistema operacional *Windows XP Home Edition*, SP2. O *desktop* utilizado como servidor foi um *Pentium* 4 (2,80GHz) com 512MB de memória RAM e sistema operacional *Windows XP Home Edition* com SP2.

5.3 Ferramentas

O *framework* de suporte a QoS é dividido em duas partes distintas que trabalham em conjunto: os roteadores que formam o *backbone* da rede sem fio são responsáveis por reservar largura de banda para os fluxos multimídia e por dar prioridade a eles; já as extremidades da transmissão realizam medições periódicas dos requisitos de QoS e acionam o processo de adaptação, caso seja necessário. Por causa dessa diferença, foram utilizadas ferramentas distintas para a implementação de cada parte.

Primeiramente, esta seção irá apresentar o ambiente instalado nos roteadores *wireless Linksys*, que teve seu sistema operacional original trocado pelo OpenWRT, que é um SO baseado em Linux. Para a implementação do mecanismo de reserva de recursos, foi utilizada a ferramenta *Traffic Control*, encontrada no OpenWRT e em diversas versões do Linux. Em seguida, será feita uma breve apresentação do ambiente de programação escolhido para as partes cliente e servidor, o *Microsoft Visual Studio .NET*.

5.3.1 OpenWRT

O OpenWRT pode ser descrito como uma distribuição Linux desenvolvida para dispositivos com recursos limitados. Ele tem como principal vantagem a de não precisar trabalhar de forma dependente do sistema operacional oferecido pelo fabricante do dispositivo, pois é possível a alteração ou instalação de novos componentes ao SO. Por ser um código livre, ele é recomendado para a pesquisa e o desenvolvimento de aplicações em roteadores sem fio; por essa razão é que foi escolhido para o desenvolvimento do projeto *ReMesh*.

O OpenWRT [OpenWRT 2005] possui uma interface amigável para a geração da imagem do *firmware* a ser instalado no dispositivo. Nesta interface pode ser escolhido o processador alvo e os pacotes que deverão estar presentes. Todas as ferramentas básicas do sistema operacional se encontram em um arquivo binário executável chamado *busybox*. Outras ferramentas são disponibilizadas de forma independente e podem ser instaladas separadamente.

Por ser uma distribuição Linux, o OpenWRT possui diversas aplicações e ferramentas originais desse sistema operacional. Dentre elas, a ferramenta *Traffic Control* é utilizada para fazer o gerenciamento de largura de banda e priorização de tráfego. A próxima seção irá explicá-la em detalhes.

5.3.2 *Traffic Control*

Como no Linux convencional, o *firmware* escolhido para os roteadores, o OpenWRT, também possui um sofisticado sistema para gerenciamento de largura de banda chamado *Traffic Control*, ou simplesmente *tc*. Esse sistema oferece suporte para vários métodos de gerenciamento, como classificação, priorização, divisão de recursos e limitação de tráfego. Nesta seção serão explicados esses métodos e os recursos disponibilizados por essa ferramenta, bem como o recurso que foi utilizado para o nosso *framework* de suporte a QoS.

A ferramenta *tc* faz parte do pacote *iproute2*, disponível para as versões do Linux 2.2 e acima, e também para o OpenWRT. O *site* [LARTC 2005] apresenta uma comunidade que dá suporte ao uso desse pacote e das ferramentas nele existentes. Nesse *site* também é obtido o documento [Hubert et al. 2003], muito importante para o entendimento da ferramenta *tc* e de seus métodos.

O *tc* permite criar disciplinas de filas (*Queueing Disciplines* ou *qdisc*) para o gerenciamento de largura de banda em um nó. Tais filas são criadas mais usualmente para trabalhar com os dados de saída de um nó, apresentando poucas funcionalidades para lidar com os pacotes que estão chegando.

Antes de começar a explicar as *qdisc*, é necessário esclarecer alguns termos que serão utilizados, para que o entendimento da ferramenta *tc* se torne mais claro. Uma *Queueing Discipline*, ou *qdisc*, é um algoritmo usado para o gerenciamento de filas de um dispositivo, filas que podem ser de entrada ou de saída de dados. Com uma *qdisc* é possível alterar o modo com que os dados que chegam ou que partem de um nó serão tratados. A *qdisc* associada ao dispositivo é chamada de *root qdisc*. As *qdisc* podem serem classificadas como *classless* ou *classful*. Uma *classless qdisc* é uma disciplina sem subdivisões internas que possam ser configuráveis. Já uma *classful qdisc* contém múltiplas classes internas.

As várias classes de uma *classful qdisc* categorizam os diferentes tipos de tráfego que passam por um dispositivo. Essas classes podem conter outras *qdisc* ou ainda outras classes. Assim, uma classe possui como "pai" uma *qdisc* ou outra classe. Uma classe folha é aquela que não contém filhos. Essa classe folha contém uma *qdisc* anexada a ela, que é responsável por enviar os dados provenientes dessa classe. Quando é criada uma classe, uma *fifo qdisc* é anexada a ela por padrão. Se uma classe filha for anexada a ela, essa *qdisc* é removida. Para as classes folhas, essa *fifo qdisc* pode ser substituída por qualquer outra *qdisc*.

Cada *classful qdisc* deve determinar para qual classe ela poderá enviar um pacote que

chegou ao dispositivo e deve ser transmitido na rede. Isso é feito por meio do uso de um classificador. Essa classificação pode ser feita com o auxílio de filtros. Um filtro contém um determinado número de condições que devem ser consultadas quando um pacote precisa ser classificado e, se tais condições coincidirem com os dados do pacote, o filtro aponta uma classe para o qual o pacote deve ser enviado. A figura a seguir apresenta uma visão geral do que acontece quando um pacote chega a um dispositivo.

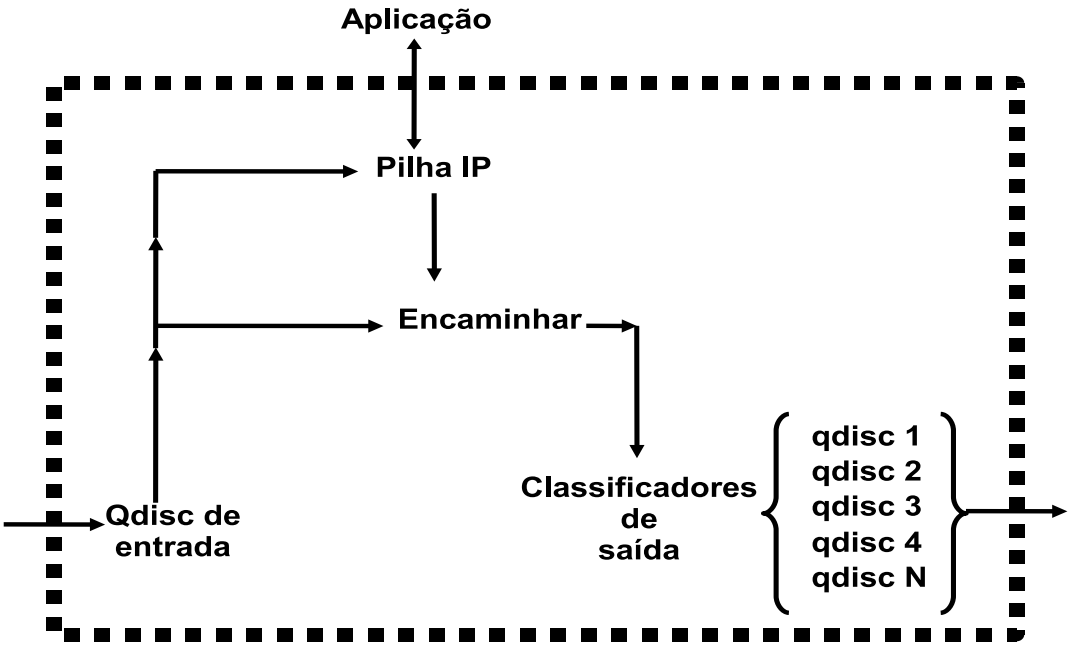


Figura 5.1: Disciplinas de filas nos dispositivos.

A seta à esquerda representa o tráfego que chega ao dispositivo através da rede. Este tráfego será inserido na fila de entrada, caso exista, onde serão aplicados os filtros. Esses filtros podem decidir que é preciso descartar alguns pacotes que estão entrando. Alguns pacotes poderão ser destinados para uma aplicação local e serem enviados para processamento na pilha IP. Os pacotes que não forem destinados para o dispositivo corrente devem ser encaminhados para o próximo nó. Assim, eles serão encaminhados para as filas de saída, que poderão conter diferentes disciplinas. Da mesma forma que na fila de entrada, os filtros também deverão ser consultados para saber em qual classe e em qual *qdisc* cada pacote deverá ser enfileirado.

As *classless qdisc* são as mais simples e permitem de forma limitada a configuração da disciplina por parte do usuário. As disciplinas mais utilizadas são *pfifo_fast*, *Token Bucket Filter* (TBF) e *Stochastic Fairness Queueing* (SFQ). Porém, apenas a SFQ será descrita nessa seção, já que esta será utilizada na implementação do protótipo. Mais informações sobre as outras *classless qdisc* podem ser encontradas em [Hubert et al. 2003].

A *Stochastic Fairness Queueing* (SFQ) é uma implementação simples da família dos algoritmos *"fair queueing"*. Na SFQ, uma conversação ou fluxo corresponde a uma sessão TCP ou a um *stream* UDP. Ela é chamada de *stochastic* porque não aloca uma fila para cada conversação, já que contém um número limitado de filas. Cada conversação é designada para uma dessas filas através de um algoritmo *hash*. O tráfego então é enviado como *Round Robin* para a rede, o que dá a cada fila uma chance de enviar seus dados.

Por causa do algoritmo *hash*, mais de uma conversação pode ir parar na mesma fila, o que faz reduzir a chance de cada uma dessas conversações enviar um pacote e diminuir, assim, a velocidade efetiva de transmissão. Para prevenir que essa situação se torne perceptível, o SFQ muda o algoritmo *hashing* freqüentemente, para que duas conversações que colidiram não fiquem muito tempo nessa situação. Na configuração da SFQ, o parâmetro *perturb* determina de quanto em quanto tempo, em segundos, a SFQ irá alterar o algoritmo *hash*.

As disciplinas acima citadas são aquelas denominadas *classless qdisc*, pois não possuem subdivisões configuráveis. As *classful qdisc* são as disciplinas que podem ter múltiplas classes dentro delas. Por permitirem a classificação dos pacotes, as *classful qdisc* são mais complexas, possuindo também mais parâmetros. As mais importantes são a *PRIO*, a *Class Based Queueing* (CBQ) e a *Hierarchical Token Bucket* (HTB). Novamente, será detalhada nesta seção apenas a *classful qdisc* utilizada na implementação do protótipo, a HTB. Mais informações sobre essas e outras *qdisc* podem ser encontradas em [Hubert et al. 2003].

A HTB é a abordagem mais apropriada quando se tem uma quantidade fixa de largura de banda que se deseja dividir entre diferentes fluxos, para dar a cada um uma largura de banda garantida, com a possibilidade de especificar o quanto de banda extra se pode pegar emprestado. Desta forma, é possível dizer que a HTB garante que a quantidade de serviço fornecida para cada classe é pelo menos a quantidade mínima requerida por ela. Quando uma classe utiliza uma quantidade menor do que a designada para ela, o restante, ou seja, o excesso de banda, é distribuído por outras classes que requisitaram serviço.

Para determinar qual classe deve processar um pacote, é usada uma "cadeia de classificadores" toda vez que a escolha precisa ser feita. Essa cadeia é composta pelos filtros anexados à *classful qdisc* que precisa decidir.

Quando uma HTB é criada como uma *root qdisc*, é preciso especificar qual será a classe chamada de *default*. Esta classe receberá todos os pacotes que não forem encaixados em algum filtro existente. A HTB, como uma *classful qdisc*, possui parâmetros para configurar

suas classes. O parâmetro ***rate*** determina qual será a largura de banda reservada para uma classe. Isso significa que todos os pacotes que forem classificados para aquela classe terão no mínimo uma quantidade ***rate*** de largura de banda garantida. O parâmetro ***ceil*** determina qual a largura de banda máxima que uma classe pode utilizar. Isso significa que, se os parâmetros ***rate*** e ***ceil*** forem iguais, a classe não poderá pegar banda emprestada de outra classe. Para que o empréstimo seja possível, é preciso que o parâmetro ***ceil*** seja maior do que o ***rate***.

As classes também podem ter prioridades diferentes. Essas prioridades afetam a forma como o excesso de largura de banda será distribuído entre as classes "irmãs". A regra é que, para as classes com prioridades mais altas, seja oferecido o excesso de banda primeiro. Porém, as regras dos parâmetros ***rate*** e ***ceil*** ainda são mantidas. As classes com prioridades mais baixas ou não serão contempladas com alguma banda extra, ou receberão um fatia pequena. Isso poderá causar atraso em seus pacotes.

O sistema de filtros da ferramenta *tc* dispõe de um variedade de possibilidades para classificação de pacotes. Basicamente, a classificação pode ser feita através da verificação dos campos do pacote, como, por exemplo, o cabeçalho IP, através das rotas existentes na tabela de rotas ou ainda através das marcações feitas por um *firewall* no pacote.

De acordo com [Hubert et al. 2003], a classificação mais utilizada é a primeira, que classifica de acordo com os valores dos campos do pacote. Esse tipo é chamado de filtro u32. Ele possui um formato simples, que consiste de dois campos: o seletor e a ação. Os seletores fazem a comparação do cabeçalho IP do pacote que está sendo processado com as diversas regras existentes, até que a primeira combinação seja encontrada. Cada regra de filtro está associada a uma ação que deve ser executada. Uma ação pode ser, por exemplo, o direcionamento do pacote para uma classe definida. Na seção que descreve a implementação do protótipo serão apresentados os filtros utilizados.

5.3.3 Microsoft Visual Studio .NET

Um dos objetivos da implementação do protótipo é que ele pudesse ser usado por dispositivos *off-the-shelf*, sem necessitar de alteração das configurações originais. Além disso, os dispositivos móveis disponíveis para os testes, como *Laptops* e *Pockets PC* da marca HP, utilizam o *Windows* como sistema operacional original. Por essas razões, foi escolhido o ambiente *Microsoft Visual Studio .NET*, que permite a implementação de aplicações tanto para *Pocket* como para *PCs* de grande porte em C# e Visual Basic, além das linguagens C++ e J# somente para *PC*. A linguagem escolhida foi o C# para os dois tipos de dispositivos.

O *Microsoft Visual Studio .NET* permite que o desenvolvedor apresente uma interface amigável para o usuário, principalmente para dispositivos de pequeno porte como *Pocket PC*. Porém, alguns problemas foram encontrados no uso deste ambiente, que serão relatados na seção 5.4.1.

5.4 Implementação do Protótipo

Para testar a implementação do protótipo do *framework*, foi utilizado o programa de videoconferência desenvolvido pelo aluno Robson Hilario da Silva [da Silva 2006], mestrando em Computação do Instituto de Computação da Universidade Federal Fluminense. O trabalho do aluno Robson propõe uma política de migração para dispositivos móveis em redes sem fio infraestruturadas e foi testado com dispositivos de pequeno porte.

Esta seção irá primeiramente apresentar os problemas encontrados na implementação e a maneira como foram ultrapassados. Em seguida fará uma descrição da implementação do *framework* tanto na parte dos roteadores como no cliente e no servidor.

5.4.1 Problemas Encontrados e Soluções Propostas

Apesar da parte cliente poder ser executada no *Pocket PC*, não foi possível realizar os testes com esse tipo de dispositivo. Até o fim da implementação e dos testes foi encontrada apenas uma versão do protocolo de roteamento OLSR para *Pocket PC*, desenvolvida por Carlos Calafate [Calafate 2002], mas que não é compatível com a versão do OLSR que é executada nos roteadores. Por essa razão, a implementação disponibilizada e os testes realizados utilizaram um *Laptop* com uma versão do OLSR compatível com a dos roteadores.

A idéia inicial do *framework* seria a de utilizar o campo Opções do cabeçalho IP para carregar o cabeçalho das mensagens com as informações importantes para os roteadores e para as extremidades. Isso permitiria que essas informações ficassem desacopladas dos dados, o que tornaria o *framework* mais transparente para a aplicação. Porém, não foi possível ter acesso às camadas mais baixas da mensagem no ambiente de desenvolvimento utilizado. Tanto no envio quanto no recebimento, as classes e métodos disponíveis apenas possibilitavam o acesso aos dados da mensagem, e não aos cabeçalhos do pacote, como o IP, TCP ou UDP.

A solução encontrada para esse problema foi colocar o cabeçalho do *framework* no

início de cada pacote transmitido. Cada quadro a ser enviado foi então fragmentado em pedaços de tamanho menor ou igual ao MTU de uma rede *Ethernet*, para que todos os pacotes que trafeguem nos roteadores tenham o cabeçalho do *framework* anexado ao início da mensagem. No cliente foi implementado um sistema de desfragmentação, que organiza e junta as partes para formar um quadro pronto a ser exibido.

O cabeçalho do *framework* é verificado pelos filtros que residem nos roteadores e desta forma é classificado para sua classe correspondente. O pacote que não tiver o cabeçalho do *framework* no início da mensagem será encaminhado para a classe Outros, pois não se encaixa como um fluxo multimídia com suporte a QoS.

Com o *driver* da câmera de vídeo disponibilizada para os testes não é possível alterar em tempo de execução a taxa de captura de quadros, para a implementação do mecanismos de adaptação proposto. Para facilitar e permitir que os testes apresentassem resultados coerentes, em vez de se utilizar um vídeo ao vivo, foi utilizado um vídeo gravado em arquivo e carregado para um *buffer* no servidor.

5.4.2 *Framework*

Como já foi citado, o *framework* se divide em duas partes: o mecanismo de reserva de recursos é implementado no *backbone* da rede *Mesh*, e as extremidades da transmissão executam o monitoramento da qualidade do tráfego e o sistema de adaptação. As próximas seções irão apresentar com detalhes cada uma dessas partes.

5.4.2.1 Roteadores

A seção 5.3.2 apresentou a ferramenta *Traffic Control* do OpenWRT, utilizada para fazer o gerenciamento da largura de banda nos roteadores e, assim, executar o mecanismo de reserva de recursos. Nesta subseção, será explicado como essa ferramenta foi utilizada e como o mecanismo de reserva de recursos funciona.

A disciplina de filas utilizada para montar as tabelas de reserva de banda foi a HTB, por ser a abordagem mais apropriada quando se tem uma quantidade fixa de largura de banda que se deseja dividir entre diferentes tipos de fluxos, dando a cada um desses tipos uma fatia garantida da banda. Cada classe que recebe os pacotes para ela encaminhados implementa o sistema de filas SFQ para organizar o envio das mensagens.

Para fazer a divisão da banda disponível nos roteadores entre os diferentes tipos de tráfego, para efeito de teste, foram criadas três classes HTB: uma classe para pacotes

de controle enviados pelo *framework* de suporte a QoS, uma classe para *streams* de vídeo e uma classe que serve para qualquer outro tipo de tráfego não prioritário. A classe de vídeo é dividida em outras quatro classes, que serão os níveis de adaptação disponíveis no *framework*. Como já foi dito, cada nível de adaptação representa uma taxa de transmissão. A figura 5.2 apresenta a configuração de cada roteador com suas *qdisc* e classes.

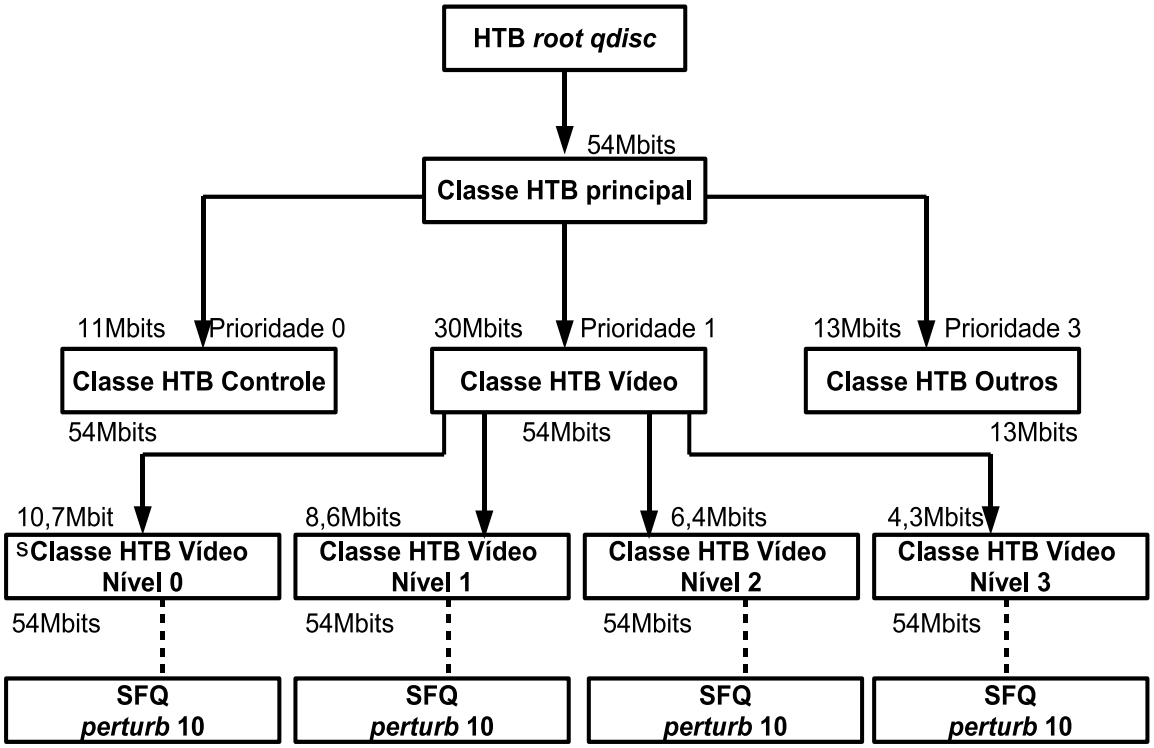


Figura 5.2: Árvore HTB montada para reserva de banda.

A largura de banda indicada na parte superior das classes corresponde à banda reservada. A banda indicada na parte inferior apresenta o valor máximo que a classe poderá pegar emprestado. É importante observar que a classe HTB Outros não pode pegar banda emprestada de suas classes irmãs, já que o valor máximo de banda é igual à quantidade reservada para ela. Porém, ela poderá emprestar banda que sobre para outra classe que necessitar.

Cada classe folha possui uma prioridade. As classes de vídeo possuem a mesma prioridade da classe pai. As prioridades afetam a forma como o excesso de largura de banda será distribuído entre as classes "irmãs". A regra é que, para as classes com prioridades mais altas, seja oferecido o excesso de banda primeiro. As classes com prioridades mais baixas ou não serão contempladas com alguma banda extra, ou receberão uma fatia menor.

O sistema de filas SFQ inserido em todas as classes folha tem seu parâmetro *perturb* igual a 10. Isso significa que, a cada 10 segundos, ele irá refazer o seu algoritmo *hash*,

redistribuindo os fluxos da classe entre as filas disponíveis.

Para que os pacotes sejam apontados para as suas respectivas classes, foram criados filtros que analisam os campos dos pacotes e fazem a combinação desses campos, comparando-os com os valores de uma tabela. Mais precisamente, eles verificam os campos do cabeçalho do *framework* e também o campo do cabeçalho IP que indica o protocolo de transporte utilizado. A tabela 5.1 indica os valores que cada campo deve apresentar para ser enviado para cada classe. Os pacotes que não se encaixarem em nenhum desses filtros, ou não tiverem o cabeçalho do *framework* no início da mensagem, serão classificados para a classe HTB Outros.

Classe	**Protocolo IP**	**Nível de adaptação**	**Tipo de dados do pacote**	**Requisição de qualidade**
Classe Vídeo Nível 0	UDP	0	Vídeo	Sim
Classe Vídeo Nível 1	UDP	1	Vídeo	Sim
Classe Vídeo Nível 2	UDP	2	Vídeo	Sim
Classe Vídeo Nível 3	UDP	3	Vídeo	Sim
Classe Controle	TCP	Acompanha o nível do fluxo de vídeo	Controle	Sim

Tabela 5.1: Valores esperados pelo filtro para indicação de cada classe do sistema.

Os pacotes de vídeo são enviados da origem para o destino utilizando o protocolo UDP. Já as mensagens de controle, tanto do *framework* quanto da aplicação cliente/servidor, são enviadas por TCP, por ser um protocolo confiável.

5.4.2.2 Cliente e Servidor

As extremidades da transmissão, o cliente e o servidor, não tomam conhecimento das reservas feitas nos roteadores. A ferramenta *tc* não distingue o quanto de banda efetivamente foi dado a cada fluxo de uma classe. A banda reservada para uma determinada classe está garantida para todos os pacotes que passarem por ela. Caso a taxa de transferência de uma classe exceda a banda reservada, ela irá pegar banda emprestada com suas classes irmãs. O que pode acontecer, caso muitos fluxos disputem banda em uma mesma classe, é que alguns pacotes se atrasem ou sejam descartados. Nesse caso, o cliente perceberá essa degradação e acionará o processo de adaptação, o que diminuirá a taxa de transferência.

Ao diminuir a quantidade de dados a ser transmitida, o fluxo também terá seu nível de adaptação alterado. Conseqüentemente, quando os pacotes daquele fluxo passarem pelos roteadores, eles serão encaminhados para a classe correspondente do seu novo nível. Isso faz com que a classe anterior utilize menos banda, favorecendo os outros fluxos que estão passando por ela.

O nó cliente realiza uma monitoração dos dados periodicamente. A cada 100 pacotes recebidos, ele verifica o *jitter* e a perda de pacotes nesse grupo. Se os valores medidos estiverem acima de um limite máximo definido pela aplicação, o processo de adaptação é acionado. Com isso, o nível de adaptação é incrementado, e um relatório de QoS é enviado para o servidor. Ao receber essa mensagem, o servidor percebe que precisa então diminuir a taxa de envio e alterar o nível de adaptação dos pacotes. Quando o primeiro pacote com o nível de adaptação alterado chegar ao cliente, esse então percebe que o relatório de QoS chegou com sucesso ao servidor e que o nível de adaptação foi alterado. Caso o cliente detecte que o relatório de QoS não chegou no servidor depois de um determinado tempo, ele então o envia novamente.

A monitoração é periódica. Caso seja detectado que o *jitter* e a perda de pacotes medidos apresentam valores abaixo do limite mínimo definido pela aplicação, o processo de adaptação é acionado novamente, o que faz o nível de adaptação diminuir. Mais uma vez, o cliente envia um relatório de QoS para o servidor avisando-o que pode aumentar a taxa de transferência. Ao receber este relatório, o servidor percebe que as condições da rede melhoraram e que pode enviar mais dados.

5.5 Conclusão

A implementação do protótipo do *framework* de suporte a QoS em redes *Mesh* permitiu descobrir problemas tecnológicos existentes atualmente que dificultam a implantação de algumas idéias propostas. A capacidade de processamento limitada dos *Pockets PC* utilizados e a dificuldade para encontrar algoritmos (como o OLSR) implementados para dispositivos *off-the-shelf* são alguns exemplos dos problemas encontrados.

O próximo capítulo irá apresentar os resultados dos testes do protótipo e fazer uma comparação com as simulações, para avaliar de forma prática o desempenho do *framework*.

Capítulo 6

Avaliação do Protótipo

6.1 Introdução

Após um estudo preliminar sobre a proposta do *framework* de suporte a QoS em redes *Mesh*, com os resultados positivos das simulações, foi implementado um protótipo na rede *Mesh* disponibilizada na Universidade através do projeto *ReMesh*. Este capítulo irá apresentar os testes realizados com o protótipo e os resultados obtidos.

Primeiramente serão descritos os cenários criados para os testes. Eles foram baseados nos cenários utilizados para as simulações. Em seguida, serão apresentados os gráficos com os resultados gerados.

Como já foi dito, a aplicação utilizada para teste foi uma cliente/servidor. O servidor é uma máquina *desktop* conectada à rede *Mesh* através da porta *Ethernet* de um dos roteadores, e o nó cliente é um *laptop* ligado à rede *Mesh* por uma interface sem fio. O protocolo OLSR é utilizado para o roteamento das mensagens entre os nós.

6.2 Cenários

Por causa da natureza das redes sem fio, alguns testes do *framework* no protótipo da rede *Mesh* não puderam ser feitos totalmente sem interferência, tendo em vista que outras redes sem fio presentes no *campus* também disputam o meio e que o simples movimentar de pessoas e elevadores já pode causar essas interferências. Mesmo assim, foram montados cenários com e sem a injeção de cargas extras na rede. A topologia dos cenários é mostrada na figura 6.1.

As linhas tracejadas indicam a rota escolhida pelo protocolo de roteamento OLSR

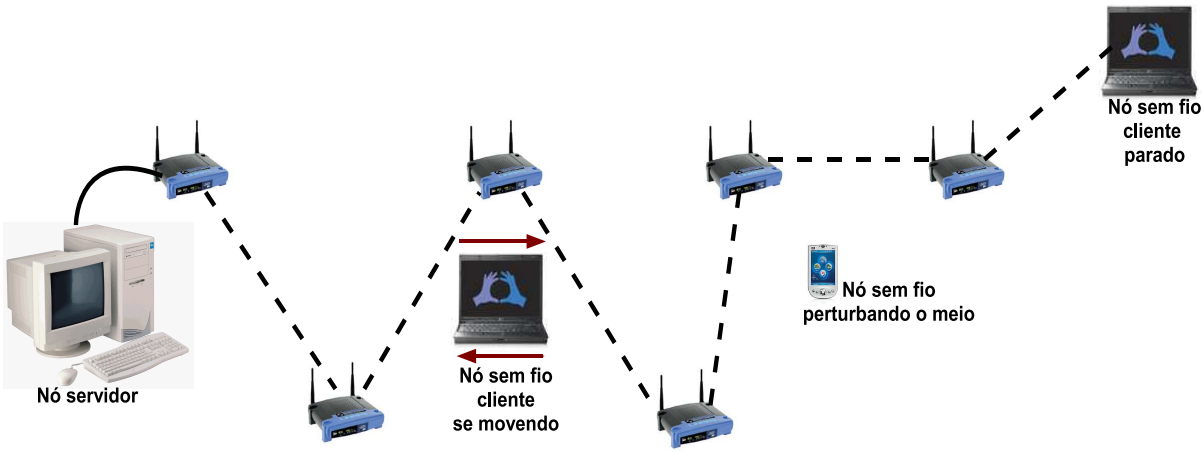


Figura 6.1: Cenário dos testes do protótipo de suporte a QoS.

para a troca de mensagens entre os roteadores. O servidor está ligado pela porta LAN a um dos roteadores *Mesh*. O cliente sem fio tanto pode ficar parado na extremidade oposta ao servidor, fazendo com que os dados trafeguem por 6 saltos da origem até o destino, quanto pode ficar se movendo, se aproximando ou se afastando do servidor. Nos testes com tráfego extra, existe mais um nó que injeta esse tráfego na rede, congestionando o meio.

Os cenários utilizados foram:

- **Cenário 1:** nó sem fio parado recebendo dados multimídia do servidor, sem tráfego extra;
- **Cenário 2:** nó sem fio se movimentando, inicialmente se afastando do servidor e depois voltando para perto dele, também sem tráfego extra;
- **Cenário 3:** nó sem fio parado recebendo dados multimídia do servidor, tendo interferência de tráfegos gerados por outro nó;
- **Cenário 4:** nó sem fio se movimentando, inicialmente se afastando do servidor e depois voltando para perto dele, tendo também interferência de tráfego gerado por outro nó;

Para cada um desses cenários foram executados oito experimentos, sendo quatro com a proposta de suporte a QoS e quatro sem o *framework*. A comparação dos resultados com e sem QoS tem como objetivo avaliar os benefícios obtidos com a proposta e confirmar os resultados da simulação. A taxa de envio de dados nos testes sem QoS foi mantida em

25fps, e nos testes com QoS variou entre 25fps e 10fps, dependendo do nível de adaptação em que o fluxo se encontrava.

Os testes em cada cenário não foram executados nos mesmos dias e horários. Logo, muitas vezes as diferenças entre cenários e experimentos se devem à não previsibilidade das condições do ambiente no momento dos testes. É importante ressaltar que, diferentemente da avaliação via simulação, testes com sistemas reais, em ambientes reais, apresentam grandes dificuldades em tentar oferecer situações semelhantes para avaliação de soluções.

Os parâmetros utilizados nos testes do protótipo do *framework* apresentaram os mesmos valores que foram definidos no ambiente de simulação:

- o peso α na amortização da medida do *jitter* e da perda de pacotes é de 40% para medições passadas e 60% para a medição corrente;
- o tamanho da **janela de monitoração** é igual a 100 pacotes;
- o ***jitter* mínimo** é de 0,25 segundos, e o ***jitter* máximo**, de 0,6 segundos;
- a **perda de pacotes mínima** é de 2,0%, e a **perda de pacotes máxima**, de 5,0%;
- o **algoritmo que procura o melhor nível de adaptação** verificará se o fluxo já esteve mais de duas vezes no "nível ótimo" antes de passar para ele. Caso se mantenha em um nível abaixo, fará isso durante 50 monitorações.

Todos os testes tiveram duração média de 20 minutos.

6.3 Resultados

As métricas utilizadas para medir o desempenho do *framework* foram *jitter* e perda de pacotes. A codificação usada para o vídeo foi a H263. Primeiramente será apresentada uma comparação dos resultados entre os testes com o *framework* e sem ele. Em seguida, serão apresentados gráficos que mostram os resultados das métricas de desempenho do *framework* ao longo dos testes.

6.3.1 Comparação de Resultados

A figura 6.2 apresenta os resultados para *jitter* e perda de pacotes em quatro experimentos realizados com o *framework* de suporte a QoS no cenário 1. Em seguida, a figura 6.3 indica

os resultados para as mesmas figuras de mérito, em quatro experimentos similares aos anteriores, para um sistema sem controle de QoS implementado.

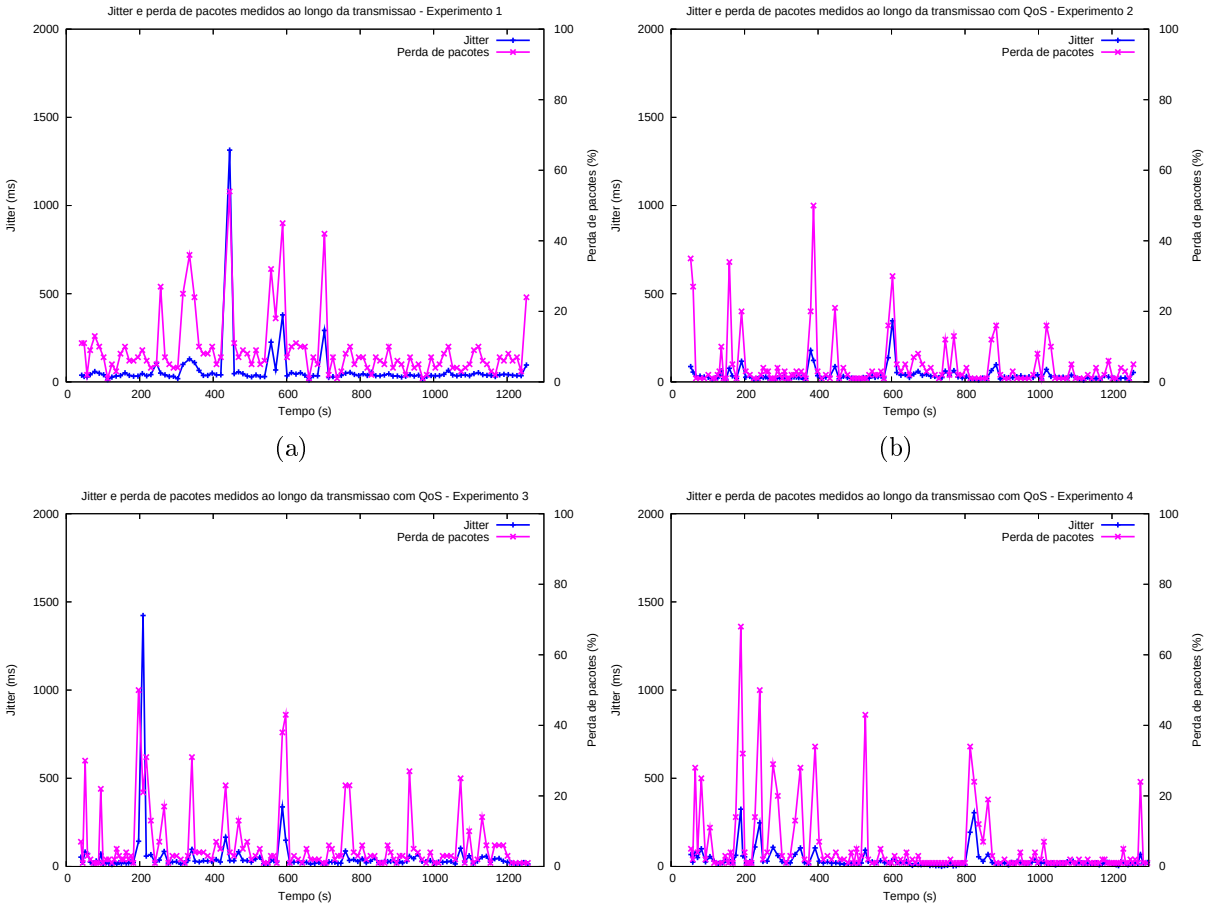


Figura 6.2: *Jitter* e perda de pacotes no cenário 1 com o *framework* - (a) Experimento 1; (b) Experimento 2; (c) Experimento 3; (d) Experimento 4.

Comparando esses gráficos, pode-se perceber que, já no cenário mais simples, o *framework* apresenta resultados bem melhores do que aqueles em uma transmissão sem nenhum mecanismo de QoS. Cada experimento foi executado em um momento diferente, e como não houve a injeção de tráfegos extra, supõe-se que os picos de perda e *jitter* foram causados por interferências inerentes ao ambiente.

Como já tinha sido observado nas simulações, é a perda de pacotes que mais sofre variações na transmissão de fluxos contínuos. Os testes sem QoS apresentam menos variações de *jitter*, pois não têm sua taxa de envio alterada em tempo de execução. Contudo, os picos em momentos de degradação são maiores do que nos testes com o *framework*. Mesmo com essa pequena variação do *jitter* nos testes sem QoS, os resultados visuais na reprodução do vídeo foram significativos, pois a porcentagem de perda de pacotes apresentou valores elevados ao longo dos testes.

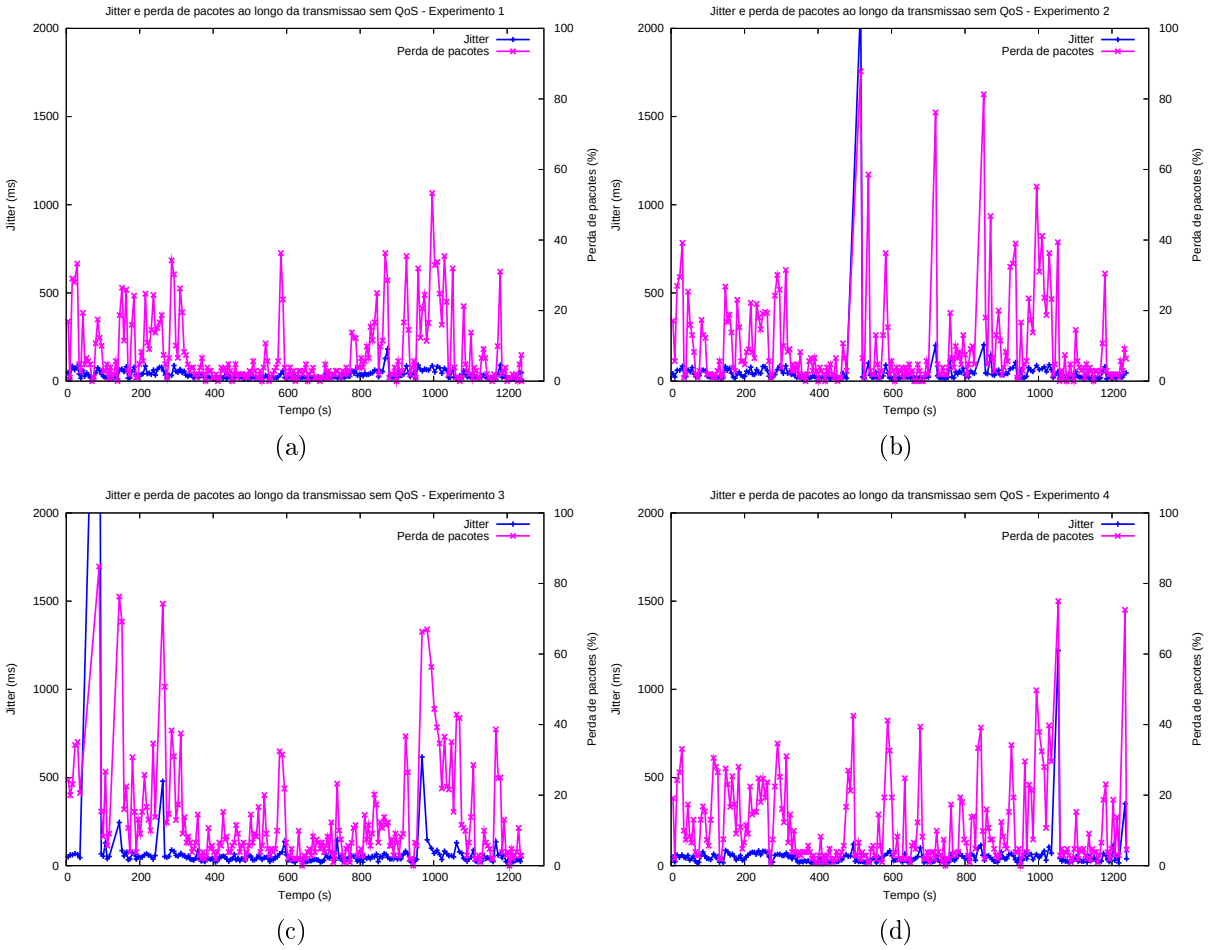


Figura 6.3: *Jitter* e perda de pacotes no cenário 1 sem QoS - (a) Experimento 1; (b) Experimento 2; (c) Experimento 3; (d) Experimento 4.

As figuras 6.4 e 6.5 mostram os resultados do cenário 2 com o *framework* e sem ele, respectivamente. No cenário 2, o cliente começa a se mover por volta do minuto 4 (240s), afastando-se do servidor. Mais ou menos no minuto 14 (840s), o nó destino chega à extremidade oposta da rede, fazendo com que os pacotes enviados pela origem passem por 6 saltos até chegar a ele. O nó móvel começa a fazer o caminho de volta para perto do nó origem em torno dos 960 segundos de execução, ou 16 minutos. Esses tempos da movimentação não são precisos em todos os experimentos, diferentemente do que ocorre no ambiente de simulação. A velocidade do nó cliente também não é sempre a mesma em todos os testes, já que a movimentação é feita por uma pessoa. Também nesse cenário não foram inseridos tráfegos extras.

Por causa da movimentação do nó cliente, os picos de perda de pacotes e de *jitter* em todos os experimentos foram maiores do que os testes com esse nó parado. Isso acontece porque, à medida que o nó cliente vai caminhando, a rota entre ele e o nó servidor precisa ser alterada. É importante lembrar que o OLSR adotado nos testes utiliza como métrica

para a escolha do melhor caminho a probabilidade de um pacote chegar ao nó vizinho com sucesso. Sendo assim, em um curto espaço de tempo a rota escolhida pode variar.

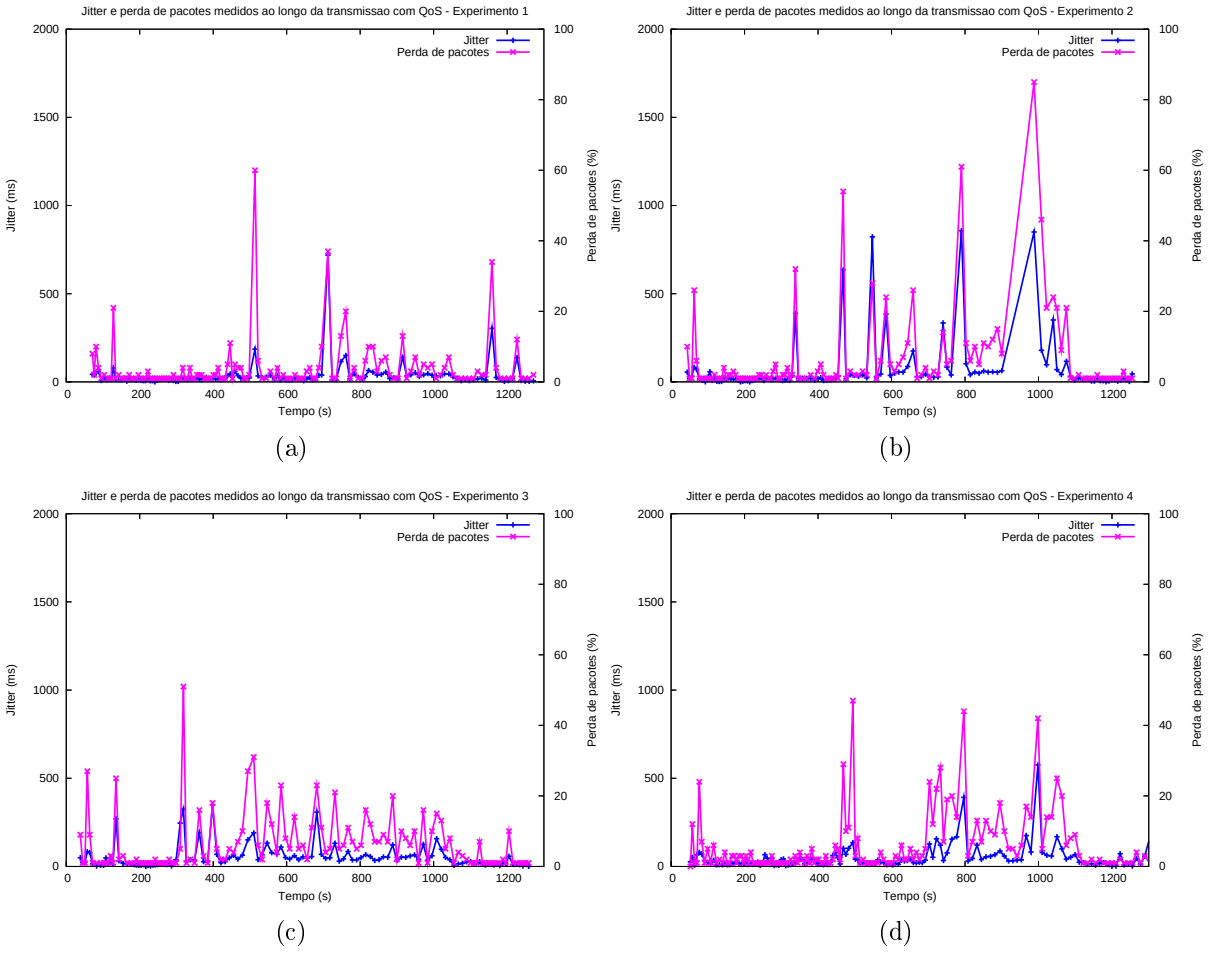


Figura 6.4: *Jitter* e perda de pacotes no cenário 2 com o *framework* - (a) Experimento 1; (b) Experimento 2; (c) Experimento 3; (d) Experimento 4.

Um ponto observado nos testes que é preciso levar em consideração é que o raio de alcance de cada nó roteador da rede *Mesh* não é constante ao longo do tempo, e não são iguais entre si. Assim, dependendo da atividade dos elevadores, da movimentação e da quantidade de pessoas que estavam nas salas e corredores durante os testes, a rota escolhida pelo OLSR não era sempre a mesma em todos os experimentos, nos mesmos locais. Isso porque o valor da métrica usada para a escolha da melhor rota também variava com o tempo.

A disposição dos roteadores da rede *Mesh* no prédio da Escola de Engenharia era tal que, algumas vezes, alguns roteadores que não se enxergavam passavam a se comunicar por apenas um salto. Isso também permitia que o caminho encontrado pelo OLSR se alterasse. Dependendo das condições do ambiente, a escolha dessa nova rota era rápida e, por isso, não causava tantos prejuízos à qualidade da transmissão. Outras vezes, a

descoberta do novo caminho causava pequenos períodos de desconexão.

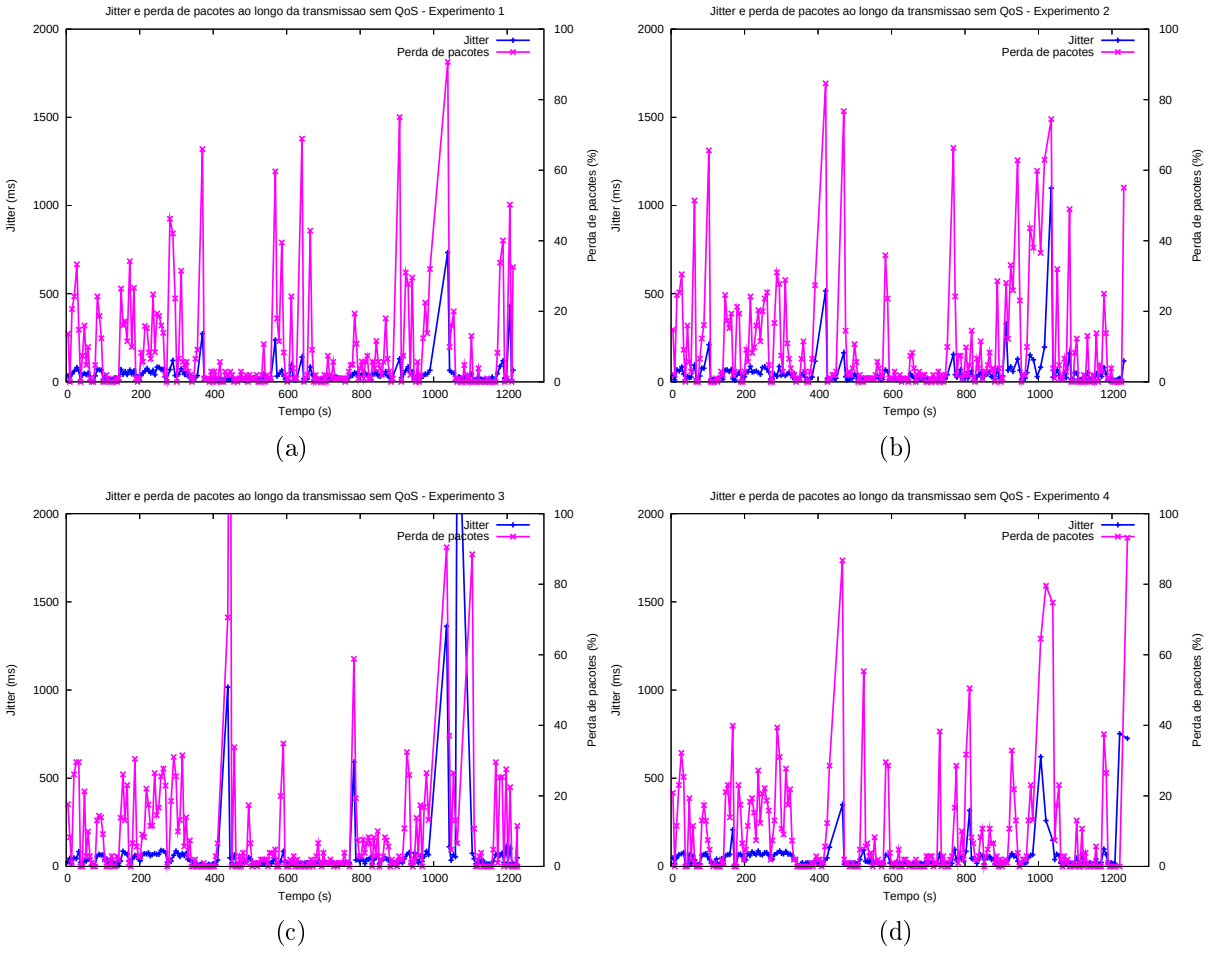


Figura 6.5: *Jitter* e perda de pacotes no cenário 2 sem QoS - (a) Experimento 1; (b) Experimento 2; (c) Experimento 3; (d) Experimento 4.

Comparando os resultados para os cenários 1 e 2, com e sem qualidade de serviço oferecida, foi possível perceber que o *framework* desempenhou um papel importante na estabilização da qualidade da transmissão. A melhora de desempenho também foi visível na reprodução do vídeo. Nos momentos de maior degradação do ambiente, o vídeo que estava sendo recebido sem QoS apresentou muitas falhas, além de ser perceptível a grande quantidade de pacotes que chegaram fora de ordem.

Os gráficos a seguir apresentam os resultados dos testes dos cenários 3 e 4, em que o ambiente sofreu perturbação provocada por dois tipos de tráfegos extra injetados na rede: (i) com 1 minuto de teste, foi disparada uma operação de FTP de um arquivo de 23Mbytes, e após dois minutos do fim da sua transferência, iniciou-se outro FTP similar - esse procedimento foi repetido até o fim da execução; (ii) aos 10 minutos, o nó cliente, que já estava recebendo um vídeo a partir da aplicação de videoconferência utilizada para os testes, fez uma requisição de um vídeo na Internet, que foi transmitido pela rede *Mesh*

sem requisição de QoS.

As figuras 6.6 e 6.7 apresentam os resultados de *jitter* e de perda de pacotes nos testes do cenário 3, com o *framework* e sem ele, respectivamente.

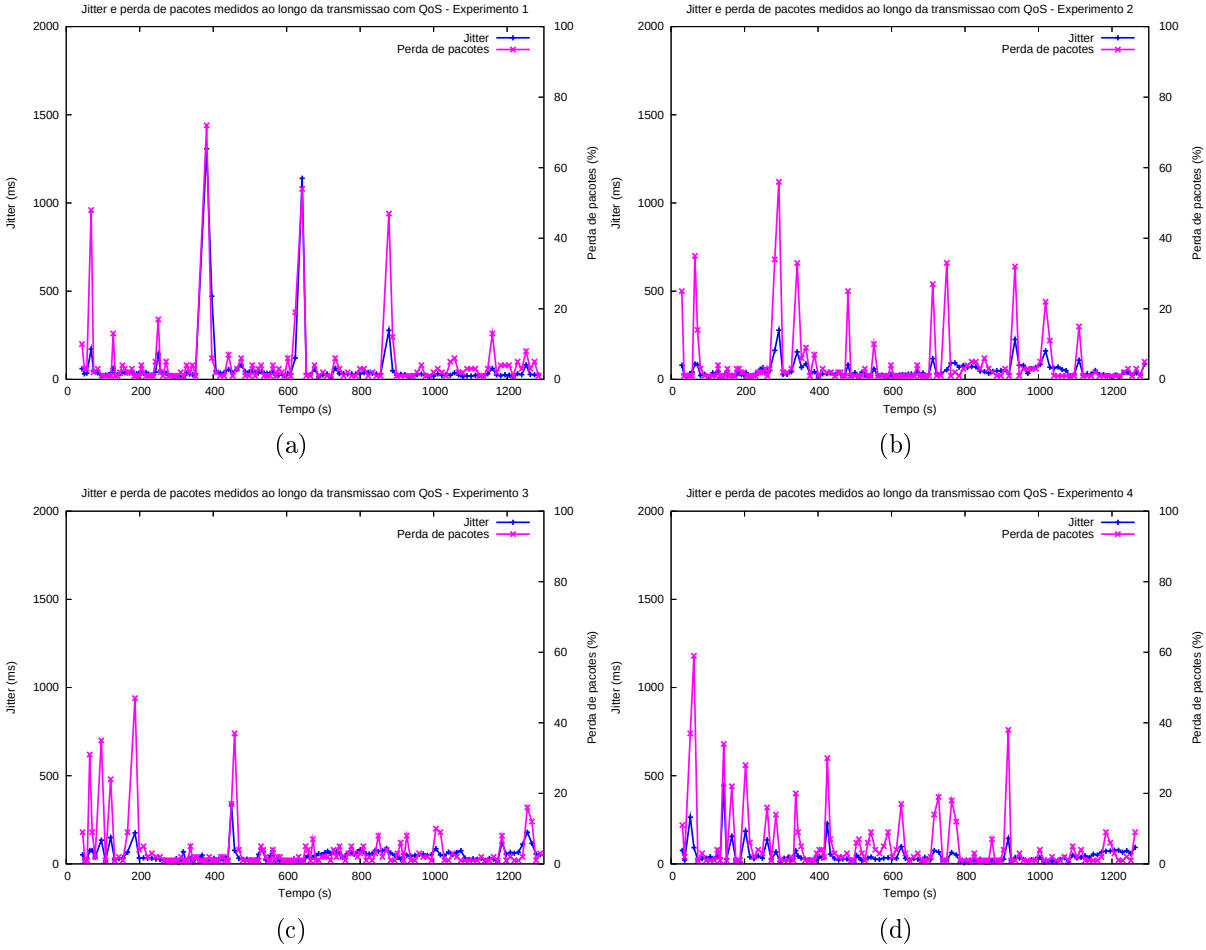


Figura 6.6: *Jitter* e perda de pacotes no cenário 3 com o *framework* - (a) Experimento 1; (b) Experimento 2; (c) Experimento 3; (d) Experimento 4.

Os resultados mais interessantes não podem ser vistos nos gráficos e foram obtidos a partir da observação da qualidade dos dois vídeos requisitados pelo nó cliente: o vídeo recebido pela aplicação de videoconferência e o vídeo requisitado na Internet. Nos testes sem QoS, nos quais os dois tráfegos estavam disputando a mesma banda, da classe HTB Outros, a visualização dos dois vídeos ficou prejudicada. Como foi explicado, a classe HTB Outros, que recebe os pacotes de fluxos sem QoS, tem uma quantidade reduzida de largura de banda reservada para si. Além disso, não é permitido a ela pegar banda emprestada com as outras classes. Assim, como os dois fluxos de vídeo não requisitaram suporte à qualidade de serviço, um prejudicou a transmissão do outro quando foram requisitados ao mesmo tempo.

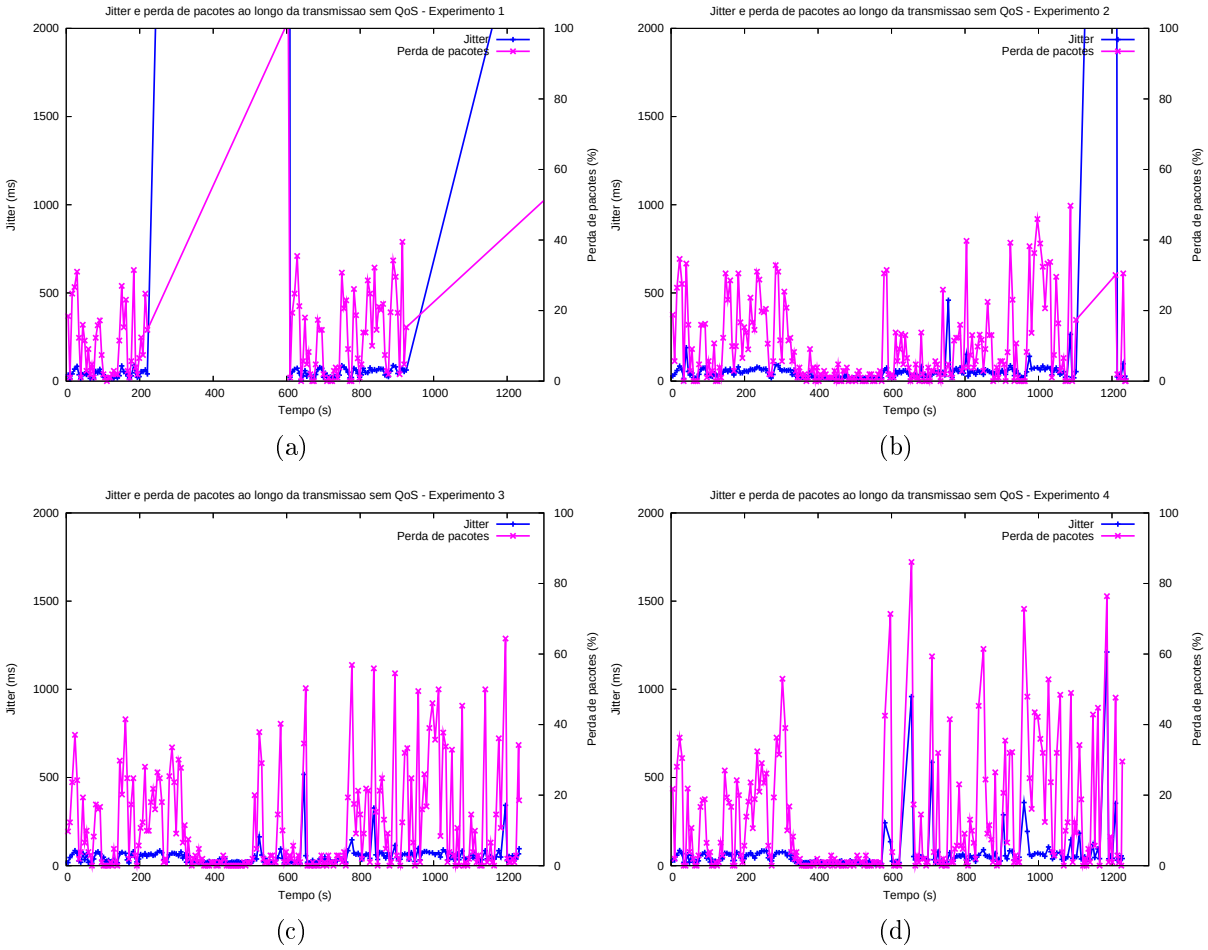


Figura 6.7: *Jitter* e perda de pacotes no cenário 3 sem QoS; (a) Experimento 1; (b) Experimento 2; (c) Experimento 3; (d) Experimento 4.

Já nos testes com QoS, em que os pacotes de cada tráfego estavam sendo encaminhados para uma classe diferente, um não afetou a transmissão do outro. O vídeo recebido pela aplicação de videoconferência estava recebendo suporte a QoS na sua transmissão, enquanto o vídeo requisitado na Internet estava sendo tratado sem priorização e sem mecanimos de adaptação. As classes que têm largura de banda reservada para os fluxos de vídeo com suporte à qualidade podem pegar mais banda emprestada com outras classes, além da fatia que lhes foi dada de direito.

Esses resultados reforçam a importância da proposta de oferecer reserva de recursos para os fluxos contínuos ao longo do *backbone* da rede *Mesh*. Essa reserva propicia uma maior priorização na transmissão para os tráfegos multimídia e para aqueles que desejarem o suporte a QoS, como os fluxos de controle.

Ainda analisando os gráficos da figura 6.7, para o caso sem QoS podem-se observar, no experimento 1 (figura 6.7(a)), períodos de desconexão, como entre 200 e 600 segun-

dos, o que gerou taxas bem elevadas tanto de *jitter* quanto de perda de pacotes. Não é possível identificar a causa exata desse efeito, mas ele provavelmente foi gerado por uma instabilidade na rede. É importante ressaltar que não foram escolhidos para serem aqui analisados apenas resultados de testes de quando o ambiente estava em boas condições, mas também de quando a rede apresentava instabilidade nos enlaces.

As figuras 6.8 e 6.9 apresentam os resultados dos testes do cenário 4 com o *framework* e para a implementação sem QoS, respectivamente. Esses experimentos foram executados em momentos em que a rede *Mesh* não apresentava transmissões estáveis, o que era perceptível ao usuário. Como nesse cenário adicionalmente o nó cliente se move, a geração de novas rotas em alguns momentos foi lenta e gerou pequenos períodos de desconexão.

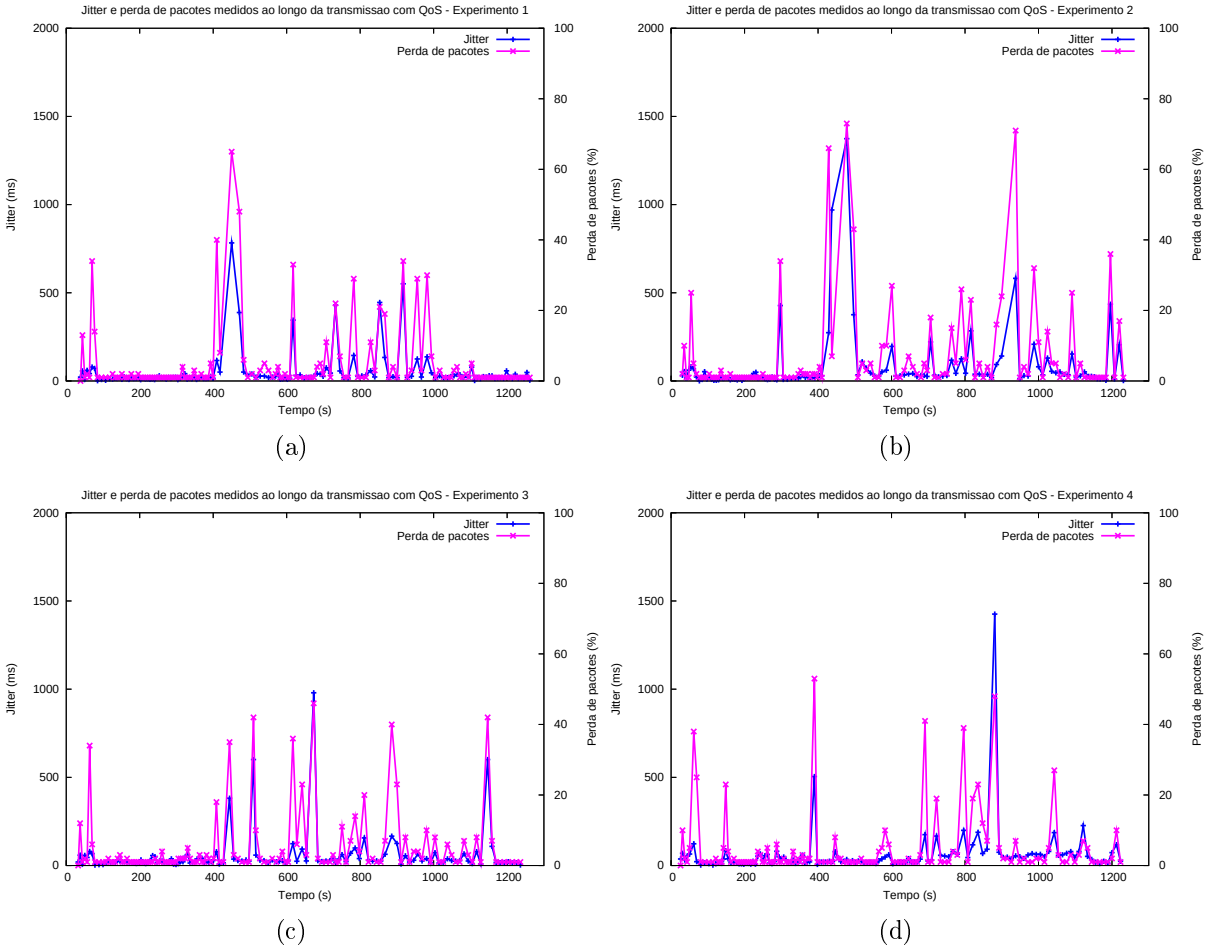


Figura 6.8: *Jitter* e perda de pacotes no cenário 4 com o *framework* - (a) Experimento 1; (b) Experimento 2; (c) Experimento 3; (d) Experimento 4.

Esses períodos de desconexão podem ser observados, por exemplo, no experimento 2 dos testes com QoS (figura 6.8(b)) entre os instantes 400 e 500s. O *framework* espera a chegada de 100 pacotes para fazer uma monitoração, e como o enlace estava com qualidade ruim, e poucos pacotes estavam chegando ao cliente, o tempo entre uma monitoração e

outra foi um pouco maior. Essa instabilidade da rede casou uma perda de pacotes de 60%, o que gerou, conseqüentemente, um pico também no *jitter* medido.

Já no experimento 4 dos testes sem QoS (figura 6.9(d)), a instabilidade da rede gerou uma porcentagem de perda de pacotes e um *jitter* bem elevados entre os segundos 790 e 1120. Isso causou uma grande degradação na qualidade do vídeo reproduzido e provocou congelamentos e atrasos na imagem.

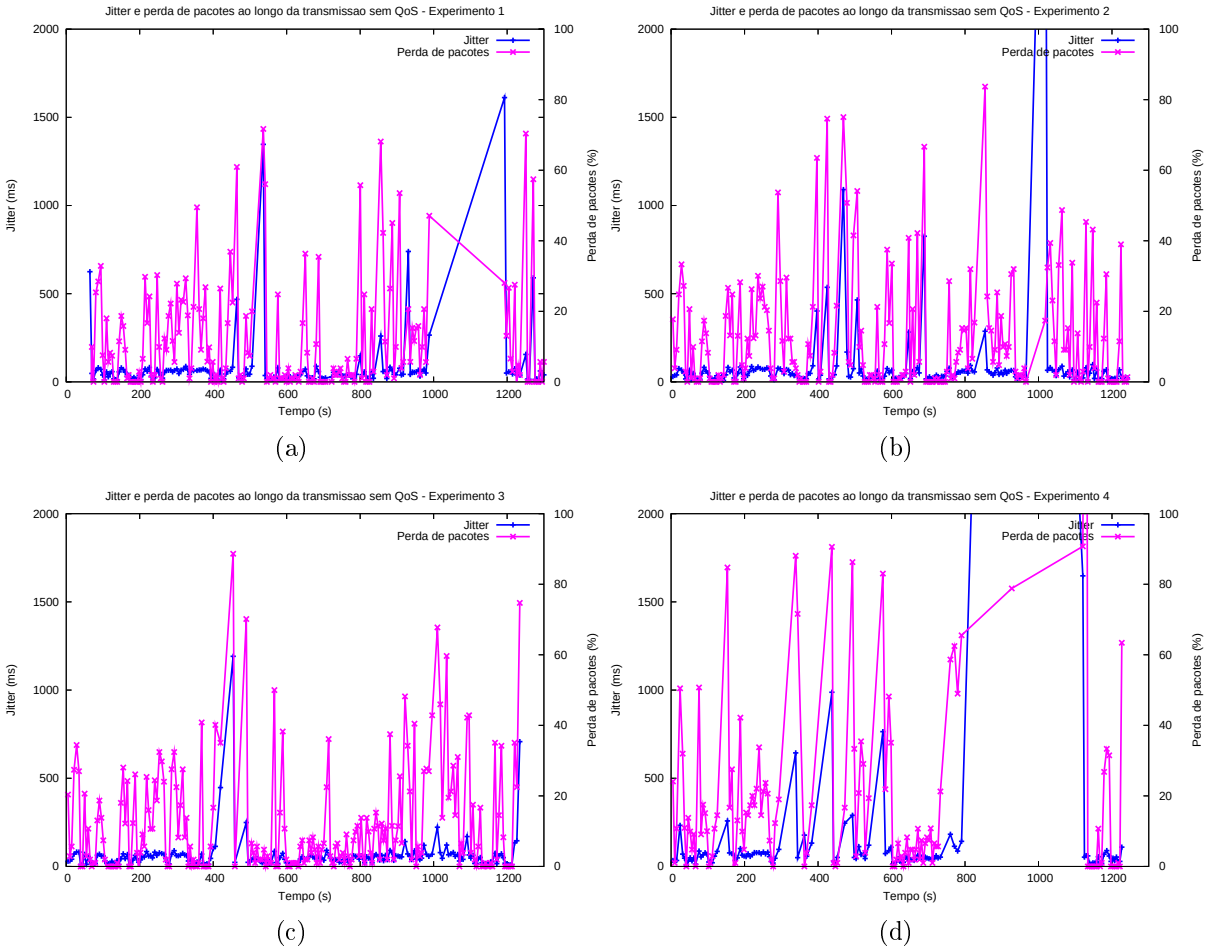


Figura 6.9: *Jitter* e perda de pacotes no cenário 4 sem QoS - (a) Experimento 1; (b) Experimento 2; (c) Experimento 3; (d) Experimento 4.

Com esses resultados foi possível comprovar a eficiência do *framework* proposto, tanto nos valores das figuras de mérito avaliadas quanto na qualidade da imagem percebida no nó cliente. Para dar suporte a QoS a fluxos multimídia em redes sem fio, é preciso monitorar a qualidade da transmissão e realmente diminuir a taxa de envio de pacotes no caso de uma degradação do ambiente. Esse processo permite que a qualidade da mídia reproduzida no cliente não apresente tantos congelamentos e atrasos. A próxima seção irá apresentar os resultados de *jitter* e de perda de pacotes do *framework* ao longo dos experimentos, acompanhados da variação da taxa de envio de dados.

6.3.2 Resultados do *Framework* ao Longo dos Testes do Protótipo

Os gráficos a seguir apresentam o processo de acionamento da adaptação agindo nos momentos de degradação da rede. Os resultados serão mostrados em dois tipos de gráficos. Primeiramente, será apresentada a evolução da taxa de envio juntamente com os valores de *jitter* calculados em cada monitoração. Depois, será apresentada a mesma taxa de envio durante a transmissão, só que acompanhada dos valores medidos da porcentagem de perda de pacotes.

Para que o *framework* acione a adaptação por causa de uma degradação da rede, é preciso que apenas um dos requisitos apresente um valor acima do limite máximo. Para reverter o mecanismo de adaptação, é preciso que ambos, o *jitter* e a perda de pacotes, apresentem valores abaixo do limite mínimo.

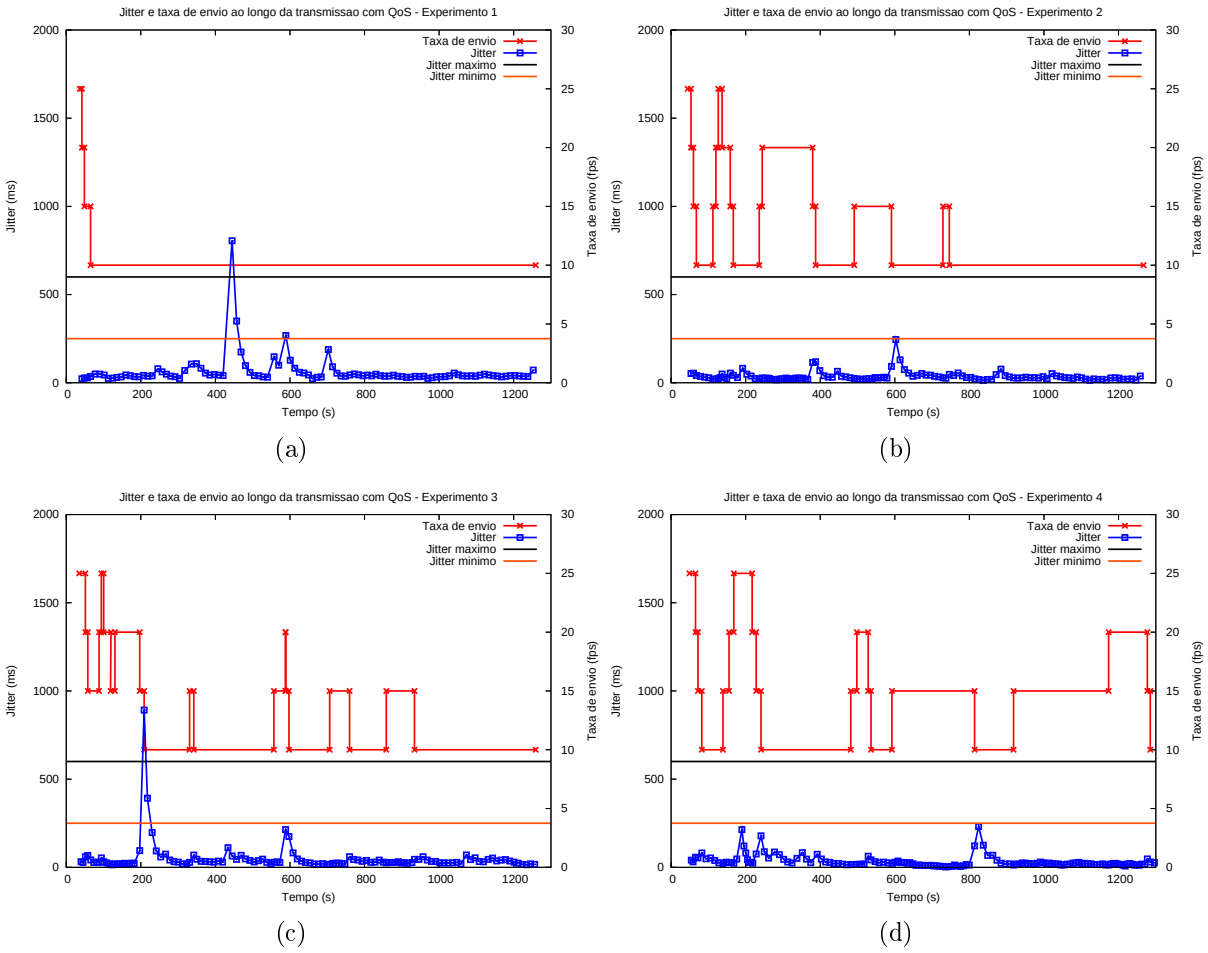


Figura 6.10: *Jitter* e taxa de envio no cenário 1 com o *framework* - (a) Experimento 1; (b) Experimento 2; (c) Experimento 3; (d) Experimento 4.

As figuras 6.10 e 6.11 apresentam os resultados para *jitter* e perda de pacotes, respectivamente, para os quatro experimentos do cenário 1. No experimento 1 (6.10(a) e

6.11(a)), mesmo com o nó cliente parado, o *jitter* e a perda de pacotes não apresentaram valores suficientes que ficassem abaixo dos limites mínimos especificados para os testes. Isso fez com que a taxa de envio dos pacotes do servidor se mantivesse em 10fps, no nível de adaptação mais baixo.

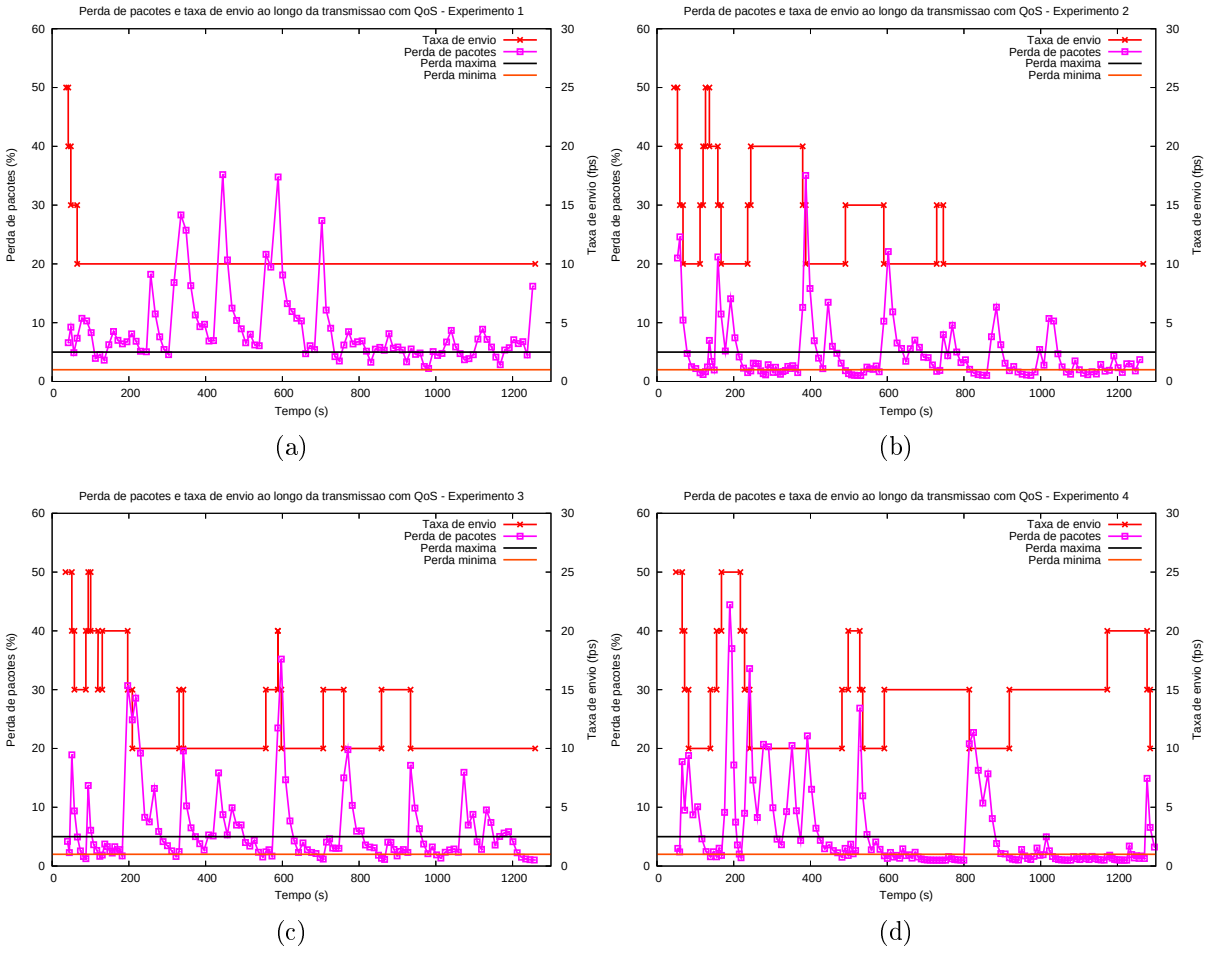


Figura 6.11: Perda de pacotes e taxa de envio no cenário 1 com o *framework* - (a) Experimento 1; (b) Experimento 2; (c) Experimento 3; (d) Experimento 4.

Nos experimentos 2, 3 e 4, o sistema de adaptação foi acionado, e o processo que procura o nível ótimo de transmissão para o fluxo também pode ser observado. Nos gráficos 6.10(b) e 6.11(b), no instante 150s, o fluxo apresentou valores de *jitter* e de perda de pacotes abaixo dos limites mínimos, permitindo passar do nível 1 para o nível 0. Porém, o nível 0 nesse momento era o "nível ótimo", e o fluxo já tinha estado nele por duas vezes e apresentado uma qualidade ruim. Assim, o *framework* decidiu manter o fluxo no nível 1. Logo em seguida, no instante 160s, a porcentagem de perda de pacotes excedeu o limite máximo, o que fez cair a taxa de envio do servidor. Assim, a decisão de não voltar ao nível 0 permitiu que a qualidade do fluxo se estabilizasse mais rapidamente, a partir do instante 250s.

Antes dos 400s, o fluxo apresentou uma perda de pacotes acima do valor limite e precisou, por isso, acionar o processo de adaptação, que diminui a taxa de transmissão. Depois da adaptação acionada, o fluxo conseguiu manter novamente a qualidade desejada pela aplicação. Mas, ao tentar aumentar a taxa de envio de pacotes, o *framework* mais uma vez percebeu que o nível ótimo, agora o nível 1, não conseguiu manter a qualidade desejada, e por isso não pôde ser mais considerado ótimo. Por essa razão, o *framework* manteve o fluxo no nível 2, a uma taxa de 15fps.

As figuras 6.12 e 6.13 apresentam a evolução do *jitter* e da perda de pacotes ao longo da execução dos quatro experimentos do cenário 2. Como já foi citado anteriormente, estes gráficos apresentam picos maiores que os gráficos do cenário 1, pois o nó cliente está se movendo, o que faz com que novas rotas precisem ser formadas.

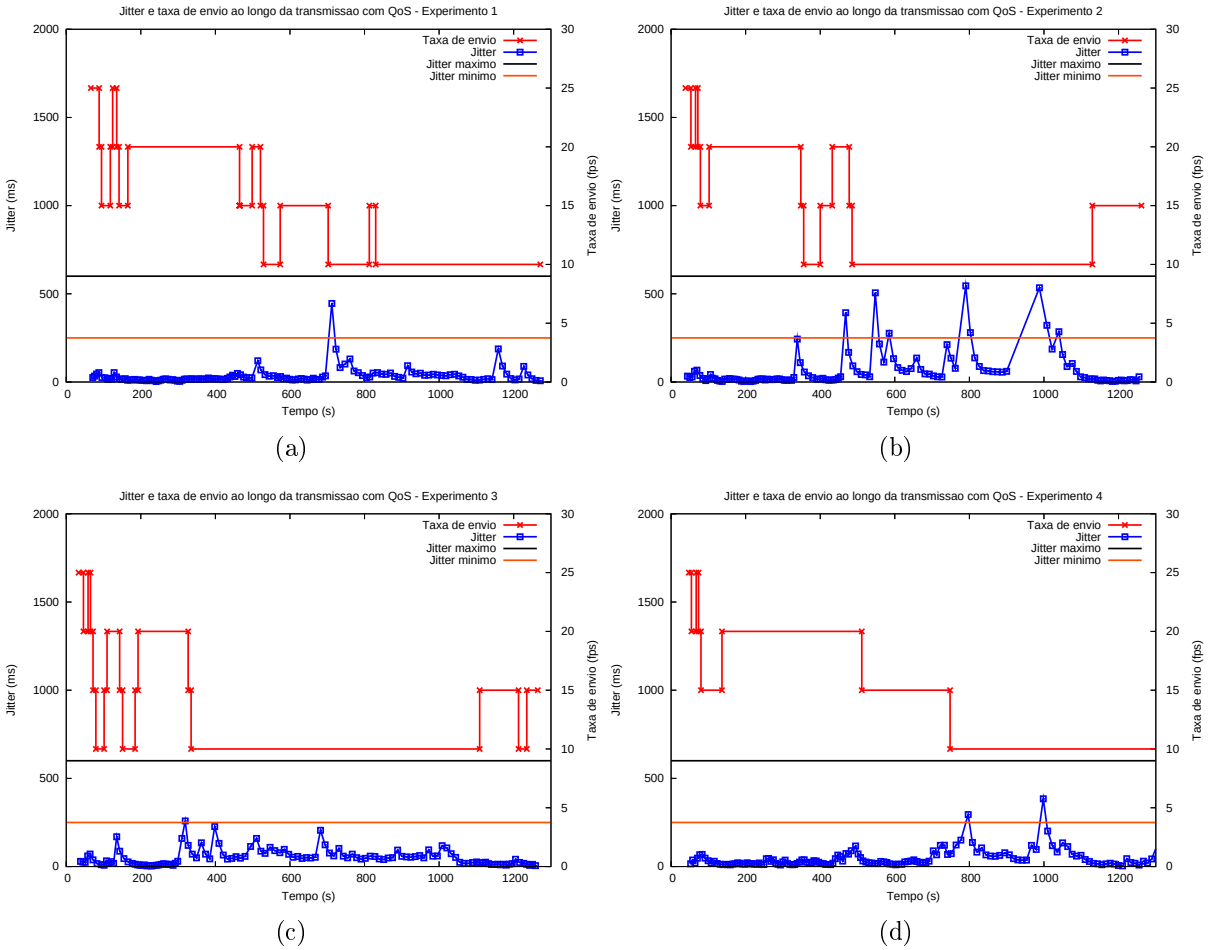


Figura 6.12: *Jitter* e taxa de envio no cenário 2 com o *framework* - (a) Experimento 1; (b) Experimento 2; (c) Experimento 3; (d) Experimento 4.

No experimento 4 (figuras 6.12(d) e 6.13(d)), o mecanismo de adaptação conseguiu estabelecer a melhor taxa de envio para o fluxo a 20fps logo no início da transmissão (145s). Porém, aos 470s, a porcentagem de perda de pacotes excedeu o limite máximo,

e o nó destino enviou um relatório de QoS para a origem, avisando-a para diminuir a taxa de envio. No entanto, de forma atípica, causada pelo canal TCP de controle, esse relatório demorou 45 segundos para chegar ao nó origem. Nesse tempo, o nó destino fez mais 4 monitorações e continuou a enviar um relatório de QoS para a origem requisitando a diminuição da taxa de envio. Isso fez com que, quando o fluxo pôde voltar para o nível 1 (530s), o *framework* não permitisse que isso acontecesse, já que o fluxo tinha ficado muito tempo nesse nível sem conseguir manter a qualidade desejada (4 monitorações).

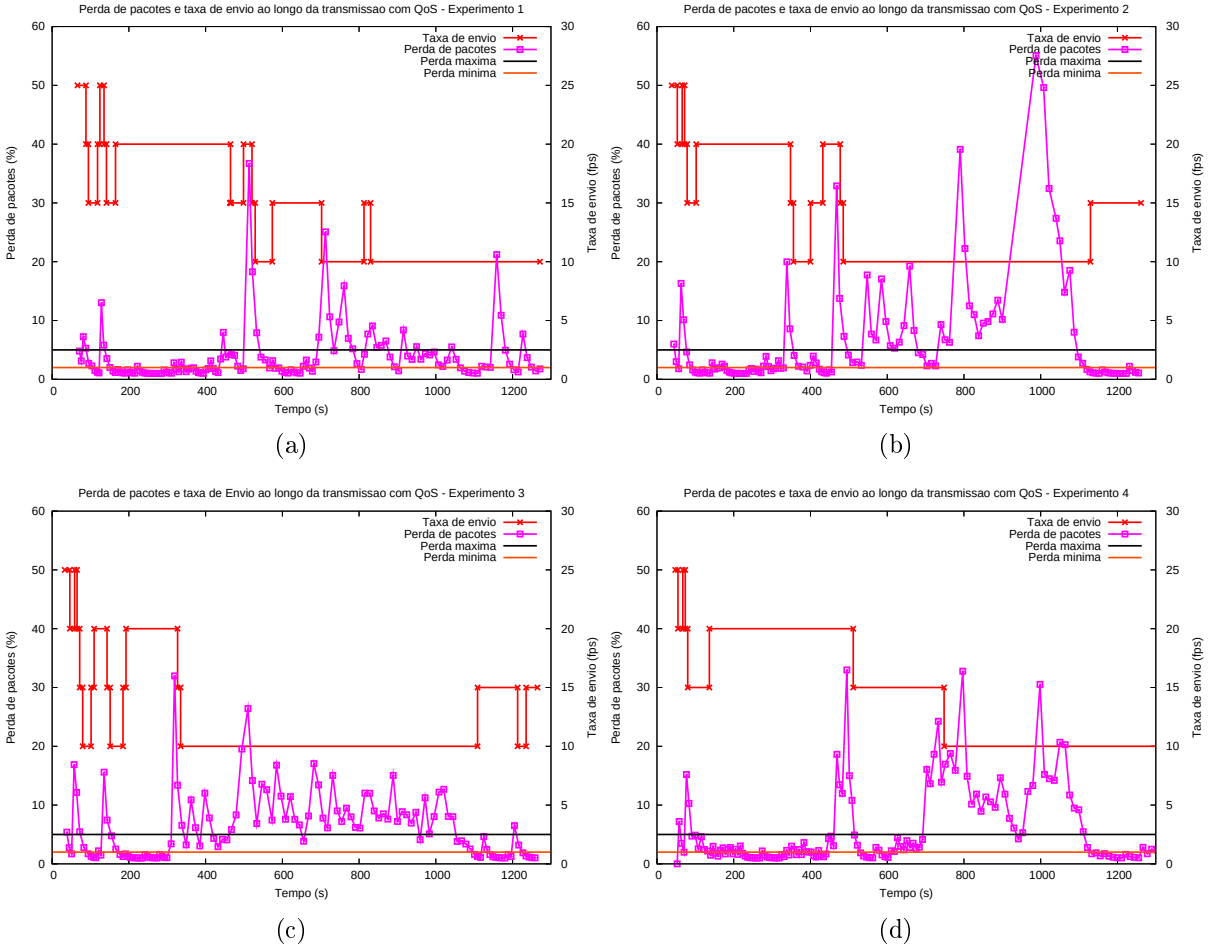


Figura 6.13: Perda de pacotes e taxa de envio no cenário 2 com o *framework* - (a) Experimento 1; (b) Experimento 2; (c) Experimento 3; (d) Experimento 4.

No mesmo experimento 4, aos 700s, a porcentagem de perda de pacotes excedeu mais uma vez o limite máximo, atingindo 16%. O nó destino então enviou um relatório de QoS para o nó origem, pedindo que passasse o nível de adaptação do fluxo de 2 para 3. Esse relatório, também de forma atípica, demorou para chegar ao nó origem (65s), o que permitiu que o nó destino fizesse mais 5 monitorações. Mais à frente, aos 1130s, quando o fluxo apresentou valores de *jitter* e de perda de pacotes abaixo do limite mínimo, o *framework* não pôde voltar ao nível de adaptação 2, já que este já tinha apresentado uma

qualidade ruim por muito tempo.

Os experimentos para os cenários 3 e 4 foram executados com um tráfego extra sendo injetado na rede para causar perturbação. Os resultados não apresentaram muitas diferenças em relação ao que já foi mostrado nos cenários anteriores. O mecanismo de monitoração e o processo de adaptação mais uma vez trabalharam em conjunto para estabilizar a qualidade do fluxo.

É importante ressaltar que não foram colocados neste trabalho apenas os melhores resultados obtidos, mas também aqueles resultados de testes em que o ambiente apresentava problemas nos enlaces. Dessa forma, foi possível avaliar o comportamento do *framework* quando a rede estava em boas condições e quando apresentava instabilidade.

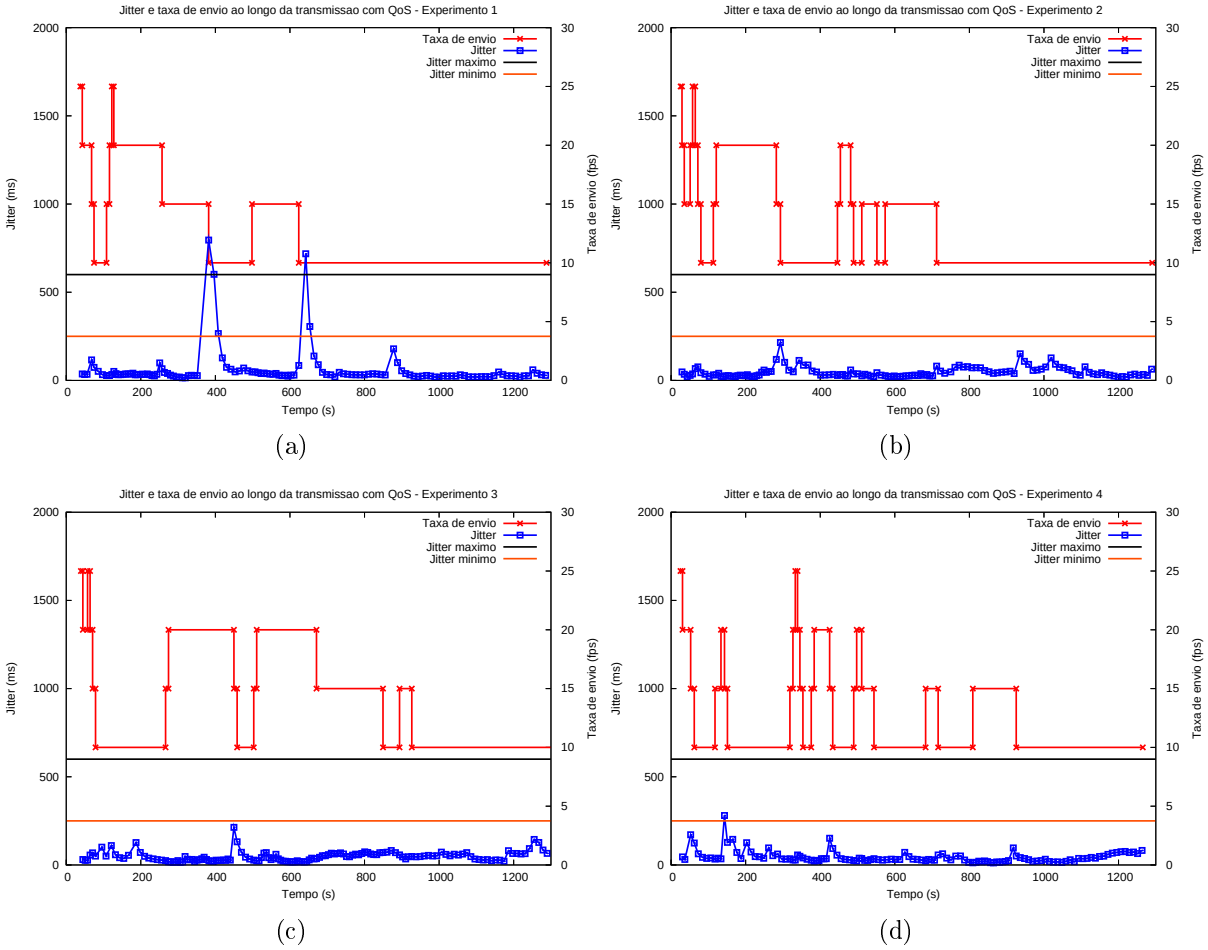


Figura 6.14: *Jitter* e taxa de envio no cenário 3 com o *framework* - (a) Experimento 1; (b) Experimento 2; (c) Experimento 3; (d) Experimento 4.

As figuras 6.14 e 6.15 mostram os gráficos do cenário 3. Os tráfegos extras gerados foram dois: (i) um FTP foi iniciado em média a cada 4 minutos; (ii) aos 10 minutos de execução, o nó cliente, que já estava recebendo um vídeo com suporte a QoS pela

aplicação de videoconferência, também requisitou um vídeo da Internet, que foi transmitido sem QoS. Nesse cenário também foram detectadas anomalias no canal TCP que causaram a demora na entrega de alguns relatórios de QoS para a origem; isso manteve o fluxo degradado nessa situação por mais tempo. Como já foi mencionado, nos dias em que os experimentos desse cenário foram executados, a rede *ReMesh* apresentava alguma instabilidade, além de não estar operando em 54Mbps, que é sua capacidade máxima. Esse fato fez diminuir a largura de banda disponível para os fluxos e aumentar a disputa pelo recurso.

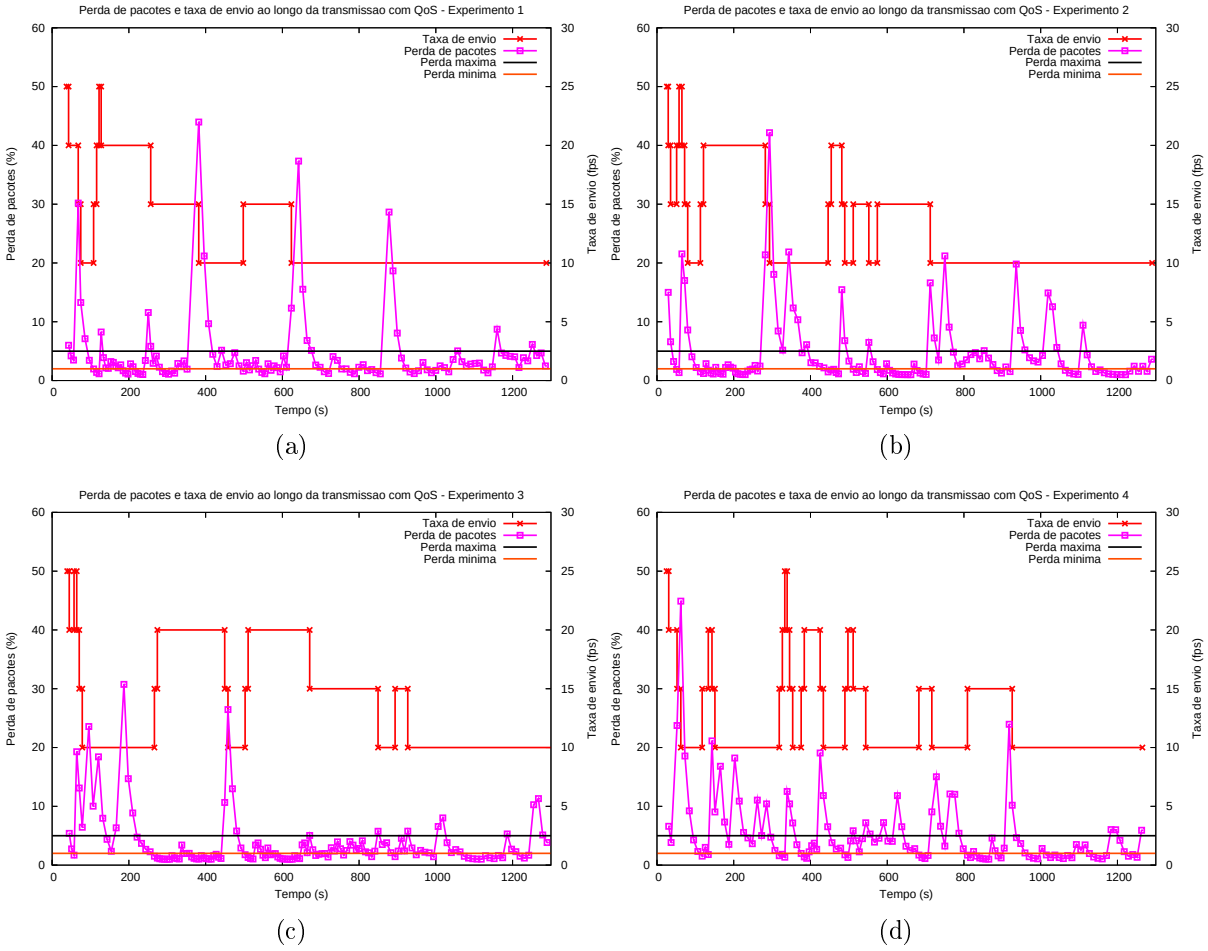


Figura 6.15: Perda de pacotes e taxa de envio no cenário 3 com o *framework* - (a) Experimento 1; (b) Experimento 2; (c) Experimento 3; (d) Experimento 4.

Analisando os gráficos dos experimentos, é possível observar que, em um ambiente real, o mesmo cenário apresentou resultados bem diferentes entre si. Isso se deve às interferências inerentes aos ambientes sem fio. Durante os testes foi visível a perturbação causada, por exemplo, por um grupo de alunos que saíam todos juntos de uma sala de aula. A potência do sinal recebido diminui momentaneamente, e muitas vezes o protocolo de roteamento tenta encontrar uma nova rota com qualidade melhor.

Os testes no ambiente real comprovaram a eficiência do *framework* proposto. Como o ambiente de simulação não reproduz as interferências inerentes às redes sem fio, alguns cenários que, no estudo preliminar, apresentaram desempenho satisfatório em taxas de transmissão mais altas, como 25fps, nos testes do protótipo precisaram atuar com taxas menores, chegando algumas vezes a manter a menor taxa, de 10fps. Os testes de simulação também não lidam com limitações tecnológicas dos dispositivos utilizados e com os atrasos gerados pelo tempo de processamento da aplicação. Esses fatores contribuem para que os resultados obtidos nos dois tipos de testes sejam um pouco diferentes para os mesmos cenários.

As figuras 6.16 e 6.17 apresentam os gráficos dos resultados do cenário 4. O tráfego extra gerado para este cenário foi o mesmo utilizado para o cenário 3.

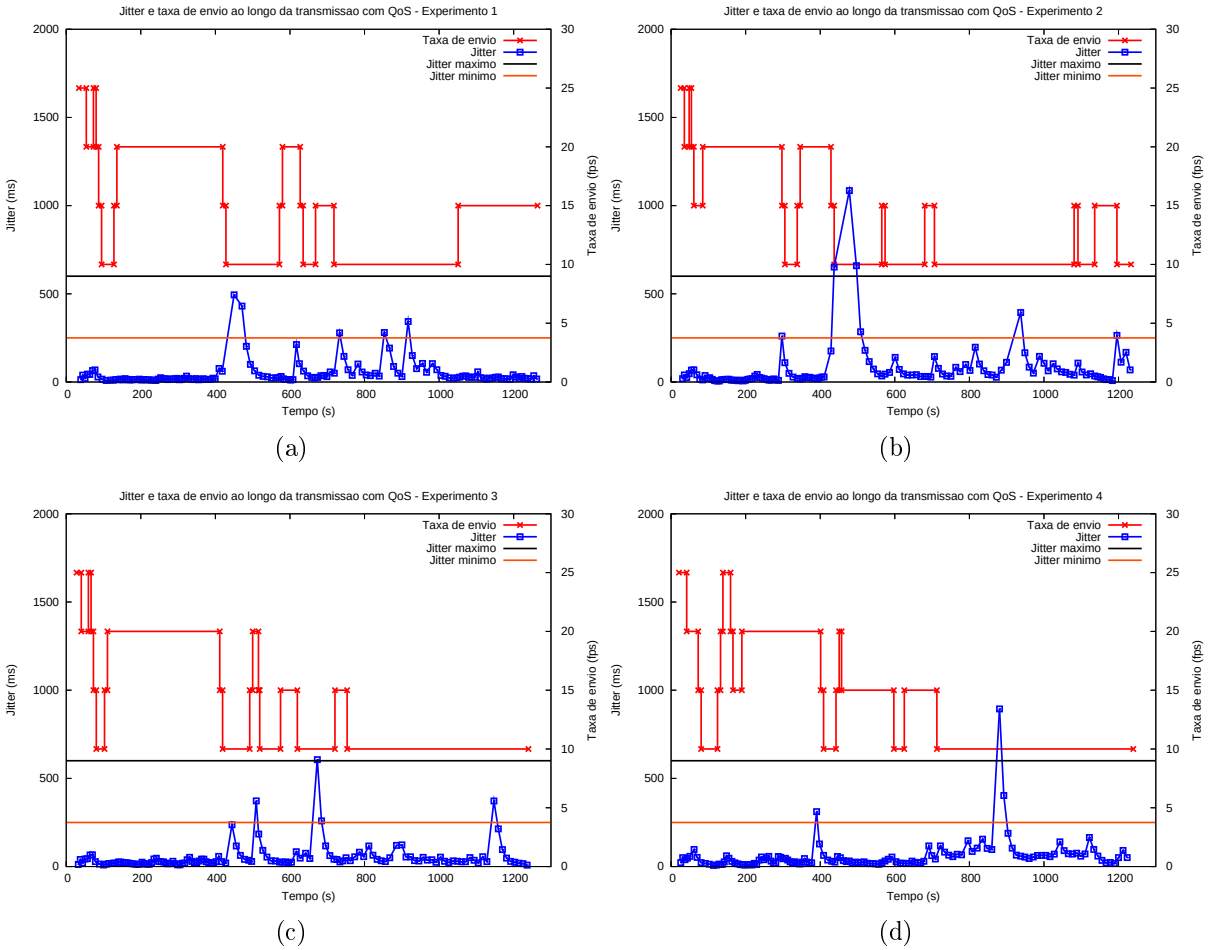


Figura 6.16: *Jitter* e taxa de envio no cenário 4 com o *framework* - (a) Experimento 1; (b) Experimento 2; (c) Experimento 3; (d) Experimento 4.

O resultado mais importante dos cenários com tráfego extra foi observado na prática: nos testes com o *framework*, a transferência de um vídeo não atrapalhou a visualização do outro, já que no *backbone* eles eram classificados como pacotes de classes distintas. Já nos

testes sem QoS, como os dois vídeos estavam disputando banda da classe de fluxos sem priorização, a visualização dos dois foi prejudicada.

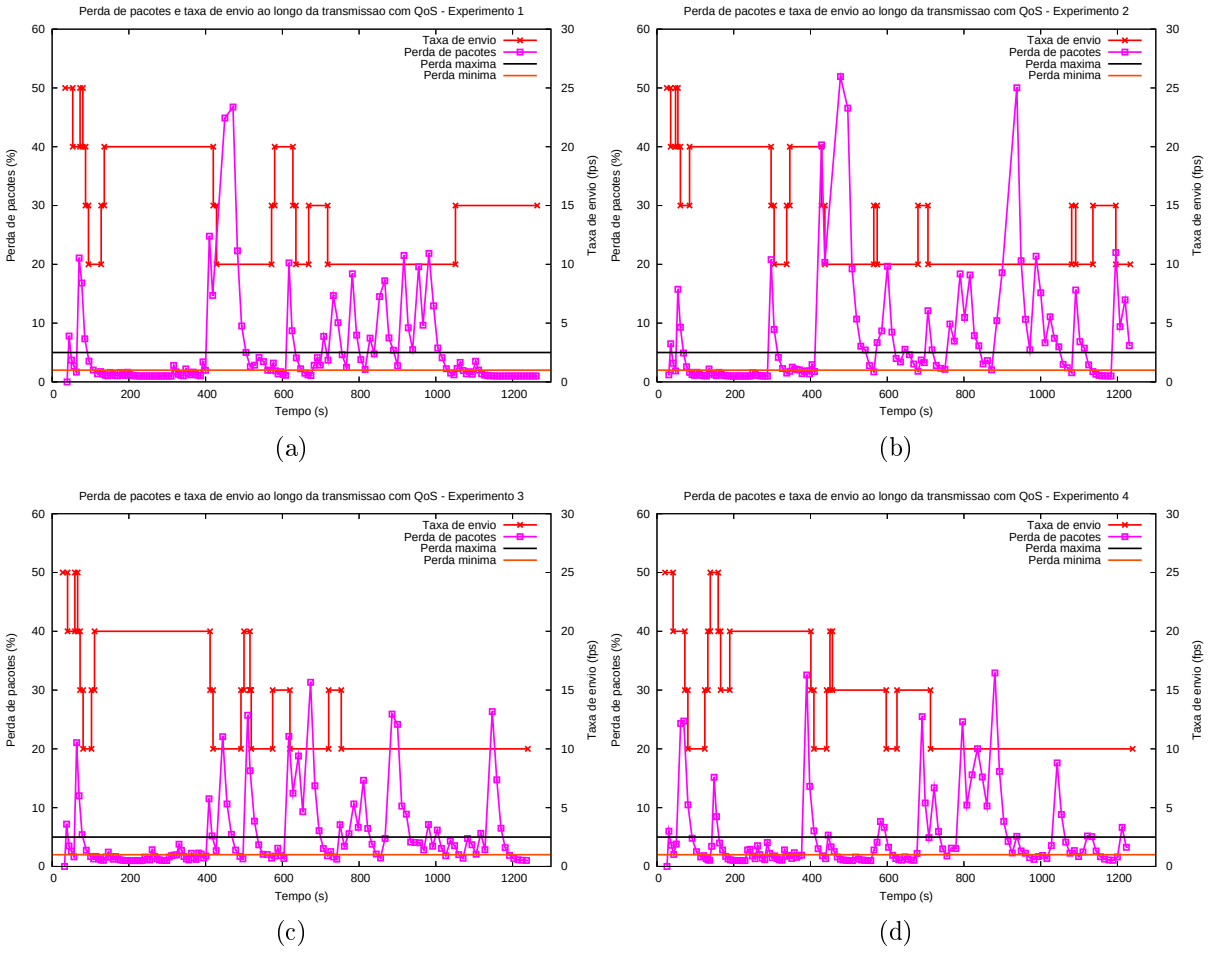


Figura 6.17: Perda de pacotes e taxa de envio no cenário 4 com o *framework* - (a) Experimento 1; (b) Experimento 2; (c) Experimento 3; (d) Experimento 4.

Os cenários com o nó em movimento apresentam uma estabilização da qualidade do fluxo no início dos testes, já que o nó cliente está perto do nó servidor. Nesses testes, pode-se perceber que a transmissão do vídeo passando apenas por um salto conseguiu manter a taxa de envio de pacotes em 20fps. Esse efeito pode ser observado nos gráficos do cenário 4 (figuras 6.16 e 6.17) e do cenário 2 (figuras 6.12 e 6.13), em que, em média nos primeiros 400s, enquanto o cliente ainda estava próximo ao servidor, a taxa de envio de pacotes se encontrava por volta dos 20fps. Após esse instante, os valores de perda de pacotes e de *jitter* começaram a variar por causa da movimentação do cliente e, conseqüentemente, do afastamento deste em relação ao servidor.

Por volta do instante 1080s, o cliente já está novamente perto do servidor. Assim, a taxa de envio tende a aumentar, chegando aos 20fps. Esse efeito pode ser percebido, por exemplo, no experimento 1 do cenário 4 (figuras 6.16(a) e 6.17(a)). Já no instante

1050s, quando o nó cliente começa a se aproximar do nó servidor, o *framework* percebe uma melhora na qualidade da transmissão e consegue aumentar a taxa de transmissão de 10fps para 15fps. Provavelmente, se o experimento tivesse continuidade, a taxa de envio de pacotes iria aumentar até chegar novamente a 20fps, que é a taxa que apresentou melhor qualidade do fluxo.

6.4 Conclusão

Os gráficos apresentados neste capítulo reforçam os resultados das simulações mostrados no capítulo 4. Da mesma forma que no estudo preliminar, o *framework* respondeu às quedas de qualidade da transmissão acionando o mecanismo de adaptação adequadamente.

Apesar dos cenários terem sido os mesmos nos testes do protótipo e da simulação, o comportamento do ambiente foi bem diferente, o que faz o *framework* agir de forma distinta em cada tipo de teste. A razão dessa diferença de comportamento pode ser atribuída a alguns fatores. Primeiramente, no ambiente de simulação, os testes dos cenários 1 e 2 foram realizados realmente sem interferência. Já no ambiente do protótipo, a movimentação de pessoas nas salas e corredores e a existência de outros dispositivos sem fio impossibilitou que os testes fossem realizados sem nenhuma interferência.

Em segundo lugar, o protocolo de roteamento utilizado na simulação foi diferente do usado nos testes do protótipo. Como já explicado, não foi possível realizar os testes do estudo preliminar com o protocolo OLSR, mas apenas com o DSR. A diferença principal entre esses dois protocolos é que o OLSR faz parte do grupo dos pró-ativos, que mantêm suas tabelas de rota periodicamente atualizadas, e o DSR é um protocolo reativo, que busca uma rota entre dois nós apenas quando se necessita transmitir dados.

A terceira justificativa para a diferença de comportamento entre os ambientes é a própria topologia da rede *Mesh*. Enquanto no ambiente de simulação o raio de alcance dos nós móveis se mantém sempre o mesmo, no ambiente real ele varia com o tempo e entre os nós, por causa de obstáculos como paredes, elevadores e o próprio movimentar de pessoas. Ainda pode ser citado o fato de que é preciso lidar com as possíveis falhas do *hardware*, o que não acontece na simulação.

Mesmo com essas diferenças, os testes do protótipo apresentaram resultados positivos. O ponto mais importante que pode ser tirado desses experimentos é a observação da reprodução do vídeo. A diferença entre a transmissão com e sem suporte a QoS é perceptível ao usuário cliente.

Capítulo 7

Conclusão

7.1 Introdução

Este trabalho buscou apresentar uma solução para fornecer qualidade de serviço a transmissões multimídia em redes *Mesh*. Os ambientes sem fio possuem características inerentes à sua natureza, como a instabilidade dos enlaces, gerada pela possível mobilidade dos nós e pelas interferências causadas por ruídos, pelo movimentar de pessoas e ainda por outros dispositivos sem fio. O próprio compartilhamento do meio pelos nós participantes pode causar colisões, atrasos e perdas dos pacotes.

As redes *Mesh* têm como principal objetivo formar um *backbone* sem fio para fornecer acesso banda larga comunitário. Dessa forma, será possível montar uma rede de baixo custo para os moradores de um bairro, por exemplo, ou ainda oferecer acesso à Internet banda larga para os alunos e professores que morarem nos arredores de sua universidade. Foi com esse último foco que o projeto *ReMesh* desenvolveu um protótipo de rede *Mesh* na Universidade Federal Fluminense.

O *framework* de suporte a QoS em redes *Mesh* proposto neste trabalho é formado por um conjunto de mecanismos que, juntos, permitem que a mídia transmitida apresente a qualidade desejada pelo usuário. Esses principais elementos são:

- A especificação de **requisitos de QoS** por parte da aplicação, que permite haver uma participação do usuário no processo de suporte a QoS;
- Um sistema de **sinalização *in-band***, que envia junto com os pacotes de dados informações de controle para o *framework* e auxilia, assim, o funcionamento dos principais mecanismos;

- O processo de **reserva de recursos**, que tem como objetivo fornecer garantias de largura de banda e priorização para a transferência de fluxos multimídia;
- O mecanismo de **monitoramento do fluxo**, que permite que os requisitos de QoS sejam avaliados durante a transmissão, comparando os resultados obtidos com os valores especificados pela aplicação; e,
- O **sistema de adaptação**, que age em conjunto com o monitoramento, permitindo que a transmissão dos dados se adapte às condições da rede e apresente o desempenho desejado pela aplicação.

Com esse sistema definido, foi realizado um estudo preliminar para avaliar a viabilidade dos mecanismos propostos, para escolher os melhores valores para os parâmetros dos algoritmos utilizados, e para verificar se o *framework* realmente apresentaria um desempenho satisfatório. Para isso, foram feitas simulações no ambiente *ns-2*. A proposta denominada INSIGNIA foi usada como base de comparação para os resultados do *framework*. O INSIGNIA é um *framework* para suporte a QoS em redes *Ad Hoc* para fluxos de tempo real. Ele também utiliza um mecanismo de reserva de recursos e propõe, como adaptação do fluxo às más condições da rede, que pacotes menos importantes sejam enviados por melhor esforço. Os resultados dessa comparação foram os melhores possíveis: o *framework* para redes *Mesh* apresentou um desempenho superior ao do INSIGNIA em todos os cenários testados.

Após a conclusão do estudo preliminar, o *framework* foi então implementado no protótipo de redes *Mesh* montado na Universidade Federal Fluminense pelo projeto *ReMesh*. Os resultados dos testes no protótipo seguiram o que a simulação já havia indicado e também permitiram observar os resultados visuais de transmissões reais.

A transmissão do vídeo apresentou realmente uma melhora significativa com o suporte a QoS fornecido pelo *framework*. Todos os testes realizados sem o *framework* tiveram uma degradação perceptível na visualização da mídia, o que gerou congelamentos no vídeo e aumentou a quantidade de pacotes que chegaram fora de ordem no destino. Já com o *framework* de suporte a QoS, a perda de pacotes e os atrasos observados não causaram uma degradação perceptível na visualização da mídia, na maior parte dos testes.

A implementação do *framework* no protótipo da rede *Mesh* também permitiu a avaliação da tecnologia atual. Inicialmente, a proposta era a de que os testes fossem feitos tanto em dispositivos de pequeno porte como também em *laptops*. Porém, não foi encontrada uma versão do protocolo de roteamento OLSR para Pocket PC que fosse compatível com a

versão então usada nos roteadores. No entanto, em um trabalho paralelo, o aluno Robson Hilario da Silva [da Silva 2006] realizou testes de transmissão de vídeo em redes sem fio infra-estruturadas com *Pockets PC*; tais testes podem ser usados como indicativo de como teriam sidos os resultados com tal tecnologia.

Acompanhando os resultados desses testes, é possível perceber que esses pequenos dispositivos ainda não têm capacidade de processamento suficiente para receber um fluxo de vídeo e visualizá-lo com qualidade, além de monitorar as condições da rede e executar um protocolo de roteamento. No trabalho citado, o nó cliente deve monitorar a potência do sinal dos pontos de acesso da rede infra-estruturada e decidir o melhor momento de fazer uma migração entre eles, enquanto recebe e visualiza quadros de vídeo. Para que fosse possível ao *Pocket* conseguir processar todas as tarefas no prazo, a taxa de envio teve que ser reduzida a apenas 1 quadro por segundo. Já nos testes com *laptop* no mesmo ambiente, a taxa de envio pôde ser de 30fps, que é a máxima utilizada.

Fazendo uma comparação com o nosso *framework* de suporte a QoS, o nó cliente também precisa fazer monitorações da qualidade da transmissão, medindo *jitter* e perda de pacotes. Além desses cálculos periódicos e da recepção e visualização do vídeo, o cliente também precisa executar o protocolo de roteamento OLSR, para manter a comunicação com o servidor enquanto se move. Assim, pode-se concluir que, com a tecnologia hoje disponível para pequenos dispositivos, não seria possível obter resultados satisfatórios nos testes com o *framework*.

A próxima seção irá apresentar os trabalhos futuros propostos para a aprimoração do *framework* de suporte a QoS em redes *Mesh*.

7.2 Trabalhos Futuros

Ao estudar as codificações existentes para vídeo, é possível perceber que o *bit rate* de saída do vídeo pode ser alterado, gerando mais ou menos volume de dados a ser enviado pela rede. Uma proposta interessante é que, em vez de alterar a taxa de captura de uma câmera ou a taxa de envio dos pacotes, o módulo de codificação do vídeo fosse alterado para que gerasse um vídeo com uma qualidade inferior, mas que o *bit rate* fosse menor, o que diminuiria a quantidade de *bytes* enviados na rede.

Em relação ao sistema de reserva de recursos, mais classes poderiam ser criadas para fornecer qualidade e largura de banda para outros tráfegos prioritários, como, por exemplo, as mensagens de controle do OLSR. Na implementação atual, as mensagens trocadas entre

os nós para montar a tabela de rotas são classificadas como um tráfego não prioritário e, por isso, vão para a classe Outros.

O principal objetivo do trabalho apresentado foi o de implementar um protótipo de um *framework* de suporte a QoS em redes *Mesh*. Para isso, foi necessário fazer algumas simplificações nos mecanismos propostos. A reserva de recursos, por exemplo, é fixa em todos os roteadores do *backbone*. A largura de banda foi dividida por classes de fluxos, e não por cada conversação. Uma outra alternativa é a de implementar uma alocação dinâmica de reservas, na qual os recursos sejam reservados apenas para os fluxos que requisitarem esse serviço, com a quantidade desejada e pelo tempo em que o fluxo estivesse ativo. Outra simplificação feita para que fosse possível utilizar as ferramentas disponíveis do sistema, sem a alteração ou inserção de novos programas, foi a ausência de um controle de admissão. Esse mecanismo permite que um fluxo não tenha permissão para transmitir seus dados, quando a rede já está saturada e não existem mais recursos disponíveis.

O sistema de monitoração também foi simplificado, apresentando um tamanho fixo de janela. Para que a monitoração pudesse acompanhar as condições da rede, seria interessante que o tamanho da janela de monitoração fosse definido de forma dinâmica. Se a rede estivesse muito instável e congestionada, o tamanho da janela seria menor, o que permitiria uma rápida percepção dos problemas do ambiente por parte do *framework*. Caso o ambiente estivesse estável, e com poucos usuários competindo pelo meio, o tamanho da janela poderia ser maior, o que possibilitaria ao *framework* executar menos monitorações.

A utilização de dispositivos com o padrão IEEE 802.11e permitiria que fosse implementado um mecanismo de reserva de recursos diretamente na camada MAC. As especificações desse padrão prevêem que haja um mecanismo responsável por fornecer qualidade de serviço através da reserva de largura de banda no nível de controle de acesso ao meio. A princípio, ainda não existem dispositivos no mercado com esse padrão.

Duas questões muito importantes, quando se trabalha em ambientes sem fio, são a segurança e o controle do gasto energético. As redes *wireless* utilizam o ar como meio de transporte dos pacotes, o que pode permitir a captação das mensagens por terceiros. Além disso, é importante haver sistemas de autenticação de usuário e criptografia dos dados, o que evita a invasão por pessoas mal intencionadas e a violação ou corrupção do conteúdo das mensagens que trafegam na rede. No entanto, para dispositivos de pequeno porte, que possuem baixa capacidade de processamento e capacidade energética limitada, o uso desses mecanismos pode prejudicar o funcionamento de outros sistemas e consumir muita energia. Dentre todas as funções que uma máquina deve processar, a comunicação

sem fio é uma das que mais consome energia. Um mecanismo que controle esse consumo para prolongar a vida do nó na rede também auxilia no processo de fornecimento de QoS, já que em níveis muito baixos de energia a recepção de pacotes fica prejudicada.

Outra questão de segurança neste trabalho é em relação a clientes maliciosos. Como a decisão de adaptação do *framework* é de responsabilidade do cliente, o servidor pode ser alvo de nós que querem se aproveitar dessa situação e requisitar uma taxa de envio maior do que as condições da rede podem suportar. Para se evitar isso, é preciso ter um mecanismo de reconhecimento dos clientes confiáveis por parte do servidor.

Na avaliação dos resultados do protótipo, pode-se perceber que, com a configuração utilizada para os limites de *jitter* e de perda de pacotes, o *jitter* não influenciou tanto no acionamento do sistema de adaptação quanto à perda. Seria preciso executar mais testes com o protótipo, utilizando diferentes faixas para os limites, e avaliar a atuação do *jitter* nesses casos, comprovando ou não a necessidade da monitoração desse parâmetro para o fornecimento de QoS. Se fosse inserida na aplicação a transmissão do áudio, é intuitivo prever que, nesse tipo de tráfego, a medição de *jitter* seja realmente indispensável.

Para que a implementação do *framework* seja mais amplamente avaliada, mais testes deveriam ser executados. Além disso, a escala desses testes deveria ser maior, inserindo mais clientes móveis que requisitassem um vídeo com suporte a QoS ou simplesmente perturbassem o meio. Isso não foi possível por falta de uma equipe trabalhando nos testes.

O sistema de adaptação poderia prover um mecanismo mais elaborado de controle, através do uso de controladores tipo PID, por exemplo. Porém, por serem estesmecanismos mais complexos, a baixa capacidade de processamento e a capacidade energética limitada dos pequenos dispositivos pode ser um problema para a execução adequada desses sistemas.

7.3 Conclusão

O desenvolvimento e a implementação da rede *Mesh* e do *framework* de suporte à qualidade de serviço permitiram conhecer, na prática, as vantagens e as limitações dos ambientes sem fio. Quando uma arquitetura é proposta para esse tipo de ambiente e apenas simulações são feitas, os resultados obtidos nem sempre correspondem à realidade.

O resultado da nossa proposta para suporte a QoS, em comparação com outros traba-

lhos estudados, reafirma alguns conceitos por esses defendidos e a eficácia dos mecanismos por nós propostos. A reserva de recursos num ambiente em que existe escassez de largura de banda, por exemplo, e em que todos os participantes da rede precisam disputá-los, torna-se importante principalmente para fluxos contínuos. Os testes com o *framework* comprovaram que a priorização desse tipo de tráfego resulta em melhor qualidade da mídia no cliente.

Referências

[Aguayo et al. 2004] Aguayo, D.; Bicket, J.; Biswas, S.; Judd, G. e Morris, R. **Link-level Measurements from an 802.11b Mesh Network**. *SIGCOMM*, Portland, Oregon, EUA, v. 34, n. 4, p. 121–132, Agosto 2004.

[Ahn et al. 2002] Ahn, G.-S.; Sun, L.-H.; Veres, A. e Campbell, A. T. **SWAN: Service Differentiation in Stateless Wireless Ad Hoc Networks**. In: *INFOCOM*, Nova York, NY, EUA, v. 2, p. 457 – 466, Junho, 2002.

[Akyildiz et al. 2005] Akyildiz, I. F.; Wang, X. e Wang, W. **Wireless Mesh Networks: a Survey**. *Computer Networks and ISDN Systems*, Amsterdam, Holanda, v. 47, n. 4, p. 445–487, Janeiro 2005.

[Bicket et al. 2005] Bicket, J.; Aguayo, D.; Biswas, S. e Morris, R. **Architecture and Evaluation of an Unplanned 802.11b Mesh Network**. In: *MOBICOM*, Colônia, Alemanha, p. 31 – 42, Agosto, 2005.

[Calafate 2002] Calafate, C. M. T. **OLSR for Windows 2000 and Pocket PC**. 2002. Disponível em: <http://www.grc.upv.es/calafate/olsr/olsr.htm>.

[Calhoun et al. 2005] Calhoun, P. R.; O'Hara, B.; Suri, R.; Kelly, S.; Williams, M. G.; Hares, S. e Winget, N. C. **Light Weight Access Point Protocol (LWAPP)**. Cisco Systems, Facetime Communications, Nokia, Nexthop Technologies, 2005. Disponível em: <http://www.ietf.org/internet-drafts/draft-ohara-capwap-lwapp-03.txt>.

[Clausen e Jacquet 1997] Clausen, T. e Jacquet, P. **Resource ReSerVation Protocol (RSVP) – Version 1 Functional Specification**. Internet Engineering Task Force, September 1997. Disponível em: <http://rfc.net/rfc2205.txt>.

[Clausen e Jacquet 2003] Clausen, T. e Jacquet, P. **Optimized Link State Routing Protocol (OLSR)**. Internet Engineering Task Force, Outubro 2003. Disponível em: <http://rfc.net/rfc3626.txt>.

[Draves et al. 2004a] Draves, R.; Padhye, J. e Zill, B. **The Architecture of the Link Quality Source Routing Protocol.** Microsoft Research, 2004.

[Draves et al. 2004b] Draves, R.; Padhye, J. e Zill, B. **Comparison of Routing Metrics for Static Multi-Hop Wireless Networks**. In: *SIGCOMM*, Portland, Oregon, EUA, p. 133–144, Agosto, 2004.

[Draves et al. 2004c] Draves, R.; Padhye, J. e Zill, B. **Routing in Multi-Radio, Multi-Hop Wireless Mesh Networks**. In: *MOBICOM*, Philadelphia, PA, EUA, p. 114–128, Setembro, 2004.

[Dynamics 2001] Dynamics. **HMIP Code from HUT**. 2001. Disponível em: <http://dynamics.sourceforge.net/>.

[Guimarães et al. 2004] Guimarães, R.; Morilo, J.; Cerdà, L.; Barceló, J.-M. e Garcia, J. **Quality of Service for Mobile Ad-hoc Networks: an Overview**. Polytechnic University of Catalonia, Julho 2004. Disponível em: <http://people.ac.upc.edu/rpaoliel/publications/report-upc-dac-2004-29.pdf>.

[Hubert et al. 2003] Hubert, B.; Graf, T.; Maxwell, G.;van Mook, R.;van Oosterhout Paul B Schroeder, M.; Spaans, J. e Larroy, P. **Linux Advanced Routing & Traffic Control HOWTO**. 2003. Disponível em: <http://www.lartc.org/>.

[INSIGNIA 2001] INSIGNIA. **INSIGNIA - Enabling Services in Mobile Ad Hoc Networks**. 2001. Disponível em: <http://comet.columbia.edu/insignia/>.

[Jacquet et al. 2001] Jacquet, P.; Mühlethaler, P.; Clausen, T.; Laouiti, A.; Qayyum, A. e Viennot, L. **Optimized Link State Routing Protocol for Ad Hoc Networks**. In: *Proceedings of the 5th IEEE Multi Topic Conference (INMIC 2001)*, Lahore, Paquistão, p. 62–68, Dezembro, 2001.

[Johnson et al. 2004] Johnson, D. B.; Maltz, D. A. e Hu, Y.-C. **The Dynamic Source Routing Protocol for Mobile Ad Hoc Networks (DSR)**. IETF MANET Working Group, Julho 2004. Expirado em Janeiro de 2005. Disponível em: <http://www.ietf.org/internet-drafts/draft-ietf-manet-dsr-10.txt>.

[Kassler et al. 2000] Kassler, A.; Burness, L.; Khengar, P.; Kovacs, E.; Mandato, D.; Manner, J.; Neureiter, G.; Robles, T. e Velayos, H. **BRENTA - Supporting Mobility and Quality of Service for Adaptable Multimedia Communication**. In: *IST Mobile Summit 2000*, Galway, Irlanda, Outubro, 2000.

[LARTC 2005] LARTC. **Linux Advanced Routing & Traffic Control**. 2005. Disponível em: <http://www.lartc.org/>.

[Lee et al. 2001] Lee, S.-B.; Ahn, G.-S. e Campbell, A. T. **Improving UDP and TCP Performance in Mobile Ad Hoc Networks With INSIGNIA**. *Communications Magazine, IEEE*, v. 39, n. 6, p. 156–165, Junho 2001.

[Lee et al. 1999] Lee, S.-B.; Ahn, G.-S.; Zhang, X. e Campbell, A. T. **INSIGNIA**. IETF MANET Working Group, Outubro 1999. Espirado em Maio de 2000. Disponível em: <http://www3.ietf.org/proceedings/99nov/I-D/draft-ietf-manet-insignia-01.txt>.

[Lee et al. 2000] Lee, S.-B.; Ahn, G.-S.; Zhang, X. e Campbell, A. T. **INSIGNIA: an IP-based Quality of Service Framework for Mobile Ad Hoc Networks**. *Journal of Parallel and Distributed Computing*, v. 60, n. 4, p. 374–406, 2000.

[Lin e Gerla 1997] Lin, C. R. e Gerla, M. **Asynchronous Multimedia Multi-hop Wireless Networks**. In: *IEEE INFOCOM*, Kobe, Japão, p. 118–125, Abril, 1997.

[MeshNet 2005] MeshNet. **Santa Barbara Mesh Network**. 2005. Disponível em: <http://moment.cs.ucsb.edu/meshnet>.

[Microsoft 2005] Microsoft. **Microsoft Mesh Networking Academic Resource Toolkit**. 2005. Disponível em: <http://research.microsoft.com/mesh/>.

[Mirhakkak et al. 2000] Mirhakkak, M.; Schult, N. e Thomson, D. **Dynamic Quality-of-Service for Mobile Ad Hoc Networks.** In: *MobiHoc*, Boston, Massachusetts, EUA, p. 137–138, Agosto, 2000.

[Mohapatra et al. 2003] Mohapatra, P.; Li, J. e Gui, C. **QoS in Mobile Ad Hoc Networks**. *IEEE Wireless Communications*, v. 10, n. 3, p. 44–52, Junho 2003.

[Nortel 2005] Nortel. **Wireless Mesh Network Solution**. 2005. Disponível em: <http://products.nortel.com/go/solution_content.jsp?parId=0&segId=0&catId=-9227&prod_id=47160&locale=en-US>.

[NS-2 2000] NS-2. **The Network Simulator – ns-2**. 2000. Disponível em: <http://www.isi.edu/nsnam/ns/>.

[OpenWRT 2005] OpenWRT. **OpenWRT Wireless Freedom**. 2005. Disponível em: <http://www.openwrt.org>.

[Perkins et al. 2003] Perkins, C. E.; Belding-Royer, E. M. e Das, S. R. **Ad Hoc On-Demand Distance Vector (AODV) Routing**. Internet Engineering Task Force, Julho 2003. Disponível em: <http://rfc.net/rfc3561.txt>.

[ReMesh 2005] ReMesh. **Universidade Federal Fluminense**. 2005. Disponível em: <http://mesh.ic.uff.br>.

[Ruiz e Garcia 2002] Ruiz, P. M. e Garcia, E. **Adaptive Multimedia Applications To Improve User-Perceived QoS in Multihop Wireless Ad-hoc Networks**. In: *IEEE International Conference on Wireless LANs and Home Networks*, Atlanta, EUA, p. 673–684, Agosto, 2002.

[Sanzgiri e Belding-Royer 2004] Sanzgiri, K. e Belding-Royer, E. M. **Leveraging Mobility to Improve Quality of Service in Mobile Networks.** In: *MobiQuitous*, Boston, Massachusetts, EUA, p. 128–137, Agosto, 2004.

[da Silva 2006]da Silva, R. H. **Uma Implementação de um Sistema Adaptativo para Comunicação Multimídia em Redes Sem Fio Infra-estruturadas**. Dissertação de Mestrado — Instituto de Computação, Universidade Federal Fluminense, Niterói, RJ, Brasil, Outubro 2006.

[Sobrinho e Krishnakumar 1999] Sobrinho, J. L. e Krishnakumar, A. S. **Quality-of-Service in Ad Hoc Carrier Sense Multiple Access Wireless Networks**. *IEEE Journal on Selected Areas in Communications*, v. 17, n. 8, p. 1353 – 1368, Agosto 1999.

[Systems] Systems, A. **ISABEL CSCW application**. Disponível em: <http://www.agora-2000.com />.

[Cisco Systems 2005] Cisco Systems, I. **Cisco Wireless Mesh Networking Solution**. 2005. Disponível em: <http://www.cisco.com/go/wirelessmesh>.

[Tsarmpopoulos et al. 2005] Tsarmpopoulos, N.; Kalavros, I. e Lalis, S. **A Low-Cost and Simple-to-Deploy Peer-to-Peer Wireless Network based on Open Source Linux Routers.** In: *TRIDENTCOM*, Trento, Itália, p. 92–97, Fevereiro, 2005.

[VMesh 2005] VMesh. **Wireless Network Testbed at University of Thessaly, Volos, Greece**. 2005. Disponível em: <http://vmesh.inf.uth.gr/>.

[Xiao 2004] Xiao, Y. **IEEE 802.11e: QoS Provisioning at the MAC Layer**. *IEEE Wireless Communications*, v. 11, n. 3, p. 72–79, Junho 2004.

[Xue et al. 2003] Xue, J.; Stuedi, P. e Alonso, G. **ASAP: An Adaptive QoS Protocol for Mobile Ad Hoc Networks**. In: *14th IEEE International Symposium on Personal, Indoor and Mobile Radio Communications*, Beijing, China, p. 7–10, Setembro, 2003.

[Zhu et al. 2004] Zhu, H.; Li, M.; Chlamtac, I. e Prabhakaran, B. **A Survey of Quality of Service in IEEE 802.11 Networks**. *IEEE Wireless Communications*, v. 11, n. 4, p. 6–14, Agosto 2004.

Livros Grátis

(http://www.livrosgratis.com.br)

Milhares de Livros para Download:

Baixar livros de Literatura
Baixar livros de Literatura de Cordel
Baixar livros de Literatura Infantil
Baixar livros de Matemática
Baixar livros de Medicina
Baixar livros de Medicina Veterinária
Baixar livros de Meio Ambiente
Baixar livros de Meteorologia
Baixar Monografias e TCC
Baixar livros Multidisciplinar
Baixar livros de Música
Baixar livros de Psicologia
Baixar livros de Química
Baixar livros de Saúde Coletiva
Baixar livros de Serviço Social
Baixar livros de Sociologia
Baixar livros de Teologia
Baixar livros de Trabalho
Baixar livros de Turismo