UNIVERSIDADE FEDERAL DA BAHIA

INSTITUTO DE MATEMÁTICA

DEPARTAMENTO DE CIÊNCIA DA COMPUTAÇÃO

CARLOS EDUARDO BENEVIDES BEZERRA

QOS PARA JOGOS MULTIJOGADOR EM REDES IEEE 802.11

Salvador

2006

CARLOS EDUARDO BENEVIDES BEZERRA

QOS PARA JOGOS MULTIJOGADOR EM REDES IEEE 802.11

Monografia apresentada ao Curso de graduação em Ciência da Computação, Departamento de Ciência da Computação, Instituto de Matemática, Universidade Federal da Bahia, como requisito parcial para obtenção do grau de Bacharel em Ciência da Computação.

Orientador: Profº . George Marconi de Araújo Lima

Salvador

2006

RESUMO

Este trabalho faz um paralelo entre os problemas enfrentados em jogos multijogador, que são tratados como aplicações de tempo-real não-crítico, e os problemas relacionados com QoS em redes IEEE 802.11. É feito um estudo das vantagens de se utilizar redes sem fio IEEE 802.11 em jogos multijogador e como as soluções encontradas para esse tipo de rede podem ser aplicadas a esse tipo de jogo. O foco do trabalho é buscar e descrever abordagens que visam manter qualidade de serviço em redes IEEE 802.11, de forma a torná-las adequadas em relação aos requisitos temporais dos jogos multijogador. As diversas abordagens estudadas encontram-se em várias camadas, desde a camada da aplicação até modificação da camada física. Além de revisar o estado da arte, procurando dar ênfase aos aspectos temporais dos jogos multijogador, este trabalho traz também uma proposta, ainda em estágio preliminar, para a construção de um *framework*, cuja implementação poderia facilitar o trabalho de projetistas de jogos. Tanto a revisão do estado da arte na área quanto a proposta do *framework* são contribuições que poderão ser utilizadas em trabalhos futuros, ajudando, desta forma, no avanço da área.

Palavras-chave: IEEE 802.11, jogos multijogador, QoS, Wi-Fi, tempo-real

ABSTRACT

This work compares the problems faced in multiplayer games, which are treated as soft real-time applications, with the problems related to QoS in IEEE 802.11 networks. It is made a study of the advantages of using these networks in multiplayer games and how the researched approaches can be applied to this kind of games. The focus of this work is to search and describe solutions that try to maintain quality of service in IEEE 802.11 networks, adjusting them to the real-time constraints of multiplayer games. The different studied approaches can be divided in different layers, from the application layer to the physical layer. Besides reviewing the state of the art, focusing on the temporal aspects of multiplayer games, this work also brings a suggestion, still in a preliminary stage, to develop a *framework* which can make the work of game developers much easier. Both the review of the state of the art of this area and the suggestion of the *framework* are contributions that can be used in future works, helping, therefore, the advance of this area.

Keywords: IEEE 802.11, multiplayer games, QoS, Wi-Fi, real-time

LISTA DE FIGURAS

SUMÁRIO

1 INTRODUÇÃO

Jogos de um só usuário têm requisitos temporais, no que se refere a velocidade de processamento gráfico, execução de sons etc. Esses requisitos se baseiam no fato de que nos jogos há algum tipo de simulação que, quase sempre, ocorre simultaneamente à entrada de comandos dos jogadores. Jogos multijogador (*multiplayer*) são aqueles em que vários jogadores participam simultaneamente, geralmente cada um em um computador, conectados através de algum tipo de rede de comunicação. Esses jogos impõem ainda outros requisitos, desta vez sobre o suporte de rede, pois as mudanças de estado nos objetos que fazem parte do jogo devem ser transmitidas aos diferentes jogadores o mais rápido possível. Logo, a rede utilizada deve ser capaz de prover essa rapidez de comunicação exigida. Em outras palavras, jogos multijogador são sistemas de tempo real distribuídos, com requisitos já pesquisados e bem definidos (7), com problemas próprios inerentes à sua natureza (8). Alguns desses problemas são comuns à classe de sistemas de tempo real - como lidar com atraso e perda de pacotes, utilização da largura de banda disponível - e outros são específicos de jogos multijogador - como impedir que haja trapaça por parte de algum participante.

Com a popularização das redes sem fio, tem-se motivado a implementação de jogos multijogador em ambientes móveis. Jogos nesse tipo de rede são considerados uma fonte atraente de futuros lucros por desenvolvedores de jogos, fabricantes de dispositivos móveis e provedores de serviços. Além disso, o surgimento e evolução de novos paradigmas de comunicação - como redes sem fio *ad hoc* - oferecem novas formas e características únicas para aplicações móveis de tempo real, sendo jogos multijogador uma das vertentes dessa nova era (9). Vale a pena ressaltar que os problemas de comunicação existentes em projetos de jogos multijogador são amplificados em ambientes sem fio, pois estes estão mais sujeitos a interferência e falhas de comunicação.

Como aplicações distribuídas, jogos multijogador são e estão na vanguarda da utilização das possibilidades que as redes de comunicação oferecem. Apesar de ter havido pesquisas a respeito do tema em simulações militares, sistemas de realidade virtual e trabalho cooperativo

com suporte de computador, muitas dessas soluções divergem dos problemas propostos pelos jogos multijogador (8). Assim, faz-se necessária uma visão detalhada dos principais aspectos relevantes para este tipo de jogos, no que se refere à rede de comunicação, sendo este o principal objetivo desta monografia.

Os requisitos que estes jogos têm com relação à rede são fortes, necessitando de baixa latência com o mínimo possível de *jitter* (variação da latência) e uma baixa taxa de perda de pacotes, sendo então encarados como aplicações de tempo-real não-crítico (soft real-time). Dessa forma, é essencial que a rede dê suporte a *QoS* (qualidade de serviço) para satisfazer as necessidades de uma aplicação dessa natureza. É especialmente desafiadora essa tarefa quando se trata de redes sem fio, devido à grande mobilidade de seu nós, às características do canal de comunicação e, no caso de redes *ad hoc*, à falta de coordenação central da comunicação (10). As redes sem fio 802.11 não garantem nenhum desses requisitos (baixa latência, baixa perda de pacotes, e o mínimo de *jitter*), pelas seguintes razões:

- **Latência**: quando uma tentativa de envio de pacote falha, há uma espera aleatória - *backoff* -, tornando imprevisível a latência para uma transmissão de dados. De fato, seria interessante que, no mínimo, houvesse um limite superior para essa latência. O padrão IEEE 802.11 não estabelece esse limite;

- **Perda de pacotes**: o meio utilizado para transmissão - canal de rádio - não é confiável, pois está sujeito a interferência e a *colisões* (dois nós da rede transmitindo simultaneamente e prejudicando ambas as transmissões, por causa de interferência mútua entre elas). É necessário, pois, um tratamento especial para esse problema, buscando um protocolo de comunicação que minimize essa perda;

- ***Jitter*** (variação de latência): esse é um problema particularmente difícil de tratar em redes sem fio IEEE 802.11, pois seus nós podem se movimentar no espaço, variando a distância entre os mesmos. Pode ocorrer também de um nó deslocar-se para uma região onde a qualidade da transmissão piore ou melhore, devido a fatores geográficos. Além disso, mesmo que os nós da rede permaneçam estáticos, o canal de rádio pode sofrer interferências externas imprevisíveis. Por todos esses motivos, essas redes são muito suscetíveis a sofrerem variação na latência de suas transmissões. Faz-se, portanto, necessária a busca por uma solução para minimizar esse problema.

Além do que foi citado acima, há outras características que devem ser consideradas. Uma delas é a arquitetura de comunicação (que pode ser ponto-a-ponto, cliente/servidor, rede de servidores etc.), pois esta interfere nos aspectos temporais e no projeto do jogo. De fato, a

escolha do tipo de arquitetura poderá causar maior ou menor latência média de comunicação, assim como prover confiabilidade (no que se refere à possibilidade de trapaça dos jogadores na rede, o que pode ser mais facilmente controlado se houver um servidor intermediando toda comunicação), simplicidade de implementação, e distribuição de processamento. Outro fator com o qual um jogo multijogador deve lidar é com o problema da extensibilidade, ou seja, da capacidade que aquele sistema tem de suportar um número crescente de participantes (jogadores). Por último, mas não menos importante, um jogo multijogador deve ser tolerante a trapaça e vandalismo, ou seja, deve ser capaz de detectar e impedir trapaça por parte dos jogadores, além de não permitir que o sistema pare de funcionar devido a ações de participantes mal-intencionados.

Como pode ser percebido, projetar um jogo multijogador requer cuidados especiais, envolvendo desde questões temporais até aspectos de segurança. Esta monografia está focada nos aspectos temporais relativos à rede de comunicação. Em particular, abordam-se soluções de comunicação baseadas em redes sem fio IEEE 802.11. Ultimamente, tem crescido de forma dramática o número de dispositivos móveis e redes sem fio (9) e, por esta razão, deve-se estudar sua aplicabilidade em jogos multijogador.

Serão explicados os fatores acima citados e serão revisadas as abordagens propostas para que as redes dêem suporte satisfatório a jogos multijogador. Isso envolve questões no nível da aplicação, como utilização eficiente da largura de banda disponível, assim como questões no nível de enlace de dados e físico, como adaptação a flutuações de qualidade do canal. Algumas destas soluções são específicas para jogos multijogador, outras são comuns a diversos tipos de aplicações de tempo real. Busca-se, também, integrar as soluções pesquisadas, formando um conjunto que pode ser aplicado em diversas camadas de comunicação, desde aplicação até física. Por último, são expostas as conclusões a que se chegou e propostos caminhos a serem seguidos nessa área em trabalhos futuros.

2 JOGOS MULTIJOGADOR

Um jogo multijogador é um jogo eletrônico no qual várias pessoas podem jogar a mesma partida ao mesmo tempo. Diferente da maioria dos outros jogos, jogos de computador são freqüentemente atividades solitárias porque o poder computacional permite a criação de oponentes artificiais. Porém, em jogos multijogador, há vários jogadores que ou competem todos entre si, ou se agrupam em times para atingir um determinado objetivo comum, tal como derrotar um inimigo, que, por sua vez, pode ser ou um jogador - ou time - humano ou artificial simulado por computador. Geralmente, jogos multijogador utilizam redes de computadores para permitir aos jogadores jogarem entre si. Pode-se também utilizar um único sistema, com vários dispositivos de entrada, ao redor do qual os jogadores se reúnem (11), porém esse não é o objetivo deste trabalho. Aqui, será tratada a questão de jogos multijogador utilizando redes de computadores.

Assim sendo, jogos multijogador são geralmente referidos como jogos com vários jogadores, cada um em um computador diferente, através de uma *LAN* (*local area network*, rede de área local) ou através da Internet. Esse tipo de jogo é também conhecido como *netplay* (11). Jogar através de uma *LAN* quase sempre é preferível a jogar através da Internet, devido à alta latência (também conhecida como *lag*) e largura de banda muito inferior àquela de uma *LAN*, que já alcançou a marca de 10 Gbps, contra 44,7 Mbps das conexões de altíssima velocidade T3, para Internet (12, 13).

Existem vários tipos de jogos com opção de modo multijogador. Cada tipo de jogo multijogador tem seus pré-requisitos próprios para funcionamento adequado. Existem, inclusive, jogos que têm apenas a opção multijogador, sem a opção de jogo solitário. Exemplos desse tipo de jogo são os *MMOGs* (*massively multiplayer online games*, jogos online massivamente multijogador), como *World of Warcraft* (14), *Ragnarok* (15), *Priston Tale* (16) e *Silk Road* (17). Além desses, há também os jogos que podem ser jogados de maneira solitária, mas que foram criados preferencialmente para serem jogados em rede, tais como *Counter-Strike* (18) e *Battlefield: 1942* (19).

A seguir, são enumeradas e descritas algumas das categorias de jogos multijogador.

2.1 GÊNEROS

Cada um dos seguintes gêneros tem seus requisitos temporais próprios. Alguns, como os baseados em turno, não possuem requisitos que os tornem aplicações de tempo-real. Outros, no entanto, possuem requisitos fortíssimos. Aqui serão considerados os fatores: atraso de pacotes, perda de pacotes e *jitter*.

Segundo (20), os principais gêneros de jogos são: ação, rpg, plataforma, simulação, esporte e estratégia.

- **Ação**: caracterizados por ter ênfase em ações que os jogadores devem executar como reflexo, rapidamente, frente a situações que exigem tal rapidez de reação (20). Inclui jogos: *FPS* (*first-person shooting*, jogos de tiro em primeira pessoa), como *Unreal Tournament* (21), jogos de ação-aventura, como *Grand Theft Auto* (22) e jogos de luta, como *Soul Calibur* (23);

- **Plataforma**: jogos em que há gravidade para baixo, restringindo o jogador a movimentar-se sobre superfícies horizontais, também chamadas de plataformas, explorando algum ambiente (20). O jogo *Super Mario Bros.* (24) é o mais conhecido desse gênero;

- ***RPG***: sigla em inglês para *role playing game*, jogo de interpretação de papéis. Geralmente põe o jogador em um mundo de fantasia ou ficção científica e dirige o jogo através de uma rica e elaborada história. Na maioria destes jogos, o jogador toma o papel de um tipo de "aventureiro", que se especializa em certo conjunto de habilidades (tais como combate, ou uso de mágicas e encantamentos). O nome que se dá a cada tipo de aventureiro é "classe" e, normalmente, os jogadores podem controlar um ou mais desses personagens (20). *Diablo II* (25) pertence a este grupo. Inclui o subgênero *MMORPG* (*massively multiplayer online RPG*, RPG online massivamente multijogador), do qual faz parte *World of Warcraft* (14);

- **Simulação**: tem por objetivo simular uma determinada experiência, tal como dirigir uma aeronave, tão realisticamente quanto possível, levando em consideração leis da física e outras limitações do mundo real. Alguns desses jogos exigem bastante leitura antes mesmo que se tente jogar, enquanto outros carecem apenas de um simples tutorial (20). Apesar de ser mais voltado para o estilo de jogo solitário, inclui alguns subgêneros com opção

de modo multijogador. Um destes é combate espacial, sendo representantes desse grupo o jogo *Descent* (26) e seus sucessores;

- **Esporte**: emulam a prática de esportes reais tradicionais, tais como futebol, corrida, boxe, golfe, basquetebol, tênis, boliche etc. Alguns têm ênfase no jogo em si, enquanto outros são mais voltados à estratégia por trás do esporte (20). Um exemplo é o jogo de corrida *Need for Speed: Most Wanted* (27);

- **Estratégia**: tem como foco o planejamento cuidadoso e o gerenciamento eficiente de recursos de forma a alcançar a vitória, e são, por isso, chamados de "jogos de pensar" (14). Tem como subgêneros: jogos de estratégia baseados em turno, como *Heroes of Might and Magic* (28), e jogos de estratégia em tempo real (ou jogos *RTS*, *real time strategy*), sendo *Age of Empires* (29) um clássico desse subgênero.

2.2 REQUISITOS TEMPORAIS

Ação é o gênero mais básico de jogos e, certamente, o mais difundido (20). Nesses jogos, os jogadores "pensam menos e agem mais", pois compreendem, geralmente, ações como esquivar-se, atacar (com alguma arma, ou mesmo desarmado) e movimentar-se no ambiente virtual, em tempo-real, com a simulação dos personagens inimigos e dos outros elementos do jogo sendo feita simultaneamente à entrada de comandos do jogador. Alguns jogos incluem também buscar itens que auxiliem o jogador, como munição, novas armas, remédios que recuperem sua saúde instantaneamente etc. O gênero ação inclui, dentre outros, jogos *FPS* (*first person shooting*, jogos de tiro em primeira pessoa).

Em jogos *FPS*, o jogador controla um personagem que possui alguma arma com a qual atira em outros personagens, sejam estes simulados por computador ou controlados por outros jogadores. Geralmente, os ambientes virtuais são simulados em três dimensões. A câmera através da qual o jogador visualiza o ambiente 3D é posicionada próximo aos olhos da representação virtual do seu personagem, dando a impressão de estar "dentro" do mesmo. É daí que vem a expressão "jogos de tiro em primeira pessoa".

Segundo (30), os jogos *FPS* são os que mais exigem da capacidade de transmissão da rede de comunicação. Sendo assim, pode-se focar atenção neste tipo de jogo e em seus requisitos temporais. Os outros gêneros também têm seus próprios requisitos, mas não chegam a ser tão severos e sensíveis a falhas quanto os dos jogos *FPS*. Assim, garantir o cumprimento de seus requisitos temporais significa garantir cumprir, também, os requisitos de vários outros gêneros de jogos multijogador.

No que se refere a esses requisitos, o **atraso** de um pacote de rede com o estado de um personagem pode mudar drasticamente o resultado da ação do jogador, que poderia tomar decisões baseadas num estado antigo, que não é mais o estado corrente do personagem de outro jogador. Um exemplo seria:

> Alice e Bob estão jogando uma partida de *Unreal Tournament* via rede. Se Alice dispõe da localização mais atual do personagem de Bob, mas Bob está tendo problemas para receber os pacotes de Alice, ela terá vantagem. Bob não poderá saber onde o personagem de Alice está com certeza e, provavelmente, a representação do personagem dela estará no lugar errado no computador de Bob. O resultado seria Bob atirar num lugar onde ele equivocadamente pensa que Alice está, desperdiçando sua munição, enquanto Alice, com uma representação correta do personagem de Bob, atiraria no lugar certo, acertando e derrotando Bob.

A **perda de pacotes** geraria um problema semelhante ao do atraso. Porém, além de fazer com que seja visto um estado antigo dos personagens que são trafegados via rede, a perda de pacotes faz com que as transições entre um estado e outro de um personagem sejam bruscas. Os estados intermediários terão sido perdidos com os pacotes que não chegaram. Quando um personagem muda de posição, por exemplo, algum computador X na rede poderia receber apenas os estados inicial e final da movimentação, pois os intermediários teriam se perdido. Isso faria, na simulação em X, com que o personagem que se movimentou permanecesse parado na posição inicial - o que seria um estado inválido - e, ao chegar o pacote da posição final, ir instantaneamente à nova posição, sem ter percorrido o caminho entre os dois pontos.

O ***jitter***, por sua vez, também prejudica partidas de jogos *FPS* via rede. Se a comunicação durante um jogo *FPS* está sofrendo *jitter*, a capacidade de mirar em um alvo móvel é negativamente impactada (31). Isso ocorre porque, quando há atraso, o jogador geralmente prevê, ele mesmo, onde seu oponente estará quando o pacote com seu tiro chegar do outro lado da rede, baseado numa predição que ele faz, instintivamente, quando tem uma noção de qual o atraso médio (32). Se esse atraso varia, essa técnica que os jogadores usam não funcionará mais como deveria.

Além disso, em jogos com mecanismo de compensação de atraso de pacotes, como *Counter-Strike* (18), o *jitter* irá causar mau funcionamento desse mesmo mecanismo (34). Isso acontece porque, em *Counter-Strike*, que faz uso da arquitetura cliente-servidor, o mecanismo de compensação de atraso utiliza o valor do atraso médio das mensagens e um histórico de estados, da seguinte forma: quando uma ação é recebida pelo servidor, este examina o seu *timestamp*

Tabela 2.1: Requisitos temporais dos jogos (7, 33)

	atraso de pacotes	***perda de pacotes***
Jogos FPS (ação, de tiro em primeira pessoa)	< 150 ms	$< 3\%$
Jogos RTS (estratégia em tempo-real)	< 500 ms	$< 5\%$

(atributo da mensagem informando qual era o instante em que a ação ocorreu no cliente) e, com base no atraso médio daquele cliente que enviou a mensagem, calcula o que aquele jogador estava vendo naquele instante (32, 35). Para exemplificar:

> Sejam Alice e Bob novamente, dessa vez em uma partida de *Counter Strike*, com atrasos de 0 ms e 500 ms, respectivamente. No instante 30000 (em milissegundos, contados a partir do início da partida), Alice vê o personagem de Bob e atira nele. Porém, no instante 30000, Bob está em outro lugar, portanto Alice erraria. Contudo, o servidor tem armazenado no histórico que, no instante 29500, Bob estava, de fato, no lugar (pois Bob mandou no instante 29500 um pacote informando onde estava, com o *timestamp* 29500) onde Alice atirou no instante 30000. Dessa forma, baseado no atraso médio que ele obteve de Bob, que é de 500 ms, o servidor calcula que, no instante em que Alice atirou (30000), a representação de Bob - no computador de Alice - estava em posição para ser atingido. Bastou somar às ações de Bob o seu atraso médio. Dessa forma, o servidor determina que o personagem de Bob seja atingido e morto.

Percebe-se, claramente, que *jitter* traz conseqüências muito ruins para esse tipo de jogo, seja com ou sem mecanismos de compensação de atraso. Verificou-se ainda que, no jogo *Counter-Strike*, esse fator tem maior peso do que o próprio atraso em si no desempenho dos jogadores (34).

De acordo com (7, 33), pode-se considerar os valores da tabela 2.1 para esses requisitos (não foram encontradas definições de limite máximo para *jitter*).

3 REDES IEEE 802.11

Neste capítulo, será feita uma contextualização a respeito do crescimento e popularização das redes sem fio, particularmente redes IEEE 802.11. Além disso, será explicado porque é útil o trabalho de buscar adequar essas redes para um funcionamento desejável dos jogos multijogador, que, como será mostrado, têm crescido muito ultimamente.

Também será feita uma breve descrição do funcionamento do protocolo de acesso ao meio definido pelo padrão IEEE 802.11, pois seu entendimento é pré-requisito para a compreensão das adaptações sobre esse mesmo protocolo, que foram pesquisadas e serão descritas no capítulo 4.

Por fim, serão listados diversos problemas que dificultam a manutenção de qualidade de serviço nessas redes sem fio. Soluções possíveis para esses problemas podem ser encontradas no capítulo seguinte.

3.1 REDES IEEE 802.11 E JOGOS MULTIJOGADOR

Redes sem fio têm se popularizado bastante nos últimos anos. Aparelhos celulares são o maior exemplo de sua utilização. Mas existe um tipo específico de rede sem fio, definido pela IEEE, que são as redes IEEE 802.11, também conhecidas como *Wi-Fi*. Esse padrão foi desenvolvido para redes locais, chamadas, nesse caso, de *WLANs* (*wireless local area networks*, redes locais sem fio), tendo alcance limitado: antenas comuns, para usuários domésticos, têm alcance máximo de cerca de 90 metros, podendo ser aumentado se forem utilizadas antenas especiais (36).

Sua aplicação se estende desde computadores pessoais e laptops até dispositivos portáteis, como *PDAs* e *PALMTOPs*. Essas redes sem fio permitiram a criação de *hot spots*, um sistema composto de um ponto de acesso (que é um dispositivo sem fio ao qual os computadores podem se conectar e ter acesso a uma rede maior) com uma antena e conexão de banda larga à Internet. *Hot spots* estão disponíveis em aeroportos, hotéis e cafés. Um *hot spot* também pode ser um

conjunto de pontos de acesso, com raios de alcance sobrepostos em certas regiões, provendo uma grande área de conectividade.

Além disso, também é possível que vários dispositivos se conectem sem que haja a necessidade de algum ponto central que intermedie a comunicação. Nesse caso, seria uma rede sem fio *ad hoc*, que é particularmente útil para jogos multijogador. Com ela, basta que os jogadores liguem seus computadores perto o suficiente uns dos outros e configurem alguns parâmetros das interfaces de rede sem fio. Daí então, as próprias interfaces se encarregarão de criar a rede.

A comunicação sem fio já está disponível até nos aviões. A *Lufthansa* foi a primeira a colocar em funcionamento o acesso sem fio nos vôos (37). Por enquanto, somente os *notebooks* fornecidos pela companhia podem utilizar a rede. A empresa espera em breve conseguir, no Departamento de Aviação Civil, a liberação para o acesso de qualquer *notebook* que utilize o padrão *Wi-Fi*.

A utilização de redes sem fio IEEE 802.11 traz diversas vantagens, tais como (38):

- não é necessária a instalação e utilização de fios;
- é uma maneira rápida, econômica e flexível (permite conexão de diversos dispositivos, de plataformas diferentes) de se montar uma rede;
- é um padrão largamente utilizado e dispositivos que o implementam devem ser certificados pela *Aliança Wi-Fi* (39), que padroniza e garante interoperabilidade de aparelhos de diferentes fabricantes que utilizem essa tecnologia;
- os dispositivos conectados podem movimentar-se com muito mais flexibilidade do que em redes cabeadas; é possível, inclusive, mudar de ponto de acesso de forma automática;
- o padrão é reconhecido em todo globo, operando igualmente em qualquer país, diferente, por exemplo, das tecnologias para telefones celulares;
- há diversos *hot spots* espalhados pelo mundo, permitindo conexão à Internet bastando ter um dispositivo equipado com interface de rede compatível com IEEE 802.11 e estar dentro do raio de alcance de algum ponto de acesso.

Além das vantagens apresentadas, redes IEEE 802.11 são especialmente interessantes em *lan-parties*, que consistem em encontros de vários jogadores - e seus respectivos computadores -, onde é montada uma *LAN* (rede local), geralmente para jogos (40). Usualmente, é necessária toda uma estrutura de cabos de rede e instalação de algum *hub*/*switch*, que faça a interconexão

das máquinas. Porém, se cada uma dispuser de um dispositivo de rede IEEE 802.11, basta que seja feita uma configuração adequada e toda a comunicação será feita por ondas de rádio.

Além disso, há diversos dispositivos para jogos que utilizam esse tipo de comunicação para se interconectarem sem o uso de fios. Alguns exemplos são:

- O *Nintendo DS* (41) (aparelho portátil para jogos eletrônicos) é compatível com *Wi-Fi* para conexão com outros dispositivos;
- O *Sony PSP* (42), outro dispositivo de jogo portátil, inclui *Wi-Fi* para se conectar a *hot spots* e fazer conexões sem fio a outros aparelhos;
- O *Xbox 360* (43), da *Microsoft*, possui um adaptador sem fio IEEE 802.11 para conectar-se a redes, inclusive sendo possível a criação de *lan-parties* compostas unicamente de vídeo games desse tipo (40);
- O modelo *premium* do *PlayStation 3* (44), *video-game* criado pela *Sony*, possui *Wi-Fi*;
- O *Nintendo Wii* (45), *video-game* fabricado pela *Nintendo* também inclui *Wi-Fi*.

Como se pode perceber, a popularização de redes sem fio IEEE 802.11 aponta na direção da aplicação dessas redes para jogos multijogador. Aliás, como foi listado acima, diversas empresas de jogos eletrônicos estão incluindo essa tecnologia de comunicação em seus aparelhos, o que reforça essa tendência. Sendo assim, é razoável estudar a aplicabilidade dessas redes frente aos requisitos específicos de jogos multijogador e buscar maneiras de garantir o cumprimento dos mesmos ou de, pelo menos, tornar o funcionamento das redes IEEE 802.11 o mais próximo possível do ideal para esses jogos.

3.2 O *DCF*

As redes IEEE 802.11 são redes sem fio que utilizam ondas de rádio para comunicação entre seus nós. Porém, existem órgãos internacionais que regulam a transmissão de comunicação em determinadas faixas de freqüência. Assim, é necessária a posse de licenças para fazer transmissões em determinadas faixas. Uma exceção dessa regra são as faixas *ISM* (*industrial, scientific and medical*, para uso médico, científico e industrial), para as quais não é necessário obter licenças. As redes IEEE 802.11 operam nessas faixas, utilizando freqüências entre 2,4 e 2,5 GHz para suas transmissões (com exceção do IEEE 802.11a, que utiliza uma freqüência em torno de 5 GHz).

Porém, se dois nós da rede transmitirem dados ao mesmo tempo, na mesma freqüência, as transmissões causarão interferência uma na outra e isso prejudicará ambas. Por isso, é necessário minimizar a ocorrência de situações desse tipo, além de garantir que o meio será utilizado de maneira eficiente.

Como em outras redes de comunicações, o padrão IEEE 802.11 define um protocolo de controle de acesso ao meio (*MAC*, *medium access control*). Esse protocolo contém duas funções de coordenação: uma é distribuída (*DCF* (36), *distributed coordination function*), podendo ser utilizada em redes *ad hoc* - redes em que não há um controle centralizado - e a outra é centralizada, executada por um ponto (*PCF*, *point coordination function*). Na primeira, os nós da rede competem pelo acesso e aquele que ganhar transmite o que tiver que transmitir. Na segunda, algum dispositivo computacional é encarregado de escalonar e dividir o acesso ao meio entre os diferentes nós. Periodicamente, ele pede a cada um dos nós que transmita seus dados.

O *DCF* é definido pelo padrão IEEE 802.11 da seguinte forma: quando um nó da rede sem fio tem um pacote a ser enviado, ele verifica o meio e, se este tiver permanecido livre por um intervalo de tempo superior a um *DIFS* (*distributed coordination function interframe spacing*, espaçamento entre quadros da coordenação distribuída), ele transmite imediatamente aquele pacote. Caso contrário, ele espera que o meio fique livre, decorra-se o *DIFS* e entra em modo de espera aleatória.

A espera aleatória, conhecida como *backoff*, consiste em esperar um tempo adicional (chamado de janela de contenção) após o *DIFS*, tempo esse pertencente ao intervalo (0, $Tslot \times 2^n + Tinit$). $Tslot$ é a duração de um *slot* (um *slot* é uma espécie de "vaga" no tempo, na qual pode ser iniciada uma transmissão); n é o número de tentativas que já foram feitas para aquele pacote a transmitir e $Tinit$ é o tamanho mínimo da janela de contenção. O valor de $Tinit$ depende da camada física. Quando a transmissão de um pacote falha, é agendada uma nova tentativa de envio, dessa vez com uma janela de contenção maior, pois o valor de n aumentou. No entanto, a janela não cresce indefinidamente: quando atinge um determinado limite, a janela de contenção permanece constante, não importando o número n de tentativas. Uma descrição visual e mais fácil de ser compreendida pode ser vista na figura 3.1. Os valores da janela de contenção dessa figura são aqueles definidos para a camada física que utiliza transmissão com *DSSS* (*direct sequence spread spectrum*, espectro de difusão de seqüência direta).

O nó que acaba de entrar em modo de espera aleatória seleciona um *slot* da janela de contenção para transmitir seu pacote. Se o canal continuar livre até a chegada daquele *slot*, a transmissão é feita, já que se supõe que outros nós selecionaram, aleatoriamente, um *slot* mais

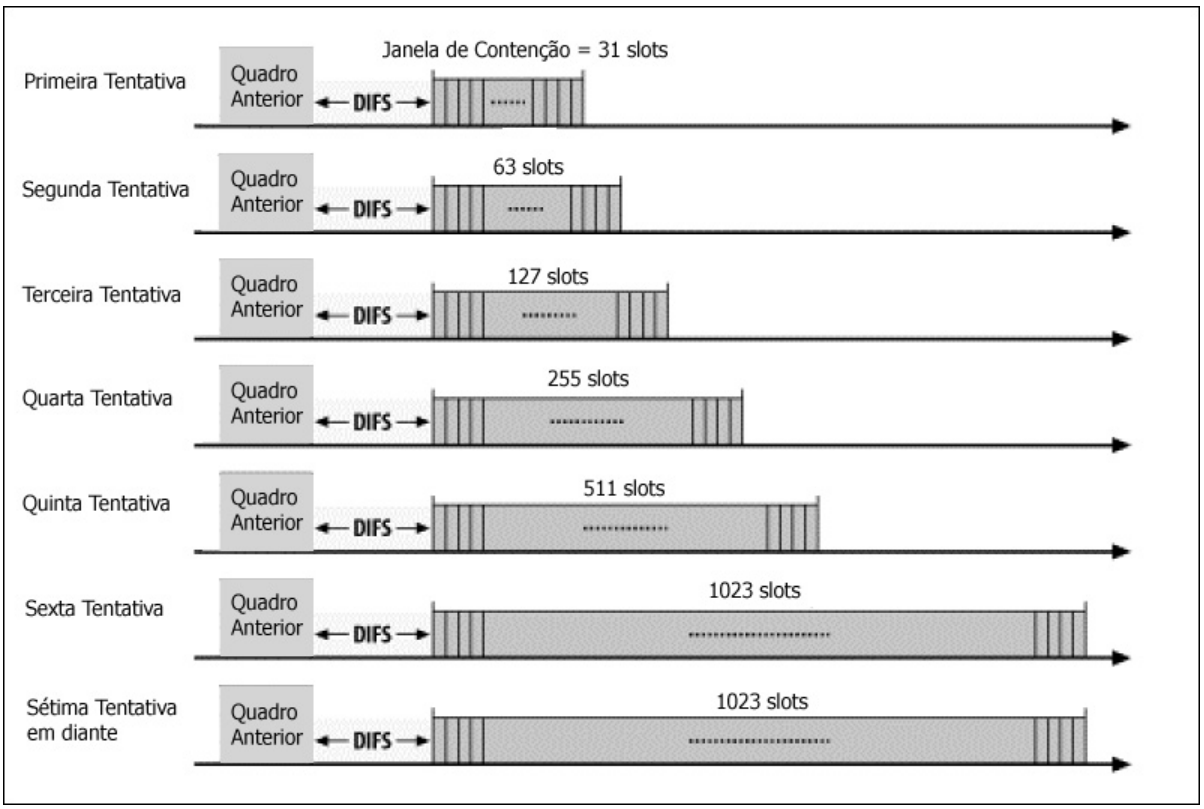

Figura 3.1: Espera aleatória com janela de contenção (1).

distante e, conseqüentemente, um intervalo maior de tempo para esperar. Se o número *n* de tentativas atingir um determinado limite, o pacote é descartado.

Quando é transmitido um conjunto de dados através da rede, o transmissor espera uma confirmação positiva (*positive acknowledgment*, ou *ACK*) vinda do receptor. Quando essa confirmação não chega, assume-se que houve erro e tenta-se refazer a transmissão.

Os quadros enviados pela camada de acesso ao meio são protegidos contra erros por um código de redundância cíclica, de forma que cada quadro é verificado no receptor para garantir que esteja de acordo com esse código. Se não estiver, percebe-se que o quadro foi corrompido na transmissão - seja por colisão com outro quadro ou por interferência externa - e não é enviado o *ACK*.

É fácil perceber que dois ou mais nós da rede podem escolher um mesmo intervalo de tempo para esperar após a passagem do *DIFS* (i.e., selecionam o mesmo *slot*). Isso pode, e irá, causar colisões, resultando em retransmissões e eventuais descartes de pacotes cujas transmissões falhem muito.

Outro detalhe é que o acesso ao meio, no caso de haver vários nós disputando, é decidido de maneira aleatória, dando prioridade, na média, a transmissões que foram enfileiradas

mais recentemente. Pacotes mais antigos, e que estão tentando novamente ser enviados, tem a possibilidade de esperar tempos maiores do que pacotes que estão tentando ser enviados pela primeira vez.

Existe também a possibilidade de haver nós escondidos (1). Nós escondidos são nós localizados dentro do raio de comunicação do receptor, porém fora do alcance do transmissor. Durante a transmissão, esse nó escondido não será capaz de perceber que o canal está ocupado e poderá enviar alguma coisa através do mesmo. Isso causará interferência no nó receptor, impedindo-o de receber os dados corretamente. Esse tipo de perturbação pode ser ignorado se é utilizada a técnica de *RTS*/*CTS* (*request to send*/*clear to send*, pedido para enviar/limpo para enviar), ou quando um ponto de acesso intermedia a comunicação, em uma rede operando em modo de infra-estrutura (5).

A técnica de *RTS*/*CTS* consiste em, antes de ser iniciada uma transmissão, o transmissor enviar um quadro *RTS* para o receptor que responde com um quadro *CTS* se o meio estiver desocupado para ele. O objetivo é fazer com que os nós vizinhos ao transmissor e ao receptor sejam informados de que haverá uma transmissão, ao perceberem o quadro *RTS* e o quadro *CTS*, respectivamente. Isso é possível porque quase todos os quadros da camada de acesso ao meio do padrão IEEE 802.11 contém uma informação de quanto tempo será necessário para que a transmissão seja concluída. No caso de quadros *RTS*, esse tempo inclui, além do próprio quadro *RTS*, todos aqueles que se esperam transmitir até que seja concluída a operação iniciada pelo ciclo *RTS*/*CTS*. Quando um nó da rede percebe um quadro sendo transmitido, aloca um *NAV* (*network allocation vector*, vetor de alocação da rede), que é o tempo durante o qual terá que esperar para poder acessar o meio. Na figura 3.2 esse funcionamento é ilustrado.

Para que o ciclo *RTS*, *CTS*, quadros de dados, *ACK* não seja interrompido por outra estação, o intervalo de tempo que se espera para transmitir quadros *CTS* e quadros *ACK* é o *SIFS* (*short interframe spacing*, espaçamento curto entre quadros). O *SIFS* tem duração menor que o *DIFS*. Assim, se há duas estações prestes a transmitir algo, sendo uma com um quadro *ACK*, por exemplo, e outra com um quadro de dados comum, a primeira acessa o meio antes, tendo maior prioridade. Porém, o quadro *RTS* tem de esperar um *DIFS*, também. A técnica de *RTS*/*CTS* é opcional e o quadro *RTS* tem tanta prioridade quanto quadros de dados comuns, pois marca o início de uma transmissão de dados.

Além do *DIFS* e do *SIFS*, há outro espaçamento entre quadros, conhecido como *PIFS* (*PCF interframe spacing*, espaçamento entre quadros do *PCF*). O *PIFS* tem duração maior que o *SIFS* e menor que o *DIFS*, dando maior prioridade a quadros que o utilizem. Ele é utilizado na função de coordenação *PCF*, dando maior prioridade a determinados quadros de acordo com

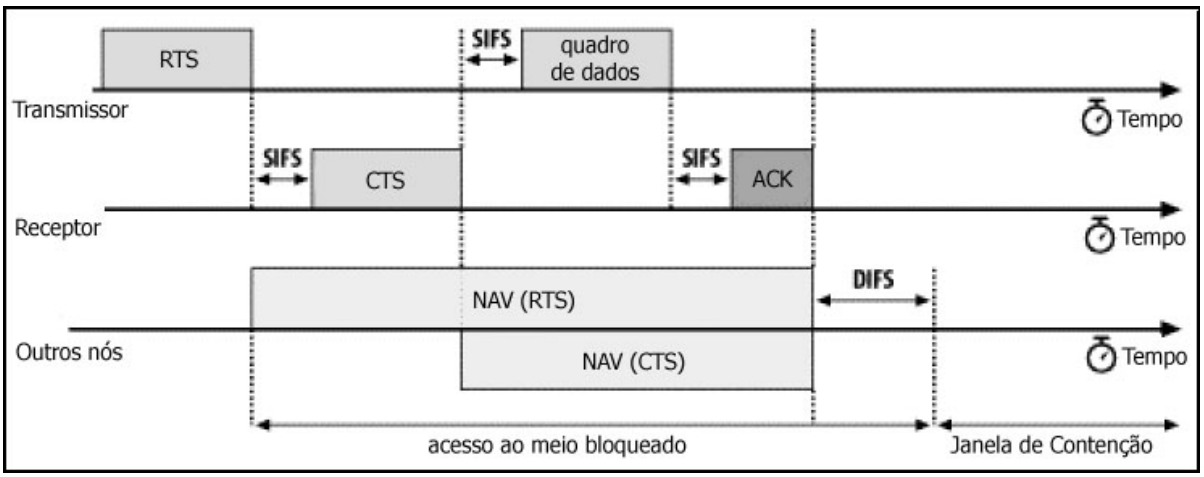

Figura 3.2: Funcionamento da técnica *RTS*/*CTS* (1).

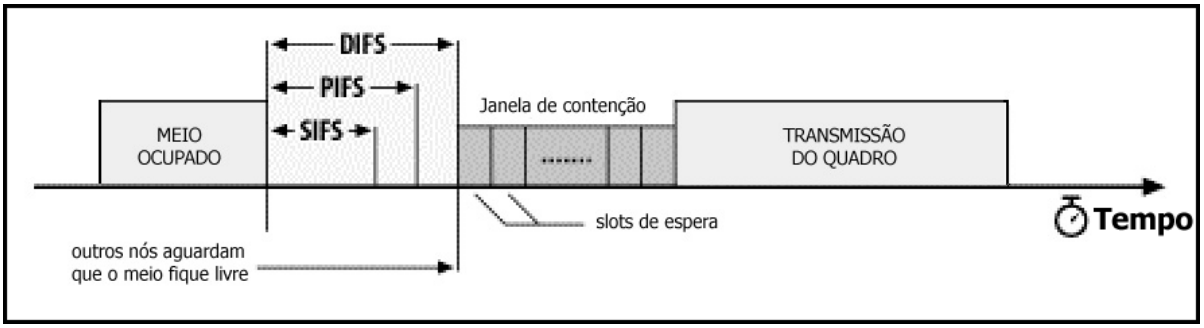

Figura 3.3: Diferentes espaçamentos entre quadros (1).

o que for decidido com o protocolo, que não será descrito aqui. Na figura 3.3, pode-se perceber as diferenças entre esses três espaçamentos diferentes.

3.3 PROBLEMAS QUE DEVEM SER RESOLVIDOS

Pelo que foi visto no capítulo 2, jogos multijogador são aplicações de tempo real com requisitos temporais bem definidos. O que precisa ser feito, então, é buscar alguma forma de prover *QoS* para esse tipo de aplicação, de forma que funcione de maneira adequada, permitindo aos jogadores uma experiência com o mínimo possível de problemas no que se refere à comunicação entre os nós da rede em que o jogo opera.

Porém, para garantir essa qualidade de serviço, é necessário atingir determinados objetivos. Esses objetivos incluem lidar com:

- atraso de pacotes;

- perda de pacotes;
- nível de utilização da largura de banda disponível;
- variação no atraso das transmissões (também conhecido como *jitter*).

Todos esses problemas tornam-se mais complicados de se resolver quando se trata de redes IEEE 802.11, que geralmente operam em modo *half-duplex* (1), sem detecção determinística de colisões e com grande vulnerabilidade a interferências e falhas de transmissão.

Um problema das redes IEEE 802.11 em relação a tráfego de dados com restrições temporais é a falta de um limite determinístico para o atraso dos pacotes. Isso se deve ao fato de que essas redes utilizam espera aleatória (*backoff*) entre uma tentativa de envio de dados mal sucedida e a subseqüente nova tentativa. Assim sendo, fica difícil escalonar as mensagens e cumprir o requisito de tempo máximo daquele conjunto de dados. Portanto, é necessário modificar - ou estender - o protocolo da camada de enlace de dados do padrão IEEE 802.11 de forma a eliminar ou minimizar esse problema.

Foi observado também que o protocolo da camada de enlace de dados do padrão IEEE 802.11 pode gerar um número alto de perda de pacotes devido a sobrecargas nos *buffers* das interfaces de rede dos nós (4). Isso acontece porque o canal de rádio - no qual essas redes operam - é muito suscetível a interferências e falhas, fazendo com que a sua confiabilidade varie bastante em função do tempo e do espaço. Quando uma mensagem de rede não é confirmada pelo receptor, o transmissor enfileira novamente aquela mensagem no *buffer* para envio. Em um ambiente propício a falhas, essa característica faz com que a perda de mensagens seja ainda maior do que o esperado, devido a sobrecarga desses *buffers*. Logo, esse protocolo deve ser otimizado, de forma a limitar o número de reenvios e fazer melhor uso do canal de comunicação (4). Essa limitação seria dinâmica, de acordo com a qualidade do canal naquele instante.

A utilização de largura de banda também deve ser otimizada, pois ela limita a quantidade de dados que podem ser trocados entre nós da rede por unidade de tempo. Violações desse limite causam mais atraso e perda de pacotes além da média em operação normal. Arquiteturas de rede devem ser construídas considerando a quantidade de largura de banda disponível (46). Nas redes IEEE 802.11, o canal através do qual os nós se comunicam - o canal de rádio - tem uma qualidade muito suscetível a variações, devido a interferências e fatores geográficos que podem tornar a comunicação mais difícil. Por exemplo, se um nó quer se comunicar com outro que se encontra a uma grande distância, o valor do *SNR* (*Signal to Noise Ratio*, razão entre sinal e barulho) resultante no receptor pode ser muito baixo, se uma alta taxa de transmissão é utilizada (5). Isso resultaria em uma alta probabilidade de ocorrência de erros na transmissão.

Ocorrendo erros na transmissão, aquela mensagem é novamente enfileirada para retransmissão, o que diminui a eficiência da utilização da largura de banda disponível.

Em tais situações, é necessário encontrar formas de adaptar-se a essas flutuações de qualidade do canal, de forma a melhorar o desempenho da comunicação, mesmo que seja necessário diminuir a taxa de transmissão ou descartar determinados pacotes. Isso aparentemente tornaria a comunicação mais lenta, mas o resultado seria, na verdade, a melhora do sistema como um todo (4).

Uma das formas de prover *QoS* para aplicações que funcionam em redes sem fio é oferecer diferenciação de serviços. Porém, nessas redes, onde a largura de banda é escassa e as condições do canal são variáveis, a diferenciação de serviços na camada de rede não será ótima se não houver suporte das camadas de mais baixo nível (6). Dessas camadas inferiores, a que oferece mais espaço para desenvolvimento de esquemas como esse - de diferenciação de serviços - é a camada de enlace de dados. Assim sendo, um dos problemas é buscar essa implementação no nível de enlace.

É especialmente complicado prover qualidade de serviço em redes sem fio *ad hoc* móveis (*MANETs*), que são redes sem fio auto-configuráveis, independentes de qualquer infra-estrutura centralizada, que podem conter dispositivos de diferentes tipos (telefones celulares que utilizem o padrão IEEE 802.11, *PDAs*, *laptops*, etc.) (46). Nela pode haver roteadores móveis, conectados também por canal de rádio, formando uma topologia arbitrária. Os roteadores estão livres para movimentarem-se aleatoriamente e organizarem-se de forma dinâmica. Assim sendo, a topologia de uma rede desse tipo pode se alterar rapidamente e de forma imprevisível (33, 47). Para manter *QoS* em redes *ad hoc* móveis, é necessário lidar com uma série de desafios (33):

- Mobilidade - A topologia da rede *ad hoc* pode mudar imprevisivelmente, resultando em *links* quebrados e rotas que não funcionam;
- Congestionamento - Tráfego de tempo-real deve chegar dentro do prazo, mesmo que a rede esteja altamente carregada;
- Meio Compartilhado - Redes sem fio operando em modo *ad hoc* não dão nenhuma garantia de *QoS*, devido ao acesso ao meio ser baseado em mecanismos de contenção e espera aleatória (*backoff*);
- Sinal de Rádio - O sinal de rádio sofre atenuação e interferência, o que dá lugar a zonas cinza (48) (zonas cinza se formam quando mensagens *HELLO*, enviadas através de *broadcast*, são recebidas e indicam alcançabilidade, mas não se consegue trocar pacotes enviados por *unicast*) e freqüente retransmissão.

4 SOLUÇÕES PARA OS PROBLEMAS APRESENTADOS

Neste capítulo, serão apresentadas as diferentes soluções pesquisadas para cada tipo de problema encontrado. Existem soluções no nível de aplicação e soluções em baixo nível (camada de enlace e camada física), estas últimas se adequando tanto ao problema do trabalho quanto a outras aplicações de tempo real. Essa característica pode possibilitar a sugestão de uma pilha de soluções, separadas por camadas da arquitetura de rede, desde a camada de aplicação até a camada física. Um exemplo disso é o problema de utilização eficiente de largura de banda, que tem soluções específicas para jogos, como também soluções de baixo nível, para melhoria do desempenho em redes IEEE 802.11 de forma geral.

4.1 SOLUÇÕES NO NÍVEL DE APLICAÇÃO

Para os problemas apresentados no capítulo anterior, que são de fundamental importância para que seja mantido *QoS* e, portanto, garantir o cumprimento dos requisitos temporais, existem soluções no nível de aplicação, independentemente de em que tipo de rede operam. É importante ressaltar que essas soluções, que minimizam o uso de largura de banda, tendem a reduzir o atraso e a perda de pacotes, dentre outros motivos, por diminuir a sobrecarga de *buffers* e por deixar o canal livre por mais tempo.

As abordagens pesquisadas foram:

- **compressão e agregação de pacotes**, que visam diminuir o uso de largura de banda pela aplicação;
- **gerenciamento de interesse**, que tem por objetivo enviar determinadas atualizações de estado apenas a nós para os quais as mesmas sejam relevantes;
- ***dead reckoning***, que é uma técnica de extrapolação que permite enviar pacotes em intervalos maiores e extrapolar os estados subseqüentes ao último estado recebido através da

rede, até que chegue uma nova atualização.

Essas soluções serão descritas a seguir.

4.1.1 COMPRESSÃO E AGREGAÇÃO DE PACOTES

Para reduzir o uso de largura de banda, pode-se comprimir e/ou agregar os pacotes de rede (2). A idéia de comprimir os pacotes vem da necessidade de diminuir a quantidade de bits que precisam ser transmitidos, reduzindo, assim a largura de banda utilizada. Essa compressão pode ser com perdas ou sem perdas. Em caso de transmissão de vídeo, por exemplo, pode haver uma certa perda de dados sem comprometer muito a qualidade da recepção, pois pode-se descartar informação irrelevante (2). No caso de jogos, isso não é desejável, pois os pacotes costumam ter informações a respeito de: posição, velocidade, direção, ações que o jogador executou etc., de forma que a perda de alguma dessas informações pode comprometer a experiência do jogo.

Contudo, existe um tipo de compressão que consiste em, após ser enviado um pacote com todas as informações de estado do jogador, permanecer enviando pacotes que contenham apenas as variações daquele estado (2). No entanto, esse tipo de compressão exige o uso de um protocolo confiável, de forma a não serem perdidos pacotes pela rede. O problema é que, geralmente, utiliza-se o protocolo *UDP* para transmissão em jogos (49) e esse protocolo não garante entrega dos pacotes.

Pode-se utilizar também agregação dos pacotes, que consiste em combinar as informações que seriam transmitidas em diversos pacotes, em apenas um grande pacote. Dessa forma, a sobrecarga causada pelos cabeçalhos pode ser bastante reduzida (pacotes *UDP* e *TCP* têm, respectivamente, 28 e 40 bytes de cabeçalho). Existem duas abordagens para executar essa técnica (2): *baseada em tempo* e *baseada em quórum*. A primeira permite que vários pacotes a serem transmitidos sejam armazenados num espaço de memória durante um certo período. Ao final desse período, todos os pacotes armazenados são combinados em apenas um e enviados através da rede. O problema dessa abordagem é que, se naquele período não houve um número significativo de pacotes armazenados, não há ganhos significativos em desempenho. A outra abordagem, baseada em quórum, permite que todos os pacotes sejam guardados até que seja atingida uma determinada quantidade, sendo, então, combinados e enviados. O problema dessa abordagem é que o tempo necessário para atingir o número mínimo de pacotes pode ser longo demais para o bom andamento do jogo. A alternativa, então, seria uma abordagem híbrida, em que qualquer das condições acima descritas dispararia a combinação e envio dos pacotes até então armazenados.

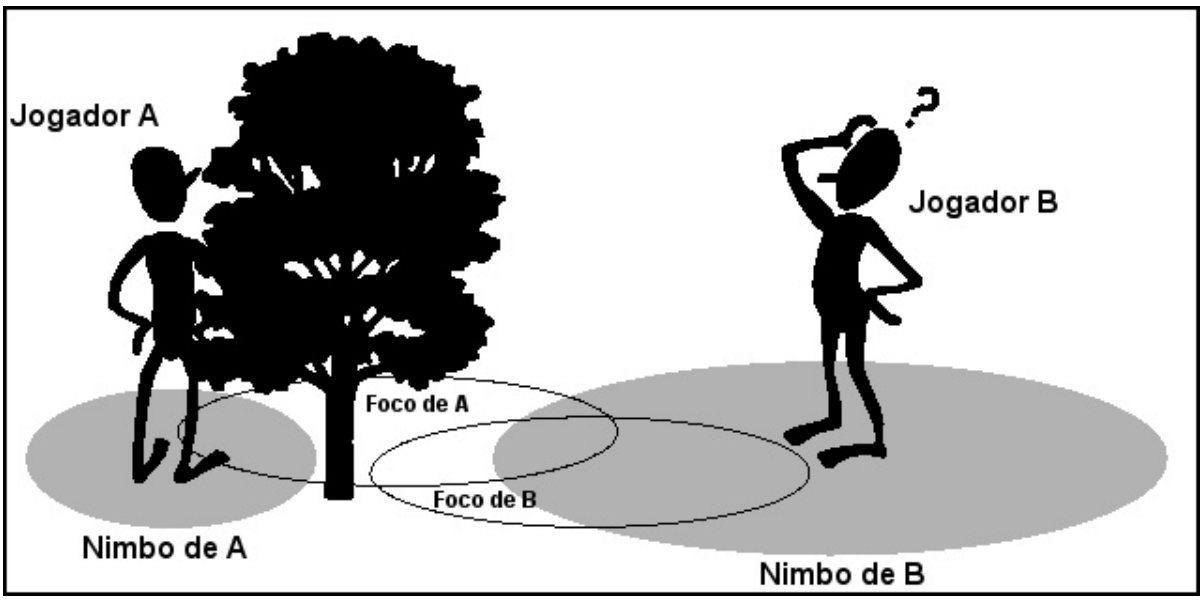

Figura 4.1: Gerenciamento de interesse com uso de foco e nimbo (2).

4.1.2 GERENCIAMENTO DE INTERESSE

As entidades presentes no jogo possuem estados, cuja mudança é dada a conhecer, geralmente, a todos os nós da rede. No entanto, em um determinado momento, a atualização de estado de uma entidade pode ser relevante apenas para uma minoria desses nós. Assim sendo, uma maneira óbvia de reduzir a utilização de largura de banda é fazer com que os pacotes de atualização de estado de entidade sejam enviados apenas àqueles nós onde há interesse naquelas informações (2). Uma forma de implementar isso é com *áreas de interesse*. Essas áreas são subespaços do ambiente virtual onde há interação dos jogadores. Quando dois jogadores estiverem posicionados no jogo, de forma que haja intersecção entre suas áreas de interesse, eles passam a enviar pacotes de estado um ao outro.

Uma forma de refinar essa técnica seria subdividir a área de interesse em *foco* e *nimbo*. O foco seria o subespaço que o jogador percebe; o nimbo seria a área dentro da qual o jogador é perceptível. Para que o jogador A perceba o jogador B, é necessário que haja uma intersecção entre o foco de A e o nimbo de B. Dessa forma, é possível que A perceba B, mas B não perceba A, vide figura 4.1.

Em um esquema que utiliza filtragem de pacotes por áreas de interesse, pode-se implementar um *gerenciador de subscrição*. Esse gerenciador se encarregaria de receber subscrições, que seriam pacotes contendo as informações dos interesses de cada nó da rede (seus focos e seus nimbos) e os pacotes com mudança de estados (esse método é mais adequado para jogos multijogador com arquitetura cliente-servidor). Aí então, bastaria encaminhar a cada nó apenas

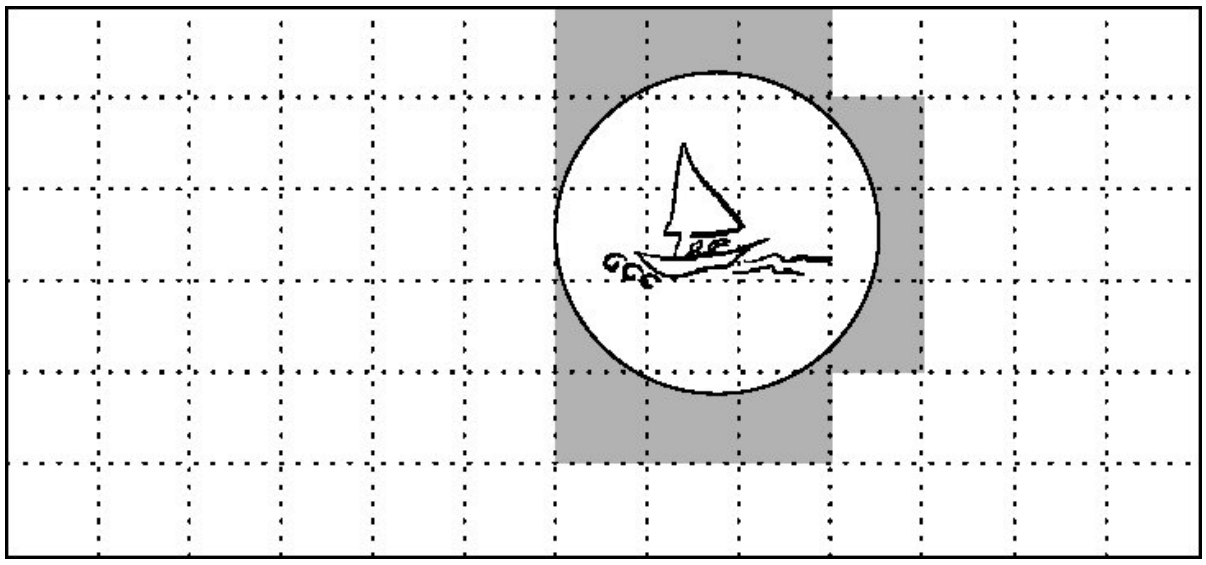

Figura 4.2: Divisão do ambiente virtual em regiões (2).

aqueles pacotes que lhe fossem relevantes.

Essa técnica é chamada de *filtragem intrínseca*, pois o processamento da informação de para onde deve ir cada pacote é feito no nível da aplicação. Outra forma seria a *filtragem extrínseca*; uma maneira de executá-la seria por meio de *multicast*. Com *multicast*, a filtragem dos pacotes seria feita pela própria rede, tendo um processamento mais rápido, pois seria baseado apenas no endereço de rede de cada nó, e não em complexos cálculos de intersecção de subespaços. O protocolo de *multicast* consiste em determinar grupos para cujos integrantes são enviados os pacotes de rede. O desafio, nesse caso, seria determinar, no nível da aplicação, como seriam esses grupos.

Uma forma seria com *agrupamento por objeto*, com cada objeto no ambiente virtual tendo um grupo de *multicast* associado a ele, para cujos nós integrantes seriam enviados pacotes de atualização de estado daquele objeto. Para tanto, seria necessário que, de acordo com a movimentação dos jogadores, o jogo os subscrevesse nos grupos de *multicast* dos objetos relevantes.

Outra forma seria com *agrupamento por região*. Nessa abordagem, o ambiente virtual seria divido em regiões e, de acordo com a movimentação de cada jogador, este seria subscrito automaticamente nos diferentes grupos de *multicast*. Aquelas regiões que tivessem intersecção com seu foco seriam aquelas em cujos grupos de *multicast* ele seria subscrito. A figura 4.2 ilustra um exemplo disso.

4.1.3 *DEAD RECKONING*

A expressão em inglês "dead reckoning" (2, 31, 50) significa, literalmente, "cálculo morto". Tem o termo "deduced reckoning" (cálculo deduzido) como uma de suas prováveis origens (51). *Dead reckoning* é uma técnica de extrapolação inicialmente desenvolvida para o domínio da aviação para estimar a posição atual de uma aeronave baseado na última posição conhecida e no vetor movimento (50). Dessa forma, pode-se deduzir, a partir de cálculos, qual a próxima posição que será ocupada pelo objeto que está se deslocando.

Com essa técnica, pode-se diminuir consideravelmente o uso de largura de banda, pois é viável transmitir pacotes com menor freqüência. Enquanto um pacote de atualização de estado não chega para uma determinada entidade no ambiente virtual, o jogo pode ir calculando, a cada quadro da renderização gráfica do jogo, onde estaria aquele objeto, se continuar na mesma rota, com as mesmas características de velocidade e aceleração. O *dead reckoning* mostrou-se bastante eficaz, pelo menos, para cálculo de posição (2).

Obviamente, esse cálculo pode ter um resultado incorreto, se o jogador remoto que controla aquela entidade em movimento a fizer mudar de direção ou velocidade de forma imprevisível. Isso resultaria numa mudança brusca de posição daquele objeto, quando fosse recebido um pacote de atualização de estado. Mesmo assim, podem-se aplicar técnicas de convergência para que a transição entre a posição calculada e a posição real recém recebida via rede seja feita de forma mais suave. De fato, *dead reckoning* consiste de duas partes: uma *técnica de predição* e uma *técnica de convergência* (2).

A técnica mais comum de predição é a utilização de polinômios com informação de posição, velocidade, aceleração etc., de forma a permitir predições mais exatas. No entanto, com polinômios de grau muito elevado (que incluem aceleração, variação da aceleração, etc.), a predição é muito sensível a erros dos termos de grau elevado, de forma que acaba se tornando inexata. O mais usual de se fazer é incluir posição, velocidade e aceleração (polinômio de grau 2). Mesmo assim, porém, quando se verifica que, em um jogo específico, há tendência a grande variação de aceleração, usa-se apenas a informação de posição e velocidade para efetuar a predição.

Uma forma de economizar ainda mais o uso de largura de banda seria o nó original, que enviou o pacote com estado da entidade, executar *dead reckoning* sobre essa mesma entidade, que é sua. Assim, só seriam enviados novos pacotes de atualização quando fosse percebido que a diferença entre a posição calculada e a posição real estivesse acima de um determinado limite. O problema, nesse caso, é que seria necessário um protocolo de comunicação confiável, caso contrário o nó receptor dos pacotes poderia perder um dos mesmos e ficar muito tempo sem

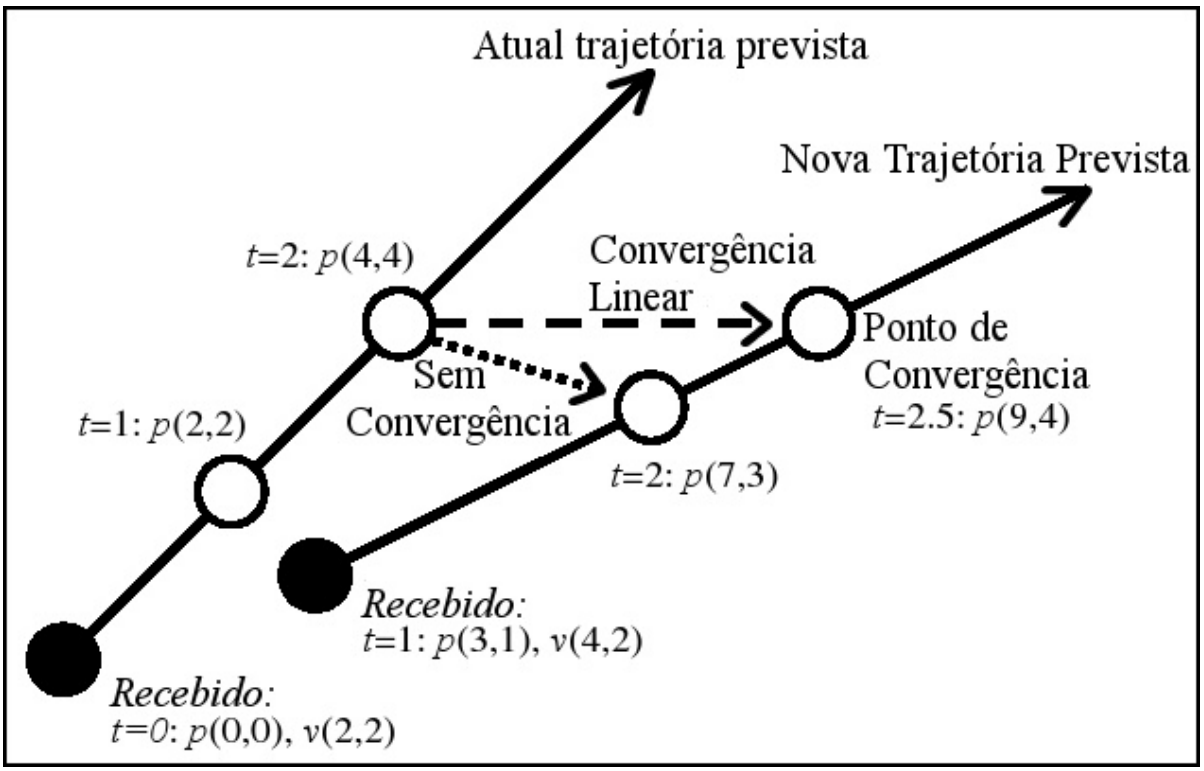

Figura 4.3: *Dead reckoning* com convergência linear e sem convergência (2).

atualização de determinada entidade, que possuiria um estado inválido, baseado numa predição possivelmente incorreta.

Um detalhe do *dead reckoning* é que, quando um nó da rede recebe um pacote de atualização de estado de entidade, a mesma pode ter uma posição calculada diferente da nova posição recebida. A forma mais simples de lidar com isso seria mudar o objeto de posição instantaneamente, sem nenhuma suavização. Outra forma é utilizando alguma técnica de convergência mais elaborada.

Uma boa técnica de convergência corrige tais erros de forma rápida e imperceptível (2). Um método de fazer isso seria usando convergência linear, fazendo o objeto mover-se da posição onde está atualmente (que foi calculada antes do recebimento do pacote de atualização) até uma posição X, sendo que X é uma futura posição do novo caminho previsto a partir do novo pacote de atualização de estado. O período no qual ocorre essa movimentação seria o *período de convergência*. Na figura 4.3, essa técnica é mais bem ilustrada. Nela, os círculos fechados indicam a posição quando foi enviado um pacote de atualização; os círculos abertos indicam posições calculadas com *dead reckoning*; a linha pontilhada indica uma mudança instantânea de posição; a linha tracejada indica o caminho percorrido durante o período de convergência, ao ser utilizada a convergência linear. Nesse exemplo, considera-se que o pacote enviado em

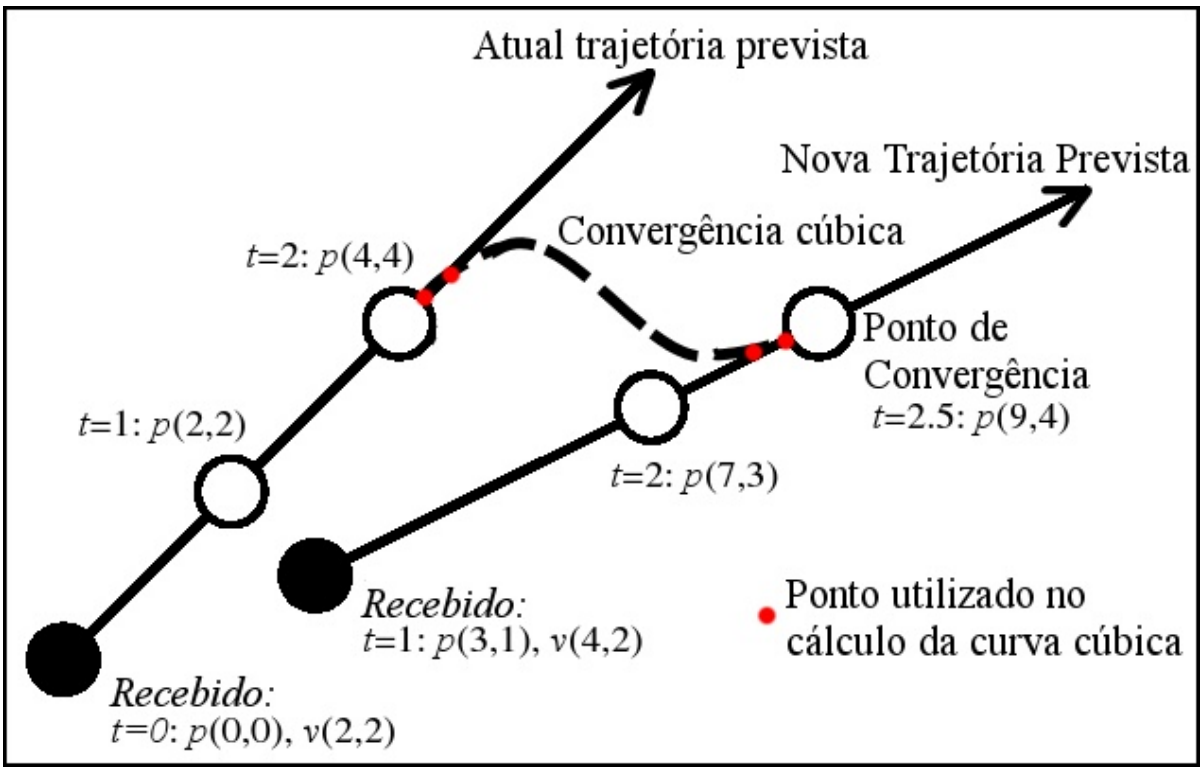

Figura 4.4: *Dead reckoning* com convergência cúbica (2).

$t = 1$ foi recebido apenas tem $t = 2$, devido a atraso. Por causa disso, a convergência foi feita a partir de $t = 2$.

Um problema da utilização de convergência linear é que os movimentos podem ser bruscos. Pode haver uma mudança de direção de forma não natural, tornando perceptível a utilização da técnica. Para resolver isso, podem-se utilizar outras curvas, como parábolas e curvas cúbicas. Neste caso, para se construir uma curva cúbica, tomam-se quatro pontos: o inicial, o final e mais dois intermediários, um pertencente ao antigo caminho previsto e o outro fazendo parte do novo caminho previsto. Com esses quatro pontos, constrói-se uma curva cúbica que será percorrida durante o período de convergência. Na figura 4.4, foi utilizada uma curva cúbica para descrever o caminho a ser percorrido durante o período de convergência. O resultado é um caminho mais natural, sem mudanças bruscas de direção.

Algo indesejável que pode ocorrer ao se utilizar o *dead reckoning* é o de um jogador interagir com um objeto cujo estado está sendo calculado, ou seja, cujo estado está sendo previsto e não é necessariamente o correto. Em um jogo *FPS*, por exemplo, um jogador J pode atirar em outro que está aparentemente em determinada posição, sendo que na verdade aquela posição foi calculada por *dead reckoning* e não condiz com a realidade. Seria necessário decidir se ele acertou ou não. Se for decidido que sim, o jogador J será beneficiado. Se não, J terá desperdiçado

munição. Em uma situação como essa, uma possibilidade seria a de delegar a decisão para o servidor (numa arquitetura cliente-servidor) ou para outro nó (numa arquitetura ponto a ponto). Pode-se também restituir a munição que foi gasta por aquele jogador J, no exemplo, e decidir que ele não acertou o outro jogador.

4.2 SOLUÇÕES EM BAIXO NÍVEL, APLICÁVEIS A REDES IEEE 802.11

Como foi apresentado anteriormente, prover *QoS* torna-se especialmente difícil em redes IEEE 802.11. Existem questões referentes a falta de limites determinísticos para atraso, flutuação de qualidade do canal, mobilidade dos nós da rede etc., o que traz uma série de complicações. O padrão IEEE 802.11 oferece mecanismos que foram feitos para prover *QoS*, que são o *PCF* (1) e, mais recentemente, a extensão IEEE 802.11e, que especifica um protocolo que visa prover *QoS*, o *EDCF* (*enhanced DCF*, *DCF* melhorado). Porém, verificou-se que esses protocolos não são ótimos. O *PCF* não demonstra ter o melhor resultado quando utilizado para manter qualidade de serviço (52) e a técnica de rajada de interferência (*blackburst*) demonstrou ser mais eficiente que o método utilizado no IEEE 802.11e (53, 54). Apesar destas referências serem de 2001 e 2003, não foram encontrados outros resultados de simulações que comparassem o *blackburst* com o *EDCF*. Sendo assim, daqui por diante serão levados em consideração os resultados encontrados nessas referências. Este trabalho busca listar soluções que melhorem o desempenho em relação àquelas definidas pelo padrão IEEE 802.11, incluindo o *EDCF*.

Existem adaptações sugeridas para o protocolo utilizado no *MAC* (camada de acesso ao meio, parte da camada de enlace de dados) do padrão IEEE 802.11. Esse protocolo foi descrito no capítulo 3 e é necessário seu completo entendimento para que se possam compreender as soluções que vêm a seguir. Algumas estendem esse protocolo, enquanto outras sugerem mudanças mais profundas na sua lógica. Foram pesquisadas:

- **rajada de interferência (*blackburst*)**, que consiste em nós que contêm dados de tempo-real a transmitir garantirem atraso máximo de envio através de competição entre eles baseada em: menor espaço entre os quadros (para obter maior prioridade sobre o canal) e preferência aos nós que estiverem esperando há mais tempo;
- **adaptação dinâmica do limite de reenvios**, de forma a balancear a perda de pacotes por sobrecarga de *buffers* - que ficam muito cheios quando há muitas retransmissões pendentes - com perda de pacotes por erros de transmissão - cuja forma de recuperação é justamente com reenvios;

- **adaptação dinâmica da taxa de transmissão**, que consiste em melhorar o uso do canal, mudando para uma taxa de transmissão mais baixa, porém mais confiável, quando necessário - taxas excessivamente altas implicariam em mais erros e seria contraproducente;

- **diferenciação de serviços, baseada em diferentes durações para o intervalos *DIFS***: dados com maior prioridade esperariam um *DIFS* mais curto.

É importante salientar que essas abordagens foram sugeridas visando melhorar o funcionamento de redes IEEE 802.11. Inclusive, suas simulações foram feitas nesse tipo de redes. No entanto, pode-se perceber que algumas delas podem ser aplicadas a redes genéricas, com excessão da diferenciação de serviços com diferentes *DIFS*, que é bastante específica, baseada no espaçamento entre quadros definido pelo padrão IEEE 802.11.

Também foram pesquisadas soluções para *MANETs*, porém esse tipo de rede sem fio abre uma vasta quantidade de problemas próprios que aumentariam excessivamente o escopo deste trabalho. *MANETs* têm problemas adicionais, geralmente relacionados a roteamento, e algumas das abordagens citadas anteriormente simplesmente não se aplicam. Por exemplo, o protocolo por rajada de interferência (descrito em 4.2.1) funciona quando se assume a não existência de nós escondidos, não havendo garantias se for de outra forma. Em *MANETs*, assume-se que a topologia da rede é arbitrária, aleatória e dinâmica, de forma a ser pouco adequada uma abordagem que necessita de uma garantia desse tipo.

Assim sendo, deve-se buscar soluções que levem em consideração as peculiaridades desse tipo de rede sem fio. Devido à sua natureza dinâmica, essas redes exigem protocolos especiais de roteamento. Um deles, o *AODV* (*ad hoc on demand distance vector*, vetor de distância sob demanda), cuja descrição detalhada pode ser encontrada em (55), aparenta ter o melhor desempenho, no geral, se comparado a outros protocolos de roteamento, pelo menos para o cenário de jogos multijogador. Em (10) foram apresentados resultados de simulações que confirmam essa assertiva. Uma adaptação desse protocolo de roteamento, chamada de *AODV-UU* (da Universidade de Uppsala), pode ser encontrada em (56) e demonstrou (10) ter melhor desempenho do que a especificação original do *AODV*. Em (10) também podem ser encontradas melhorias possíveis sobre esse protocolo, de forma a cumprir os requisitos temporais dos jogos multijogador.

A seguir, as abordagens pesquisadas serão descritas, com exceção daquelas voltadas para *MANETs*.

4.2.1 RAJADA DE INTERFERÊNCIA (*BLACKBURST*)

É desejável que um nó da rede - que tenha dados com requisitos temporais a transmitir - ganhe acesso indisputado ao canal de rádio compartilhado em intervalos regulares de tempo. Utilizando a técnica de *rajada de interferência* (3, 52, 57), é possível garantir que, uma vez que um nó transmite seu primeiro pacote com sucesso, as transmissões subseqüentes não colidirão com outros pacotes de tempo-real (3). Isso acontece porque essa técnica acaba dando acesso aos nós em *TDMA* (*time division multiple access*, acesso múltiplo por divisão do tempo), pois o intervalo de tempo, $Tsch$, entre os instantes de acesso dos nós é igual para todos. Se dois nós, n1 e n2, acessaram o meio nos instantes $T1$ e $T2$, respectivamente, o próximo acesso será agendado para $(T1 + Tsch)$ e $(T2 + Tsch)$, não havendo colisão entre essas transmissões de tempo-real. Esse esquema pode ser perturbado por transmissões de dados de baixa prioridade, que não utilizam rajadas de interferência, mas o protocolo foi definido de forma que é esperado que as transmissões de tempo-real se reposicionem no tempo e seu escalonamento ajuste-se automaticamente.

Considera-se uma rede sem fio constituída de nós de transmissão em tempo-real e nós de dados de menor prioridade. Assume-se também que todos os nós podem detectar as transmissões uns dos outros, sem a presença de nós escondidos (1). Os nós de dados de menor prioridade regulam seu acesso ao meio de acordo com o *DCF*, que foi brevemente descrito no capítulo 3. Os nós de tempo-real seguem outro esquema de acesso, que será descrito a seguir, e que lhes dá prioridade em relação aos nós de dados que não são de tempo-real.

Para explicar o funcionamento dessa técnica, é necessário fazer algumas definições:

- τ: é o maior atraso de propagação entre pares de nós. Inclui o atraso de sensoriamento, assim como o tempo gasto pelas interfaces de rede sem fio para executar as operações requisitadas;

- $Tshort$, que é o menor intervalo entre os quadros das transmissões. Pode ser comparado ao *SIFS* (*short interframe spacing*, espaçamento curto entre quadros), do padrão IEEE 802.11. O *SIFS* é utilizado para transmissões rápidas, que são parte de transmissões maiores (1). Exemplo disso seriam os quadros de confirmação positiva, ou *ACK*;

- $Tmed$, que é o intervalo utilizado pelas transmissões de tempo-real. Pode ser comparado com o *PIFS* (*point coordination function interframe spacing*, espaçamento entre quadros da coordenação por um ponto) do padrão IEEE 802.11 (1). Da mesma forma que o *PIFS* deste padrão, o $Tmed$ é utilizado para que transmissões de tempo-real tenham maior

prioridade sobre outros tipos de transmissão. $Tmed > Tshort + 2\tau$;

- $Tlong$, que é o intervalo utilizado pelas transmissões de dados que não são de tempo-real. É comparável ao *DIFS*, que já foi descrito. $Tlong > Tmed + 2\tau$;
- $Tsch$: tempo, constante e igual para todos os nós, entre a última tentativa bem sucedida de acesso ao meio e a próximo instante de acesso agendado;
- $Tpkt$: tempo mínimo exigido para a transmissão de um pacote de tempo-real; se o pacote for pequeno, a transmissão pode ser complementada com informação inútil para que seja utilizado o tempo mínimo;
- $Tbslot$: é a duração mínima de uma rajada de interferência. Rajadas de duração maior serão múltiplas de $Tbslot$ (cada rajada é constituído por um número inteiro de blocos de interferência (*black slots*) (3)). $Tbslot > 2\tau$;
- $Tobs$: é um tempo de observação durante o qual o meio é verificado para garantir que se pode enviar o pacote de tempo-real, após uma rajada de interferência ou após a verificação de que o canal permaneceu livre por pelo menos $Tmed$ segundos. $Tobs > 2\tau$;
- $Tunit$: é um parâmetro do sistema para calcular a duração de uma rajada em função do tempo durante o qual um nó esperou para transmitir.

É interessante observar que podem ser utilizados os intervalos de tempo *DIFS*, *PIFS* e *SIFS* como $Tlong$, $Tmed$ e $Tshort$, respectivamente. Isso acontece porque as características acima são satisfeitas por esses intervalos definidos pelo padrão IEEE 802.11. De fato, tem-se que (3, 52):

1. $DIFS > PIFS + 2\tau$;
2. $PIFS > SIFS + 2\tau$.

Dessa forma, ao implementar o método de rajada de interferência sobre o *MAC* do padrão IEEE 802.11, tem-se de antemão os intervalos necessários já definidos. Assim sendo, a técnica pode ser implementada como uma extensão desse padrão.

A operação do protocolo com utilização de rajadas de interferência exige dos nós da rede que farão transmissão de dados em tempo-real que:

1. tentem acessar o meio com intervalos iguais e constantes, $Tsch$, contados a partir da última tentativa com sucesso;

2. sejam capazes de causar interferência (rajadas) no canal de rádio, a fim de o tomarem para si.

O *instante de acesso* de um nó de tempo-real é definido como o instante no tempo em que aquele nó adquire acesso ao meio para transmitir um pacote. Quando um nó tem um instante de acesso, ele executa duas ações:

1. transmite um pacote, que deve demorar pelo menos $Tpkt$ segundos;
2. agenda o próximo instante de acesso para ocorrer $Tsch$ segundos depois.

Seja um nó de tempo-real que tem um instante de acesso agendado para ocorrer no instante t. Se foi percebido que o canal permaneceu livre no intervalo $(t - Tmed, t]$ e continua assim nos $Tobs$ segundos subseqüentes, o nó ganha seu instante de acesso imediatamente após a passagem desses intervalos de tempo.

Caso contrário, ou seja, se houve alguma perturbação ou transmissão de dados no intervalo $(t - Tmed, t + Tobs)$, o nó aguarda que o canal fique livre e que esse tempo livre dure $Tmed$ segundos. Ao se passarem esses $Tmed$ segundos com o canal desocupado, o nó entra em um *período de contenção por rajadas de interferência*. Nesse momento, ele perturba o canal com um número inteiro de blocos de rajada (blocos consecutivos, que são pulsos de energia com duração fixa e pré-determinada). Esse número é proporcional ao tempo durante o qual o nó esperou que o canal fosse liberado. Especificamente, se ele esperou por d segundos, transmite uma rajada de duração, em segundos, de $Tbslot \times \lceil \frac{d}{Tunit} \rceil$. A figura 4.5 ilustra a lógica utilizada pela técnica de rajada de interferência descrita. A primeira, segunda e terceira linhas representam, respectivamente, o que trafega no canal de rádio e o que é enviado pelos nós de tempo-real 1 e 2. Cada bloco em cinza representa uma rajada de interferência, cuja duração é múltipla de $Tbslot$.

Após transmitir sua rajada de interferência, o nó espera por $Tobs$ segundos para verificar se há alguma outra estação também transmitindo uma rajada, porém mais longa. Uma rajada mais longa significaria que a estação que a transmitiu esperou por um tempo maior, tendo maior prioridade. Se o canal permaneceu livre por $Tobs$ segundos imediatamente após a rajada, significa que o nó que a transmitiu ganhou acesso ao meio e, então, ele transmite seu pacote. Caso contrário, ele espera por um novo período livre de $Tmed$ segundos e recomeça o algoritmo, enviando nova rajada de interferência, desta vez maior por ter esperado mais tempo.

O intervalo de observação, $Tobs$, tem que ser obrigatoriamente menor do que $Tmed$, pois

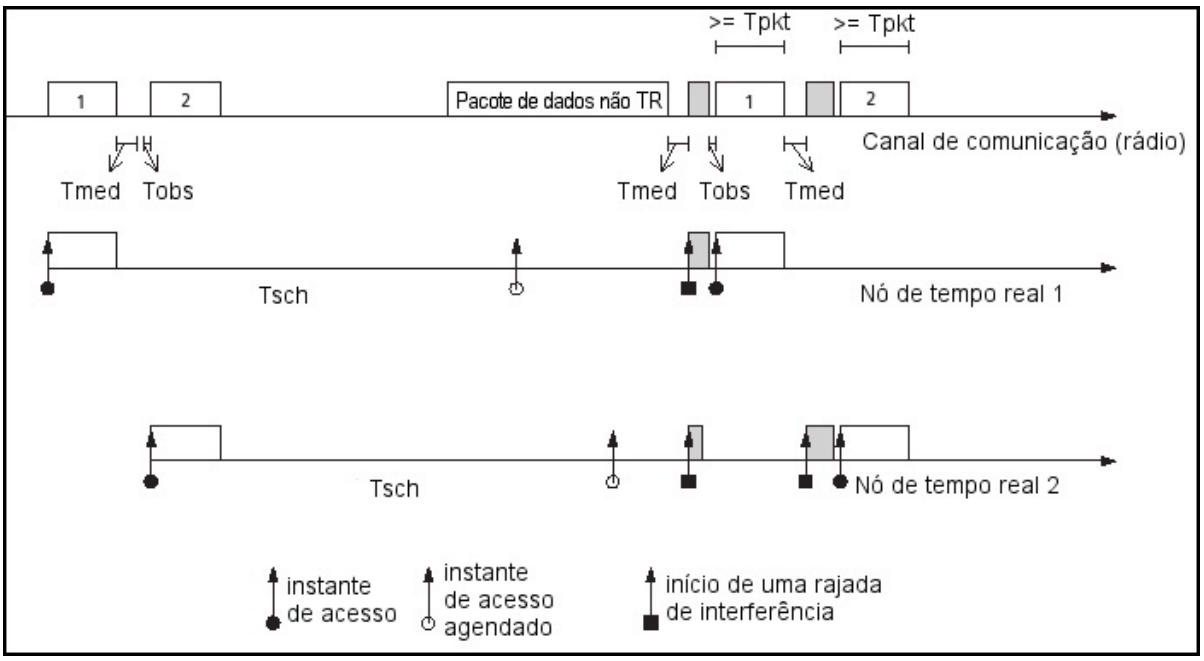

Figura 4.5: Exemplo de funcionamento da técnica de rajada de interferência (3).

pode haver outros nós de tempo-real aguardando novo intervalo $Tmed$ para transmitirem novas rajadas. Em suma, tem-se que $2\tau < Tobs < Tmed$.

O valor $Tunit$ é utilizado para calcular, em função do tempo esperado para transmitir, quantos blocos de rajada devem ser enviados para perturbar o canal. Para tornar mais claro, seja uma situação em que haja dois nós, N1 e N2, que já enviaram seus primeiros pacotes e estão sincronizados, sem colisão entre suas transmissões. Sabe-se, pelo que foi explicado, que o tempo necessário para transmitir um pacote de tempo-real é de, no mínimo, $(Tpkt + Tmed + Tobs)$ segundos. Assim sendo, essa será a diferença mínima de tempo entre os instantes de acesso agendados por N1 e N2. Seja, então, uma transmissão de dados de baixa prioridade que impeça que N1 e N2 enviem seus pacotes nos tempos esperados. Ocorrendo isso, ambos seguirão o algoritmo de rajada de interferência descrito, aguardando que o canal seja liberado e permaneça livre por $Tmed$ segundos. Como há essa diferença nos instantes de acessos agendados de N1 e N2, e os intervalos de agendamento são iguais a $Tsch$ para todos os nós, os tempos aguardados por N1 e N2 também diferirão de $(Tpkt + Tmed + Tobs)$ segundos, no mínimo. Daí, é fácil concluir que haverá apenas um vencedor, sempre, para o período de contenção por rajada de interferência (os tempos de rajada sempre diferirão), se o valor utilizado para $Tunit$ for menor ou igual à soma $(Tpkt + Ttmed + Tobs)$ na fórmula do tempo de rajada, $Tbslot \times \lceil \frac{d}{Tunit} \rceil$, onde d é o tempo durante o qual o nó que envia a rajada esperou que o canal fosse liberado.

Esse protocolo tem duas características importantes. Uma delas é que o nó que esperou

por mais tempo para acessar o canal vence o próximo período de contenção por rajada de interferência. Isso efetivamente garante que nós de tempo-real acessam o meio para transmitir seus pacotes em *round-robin*, todos tendo oportunidade de enviar seus dados. A outra característica é que os nós de tempo-real têm prioridade sobre os nós de dados que não são de tempo-real. Isso acontece porque o canal nunca ficará livre por um *DIFS* ($Tlong$) até que todos os nós de tempo-real que estejam competindo com rajadas de interferência tenham ganhado acesso.

É demonstrado em (3) que uma vez que o comprimento máximo dos pacotes de dados é conhecido, podem-se determinar condições para que haja um limite determinístico para o atraso de acesso ao meio destes nós. Seus mecanismos de contenção livres de colisão garantem isso. Essas condições se baseiam em certos parâmetros, tais como a duração das perturbações causadas por tráfego de dados que não sejam de tempo-real (cujo acesso ao meio foi obtido de acordo com o *DCF*, do padrão IEEE 802.11), os valores de $Tbslot$, $Tunit$, $Tsch$ etc. Essa demonstração é relativamente longa e seus detalhes fogem do escopo deste trabalho. Porém, tendo estabelecidas as condições necessárias para que o atraso seja limitado a um certo valor, baseado nisso cada novo pedido de conexão pode ser aceito ou recusado, de forma a continuar havendo essa garantia.

Verificou-se ainda que, quando há muito tráfego de dados de tempo-real na rede, a utilização da técnica de rajada de interferência prejudica bastante as transmissões de baixa prioridade, causando inanição (*starvation*) nas mesmas (53), para as quais não são dadas muitas oportunidades de acesso ao meio. No entanto, segundo simulações executadas e cujos resultados foram exibidos em (53), o protocolo utilizado pela extensão IEEE 802.11e, o *EDCF*, tem desempenho pior. Ele causa inanição no tráfego de baixa prioridade da mesma forma, além de ser menos eficiente que a técnica de rajada de interferência, pois esta diminui consideravelmente o número de colisões. A vantagem do IEEE 802.11e é que ele não faz as restrições exigidas para o uso de rajada de interferência. Apesar disso, vale ressaltar que o *EDCF* é bastante superior ao *PCF*, segundo os resultados encontrados nessas mesmas simulações (53), no cenário de aplicações multimídia.

4.2.2 ADAPTAÇÃO DINÂMICA DO LIMITE DE REENVIOS DE PACOTES

Pacotes podem se perder em redes IEEE 802.11, devido tanto a sobrecarga de *buffers*, quanto a um número de erros de envio acima do limite de reenvios. Em jogos costuma ser recomendado e utilizado *UDP* (49), um protocolo sem qualquer mecanismo embutido de controle de fluxo, o que irá fazer com que a aplicação despeje tantos pacotes quantos quiser no

buffer da rede, independentemente do seu estado atual de ocupação. Quando o *buffer* estiver cheio, novos pacotes que cheguem serão simplesmente descartados, constituindo sobrecarga. Quando o canal sofre uma alta taxa de erros de transmissão ou colisões, os pacotes exigem um número maior de reenvios, em média, para que sejam transmitidos corretamente ao receptor. Cada pacote a ser reenviado volta a ser enfileirado no *buffer* de envio, aumentando ainda mais a probabilidade de sobrecarga.

Este problema pode ser modelado utilizando um sistema de filas, com um modelo de tráfego estocástico, baseado nas *cadeias de Markov* (4). Porém, o detalhamento dessa análise foge do escopo deste trabalho e somente seus resultados serão utilizados aqui. Seja Lr o limite de reenvios para cada pacote. Sejam $pS(Lr)$ e $pD(Lr)$, respectivamente, as médias de perda de pacotes por sobrecarga e por descarte de pacotes após várias tentativas mal sucedidas de envio, ambas em função do limite de reenvios Lr. Foi evidenciado em (4), através de cálculos e confirmado em diversas simulações, que, dada uma média de ocorrência de erros de transmissão pE, $pS(Lr)$ cresce monotonicamente em função de Lr, ao mesmo tempo em que $pD(Lr)$ diminui. Assim sendo, pode ser calculado um valor de Lr ótimo, tal que a soma $pT(Lr) = pS(Lr) + pD(Lr)$ seja mínima, onde $pT(Lr)$ é a perda total média de pacotes.

Interessantemente, esse valor ótimo encontra-se justamente na intersecção entre as duas funções $pS(Lr)$ e $pD(Lr)$. Lr assume valores discretos, logo pode não haver um valor para o qual $pS(Lr)$ e $pD(Lr)$ sejam exatamente iguais. Porém, pode-se fazer um arredondamento e encontrar o valor ótimo de Lr da seguinte forma:

$$Lr = \min_{x} |pS(x) - pD(x)|$$

Visualizando a figura 4.6, esse comportamento pode ser entendido melhor. Fica bastante claro que há um valor ótimo para Lr, com uma diminuição significativa no total de pacotes perdidos. Quando o limite de reenvios é alto, mais pacotes são descartados, pois os *buffers* estão geralmente sobrecarregados. Por outro lado, se o limite for muito baixo, pacotes serão descartados precipitadamente, quando apenas mais um ou dois reenvios seriam suficientes para que a transmissão ocorresse com sucesso. Esse gráfico é resultado de simulações, cujos detalhes podem ser encontrados em (4), feitas com o simulador ns-2 (58). Foi simulada uma rede sem fio IEEE 802.11 *ad hoc* com dois nós, um transmissor e um receptor. As características da transmissão eram: taxa de bits constante (*CBR*, *constant bit rate*), a 3,52 Mbps, pacotes de 1000 bytes, *buffers* nas interfaces de rede com capacidade para 50 pacotes e canal de comunicação suportando até 11 Mbps, com probabilidade de erro $pE = 0,4$.

Levando esses resultados em consideração, pode-se resumir da seguinte forma: o valor

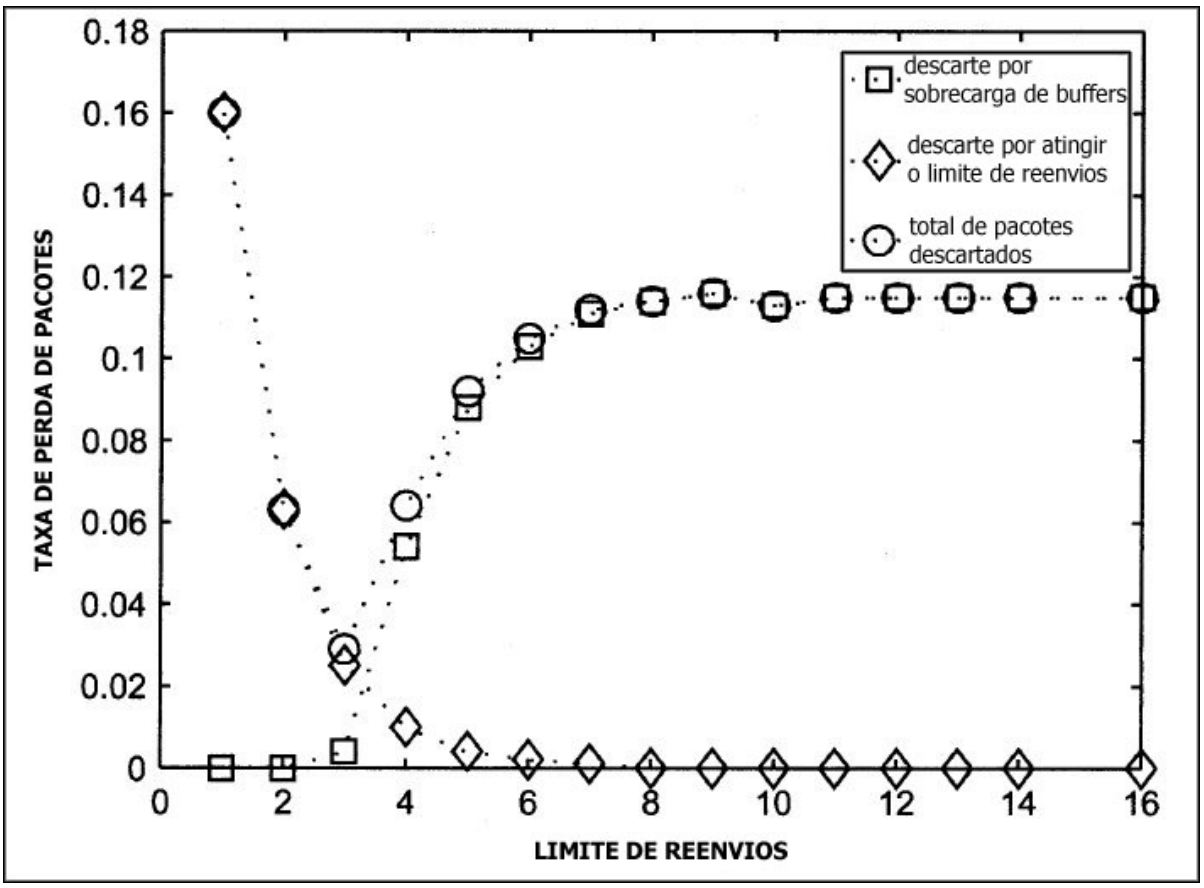

Figura 4.6: Perda de pacotes em função do limite de reenvios (4).

ótimo para o limite de reenvios sempre será aquele para o qual há um equilíbrio entre a perda de pacotes por sobrecarga do *buffer* e a perda por descarte após várias tentativas de envio mal sucedidas.

Percebeu-se, também, que a utilização desse valor ótimo diminui drasticamente a perda total média de pacotes nas transmissões. Em algumas simulações, diminui-se em cinco vezes a média de perda de pacotes (4). Essa perda média também se mostrou bastante sensível, com grande variação, a pequenos desvios do valor ótimo encontrado.

Além da média de ocorrência de erros de transmissão, o valor de *Lr* se mostrou dependente também da taxa de transmissão utilizada, em bps. Quanto maior era taxa de transmissão simulada, muito maior era a perda de pacotes média. Porém, essa variação da perda média foi bem menor ao ser utilizado o *Lr* ótimo, o que implica que, quanto maior o tráfego, maior o ganho desse algoritmo adaptativo. Porém, independentemente disso, o valor ótimo de *Lr* continua sendo aquele que torna $pS(Lr)$ e $pD(Lr)$ o mais próximos possível.

No entanto, é necessário definir o algoritmo que ajusta esse valor dinamicamente, durante o funcionamento da rede, de acordo com as mudanças percebidas no canal e na transmissão.

Uma forma bastante simples é a seguinte:

1. Monitore o estado da rede e calcule as médias de *pS* e *pD* dos últimos instantes;
2. Se $pD > pS$, eleve o valor de *Lr*;
3. Se $pD < pS$, reduza o valor de *Lr*;
4. Volte para 1.

Porém, após diversas simulações, foi encontrado um algoritmo mais refinado que obtém melhores resultados (4):

Listing 4.1: Algoritmo de adaptação de limite de reenvios *RTRA*

```
repita {
    monitore a rede e obtenha pS e pD;

    //se a soma de pS e pD for pequena (<pL1),
    //como no caso do canal estar em boas condições,
    //diminua Lr, até um valor mínimo R pré-definido.

    se (pS + pD < pL1) {
        se (Lr > R) {
            reduza Lr;
        }
    }

    //se a diferença entre pS e pD for pequena (<pL2),
    //não faça nada, pois as duas taxas de perda já estão
    //razoavelmente balanceadas

    se (|pS - pD| < pL2) {
        continue;
    }

    //e, por fim, a adaptação normal

    se (pS < pD) {
```

```
        incremente Lr em 1;
    }
    senão {
        decremente Lr de 1;

        //se pS >> pD, como no caso em que
        //o tráfego aumenta muito repentinamente,
        //é necessário reduzir rapidamente Lr,
        //para aliviar o congestionamento

        se (pS > q*pD) {
            decremente novamente Lr de 1;
        }
    }
}
```

Foram encontrados nas simulações valores para $pL1$, R, $pL2$ e q de 0,0001, 4, 0,01 e 10, respectivamente, tendo bom desempenho. Este algoritmo, batizado de *RTRA* (*real-time retry limit adaptation*, adaptação em tempo-real do limite de reenvios) tem resultados muito importantes. Um deles é a diminuição bastante significativa de perda de pacotes. A outra é a conseqüente elevação da eficiência de uso da largura de banda, deixando de desperdiçar o canal com reenvio de pacotes que talvez já sejam obsoletos, devido à demora para realizar novas tentativas.

4.2.3 ADAPTAÇÃO DINÂMICA DA TAXA DE TRANSMISSÃO

As redes IEEE 802.11 dão suporte a diferentes taxas de transmissão, permitindo que os nós selecionem aquela que for mais adequada naquele momento, levando em consideração os requisitos de qualidade de serviço e as condições do canal de rádio, de forma a melhorar o desempenho do sistema como um todo.

Se, por exemplo, um nó quer se comunicar com outro através de uma distância relativamente grande, então o *SNR* (*signal to noise ratio*, taxa de sinal em relação à interferência) resultante será muito baixo no receptor, ou seja, o sinal chegará de maneira fraca e propícia a erros, se uma alta taxa de transmissão for empregada. Em tais situações, uma taxa de transmissão mais confiável, mesmo que mais baixa, é necessária para que possa haver melhor comunicação. No entanto, se a qualidade do canal é suficientemente boa, então é desejável que se transmita

com taxas mais elevadas. Isso é essencial para dar suporte a serviços, tais como jogos multijogador, que podem exigir alta velocidade de comunicação, assim como para maximizar a taxa de transferência efetiva do sistema. Deve-se, portanto, utilizar a maior taxa de tranmissão possível, porém levando em consideração as condições do canal.

Há uma técnica com esse fim utilizada em dispositivos para redes sem fio *WaveLAN II*, da *Lucent* (59). Os autores descrevem um método automático para mudar entre duas taxas de transmissão, sendo a taxa mais elevada a padrão. O dispositivo muda automaticamente para a taxa menor depois de dois erros de transmissão consecutivos e volta para a taxa menor depois de dez transmissões com sucesso ou depois de um intervalo. A partir desse método, houve uma adaptação e alguns melhoramentos, resultando em um método que foi descrito em (5), compatível com a atual especificação da camada de acesso ao meio do padrão IEEE 802.11. Com ele, é possível detectar se a qualidade do canal está melhorando ou piorando e, baseado nessa informação, mudar para uma taxa mais alta ou mais baixa, respectivamente.

O padrão IEEE 802.11 não especifica como deve ser feita a mudança de taxa de transmissão em caso de camadas físicas capazes de múltiplas taxas. São especificadas apenas quais taxas podem ser utilizadas para enviar os quadros da camada de acesso ao meio. Além disso, não há qualquer protocolo para informar ao transmissor a respeito da qualidade atual do canal de comunicação, ou da taxa recomendada de transmissão. Assim sendo, o algoritmo sugerido em (5) de adaptação dinâmica da taxa de transmissão decide qual taxa utilizar baseado em informações disponíveis localmente, de maneira a obter o melhor desempenho possível para o sistema sem ser necessário que o receptor dê informações a respeito da qualidade do canal.

Como foi descrito, o padrão IEEE 802.11 especifica que, quando o receptor recebe um pacote e verifica que ele chegou sem erros, devolve uma confirmação ao transmissor, com um quadro *ACK*. A idéia do algoritmo proposto em (5) é que, se o transmissor não receber o *ACK* para um determinado quadro enviado para um determinado receptor, a qualidade do canal de comunicação para aquele receptor está ruim e, portanto, deve ser utilizada uma taxa de transmissão mais baixa em futuros envios para ele. Por outro lado, se o transmissor teve sucesso (i.e., recebeu confirmação) ao enviar múltiplos quadros a determinado receptor, assume-se que a qualidade do canal de comunicação está boa e, então, uma taxa mais elevada de transmissão deve ser utilizada nas próximas vezes.

Uma limitação deste algoritmo é que é desconsiderada a possibilidade de haver nós escondidos. Porém, como já foi visto, isso pode ser ignorado quando se utiliza a técnica *RTS*/*CTS* ou quando a rede opera em modo de infra-estrutura com um ponto de acesso intermediando a comunicação.

A implementação é feita com a utilização de dois contadores para cada endereço *MAC* de destino, um para transmissões bem sucedidas e outro para transmissões que falharam. Se um quadro é transmitido com sucesso, o primeiro contador é incrementado em uma unidade e o segundo é zerado; analogamente, se uma transmissão falha, o contador de falhas é incrementado em um e o de sucessos é zerado. Se o contador de falhas atinge certo valor limite F, então a taxa de transmissão para aquele endereço *MAC* de destino é diminuída e os contadores zerados. Similarmente, se o contador de sucessos atinge um determinado valor S, a taxa de transmissão para aquele destino é aumentada e os contadores são zerados.

É fácil perceber que os valores F e S influenciam na eficiência desse algoritmo. Se a qualidade do canal está variando muito, é desejável que a taxa de transmissão se adapte rapidamente como reação a essa variação. Porém, se o canal muda devagar de estado, essa adaptação deve ser lenta, para que não haja mudanças precipitadas na taxa de transmissão, o que seria propício a erros. Para o primeiro caso, devem-se utilizar valores baixos para F e S. No segundo, valores mais altos, para que seja necessário mais tempo para que uma mudança ocorra.

Foram feitas simulações com diferentes valores para essas variáveis e em (5) seus resultados são mostrados. Percebeu-se, no entanto, que o valor ideal para o limite de falhas consecutivas F é 1 (um), ou seja, o transmissor deve mudar para uma taxa de transmissão mais baixa depois de uma falha, independentemente de quão rápido a qualidade do canal esteja variando. A eficiência disso se explica pelo fato de que, mesmo quando o canal varia devagar, uma taxa mais baixa de transmissão implica em uma maior chance de sucesso, especialmente quando o canal realmente se deteriorou (5). Na prática, a conseqüência disso é que, sempre que uma transmissão falhar, a nova tentativa será feita com uma taxa mais baixa - se a retransmissão for imediatamente após a primeira transmissão -, o que, de fato, é o mais aconselhável. Com essa taxa baixa é provável que haja várias transmissões com sucesso, o que, naturalmente, fará com que o algoritmo de adaptação a eleve novamente, se a qualidade do canal assim permitir.

No entanto, o valor ideal para o limite de sucessos consecutivos S antes de aumentar a taxa de transmissão verificou-se bastante sensível à velocidade com que a qualidade do canal aumentava ou diminuía. O seu valor ideal era diferente para diferentes variações do canal. Nas simulações apresentadas em (5), utilizou-se o parâmetro de *dispersão Doppler* (5, 60, 61) como forma de determinar essa velocidade da variação do canal. Se essa dispersão é alta, o canal oscila muito sua qualidade, caso contrário, varia mais vagarosamente.

A figura 4.7 mostra os resultados encontrados nessas simulações. Percebe-se que, quando o valor de S é fixo, o desempenho é inferior a quando se utiliza o algoritmo com o valor de S sendo variável. São mostrados gráficos com S fixo com valor de 3, com valor de 10 e com

valor adaptativo (S variando de acordo com um algoritmo a ser descrito a seguir) e com taxa de transmissão baseada nos pacotes recebidos com sucesso (linha pontilhada). A simulação foi feita com um simulador construído em $C++$ que modela a camada de enlace de dados do padrão IEEE 802.11 e a camada física definida na extensão IEEE 802.11a. Foi simulado apenas o *DCF*, já que o *PCF* geralmente não é implementado em dispositivos comerciais. Os nós são considerados estacionários, embora a qualidade do canal varie com o tempo para que seja testada a técnica proposta. Informações mais detalhadas a respeito dos parâmetros da simulação podem ser encontradas em (5).

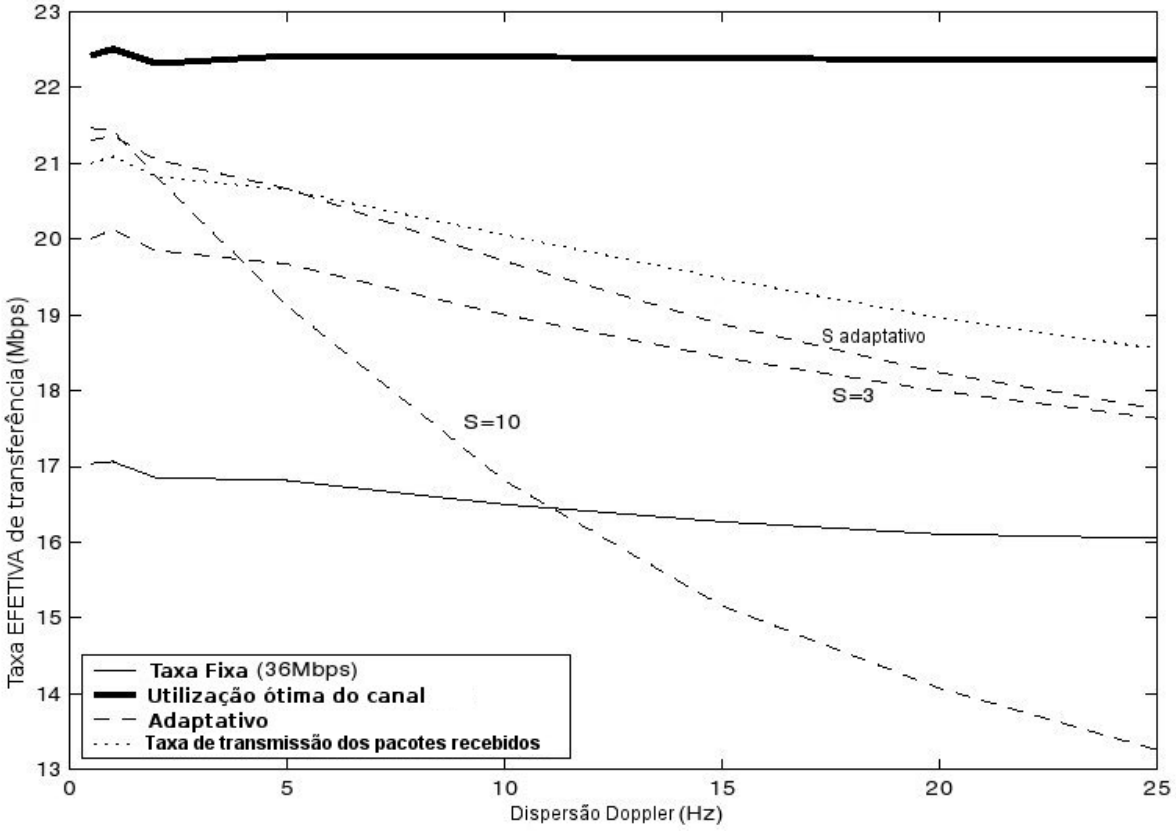

Figura 4.7: Taxa de transferência efetiva em função da dispersão Doppler (5).

Com base no que foi visto, é necessário que o algoritmo modifique o valor de S dinamicamente. Uma forma de fazer isso seria utilizando a camada física para medir a dispersão Doppler. Mas isto seria complexo e incompatível com as camadas físicas do padrão IEEE 802.11. Há uma possível solução, mais simples, utilizando uma pequena máquina de estados finita.

Como já foi dito, rápidas variações do canal exigem baixos valores de S e variações mais lentas exigem valores altos para S. Logo, são pré-definidos dois valores para S: $S1$ e $S2$, um baixo e um alto, respectivamente. A idéia é de que, após S transmissões bem sucedidas, muda-se para uma taxa de transmissão maior, porém a máquina entra no estado "dispersão?" e espera

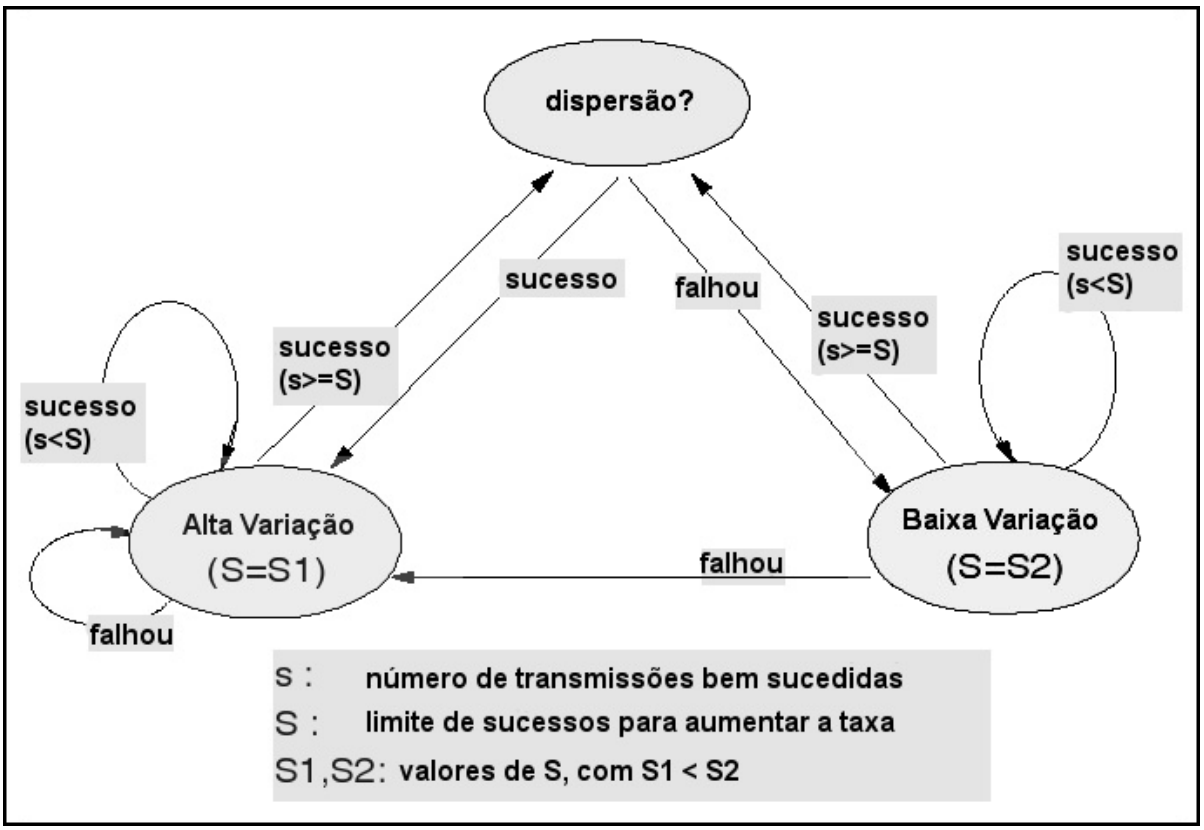

Figura 4.8: Máquina de estados construída para modificar a taxa de transmissão (5).

pelo resultado da próxima transmissão, cuja taxa será mais alta do que a última. Se for bem sucedida, significa que o canal melhorou rapidamente e, conseqüentemente, o valor utilizado será $S1$. A máquina, então, vai para o estado "alta variação", alterando o valor de S para $S1$. Por outro lado, se essa transmissão falhar, assume-se que a variação do canal é baixa, se é que houve variação, e a decisão de aumentar a taxa de transmissão foi precipitada. Com isso, a máquina vai para o estado "baixa variação" e o valor de S passa a ser $S2$, mais alto.

Na figura 4.7, a curva "S adaptativo" mostra o desempenho do algoritmo de adaptação, avaliado em simulações, utilizando a máquina de estados descrita, sendo $S1 = 3$ e $S2 = 10$. O diagrama de estados dessa máquina pode ser visto na figura 4.8.

Uma última evolução do algoritmo para determinar a taxa de transmissão ideal é levando em conta a taxa com que o outro nó lhe envia dados. Sejam dois nós, X e Y. Se X está utilizando uma taxa de transmissão tX, mas recebe um quadro enviado a uma taxa tY, tal que $tY > tX$, então X ajusta sua taxa para tY também. A linha pontilhada no gráfico da figura 4.7 mostra os resultados obtidos em simulações, utilizando esse melhoramento. Percebe-se que isto é mais eficaz quando a variação da qualidade do canal é mais rápida.

4.2.4 DIFERENCIAÇÃO DE SERVIÇOS BASEADO EM DIFERENTES *DIFS*

Como foi visto no capítulo 3, a respeito do funcionamento do protocolo de acesso ao meio do padrão IEEE 802.11, quadros de confirmação positiva *ACK*, assim como quadros *CTS*, conseguem maior prioridade para serem transmitidos, simplesmente pelo fato de esperarem um intervalo de tempo menor. Um possível mecanismo de diferenciação de serviços (i.e., atribuir diferentes prioridades a diferentes dados a trafegarem na rede) seria utilizando o mesmo princípio básico (6).

Quadros de dados, assim como quadros *RTS* - que marcam o início de transmissão de dados, quando é utilizada a técnica de *RTS*/*CTS* - podem ter seus tempos de espera modificados, de acordo com a prioridade dos dados a transmitir. Nesta abordagem (6), a cada nível de prioridade corresponde um *DIFS* diferente. Um nó cujo dado a transmitir tenha prioridade j esperará $DIFS[j]$ antes de enviar aquele pacote, sendo que $DIFS[j+1] < DIFS[j]$. Para evitar colisões de quadros de mesma prioridade, o mecanismo de espera aleatória é mantido e, para garantir que dados de maior prioridade sejam enviados primeiro, é feita a seguinte restrição: a janela de contenção deve ter duração máxima de $DIFS[j-1] - DIFS[j]$, após a passagem de $DIFS[j]$. Dessa forma, nenhum dado de prioridade $j-1$ poderá iniciar sua transmissão enquanto houver quadros de prioridade j ou maior a serem transmitidos.

Tráfego de baixa prioridade terá que esperar sempre que houver quadros de maior prioridade enfileirados, sendo suscetíveis a sofrer inanição (*starvation*). Uma maneira de resolver isso é relaxando a regra anterior e permitir que a janela de contenção aumente, podendo ser maior que $DIFS[j-1] - DIFS[j]$. Assim, seria possível, mesmo que menos provável, um nó com menor prioridade acessar o meio em detrimento de um nó com maior prioridade, evitando inanição de determinados fluxos de dados. Essa seria também uma maneira de fazer com que a prioridade de determinado pacote também fosse diminuindo a cada tentativa de envio mal sucedida, já que sua janela de contenção poderia ultrapassar aquela de outros pacotes. Essa técnica, apesar de depender de falha de envios, pode ser útil, se o requisito referente a atraso de envio for mais severo do que o de perda de pacotes.

Um exemplo ilustrativo é exibido na figura 4.9. O pacote P3, apesar de ter maior prioridade ($DIFS[3]$ é o mais curto de todos), não foi transmitido com sucesso por duas vezes consecutivas, aumentando sua janela de contenção e, conseqüentemente, sendo possível esperar mais que os pacotes P2 e P1. De certa forma, implicitamente a prioridade de P3 diminuiu. Outro detalhe importante é que P2 pode selecionar um *slot* no final de sua janela de contenção, esperando mais do que P1, que é menos prioritário. Essa possibilidade é menos provável de ocorrer,

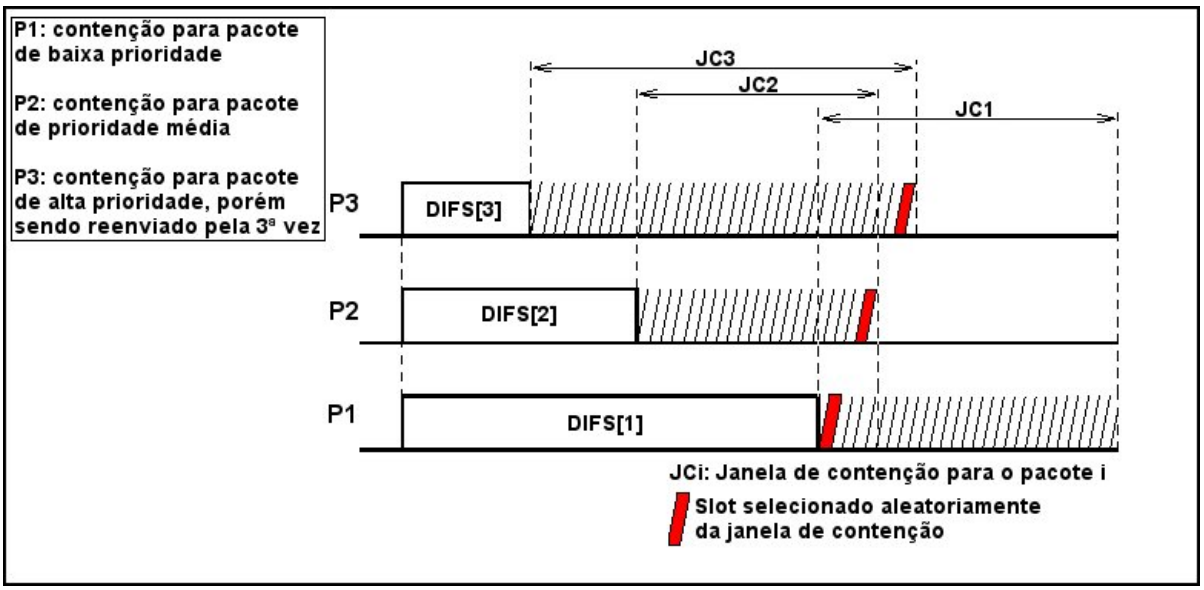

Figura 4.9: Diferenciação de serviços com diferentes *DIFS*.

pois seria necessário que, além de P2 escolher um *slot* no final de sua janela de contenção, P1 coincidentemente escolhesse um *slot* no início de sua janela. Essa possibilidade é ilustrada na figura, com os *slots* escolhidos por P1 e P2 destacados. Com tais ajustes no mecanismo de diferenciação de serviços aqui descrito, evita-se que determinados fluxos de menor prioridade nunca consigam acesso ao meio.

Foram realizadas simulações e seus resultados foram apresentados em (6). Algumas das conclusões foram:

- Esse mecanismo oferece um vasto conjunto de possíveis prioridades relativas, podendo haver prioridades iguais, quando os *DIFS* não mudam de um pacote para o outro, nem as janelas de contenção máximas. Esse conjunto é teoricamente infinito, já que os diferentes valores para os *DIFS* podem ser tão altos quanto se desejar - embora seja necessário manter num intervalo viável para aplicações práticas -, permitindo um grande número de combinações possíveis;

- A diferenciação de serviços com mudanças no *DIFS* não apresentou perda de eficiência, ou seja, a soma das velocidades de transmissão dos fluxos de diferentes prioridades é aproximadamente a mesma velocidade de transmissão conseguida quando não é utilizada a técnica aqui descrita.

O resultado dessas simulações pode ser visto no gráfico da figura 4.10. Cada nó da rede simulado pretende enviar um fluxo constante de pacotes *UDP*, a 231200 bytes/s, que está um

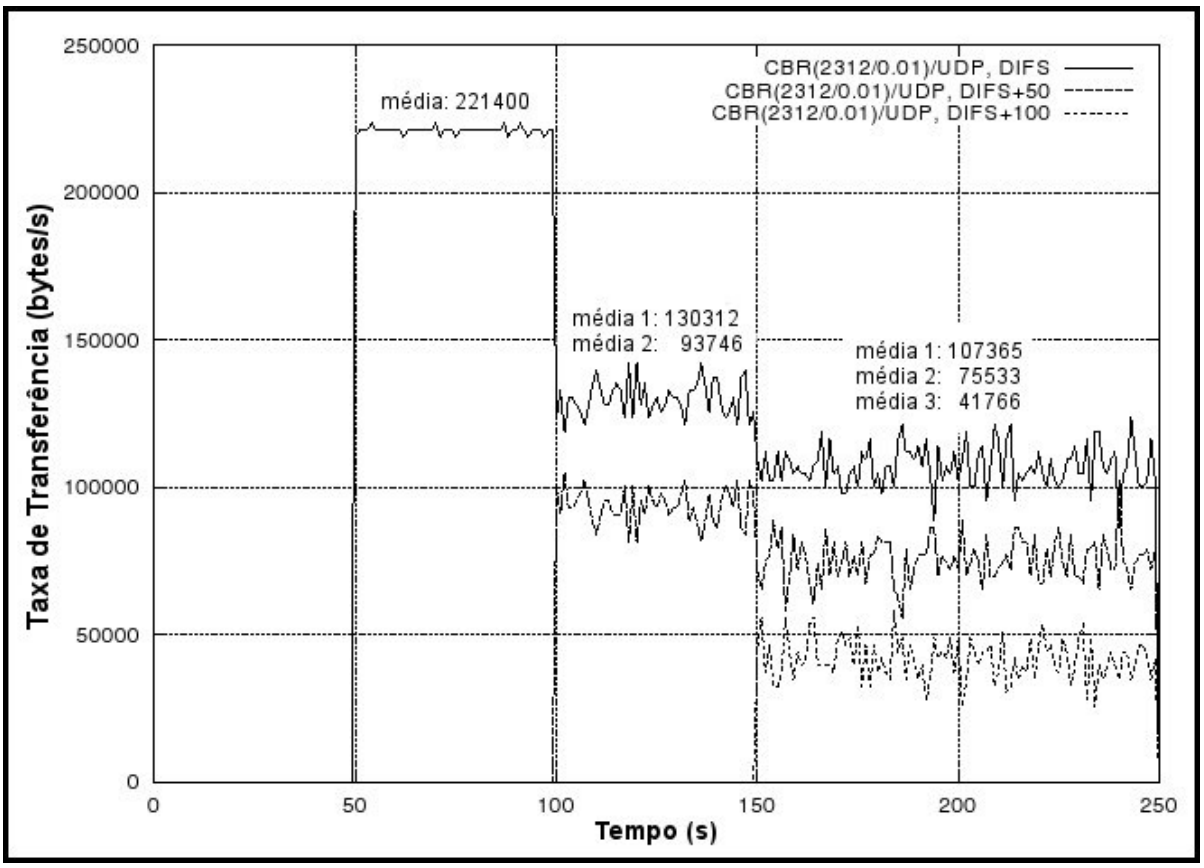

Figura 4.10: Diferentes prioridades com diferentes *DIFS* (6).

pouco além do que o canal foi feito para suportar, que é em torno de 220000 bytes/s. São exibidos os resultados de três situações. Na primeira, há um único nó presente, não precisando competir pelo meio. Ele atinge uma média total de 221400 bytes/s. Na segunda, há dois nós, com prioridades distintas, cujos *DIFS* diferem por 50 μs e, somando suas médias de taxa de transmissão, obtém-se 224058 bytes/s. Na terceira situação, há três nós presentes na rede - com *DIFS*[3] = 50 μs, *DIFS*[2] = 100 μs e *DIFS*[1] = 150 μs - com taxa de transmissão total de 224684 bytes/s. Pode-se perceber que a diferenciação de serviços com diferentes *DIFS* funciona bem, sem perda de eficiência, pois a taxa de transmissão total alcançada pelos três nós foi aproximadamente igual à capacidade total do canal. Além disso, a taxa de transmissão de cada fluxo se manteve relativamente estável, sem grandes quedas ou picos.

5 CONCLUSÃO

5.1 AVALIAÇÃO DAS ABORDAGENS APRESENTADAS

Foram apresentadas soluções que, apesar de não garantirem com absoluta certeza que esses requisitos temporais possivelmente presentes nos dados a transmitir sejam cumpridos, diminuem bastante determinados efeitos indesejados que possam ocorrer. Algumas técnicas buscam, no nível da aplicação, diminuir o consumo de largura de banda, de modo a diminuir indiretamente o atraso e a perda de pacotes - que pode acontecer se houver sobrecarga dos *buffers* das interfaces de rede, por exemplo.

As técnicas que se pode implementar na camada de aplicação basicamente buscam eficiência no uso da largura de banda disponível. Uma delas, a de compressão e agregação de pacotes, busca diminuir o tamanho dos mesmos, agregá-los ou, ainda, enviar o mínimo possível de informação, que seria apenas o que mudou no estado das entidades do jogo, após terem sido enviados pacotes com o estado completo. Dessa forma, diminui-se bastante o tráfego na rede e diminui-se, também, o atraso que poderia haver por causa de uma transmissão além da capacidade máxima do canal de comunicação por rádio.

Outras técnicas possíveis de se implementar nos jogos são as de *dead reckoning* e a de gerenciamento de interesse. A primeira busca diminuir o tráfego de pacotes de atualização de estado e deixar que o jogo extrapole os estados seguintes ao recebimento de um desses pacotes, enquanto espera-se por novas atualizações oriundas da rede. O gerenciamento de interesse tem por objetivo fazer com que haja comunicação apenas quando necessário, de forma que um determinado jogador não receba informações que lhe são desnecessárias, como aquelas referentes a outro jogador que esteja em uma posição do ambiente virtual totalmente diferente da sua.

No entanto, em redes IEEE 802.11, a aplicabilidade da técnica de gerenciamento de interesse é discutível, já que, mesmo que um determinado jogador X envie um pacote de atualização do seu estado apenas ao jogador Y, o meio estaria ocupado da mesma forma que se ele fizesse

um *broadcast* para todos os outros jogadores. Porém, essa técnica pode ser útil numa situação em que nem todos os jogadores estão presentes naquela rede local, e, portanto, não compartilham o mesmo canal de comunicação. Um exemplo disso seria o de três jogadores estarem numa mesma rede local sem fio, conectados via ponto de acesso à Internet, onde está ocorrendo o jogo com mais três outros jogadores. Seria interessante que os computadores que não fazem parte da rede local sem fio mandassem a ela apenas pacotes que interessassem aos jogadores desta rede, economizando a largura de banda disponível no canal de rádio dos últimos.

Como foi mencionado, há adaptações possíveis para o *DCF* do padrão IEEE 802.11, que obtêm resultados bastante significativos. Na técnica de rajada de interferência, por exemplo, consegue-se diminuir bastante o número de colisões entre os pacotes e, se houver apenas nós de tempo-real (i.e., que utilizam a técnica de rajada de interferência), até um determinado limite máximo de nós, garante-se que não haverá, em momento algum, colisão entre as transmissões oriundas de diferentes origens.

Outra grande otimização que se mostrou possível é o ajuste dinâmico do limite de reenvios de pacotes que não foram transmitidos com êxito na primeira tentativa. Com isso, consegue-se balancear a perda de pacotes e a sobrecarga dos *buffers* das interfaces de rede, alcançando um comportamento próximo do ideal no que se refere a essa perda. Alta perda de pacotes e sobrecarga dos *buffers* gerariam mais atraso nas transmissões, além da perda de atualizações de estado, tornando o funcionamento do jogo menos suave.

Também foi possível perceber que uma taxa de transmissão ótima não é necessariamente a taxa de transmissão mais elevada. Com uma taxa de transmissão mais baixa, consegue-se fazer que a comunicação ocorra de forma mais confiável, necessitando menos de reenvios. Taxas mais altas desperdiçariam muito tempo e largura de banda com reenvios em excesso, tornando-se contraproducente. Com um algoritmo simples e uma máquina de estados de lógica bastante fácil de compreender, conseguiu-se uma eficiência no uso do canal de rádio próxima do nível ótimo.

Apesar do *DCF* não definir um esquema de diferenciação de serviços, uma maneira simples de implementar tal mecanismo seria utilizando diferentes durações de *DIFS* para fluxos de diferentes prioridades. Foi observado, através dos resultados exibidos em (6), que tal mecanismo é eficiente, ou seja, não desperdiça largura de banda disponível: o somatório das taxas de transmissão dos diferentes fluxos de dados é aproximadamente igual à capacidade máxima de transmissão do canal. Além disso, verificou-se também que a taxa de transmissão de cada fluxo permanece relativamente estável, sem grandes picos ou quedas.

Em suma, mesmo que não seja possível garantir o cumprimento de certas exigências, é

bastante factível otimizar a comunicação em redes sem fio IEEE 802.11. Dessa forma, pode-se ter uma expectativa otimista no que se refere a manter a qualidade de serviço exigida mesmo pelos jogos com requisitos mais severos.

5.2 TRABALHOS FUTUROS

É possível, também, propor novas soluções baseadas nas que foram listadas aqui. Pode-se fazer um conjunto de soluções, que vão desde a camada de aplicação até a camada física da pilha de protocolos utilizados pelas redes IEEE 802.11. Outra possibilidade é a de implementar abordagens híbridas, buscando fundir o que há de melhor em algumas das soluções aqui descritas.

Por exemplo, pode-se buscar a implementação de um *framework* para jogos multijogador, baseado no uso em conjunto de várias das soluções descritas anteriormente, que teria um aspecto como o ilustrado na figura 5.1. Basicamente, esse *framework* teria todas as abordagens já pré-implementadas, divididas em camadas, desde a de aplicação até a física. Para alguém que estivesse desenvolvendo um jogo multijogador voltado para redes IEEE 802.11, bastaria criar a lógica do jogo e fazer uso das funcionalidades desse *framework*, poupando bastante trabalho e obtendo um desempenho excelente, supondo que as abordagens funcionem bem em conjunto.

Na sugestão da figura, ficou de fora o esquema de diferenciação de serviços por diferentes *DIFS*, descrito em 4.2.4, pois ele é incompatível com a técnica de rajada de interferência, que não utiliza espaçamento *DIFS* para separar seus quadros, mas sim um espaçamento menor, comparável ao *PIFS* (uma alternativa seria utilizar o esquema de diferenciação de serviços por diferentes *DIFS* ao invés da rajada de interferência, porém esta se mostrou mais eficiente quando comparada com vários outros esquemas de diferenciação de serviços (52, 53, 57)).

Seguindo essa sugestão, o jogo executaria *dead reckoning* para não necessitar de muitos pacotes de atualização de estado, já que os estados seguintes à última atualização poderiam ser extrapolados. Além disso, os pacotes seriam endereçados apenas àqueles destinos que tivessem interesse nos mesmos. Esses mesmos pacotes, antes de serem enviados, seriam comprimidos e/ou agregados. Ao chegar à camada de enlace de dados, cada pacote teria um limite máximo de reenvios, antes de ser descartado, caso houvesse erros de transmissão. Esse limite seria determinado pelo algoritmo descrito em 4.2.2. Para dar melhores chances do pacote não sofrer muito atraso para chegar ao destino, ele seria enviado seguindo o método da rajada de interferência, pois o mesmo escalona as transmissões implicitamente em *round-robin*. A taxa de transferência utilizada nessa transmissão teria sido previamente calculada de acordo com a técnica descrita

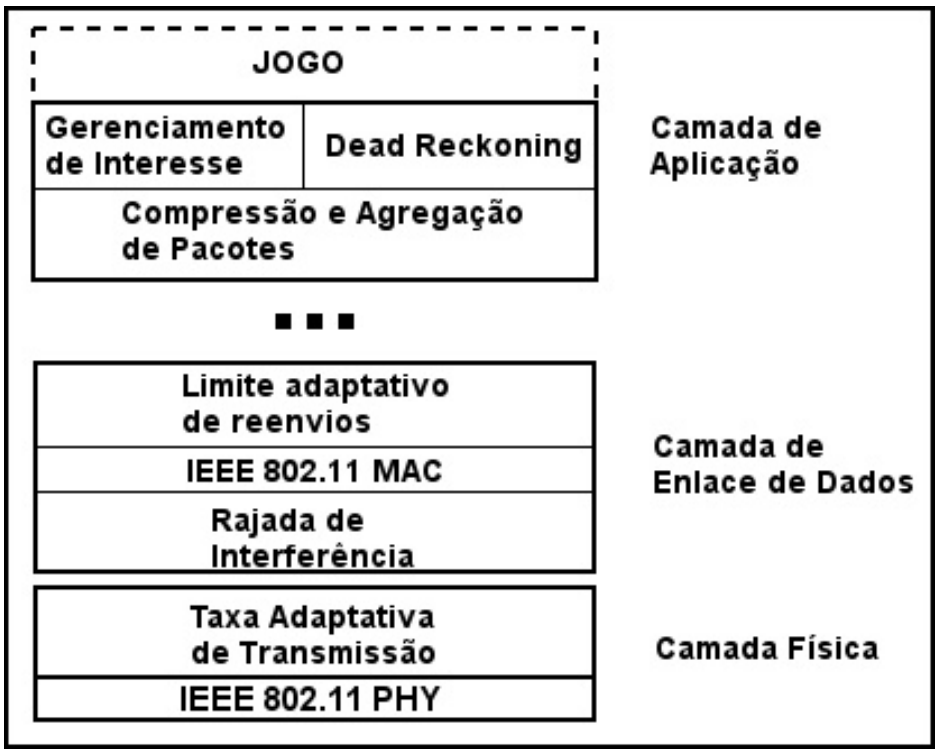

Figura 5.1: Possível *framework* para jogos multijogador sobre redes IEEE 802.11.

em 4.2.3, reduzindo bastante a probabilidade de haver erro de transmissão, porém sem reduzir desnecessariamente a velocidade de envio.

Além da implementação desse *framework* com as soluções funcionando em conjunto, da maneira descrita, seria recomendável fazer simulações para verificar a eficácia dessa união. Poderiam ser simulados não só o conjunto que foi descrito como também diferentes combinações de abordagens. Em função dos resultados encontrados, seria sugerida a combinação que se mostrasse mais adequada para o funcionamento de jogos multijogador sobre redes IEEE 802.11.

Por fim, outra sugestão de trabalho futuro seria a busca por uma adaptação na técnica de rajada de interferência, baseando-se no esquema de diferenciação de serviços com diferentes *DIFS*. Utilizando a rajada de interferência, como foi descrito em 4.2.1, pode haver períodos de competição entre os nós de tempo-real se alguma perturbação externa impedir que um ou mais desses nós envie seu pacote no instante agendado. Nesse caso, cada nó de tempo-real que teve seu escalonamento impedido de ocorrer, ao entrar no período de contenção, envia uma rajada de interferência de duração proporcional ao tempo que esperou até então.

No entanto, essa abordagem implica em prioridade igual para todos os fluxos de dados. Para o cálculo da duração da rajada de interferência, poderia ser levado em conta, além do tempo que o nó esperou, também a prioridade dos dados a serem enviados. Com dados mais

prioritários, a rajada seria mais longa, assim como, analogamente, dados com menor prioridade teriam uma rajada mais curta. Como foi descrito em 4.2.1, nós que enviaram rajadas mais longas obtêm acesso ao meio em detrimento de nós que enviaram rajadas mais curtas. Um problema dessa adaptação é que funcionaria apenas quando houvesse perturbações externas e conseqüentes períodos de contenção por rajada de interferência, onde as prioridades dos fluxos de dados interfeririam. Mesmo assim, seria uma possível melhoria da técnica, sendo razoável simular seu funcionamento.

5.3 CONSIDERAÇÕES FINAIS

É importante ressaltar que o meio através do qual os nós em uma rede sem fio se comunicam, particularmente redes IEEE 802.11, é, usualmente, o canal de rádio, que tem uma série de vulnerabilidades. Uma delas é a possibilidade de haver colisões entre transmissões de diferentes nós da rede. Outra vulnerabilidade é a interferência que pode estar havendo no meio, e a qualidade da comunicação, que depende muito de fatores geográficos, distância etc. Em particular, o mecanismo de recuperação de erros de transmissão definido para o padrão IEEE 802.11 pode gerar um novo problema, que é a sobrecarga dos *buffers* das interfaces de rede, como foi demonstrado através de simulações cujos resultados foram encontrados em artigos. Assim sendo, é de se esperar o fato de que nenhuma solução encontrada garante com absoluta certeza o cumprimento de requisitos temporais para a transmissão dos dados.

O *DCF* não define diferenciação de serviços ou qualquer outro mecanismo de prover *QoS* ou de garantir o cumprimento de requisitos temporais. Na verdade, não há sequer garantias de que determinado pacote será transmitido com sucesso, quando houver sobrecarga no tráfego da rede sem fio, pois o *DCF* especifica a utilização de espera aleatória (*backoff*) para decidir qual pacote será enviado primeiro, no caso de vários nós competirem pelo acesso.

No entanto, tendo em vista todas as soluções aqui apresentadas, pode-se tirar diversas conclusões. Uma delas é que é possível fazer várias melhorias no protocolo *DCF* do padrão IEEE 802.11, atingindo resultados bastante significativos e animadores, levando em conta todos os desafios encontrados para efetuar transmissões através de ondas de rádio.

Em resumo, este trabalho descreveu algumas das principais técnicas usadas no suporte a jogos multijogador em redes sem fio, caracterizando-as com foco nos aspectos temporais, tão importantes para esta classe de jogo. Além disso, foi possível observar problemas que podem ainda ser explorados em pesquisas futuras, contribuindo assim para direcionar possíveis inovações nesta área.

REFERÊNCIAS BIBLIOGRÁFICAS

1 GAST, M. *802.11 Wireless Networks: The Definitive Guide*. [S.l.]: O'Reilly, 2002. ISBN 0-596-00183-5.

2 SMED, J.; KAUKORANTA, T.; HAKONEN, H. *A Review on Networking and Multiplayer Computer Games*. [S.l.], April 2002. Disponível em: <http://citeseer.ist.psu.edu/article/smed02review.html>.

3 SOBRINHO, A. S. K. João L. Real-time traffic over the ieee 802.11 medium access control layer. *Bell Labs Technical Journal*, p. 172–187, 1996.

4 LI, Q.; SCHAAR, M. van der. Providing adaptive qos to layered video over wireless local area networks through real-time retry limit adaptation. *IEEE Transactions on Multimedia*, v. 6, p. 278–290, April 2004.

5 CHEVILLAT, P. et al. A dynamic link adaptation algorithm for ieee 802.11a wireless lans. In: *Proc. IEEE Int'l Conf. Comm. (ICC'03)*. [s.n.], 2003. v. 2, p. 1141–1145. Disponível em: <http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=1204543>.

6 AAD, I.; CASTELLUCCIA, C. Differentiation mechanisms for IEEE 802.11. In: *INFOCOM*. [s.n.], 2001. p. 209–218. Disponível em: <http://citeseer.ist.psu.edu/aad01differentiation.html>.

7 MIRKOVIC, J. et al. Measuring impact of dos attacks. In: *Proceedings of the DETER Community Workshop on Cyber Security Experimentation*. [s.n.], 2006. Disponível em: <http://www.cs.purdue.edu/homes/fahmy/emist/abstracts/metric.pdf>.

8 SMED, J.; KAUKORANTA, T.; HAKONEN, H. Aspects of networking in multiplayer computer games. In: SING, L. W.; MAN, W. H.; WAI, W. (Ed.). *Proceedings of International Conference on Application and Development of Computer Games in the 21st Century*. [s.n.], 2001. p. 74–81. Disponível em: <http://citeseer.ist.psu.edu/smed01aspects.html>.

9 FARKAS, K. et al. Dominating set based support for distributed services in mobile ad hoc networks. In: *Network Operations and Management Symposium, 2006. NOMS 2006. 10th IEEE/IFIP*. [S.l.: s.n.], 2006. p. 327–338.

10 BUDKE, D. et al. Real-time multiplayer game support using qos mechanisms in mobile ad hoc networks. In: *Proceedings of Third Annual Conference on Wireless On demand Network Systems and Services (WONS 2006)*. [s.n.], 2006. Disponível em: <http://www.ibr.cs.tu-bs.de/users/wellnitz/papers/wons2006/wons2006.pdf>.

11 MULTIPLAYER game. Último acesso em 3 de dezembro de 2006. Nota: Multiplayer game é um artigo da enciclopédia digital online wikipedia (http://wikipedia.org). Disponível em: <http://en.wikipedia.org/wiki/Multiplayer>.

12 T-CARRIER. Último acesso em 3 de dezembro de 2006. Nota: T-carrier é um artigo da enciclopédia digital online wikipedia (http://wikipedia.org). Disponível em: <http://en.wikipedia.org/wiki/T-carrier>.

13 10 gigabit Ethernet. Último acesso em 3 de dezembro de 2006. Nota: 10 gigabit Ethernet é um artigo da enciclopédia digital online wikipedia (http://wikipedia.org). Disponível em: <http://en.wikipedia.org/wiki/10_gigabit_Ethernet>.

14 WORLD of Warcraft. Último acesso em 3 de dezembro de 2006. Nota: World of Warcraft é um exemplo de jogo de RPG online massivamente multijogador (MMORPG). Disponível em: <http://www.worldofwarcraft.com>.

15 RAGNAROK. Último acesso em 3 de dezembro de 2006. Nota: Ragnarok é um exemplo de jogo de RPG online massivamente multijogador (MMORPG). Disponível em: <http://games.levelupgames.com.br/ragnarok/>.

16 PRISTON Tale. Último acesso em 3 de dezembro de 2006. Nota: Priston Tale é um exemplo de jogo de RPG online massivamente multijogador (MMORPG). Disponível em: <http://www.priston.com.br/>.

17 SILK Road Online. Último acesso em 3 de dezembro de 2006. Nota: Silk Road Online é um exemplo de jogo de RPG online massivamente multijogador (MMORPG). Disponível em: <http://www.silkroadonline.net>.

18 COUNTER-STRIKE. Último acesso em 3 de dezembro de 2006. Nota: Counter-Strike é um exemplo de jogo FPS multijogador. Disponível em: <http://www.counter-strike.net>.

19 BATTLEFIELD: 1942. Último acesso em 3 de dezembro de 2006. Nota: Battlefield: 1942 é um exemplo de jogo FPS multijogador. Disponível em: <http://www.counter-strike.net>.

20 COMPUTER and video game genres. Último acesso em 3 de dezembro de 2006. Nota: Computer and video game genres é um artigo da enciclopédia digital online wikipedia (http://wikipedia.org). Disponível em: <http://en.wikipedia.org/wiki/Computer_and_video_game_genres>.

21 UNREAL Tournament. Último acesso em 3 de dezembro de 2006. Nota: Unreal Tournament é um exemplo de jogo FPS multijogador. Disponível em: <http://www.unrealtournament.com/>.

22 GRAND Theft Auto. Último acesso em 3 de dezembro de 2006. Nota: Grand Theft Auto é um exemplo de jogo de ação-aventura. Disponível em: <http://www.rockstargames.com/grandtheftauto/>.

23 SOUL Calibur. Último acesso em 3 de dezembro de 2006. Nota: Soul Calibur é um exemplo de jogo de ação de luta. Disponível em: <http://www.soulcalibur.com/>.

24 SUPER Mario Bros. Último acesso em 3 de dezembro de 2006. Nota: Super Mario Bros. é um exemplo de jogo de plataforma. Disponível em: <http://www.nintendo.com/gamemini?gameid=9b8f1efe-3ac4-4d81-9a17-e7458e1891a7>.

25 DIABLO II. Último acesso em 3 de dezembro de 2006. Nota: Diablo II é um exemplo de jogo de RPG. Disponível em: <http://www.blizzard.com/diablo2/>.

26 DESCENT. Último acesso em 3 de dezembro de 2006. Nota: Descent é um exemplo de jogo de simulação de combate espacial. Disponível em: <http://www.descent2.com/>.

27 NEED for Speed: Most Wanted. Último acesso em 3 de dezembro de 2006. Nota: Need for Speed: Most Wanted é um exemplo de jogo de esporte, de corrida. Disponível em: <http://www.ea.com/nfs/mostwanted/us/>.

28 HEROES of Might and Magic. Último acesso em 3 de dezembro de 2006. Nota: Heroes of Might and Magic é um exemplo de jogo de estratégia baseado em turno. Disponível em: <http://www.heroesofmightandmagic.com/>.

29 AGE of Empires. Último acesso em 3 de dezembro de 2006. Nota: Age of Empires é um exemplo de jogo de estratégia em tempo-real. Disponível em: <http://www.microsoft.com/games/empires/>.

30 LIU, Y. E. et al. Capability of ieee 802.11g networks in supporting multi-player online games. In: *Proceedings of the 2nd IEEE International Workshop on Networking Issues in Multimedia Entertainment (NIME 2006)*. [s.n.], 2006. v. 2, p. 1193–1198. Disponível em: <http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=1593227>.

31 CRONIN, E.; FILSTRUP, B.; JAMIN, S. Cheat-proofing dead reckoned multiplayer games (extended abstract). In: *Proceedings of Application and Development of Computer Games 2003*. [s.n.], 2003. Disponível em: <http://warriors.eecs.umich.edu/games/papers/adcog03-cheat.pdf>.

32 BOND, M. *Boom! Headshot! (Building Neo-Tactics on Network-Level Anomalies in Online Tactical First-Person Shooters)*. Disponível em: <http://www.cl.cam.ac.uk/ mkb23/research/Boom-Headshot.pdf>.

33 BUDKE, D. *Quality of Service for Multiplayer Game Provisioning in Mobile Ad Hoc Networks*. Dissertação (Mestrado) — Swiss Federal Institute of Technology Zurich, September 2005. Disponível em: <http://www.tik.ee.ethz.ch/ farkas/publications/QoS-master_thesis.pdf>.

34 DICK, M.; WELLNITZ, O.; WOLF, L. Analysis of factors affecting players' performance and perception in multiplayer games. In: *Proceedings of 4th ACM SIGCOMM workshop on Network and system support for games*. [S.l.: s.n.], 2005. p. 1–7.

35 BERNIER, Y. W. Latency compensating methods in client/server in-game protocol design and optimization. In: *Proceedings of the Game Developers Conf.* [s.n.], 2001. Disponível em: <http://www.resourcecode.de/stuff/clientsideprediction.pdf>.

36 INFORMATION technology- Telecommunications and information exchange between systems- Local and metropolitan area networks- Specific requirements- Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications. 2003. ANSI/IEEE Std 802.11, 1999 Edition (R2003).

37 GRIFFITH, E. *Lufthansa and Cisco Put Wi-Fi in the Plane*. January 2003. Último acesso em 3 de dezembro de 2006. Nota: Lufthansa and Cisco Put Wi-Fi in the Plane é um artigo escrito para o sítio wi-fiplanet.com. Disponível em: <http://www.wi-fiplanet.com/news/article.php/1570531>.

38 WI-FI. Último acesso em 3 de dezembro de 2006. Nota: Wi-Fi é um artigo da enciclopédia digital online wikipedia (http://wikipedia.org). Disponível em: <http://en.wikipedia.org/wiki/Wi-Fi>.

39 WI-FI Alliance. Último acesso em 3 de dezembro de 2006. Disponível em: <http://www.wi-fi.org>.

40 LAN party. Último acesso em 3 de dezembro de 2006. Nota: Lan party é um artigo da enciclopédia digital online wikipedia (http://wikipedia.org). Disponível em: <http://en.wikipedia.org/wiki/Lan_party>.

41 NINTENDO DS. Último acesso em 3 de dezembro de 2006. Nota: Nintendo DS é um artigo da enciclopédia digital online wikipedia (http://wikipedia.org). Disponível em: <http://en.wikipedia.org/wiki/Nintendo_DS>.

42 SONY PSP. Último acesso em 3 de dezembro de 2006. Nota: Sony PSP é um artigo da enciclopédia digital online wikipedia (http://wikipedia.org). Disponível em: <http://en.wikipedia.org/wiki/Sony_PSP>.

43 XBOX 360. Último acesso em 3 de dezembro de 2006. Nota: Xbox 360 é um artigo da enciclopédia digital online wikipedia (http://wikipedia.org). Disponível em: <http://en.wikipedia.org/wiki/Xbox_360>.

44 PLAYSTATION 3. Último acesso em 3 de dezembro de 2006. Nota: Playstation 3 é um artigo da enciclopédia digital online wikipedia (http://wikipedia.org). Disponível em: <http://en.wikipedia.org/wiki/Playstation 3>.

45 WII. Último acesso em 3 de dezembro de 2006. Nota: Wii é um artigo da enciclopédia digital online wikipedia (http://wikipedia.org). Disponível em: <http://en.wikipedia.org/wiki/Wii>.

46 MEYER, C. et al. *Multiplayer Gaming in Mobile Ad Hoc Networks*. March 2006. Disponível em: <http://www.tik.ee.ethz.ch/ farkas/publications/Clowns-semester_thesis.pdf>.

47 MOBILE ad-hoc network. Último acesso em 3 de dezembro de 2006. Nota: Mobile ad-hoc network é um artigo da enciclopédia digital online wikipedia (http://wikipedia.org). Disponível em: <http://en.wikipedia.org/wiki/Mobile_ad-hoc_network>.

48 LUNDGREN, H.; NORDSTROM, E.; TSCHUDIN, C. The gray zone problem in ieee 802.11b based ad hoc networks. In: *ACM SIGMOBILE Mobile Computing and Communications Review*. [S.l.: s.n.], 2002. v. 6, p. 104–105.

49 JAMIN, S. *Networking Multiplayer Games*. 2005. Disponível em: <http://ai.eecs.umich.edu/soar/Classes/494/talks/lecture-15.pdf>.

50 GAUTIER, L.; DIOT, C.; KUROSE, J. End-to-end transmission control mechanisms for multiparty interactive applications on the internet. In: *IEEE INFOCOM'99*. [S.l.: s.n.], 1999. v. 3, p. 1470–1497.

51 DEAD reckoning. Último acesso em 3 de dezembro de 2006. Nota: Dead reckoning é um artigo da enciclopédia digital online wikipedia (http://wikipedia.org). Disponível em: <http://en.wikipedia.org/wiki/Dead_reckoning>.

52 LINDGREN, A.; ALMQUIST, A.; SCHELÉN, O. Quality of Service schemes for IEEE 802.11 - a simulation study. In: *Proceedings of the Ninth International Workshop on Quality of Service (IWQoS 2001)*. [s.n.], 2001. Disponível em: <http://citeseer.ist.psu.edu/lindgren01quality.html>.

53 LINDGREN, A.; ALMQUIST, A.; SCHELÉN, O. Evaluation of Quality of Service schemes for IEEE 802.11 wireless LANs. In: *Proceedings of the 26th Annual IEEE Conference on Local Computer Networks (LCN 2001)*. [s.n.], 2001. Disponível em: <http://citeseer.ist.psu.edu/lindgren01evaluation.html>.

54 LINDGREN, A.; ALMQUIST, A.; SCHELéN, O. Quality of Service schemes for IEEE 802.11 wireless LANs - an evaluation. *In Special Issue of the Journal on Special Topics in Mobile Networking and Applications (MONET) on Performance Evaluation of Qos Architectures in Mobile Networks*, v. 8, n. 3, p. 223–235, June 2003. Disponível em: <citeseer.ist.psu.edu/lindgren03quality.html>.

55 PERKINS, C. *Ad-hoc on-demand distance vector routing*. 1997. Disponível em: <http://citeseer.ist.psu.edu/article/perkins97adhoc.html>.

56 AODV-UU. Último acesso em 3 de dezembro de 2006. Nota: AODV-UU é uma implementação do protocolo de roteamento AODV, feita pela Universidade de Uppsala, Suécia. Disponível em: <http://core.it.uu.se/core/index.php/AODV-UU>.

57 PRASAD, A. R. Performance comparison of voice over ieee 802.11 schemes. In: *Proc. of the 50th IEEE Vehicular Technology Conference*. [S.l.: s.n.], 1999. v. 5, p. 2636–2640.

58 THE network simulator - ns-2. Último acesso em 3 de dezembro de 2006. Disponível em: <http://www.isi.edu/nsnam/ns/>.

59 KAMERMAN, A.; MONTEBAN, L. Wavelan-ii: A high-performance wireless lan for the unlicensed band. *Bell Labs Technical Journal*, p. 118–133, 1997.

60 GUIMARÃES, D. *TP105 – Caracterização do Canal e Técnicas de Melhoria de Desempenho em Comunicações Móveis*. June 2004. Disponível em: <http://cict.inatel.br/nova2/docentes/dayan/TP105/Slides/TP105_Slides.pdf>.

61 DOPPLER Spread. Último acesso em 3 de dezembro de 2006. Disponível em: <http://www.see.ed.ac.uk/ dil/thesis_mosaic/subsubsection2_8_2_1_6.html>.