BlueNetS: Uma Ferramenta para Estudo do Desempenho do Bluetooth *

Werner Priess(1), José Ferreira de Rezende(2),
Luiz Fernando Rust da C. Carmo(1), Luci Pirmez(1)

(1) Núcleo de Computação Eletrônica (NCE)/UFRJ
(2) Grupo de Teleinformática e Automação (GTA)/COPPE/UFRJ

Abstract

Bluetooth radio technology allows eletronic devices to communicate in a spontaneous way, forming small ad hoc networks. Designed to be part of a variety of portable devices, it is a promising technology towards ubiquitous computing. This paper introduces an extension to ns-2, BlueNetS (Bluetooth Network Simulator), which is being developed to study Bluetooth performance when applied to various scenarios. More specifically, we intend to use this tool to study limitations of the technology in providing quality of service (QoS) to applications and propose mechanisms to overcome these limitations. Some results to validate the tool are presented, as well as a first proposal to provide QoS to delay-sensitive applications.

Keywords: Wireless communication, ad hoc networks, quality of service, Bluetooth, piconets.

Resumo

A tecnologia de rádio Bluetooth permite que dispositivos eletrônicos próximos se comuniquem de maneira espontânea, formando pequenas redes *ad hoc*. Projetada para integrar diversos equipamentos portáteis, a tecnologia está possibilitando um grande avanço em direção à computação ubíqua. Este artigo introduz uma extensão ao simulador ns-2, o BlueNetS (*Bluetooth Network Simulator*), que está sendo desenvolvida para avaliar o desempenho da tecnologia Bluetooth em diversos cenários. Mais especificamente, pretende-se usar esta ferramenta para detectar limitações da tecnologia no fornecimento de qualidade de serviço (QoS) às aplicações e propor mecanismos para supri-las. São apresentados alguns resultados de simulações de validação da ferramenta, bem como uma primeira proposta para provimento de QoS a aplicações sensíveis ao retardo.

Palavras-chave: Comunicação sem fio, redes *ad hoc*, qualidade de serviço, Bluetooth, *piconets*.

1 Introdução

O aperfeiçoamento de dispositivos móveis, tais como telefones celulares, PDAs (*personal digital assistants*) e *laptops*, e o desenvolvimento de novas tecnologias de rádio, como Bluetooth e IEEE 802.11, trouxeram à realidade as idéias de computação ubíqua, pervasiva e onipresente. Particularmente, Bluetooth emergiu como forma de permitir a comunicação entre dispositivos eletrônicos sem o uso de cabos de conexão e, posteriormente, passou a ser empregado, também, como ponto de acesso a serviços de dados e voz e na formação de redes *ad hoc* entre dispositivos pessoais. Estas últimas, também chamadas de PANs (*personal area networks*), devido a seu curto alcance, potencializam a existência da computação onipresente.

Há propostas para a utilização do Bluetooth nos mais variados cenários, dentre os quais se incluem as redes residenciais, a automação industrial e as redes de sensores. O Bluetooth viabiliza, também, a confecção de redes temporárias e de baixo custo, citando-se o exemplo da organização de feiras, exposições e conferências. Com a utilização de *laptops*, PDAs, impressoras, projetores e outros equipamentos com interface Bluetooth, prepara-se rapidamente o ambiente para o evento desejado. Qualquer usuário, chegando ao local do evento, pode conectar-se transparentemente à rede e trocar informações com outros participantes. Pode-se, ainda, instalar um ponto de acesso à Internet ou, caso isto não seja viável, os usuários podem usar telefones celulares para estabelecer a conexão externa. Neste cenário, várias pequenas redes, chamadas *piconets*, estariam ativas simultaneamente.

Desde o lançamento da primeira versão aberta da especificação Bluetooth, muitos estudos foram realizados avaliando a tecnologia [1], [2] e propondo a incorporação de novos mecanismos [3], [4], [5]. A maioria dos trabalhos se baseia em simuladores próprios, não disponíveis para uso público. Visando estudar o comportamento

*Este trabalho foi realizado com recursos do NCE, CNPq, CAPES, COFECUB e FAPERJ.

do Bluetooth em diversos cenários, este trabalho apresenta e valida uma ferramenta de simulação do Bluetooth, BlueNetS (*Bluetooth Network Simulator*), baseada no *Network Simulator* (ns-2) [6]. No estágio de desenvolvimento atual, BlueNetS incorpora as características do Bluetooth necessárias para o estudo de mecanismos de escalonamento e encaminhamento de pacotes na rede. Então, a partir deste simulador, são coletados alguns resultados relativos ao desempenho da tecnologia e são propostos alguns mecanismos para melhorá-lo. O trabalho mostra, ainda, o estágio de desenvolvimento da tecnologia, identificando algumas questões em aberto na especificação e avaliando algumas propostas para resolvê-las.

O artigo está estruturado da seguinte maneira. A próxima seção conceitua o Bluetooth e descreve, sucintamente, sua especificação. Aborda, também, as *scatternets* e sua principais características. A seção 3 apresenta um resumo da pesquisa realizada sobre a tecnologia. A seção seguinte mostra as principais características do simulador desenvolvido. A seção 5 apresenta os resultados obtidos através da simulação de alguns cenários. Finalmente, as conclusões e algumas considerações sobre o futuro deste trabalho fazem parte da seção 6.

2 Especificação Bluetooth

Bluetooth [7], [8] é um padrão de interface de rádio de curto alcance, criado para permitir a comunicação entre dispositivos eletrônicos sem a utilização de cabos. Idealizada para ser incorporada a pequenos equipamentos como *notebooks*, PDAs, fones de ouvido, telefones celulares e sensores, a tecnologia Bluetooth precisa satisfazer os seguintes requisitos: baixo custo, baixo consumo de energia e interface de tamanho reduzido. Em 1998, um grupo de empresas dos ramos de telecomunicações e informática criou o Bluetooth SIG (*Special Interest Group*) [7], com o objetivo de padronizar o desenvolvimento do Bluetooth. O SIG lançou uma especificação industrial aberta, atualmente na versão 1.1, contendo duas partes: o núcleo e os perfis. O núcleo define as características do rádio e a pilha de protocolos para a comunicação entre dois dispositivos. Os perfis definem cenários para as aplicações e especificam quais protocolos de nível mais alto da pilha de protocolos do Bluetooth devem ser implementados para determinada aplicação. A figura 1 apresenta as camadas da pilha de protocolos de maior relevância para o entendimento deste trabalho.

2.1 Rádio Bluetooth

O rádio Bluetooth opera na banda ISM (*Industrial, Scientific and Medical*) de 2.4 GHz, disponível mundialmente sem a necessidade de licença, e emprega a técnica de saltos de freqüência com espalhamento espectral (FHSS) na transmissão. O espectro de freqüências é dividido em 79 canais de rádio-freqüência (23 em alguns países), cada um com 1 MHz de largura de banda, e a freqüência de saltos é de 1600 por segundo.

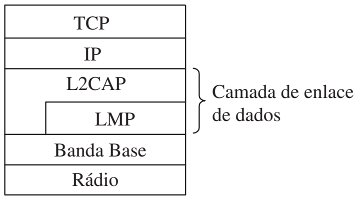

Figura 1: Pilha de protocolos do Bluetooth.

2.2 Banda Base

A banda base é a camada física do Bluetooth. Entre outras funções, destaca-se a criação das *piconets* e dos enlaces. A *piconet* é uma pequena rede, de até oito integrantes, formada pelos dispositivos que compartilham o mesmo canal, ou seja, a mesma seqüência de saltos de freqüência. Um dos dispositivos assume o papel de mestre e os outros se comportam como escravos. O mestre de uma *piconet* é quem dita a seqüência de saltos e a fase nesta seqüência, através do seu endereço único de 48 bits e do seu relógio, respectivamente. As informações de endereço e relógio são trocadas entre mestre e escravos através do procedimento de investigação (*inquiry*), sempre iniciado pelo mestre. Segue-se uma fase de estabelecimento de conexão (*paging*).

Há dois tipos de enlaces possíveis em uma *piconet*: síncrono, orientado a conexão (SCO - *Synchronous Connection Oriented*) e assíncrono, sem conexão (ACL - *Asynchronous Connectionless*). Ambos seguem o esquema de duplexação por divisão no tempo (TDD) na transmissão, onde cada enlace é dividido em intervalos de tempo de 625 microsegundos chamados *slots*. O SCO é um enlace ponto-a-ponto, simétrico, entre o mestre e um escravo, mantido pelo mestre em *slots* reservados a intervalos regulares. Sua aplicabilidade principal é o tráfego de voz. O ACL é um enlace ponto-a-multiponto, entre o mestre e todos os escravos da *piconet*. Quando usado juntamente com enlaces SCO, o ACL só pode ocupar os *slots* não reservados para o SCO. É usado, tipicamente, para transmitir dados no esquema de melhor esforço. Mestre e escravos transmitem alternadamente, com o mestre sempre iniciando suas transmissões em *slots* com numeração par e os escravos em *slots* com numeração ímpar. Um escravo somente poderá transmitir em um *slot* se tiver sido endereçado pelo mestre no slot anterior. Os pacotes podem ocupar 1, 3 ou 5 *slots*.

Além do esquema de saltos de freqüência, a banda base dispõe de técnicas de retransmissão de pacotes (ARQ - *Automatic Repeat Request*) e de correção de erros (FEC - *Forward Error Correction*), para fornecer robustez e confiabilidade ao Bluetooth. No caso do

ARQ, os pacotes de banda base são retransmitidos até que o transmissor receba uma confirmação do recebimento (ACK). O mecanismo de FEC é opcional, com taxa a 2/3, para proteger os dados e está sempre presente, com taxa a 1/3, para proteger o cabeçalho.

Como uma das preocupações do Bluetooth é a economia de energia, foram definidos quatro modos de operação para os escravos: ativo, *sniff*, *hold* e *park*. Os três últimos caracterizam modos de baixo consumo de energia. No modo *sniff*, o escravo dorme e acorda periodicamente para escutar transmissões do mestre. No modo *hold*, o escravo entra em baixo consumo por um intervalo de tempo fixo, após o qual volta à atividade normal. Não há uma periodicidade como no modo *sniff*. Finalmente no modo *park*, o escravo libera seu endereço, mas permanece sincronizado à piconet. Como uma piconet pode ter no máximo sete escravos ativos, o modo *park* permite que mais escravos continuem sintonizados à piconet, para uma possível reativação futura.

2.3 Protocolo de Gerência do Enlace (LMP)

O LMP encontra-se na camada de enlace de dados e é responsável pela configuração das propriedades do enlace. Por exemplo, os procedimentos de autenticação e criptografia e os modos de operação de baixo consumo de energia são definidos na banda base, mas são configurados e ativados através de transações LMP entre dois dispositivos.

2.4 Protocolo de Adaptação e Controle do Enlace Lógico (L2CAP)

O L2CAP é o protocolo da camada de enlace de dados que fornece serviços para as camadas superiores e esconde destas os detalhes de implementação das camadas inferiores. Cada fluxo L2CAP trafega em um canal lógico estabelecido entre as duas extremidades. Os canais podem ser sem conexão ou orientados à conexão. Pode haver vários canais lógicos entre duas estações, mas todos multiplexados pelo L2CAP em um mesmo enlace ACL. O L2CAP também realiza a segmentação e remontagem de pacotes (SAR), função necessária para permitir a utilização, por parte das camadas superiores, de pacotes maiores que o tamanho máximo aceito pela banda base. Finalmente, o processo de estabelecimento de uma conexão L2CAP permite a troca de informações relativas à QoS entre duas unidades Bluetooth. O L2CAP é responsável pelo monitoramento dos canais lógicos existentes, visando manter a QoS.

2.5 *Scatternets*

Várias *piconets* podem coexistir em uma mesma área de cobertura, pois seus padrões de saltos de freqüência são mutuamente ortogonais. Entretanto, pode haver a interconexão entre *piconets*, criando uma topologia conhecida por *scatternet*. A *scatternet* se forma quando ao menos um nó participa de duas ou mais *piconets*. Esta participação se baseia em uma divisão no tempo, ou seja, em qualquer instante o nó pode estar ativo em somente uma *piconet*. O nó ponte (*bridge* ou *interpiconet*) pode trabalhar com aplicações independentes em cada *piconet* ou funcionar realmente como uma "ponte", encaminhando pacotes de uma *piconet* para outra. O primeiro caso pode ser exemplificado com a estação ponte transmitindo áudio e vídeo para estações em redes distintas. O segundo pode ser caracterizado por um um cenário onde PDAs estejam conectados a um *notebook* e este, a um telefone celular com acesso discado à Internet. Os PDAs podem fazer acesso à Internet, caso o notebook realize o encaminhamento dos pacotes. Está, portanto, caracterizada uma rede *ad hoc* com múltiplos saltos, conforme mostrado na figura 2.

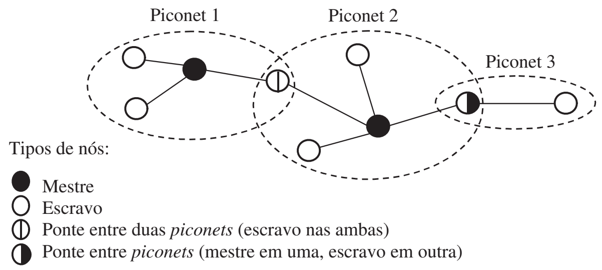

Figura 2: Exemplo da estrutura de uma *scatternet*.

Ao mesmo tempo em que propiciam versatilidade ao Bluetooth, as *scatternets* trazem consigo novas questões relativas à sua implementação. A especificação Bluetooth as define, mas mantém o foco na operação de uma *piconet* individual. As pesquisas relacionadas às *scatternets* se concentram na formação da topologia, no escalonamento *interpiconet* e no roteamento ou encaminhamento de pacotes.

A topologia da *scatternet* influi diretamente em parâmetros da rede como vazão, retardo fim-a-fim e consumo de energia. Além destes fatores, os algoritmos de formação de *scatternets* devem se preocupar com o tempo gasto para alcançar uma topologia estável. O algoritmo de escalonamento *interpiconet* deve funcionar em sincronismo com o escalonamento *intrapiconet*, de forma que o nó ponte somente seja escalonado pelo mestre de uma *piconet*, quando estiver presente nesta rede. Quanto ao encaminhamento de pacotes, existem duas soluções concorrentes: o uso do roteamento tradicional na camada IP ou o encaminhamento de pacotes realizado pelas camadas inferiores do Bluetooth. Recentemente, o grupo de trabalho PAN [7] do Bluetooth SIG especificou a BNEP (*Bluetooth Network Encapsulation Protocol*), uma camada intermediária entre L2CAP e IP. Seus objetivos são fornecer uma abstração da topologia para o nível de rede IP e realizar o encaminhamento de pacotes por múltiplos saltos. Isto permitirá, por exemplo, que uma rede composta por três *piconets* se comporte como um domínio de difusão único para o protocolo IP.

3 Pesquisas em Bluetooth

Cerca de um ano após a sua formação, o Bluetooth SIG tornou pública a versão 1.0A da especificação Bluetooth. A especificação, atualmente na versão 1.1, tem sido foco de muitas pesquisas, tanto por parte das empresas que compõem o SIG e da indústria de uma forma geral, como pelo meio acadêmico. A pesquisa em Bluetooth pode ser dividida em dois grupos principais, segundo a topologia: estudos em *piconets* e em *scatternets*.

Um dos pontos de maior enfoque nas *piconets* é o mecanismo de escalonamento *intrapiconet*. O estudo de técnicas de escalonamento não é recente. Em 1992, Liu et al. [9] publicaram um estudo sobre mecanismos de escalonamento e mostrou que as técnicas de escalonamento exaustivas são ótimas em sistemas simétricos. O esquema de acesso ACL é baseado em *polling*, mas com algumas diferenças em relação ao escalonamento tradicional. No Bluetooth, após a transmissão do mestre para o escravo, o escravo tem sempre a chance de transmitir pacotes no sentido reverso. Além disso, o mestre conhece as suas filas, mas não as dos escravos. No caso ideal, supõe-se que o mestre conheça as filas de todos os seus escravos, o que evita o desperdício de *slots* [10]. Apesar de não ser uma abordagem realista, vale como parâmetro de comparação. Capone et al. [3] mostram um estudo comparativo entre esquemas de *polling* e propõem uma variante para o *Exhaustive Round Robin* (ERR), o *Limited Weighted Round Robin* (LWRR). Das et al. [5] introduzem o *Adaptive Flow-based Polling* (AFP). Ambas as propostas variam a ordem de serviço dos escravos, conforme a disponibilidade de dados em suas filas, minimizando a passagem por escravos que não tenham dados a transmitir. Em contrapartida, necessitam da criação de parâmetros adicionais a serem transmitidos com os dados. Johansson et al. [11] comparam os algoritmos *Round Robin* (RR), *Exhaustive Round Robin*, chamado de *Exhaustive Polling* pelos autores, e *Fair Exhaustive Polling*, por eles introduzido, concluindo que o desempenho aumenta com a utilização de pacotes *multi-slots*. Entretanto, o estudo adota simplificações, tais como probabilidades de erro constantes e buffers ilimitados.

Os mecanismos de segmentação e remontagem (SAR), retransmissão (ARQ) e correção de erros (FEC) também são alvos de estudos. SAR influi no percentual de ocupação do meio físico. Já ARQ e FEC estão relacionados com problemas de interferência e evitam que os efeitos desta cheguem à camada de transporte. Das et al. [5] propõem um esquema de SAR que tenta minimizar o desperdício de banda, escolhendo pacotes de banda base convenientes (*best-fit*) para segmentar os pacotes L2CAP. Na pesquisa de Kalia et al. [10], estuda-se a possibilidade de transmitir o segundo pacote de uma fila antes do primeiro, caso este não se encaixe nos *slots* disponíveis. Para que o receptor saiba o pacote que está recebendo, é necessário o envio da informação adicional. Muitos dos erros são causados por interferência, a qual pode ocorrer tanto através da existência de múltiplas *piconets* na mesma área [2], [4], quanto pela coexistência de tecnologias que operam na mesma faixa de freqüência como outras redes Bluetooth e IEEE 802.11 [12].

As publicações específicas sobre *scatternets* são menos numerosas, em parte devido ao recente amadurecimento do assunto e também devido aos trabalhos proprietários, não publicados, do Bluetooth SIG. De modo geral, os trabalhos publicados se concentram em três áreas: a formação das *scatternets*, o escalonamento *interpiconet* e o encaminhamento de pacotes entre os nós. Apesar de fazerem parte da especificação Bluetooth, as *scatternets* não foram exploradas neste artigo.

4 BlueNetS: A Extensão Bluetooth para o Ns-2

Este trabalho faz parte de um esforço maior que é identificar, através de simulações, as deficiências do Bluetooth em prover, quando necessário, qualidade de serviço (QoS) ao tráfego e propor mecanismos para supri-las. Foi encontrado apenas um simulador com código aberto disponível: o BlueHoc [13], baseado no simulador *Network Simulator* (ns-2) [6]. O BlueHoc implementa diversas funcionalidades do Bluetooth, mas até o início deste trabalho permaneciam algumas limitações da ferramenta: tráfego apenas no sentido mestre-escravo, limite de apenas um tráfego por escravo, poucas aplicações suportadas, apenas um algoritmo de escalonamento *intrapiconet*, entre outras. No lugar de se modificar o código para estender suas funcionalidades, optou-se pelo desenvolvimento de um novo simulador, baseado nos recursos já existentes do ns-2.

O objetivo desta ferramenta é simular o funcionamento das camadas física e de enlace da pilha Bluetooth, a partir do instante em que as conexões estejam estabelecidas, ou seja, a partir do início da transferência de dados. As seguintes funcionalidades já estão implementadas e em uso: TDD, configuração de enlaces ACL, algoritmos de escalonamento *intrapiconet*, encaminhamento de pacotes nos sentidos mestre-escravo e escravo-mestre, mecanismos de ARQ e FEC, segmentação e remontagem (SAR) e modelo de erros provocados pela distância entre as estações. Estão sendo desenvolvidas: algoritmos de escalonamento *interpiconet*, troca de função entre mes-

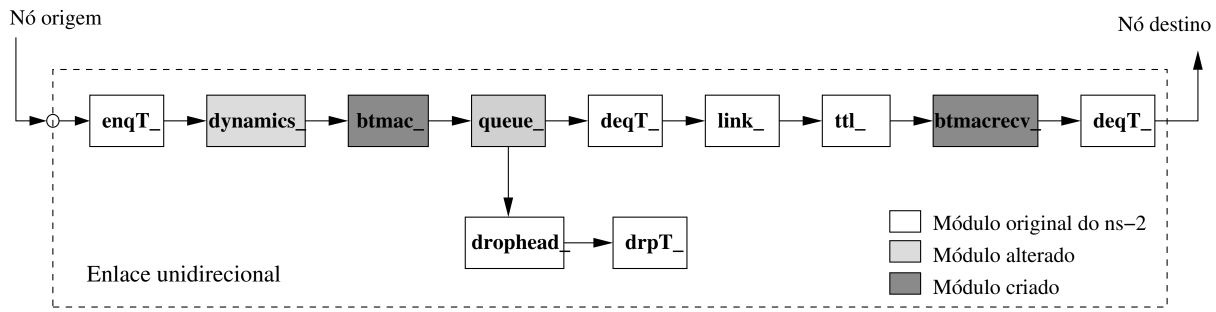

Figura 3: Componentes do enlace unidirecional ns-2 com modificações.

tre e escravo, modos de baixo consumo, negociação de QoS e modelo de erros provocados por interferência externa.

4.1 Características do Ns-2

O ns-2 é um simulador de redes orientado a objetos, baseado em eventos discretos. Seu código é escrito em duas linguagens: C++ e OTcl. O simulador é formado por vários componentes, escritos em C++, os quais podem ser combinados para fornecer topologias de rede específicas. Os componentes incluem, entre outros, nós, aplicações, agentes, protocolos de roteamento, filas, modelos de erro, enlaces, pacotes e temporizadores. A linguagem OTcl é usada como interface para a criação de cenários de simulação. Para cada classe C++ existe uma classe correspondente em OTcl. Assim, o usuário usa comandos OTcl para combinar os componentes desejados e formar o seu *script* de simulação.

O ns-2 está em constante evolução. Cada nova versão incorpora novos módulos. Neste trabalho, utilizou-se a versão 2.1b8a.

4.2 Modificações ao Simulador

Normalmente, em simulações de redes móveis com o ns-2, a inclusão de novas funcionalidades é feita no componente nó móvel. A ferramenta BlueNetS seguiu outra abordagem, onde foram usados os módulos básicos de redes fixas para a criação das *piconets*. O enfoque passou para o enlace entre nós do ns-2 (classe Simplex-link). Durante uma simulação, o enlace entre duas estações pode ser desativado ou ativado, simulando uma queda ou restabelecimento de conexão, respectivamente. A alternância entre ativação e desativação dos enlaces unidirecionais pode ser empregada na simulação do TDD entre o mestre e um escravo. Esta e outras características do Bluetooth foram modeladas, alterando-se apenas a estrutura do enlace unidirecional do ns-2. A figura 3 mostra os módulos que compõem o referido enlace após as modificações propostas.

Basicamente, o enlace unidirecional entre dois nós no ns-2 é composto dos módulos: queue_, simulando a fila de pacotes de saida do nó origem do enlace, link_, modelando as características de banda e retardo do enlace, drophead_, processando o descarte de pacotes do topo da fila e ttl_, verificando o *time to live* de cada pacote. Os módulos enqT_, deqT_, drpT_ e rcvT_ monitoram e registram, respectivamente, a entrada de pacotes na fila, sua saída da fila, seu descarte e seu recebimento pelo nó destino. A seguir, são destacadas as alterações realizadas no enlace.

4.2.1 Módulo btmac_

Este módulo é o responsável pela escolha do tipo de pacote banda base a ser utilizado na transmissão. Os principais tipos de pacote ACL (DHx, DMx, ID e Null) estão disponíveis para simulação. A escolha do pacote pode ser feita pelo usuário através de um parâmetro de configuração ou pode-se deixar que o simulador a realize, mediante uma política de *best-fit*, escolhendo o tamanho de pacote que melhor se ajuste à quantidade de dados disponíveis. Esta escolha está atrelada, também, ao uso ou não de FEC.

Caso o tamanho do pacote recebido seja maior que a capacidade do pacote ACL selecionado, o módulo btmac_ realiza a segmentação do pacote e encaminha os segmentos resultantes para o módulo queue_.

4.2.2 Módulo queue_

Queue_ simula a fila de pacotes de saída do nó (*host controller buffer*). No ns-2 sem modificações, quando um enlace é desativado, a fila é automaticamente esvaziada. No BlueNetS, a queda de um enlace indica que o nó correspondente não está sendo escalonado pelo mestre naquele instante. Portanto, modificou-se a estrutura de queue_ para que os pacotes não fossem descartados.

Neste módulo, foram incluídos o modelo de erros e o mecanismo de ARQ. No atual estágio da ferramenta, o único tipo de erro possível é o causado pela distância entre o transmissor e o receptor. Este modelo de erros foi adaptado do simulador BlueHoc [13], o qual é baseado no trabalho de Kumar e Gupta [1]. Ao invés de trabalharem com a simulação em nível de bit (BER - *bit error* *rate*), os desenvolvedores do BlueHoc tratam a taxa de erros por pacote (FER - *frame error rate*). Tais taxas de erro foram obtidas com simulações em um sistema de comunicação que incorporava filtro Gaussiano, modulador FM, canal de rádio em ambiente fechado (*indoor* [14]), detetor de quadratura e FEC. A potência do transmissor era de 1mW, usada para transmissões até 10 metros.

No momento da transmissão de um pacote, o modelo de erros é aplicado. Se a transmissão falhar, verifica-se a possibilidade de efetuar uma retransmissão. Se houver esta possibilidade, o pacote é mantido na fila, aguardando nova transmissão. Caso contrário, é descartado. Se o pacote descartado for um segmento de um pacote maior, todos os segmentos correspondentes a este também deverão ser descartados.

4.2.3 Módulo btmacrecv_

Os pacotes que atravessam o enlace com sucesso são recebidos pelo btmacrecv_. Este módulo realiza a remontagem de pacotes. Ele acumula os segmentos de um pacote até o recebimento do último segmento, quando, então, encaminha o pacote completo para o próximo módulo.

4.2.4 Código Complementar em Tcl

Para fornecer maior flexibilidade ao simulador, a função de escalonamento foi deixada como uma extensão em Tcl. Assim, incorporam-se novas políticas de escalonamento sem a necessidade de alteração estrutural no simulador. Foram testados alguns algoritmos de escalonamento, conforme será visto na seção de simulações.

5 Simulações

As simulações foram divididas em dois grupos: simulações de validação e simulações de mecanismos adicionais à especificação Bluetooth. O primeiro grupo mostra alguns dos resultados obtidos com o objetivo de averigüar o correto funcionamento da ferramenta. O segundo destaca alguns acréscimos feitos à ferramenta para melhorar o desempenho do enlace Bluetooth, quando submetido a determinadas condições.

5.1 Validação

Para validar a ferramenta, foi usada uma abordagem comparativa. Alguns cenários e resultados apresentados em artigos ([3], [15], [1], [16]) foram usados como parâmetros de comparação. Nos casos de funcionalidades implementadas pelo BlueHoc, este também foi usado para comparação. Além disso, foram testados alguns valores apresentados na própria especificação Bluetooth. Excetuando-se os módulos acrescentados ou modificados no ns-2, as outras funcionalidades empregadas como, por exemplo, as aplicações e os protocolos TCP e UDP, foram as existentes no ns-2, versão 2.1b8a.

Inicialmente, criou-se o cenário de uma *piconet* com 2 nós, onde, em intervalos fixos, era incorporado um novo escravo à rede, até o limite de 7 escravos. O objetivo era testar o compartilhamento do canal na *piconet*, medindo-se a vazão alcançada por cada fluxo e o retardo correspondente. Usou-se o algoritmo de escalonamento *Round Robin* (RR) e um tráfego FTP/TCP New Reno no sentido mestre-escravo, para cada escravo presente na rede. O FTP gerava pacotes de 1000 bytes. Os escravos estavam a menos de 5 metros do mestre, situação na qual, conforme será abordado adiante, não há erros provocados pela distância mestre-escravo. Posteriormente, realizou-se uma simulação análoga, onde a fonte FTP foi substituída por uma de taxa de geração de bits constante (CBR), com pacotes de 512 bytes e taxa de 650 kbps. Foram realizadas 5 rodadas de simulação, com os nós se associando à piconet a cada 8 segundos. Os resultados da vazão obtida pelo FTP e CBR são mostrados, respectivamente, nas figuras 4 e 5, enquanto o retardo em ambos os tipos de tráfego é apresentado na figura 6.

As simulações apresentaram resultados coerentes tanto em termos de ocupação e compartilhamento do canal, quanto em termos de retardo.

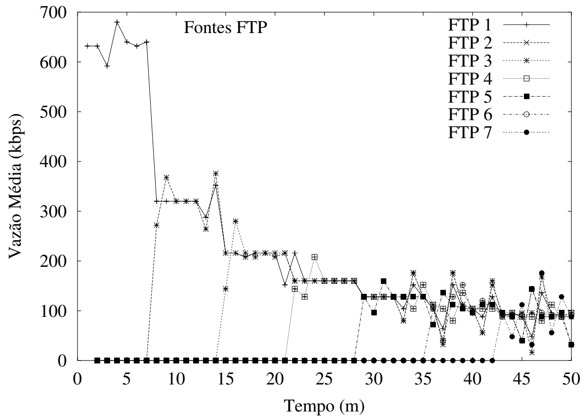

Figura 4: Variação da vazão de cada tráfego FTP, conforme a chegada de escravos.

Para estudar o comportamento do modelo de erros gerados pela distância entre o mestre e um escravo, foi criada uma *piconet* com 2 nós, onde se variou a distância entre os mesmos de 4 até 19 metros. Havia uma aplicação FTP/TCP New Reno do mestre para o escravo, gerando continuamente pacotes de 1000 bytes. Foram executadas 5 rodadas de 50 segundos cada, para cada distância medida. O mesmo cenário foi usado com um tráfego CBR do mestre para o escravo, com pacotes de 512 bytes e taxa de 400 kbps. Os resultados da vazão média obtida em cada distância, com intervalo de confiança de 95% nas duas simulações, são mostrados na figura 7.

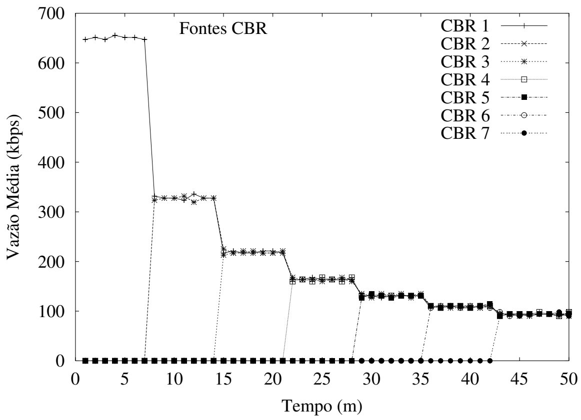

Figura 5: Variação da vazão de cada tráfego CBR, conforme a chegada de escravos.

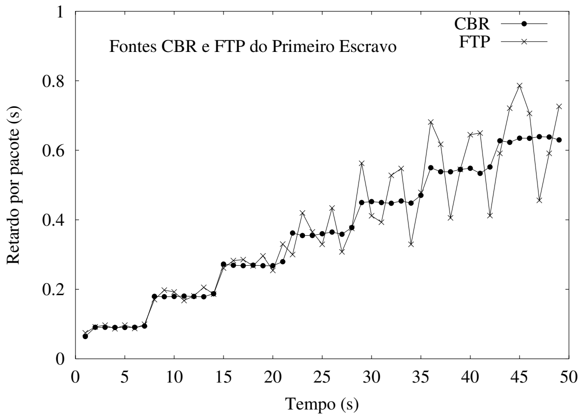

Figura 6: Retardo médio por segundo dos pacotes do primeiro escravo.

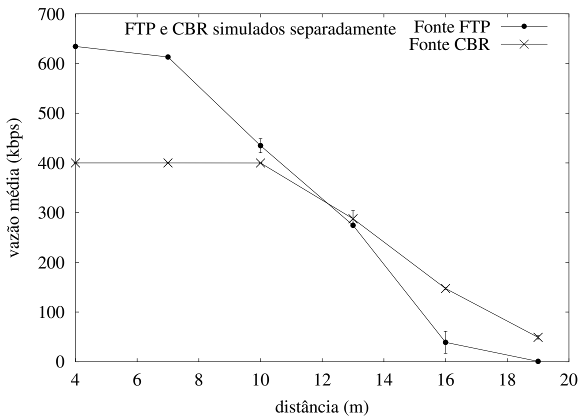

Figura 7: Vazão média de FTP/CBR em função da distância.

Pela figura 7, percebe-se que a partir de 6 metros de distância já começa a ocorrer degradação no desempenho do tráfego. Devido ao comportamento do TCP, as perdas causadas pela distância reduzem severamente o desempenho do FTP a partir dos 8 metros. Deve-se ressaltar que como foram usados pacotes de 1000 bytes para o FTP, o procedimento de segmentação da camada L2CAP o divide em segmentos de 5 slots, formando 3 pacotes DH5. Pacotes menores ou utilizando FEC são mais resistentes ao erro. Entretanto, devido a seu pequeno tamanho (por exemplo, DH1 armazena 27 bytes de dados e DM1, apenas 17), podem gerar muitos segmentos para um pacote da camada superior. Como a perda de apenas um segmento acarreta o descarte de todo o pacote, nem sempre é vantajosa a utilização de pacotes menores ou com FEC.

Os mecanismos de ARQ e FEC foram criados para fornecer maior robustez à transmissão sem fio. O uso de FEC 2/3 para os dados pode reduzir a vazão de um fluxo, já que parte do pacote é ocupada por código de correção de erros. Ao mesmo tempo, o emprego do ARQ com a possibilidade de muitas retransmissões do mesmo pacote pode aumentar o retardo. O estudo seguinte visa identificar em que situações a utilização de ARQ e FEC é favorável.

Foi criado um cenário com dois nós, com uma aplicação FTP do mestre para o escravo. Foi escolhido o FTP devido à suscetibilidade a erros do TCP ao interpretá-los como congestionamento. Variou-se a distância entre o transmissor e o receptor, testando-se em cada caso várias combinações de ARQ e FEC 2/3 para dados. Há dois valores possíveis para FEC: 0, sem código de correção de erros, e 1, com correção de erros. Um valor de ARQ = X representa que o mesmo pacote pode ser retransmitido até X vezes antes de ser descartado. Foi medida a vazão média alcançada em cada situação, com intervalo de confiança de 95%. Os resultados mais relevantes são mostrada na figura 8. Alguns valores foram suprimidos para facilitar a compreensão do gráfico.

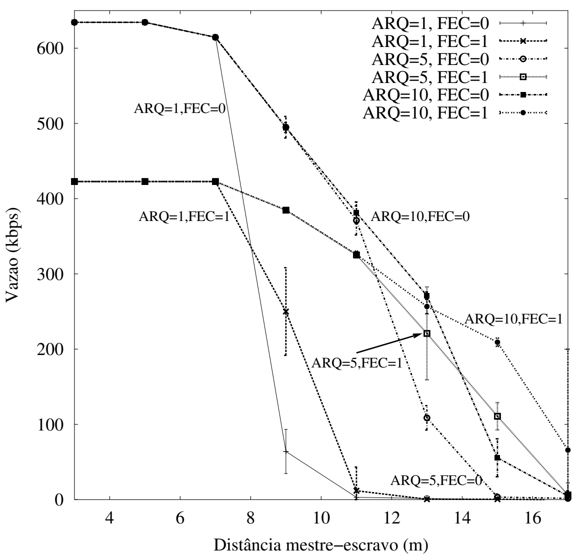

Figura 8: Vazão x distância para várias combinações de ARQ e FEC.

Os resultados mostram que até 8m de distância todos os tráfegos sem FEC (FEC=0) produzem uma vazão maior que os com FEC. Entretanto, já começam a degradar ligeiramente a partir de 6m. No caso de ARQ=1, imaginado para tráfegos de voz, devido à preocupação com o retardo, ocorre uma queda vertiginosa próximo aos 8m. A partir desta distância, compensaria o uso de FEC, pois a queda é menos brusca. Apesar de a potência de 1mW ser dimensionada para transmissões até 10m, consegue-se uma ocupação de até 300kbps em torno de 13m, usando-se 5 ou mais retransmissões. Se o tráfego não tiver restrições de retardo, esta pode ser uma solução.

5.2 Modificações Propostas

Esta seção aborda, inicialmente, os algoritmos de escalonamento *intrapiconet* desenvolvidos para a ferramenta. São eles: *Round Robin* (RR), *Deficit Round Robin* (DRR), *Deficit Round Robin* Ideal (DRR-I) e *Exhaustive Round Robin* Modificado (ERR-M). No RR, o mestre escalona, em uma ordem fixa, os escravos exatamente uma vez por rodada. O DRR tenta garantir justiça ao escalonamento atribuindo um *quantum* (Q) para os nós. Um nó somente poderá transmitir se tiver crédito suficiente. Uma característica inerente ao Bluetooth é o fato de que toda transmissão no sentido mestre-escravo é seguida de outra no sentido inverso. Como o mestre controla o escalonamento e ele tem conhecimento apenas de suas filas (tráfego do mestre para o escravo), algum escravo, mesmo tendo dados a transmitir, pode não ser escalonado, devido à falta de dados no sentido mestre-escravo. Para resolver este problema, estipulou-se um intervalo de tempo máximo em que um escravo pode ficar sem ser escalonado. O DRR-I é uma implementação do DRR em que o mestre conhece, a todo instante, suas filas e as dos escravos. Apesar de não ser uma implementação realista, serve como parâmetro de comparação. Finalmente, o ERR-M continua atendendo um par mestre-escravo até suas filas esvaziarem, quando, então, escalona o próximo escravo cuja soma das filas nos sentidos mestre-escravo e escravo-mestre seja a maior. Diferentemente do DRR-I, no ERR-M a informação sobre a fila de um escravo só é obtida quando este é escalonado. Na implementação ERR-M, usa-se ainda um parâmetro (M) para indicar a quantidade máxima de pacotes que um nó pode transmitir em seqüência naquela rodada. Assim, evita-se que um nó com muitos dados a transmitir retenha indefinidamente o direito de transmissão.

Na implementação, o *quantum* Q usado no DRR foi definido em número de slots. Por exemplo, se Q=6, em uma rodada pode-se transmitir um pacote de 5 slots (DH5 ou DM5), acumulando-se uma unidade para a próxima rodada, ou podem ser transmitidos dois pacotes de 3 slots (DH3 ou DM3). No caso do ERR-M, o limite M indica a quantidade máxima de pacotes que um nó pode transmitir de uma determinada fila em uma única rodada. Nesta simulação, usou-se Q=8 para DRR e DRR-I e M=5 para o ERR-M.

Para testar os algoritmos de escalonamento, foi criada uma *piconet* com 6 estações e aplicações heterogêneas. A figura 9 mostra a topologia da simulação.

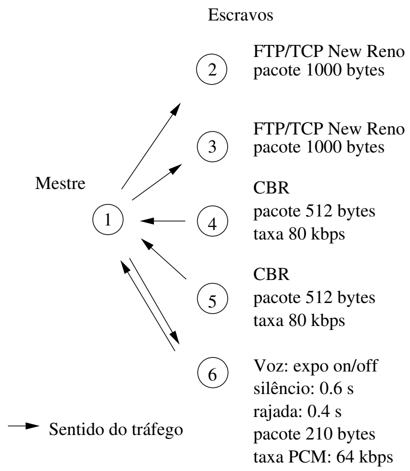

Figura 9: Cenário para simulação de escalonamento.

Foram realizadas 5 rodadas de simulação para cada algoritmo de escalonamento. As estações estavam a, no máximo, 5 metros do mestre, para evitar erros provocados pela distância entre transmissor e receptor. Todos os tráfegos se iniciaram no mesmo instante e ficaram ativos até o fim da simulação. Entre o mestre e o nó 6 existe um tráfego de voz em cada sentido. A voz foi configurada sobre um enlace ACL, como se fosse tráfego de melhor esforço. O desempenho do tráfego de voz sobre enlace ACL será explorado mais adiante. Finalmente, as filas dos nós armazenam até 20 pacotes. O tamanho da fila de saída de um enlace ACL estabelece um compromisso entre o retardo dos pacotes e o descarte dos mesmos. Uma fila muito grande evita que pacotes sejam descartados, mas pode provocar o aumento do retardo dos pacotes. Por outro lado, uma fila pequena limita o retardo, mas pode provocar o descarte de pacotes, caso a fila se encha. Como esta simulação é comparativa, o importante é que todos os algoritmos usem o mesmo tamanho de fila.

Ao término de cada rodada foi computada a vazão média e o retardo médio de cada tráfego. Tirou-se, então, a média de todas as rodadas para cada algoritmo de escalonamento. Estas informações estão representadas na tabela 5.2, juntamente com os valores agregados.

Os resultados da tabela 5.2 confirmaram o esperado. Os algoritmos ERR-M e DRR-I obtiveram vazão agregada maior que o RR. Por outro lado, o DRR-I teve os menores retardos médios. Isto está relacionado com o fato de o DRR-I conhecer o tamanho de todas as filas dos enlaces ACL. Dependendo do *quantum* e do limite configurados para o DRR e o ERR-M, respectivamente, alguns tráfegos podem ser favorecidos. Os tráfegos FTP (1-2 e 1-3) apresentam vazão maior nos casos do DRR e do ERR-M, pois como os outros tráfegos (CBR e Expo)

Tráfego	RR		DRR-I		DRR		ERR-M	
	Vazão méd(kbps)	Retardo médio(s)	Vazão méd(kbps)	Retardo médio(s)	Vazão méd(kbps)	Retardo médio(s)	Vazão méd(kbps)	Retardo médio(s)
FTP (1-2)	166.40	0.344	193.60	0.298	166.50	0.345	187.38	0.309
FTP (1-3)	166.40	0.342	192.70	0.299	167.90	0.344	184.40	0.308
CBR (4-1)	90.93	0.026	90.04	0.021	89.60	0.024	90.76	0.053
CBR (5-1)	91.02	0.028	90.53	0.022	90.50	0.026	91.02	0.052
Expo(1-6)	21.04	0.010	25.50	0.017	23.36	0.021	24.86	0.048
Agregado	539.79	-	592.37	-	528.13	-	578.42	-

Tabela 1: Vazão e retardo médios para cada mecanismo de escalonamento.

apresentam taxas baixas, pode-se aumentar a quantidade de pacotes FTP transmitidos em seqüência antes de se escalonar o próximo nó.

O uso de voz sobre enlace SCO reserva slots a intervalos fixos, fornecendo garantias de vazão e de retardo limitado ao tráfego. Entretanto, restringe as aplicações baseadas em ACL. Por exemplo, para voz configurada com pacotes HV3, o intervalo entre dois pacotes consecutivos é de 6 slots, ficando apenas 4 disponíveis para dados. Considerando-se que existem períodos de silêncio no tráfego de voz, o desperdício de banda torna-se ainda maior.

Este trabalho propõe um esquema de escalonamento para permitir o compartilhamento de tráfegos de voz e dados sobre um enlace ACL, garantindo, ainda assim, retardo limitado para a voz. O algoritmo de escalonamento DRR-QoS baseia-se no DRR, onde se introduziu um parâmetro IPol (intervalo de *polling*) para controlar o intervalo entre escalonamentos sucessivos do tráfego de voz. Ao invés de garantir slots fixos como o SCO, fornece um retardo aproximado. Ao se esgotar o tempo IPol, o mestre espera o término da transmissão corrente e obrigatoriamente escalona o nó cujo tráfego está associado ao IPol. No pior caso, a transmissão corrente pode ocupar 10 slots (5 slots em cada sentido), permitindo uma variação no retardo daquele tráfego de até 6,25ms.

Nesta simulação, a voz foi modelada como um processo que alterna entre períodos de transmissão e silêncio, seguindo distribuições exponenciais com média de 0.4 e 0.6s, respectivamente [17]. Considerando que um codificador PCM gera dados a uma taxa de 64kbps, pode-se dizer que no período de transmissão de voz, os pacotes, com tamanho de 210 bytes, chegam, em média, a cada 26ms. Configurou-se o IPol=20ms para o DRR-QoS. Assim, os pacotes tendem a ser servidos logo que chegam, evitando-se o aumento do retardo pelo acúmulo de pacotes na fila.

O cenário usado foi uma *piconet* com 8 nós, conforme mostrado na figura 10. Foram configurados vários tráfegos heterogêneos, inclusive com tráfegos em ambos os sentidos para alguns pares mestre-escravo. Assim, conseguiu-se aumentar a duração de cada rodada do escalonamento. O objetivo era monitorar o retardo fim-a-fim dos pacotes de voz, obtido através das diversas políticas de escalonamento e compará-lo com o obtido através do DRR-QoS. O valor de IPol utilizado foi 20ms. O resultado comparativo é mostrado na figura 11. Para facilitar o entendimento desta figura, os resultados obtidos com o DRR-QoS estão separados na figura 12.

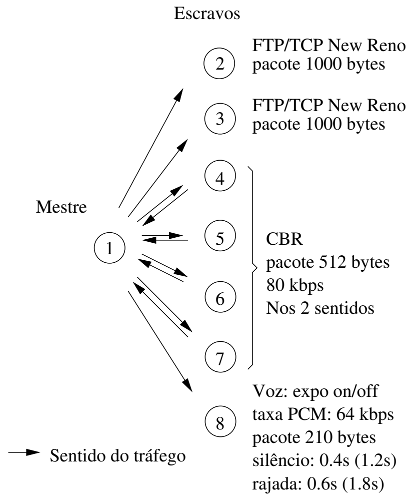

Figura 10: Cenário para teste do DRR-QoS.

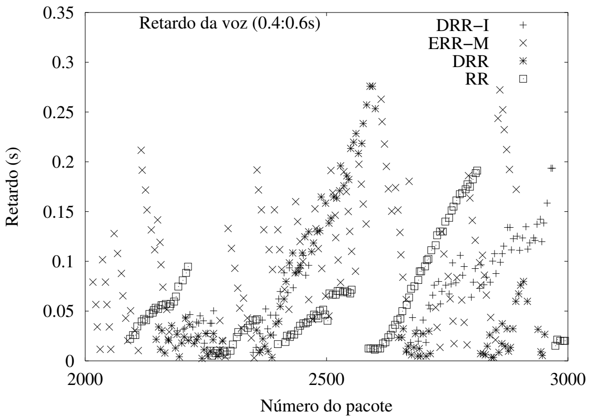

Figura 11: Retardo dos pacotes de voz (tráfego: 0,4s; silêncio: 0,6).

Os resultados mostram que o retardo dos pacotes de voz passa de 250ms em alguns mecanismos de escalonamento, mas se mantém limitado a cerca de 25ms ao longo de toda a simulação com o DRR-QoS.

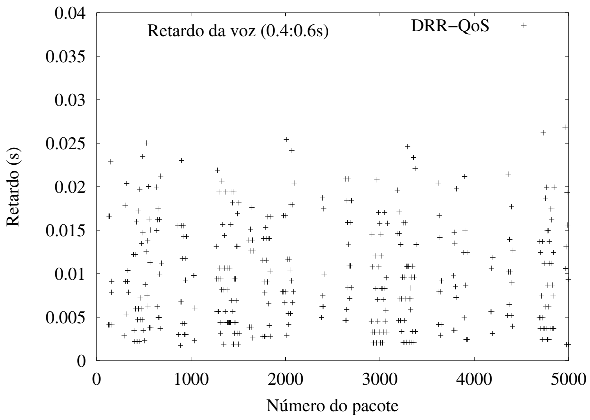

Figura 12: Retardo de voz no DRR-QoS (tráfego: 0,4s; silêncio: 0,6).

O mesmo cenário foi usado para testar outra modelagem de voz, onde a taxa é mantida a 64kbps, mas os períodos das distribuições exponenciais de tráfego e silêncio são de 1,2 e 1,8s, respectivamente [18], [19]. Apesar de a proporção de tráfego/silêncio ser mantida inalterada em relação ao outro modelo, este apresenta períodos mais longos de tráfego e, portanto, rajadas mais longas. Isto tende a gerar um retardo ainda maior para as aplicações em geral, incluindo a voz, conforme mostra a figura 13. Os resultados obtidos com o DRR-QoS são mostrados separadamente na figura 14.

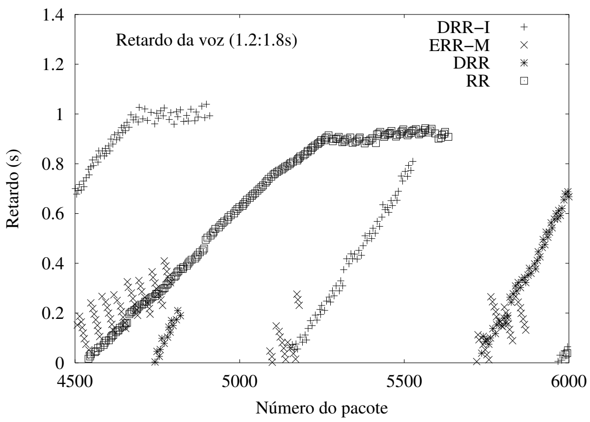

Figura 13: Retardo dos pacotes de voz (tráfego: 1,2s; silêncio: 1,8).

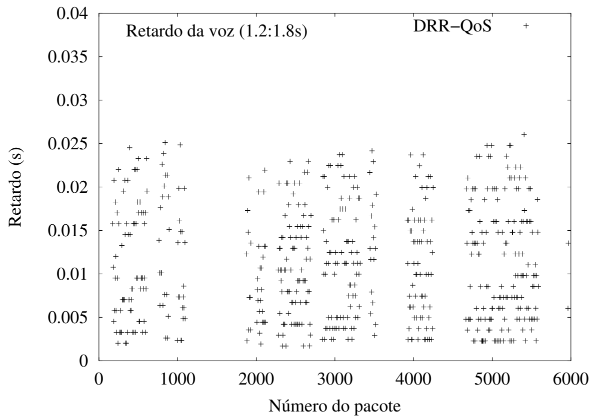

Figura 14: Retardo de voz no DRR-QoS (tráfego: 1,2s; silêncio: 1,8).

Novamente, o retardo dos pacotes usando o algoritmo DRR-QoS ficou limitado em cerca de 25ms, comprovando a eficácia do sistema. Apesar de o DRR-QoS ter se comportado bem do ponto de vista da limitação do retardo, é importante verificar a influência do esquema nos outros tráfegos da mesma piconet. Para isso, mediu-se a vazão agregada dos tráfegos ao longo da simulação. O resultado é apresentado na figura 15. A vazão agregada obtida com o DRR-QoS é, na média, um pouco inferior à obtida com os outros algoritmos de escalonamento. Em alguns momentos a vazão agregada cai para a faixa dos 600kbps, o que, ainda assim, é muito superior à alcançada com o uso de enlace SCO.

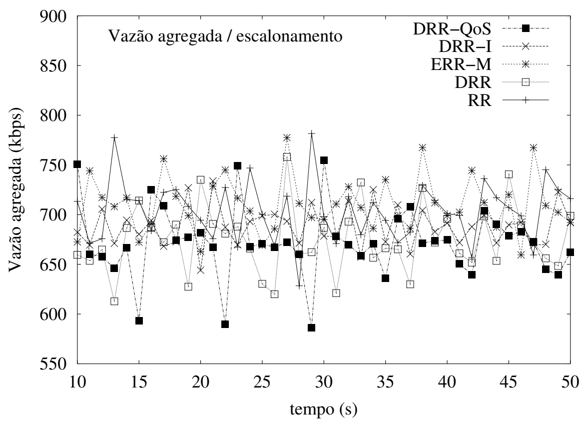

Figura 15: Vazão agregada da piconet (tráfego: 1,2s; silêncio: 1,8).

Atualmente, existem codificadores ADPCM para voz que geram amostras a taxas de 16 a 40kbps. Há, também, codificadores recentes baseados em quadros que atingem taxas de 8kbps [19]. Empregando-se este tipo de tecnologia nas *piconets*, o parâmetro IPol poderia ser aumentado, permitindo uma maior ocupação do canal por tráfegos de melhor esforço.

6 Conclusões e Trabalho Futuro

Bluetooth é uma tecnologia promissora para a formação de pequenas redes móveis *ad hoc*. Com o objetivo de estudar o seu desempenho em determinadas condições, foi desenvolvida a ferramenta BlueNetS, uma extensão ao simulador ns-2. Um aspecto a ser ressaltado sobre a ferramenta é o fato de ter sido implementada aproveitando-se ao máximo as funcionalidades disponíveis no ns-2. Apenas dois novos módulos foram incorporados e dois existentes foram modificados. Foram apresentadas as principais funcionalidades já implementadas, incluindo o escalonamento *intrapiconet*, mecanismos de correção de erros e retransmissão de pacotes, segmentação e remontagem, entre outras. Em seguida, apresentaram-se alguns resultados de validação da ferramenta, testando o compartilhamento do canal, o modelo de erros e os mecanismos de ARQ e FEC. Realizaram-se, também, algumas simulações com os algoritmos de escalonamento *intrapiconet* implementados, incluindo um algoritmo de escalonamento baseado no DRR, o DRR-QoS, modificado para atender a aplicações sensíveis ao retardo, como tráfego conversacional. O algoritmo mostrou ser plenamente viável a utilização de tráfegos de voz em enlaces ACL, concomitantemente aos tráfegos de melhor esforço.

Este artigo faz parte de um trabalho mais abrangente que prevê o estudo das *scatternets*. Algumas publicações existentes avaliam a formação das *scatternets*, sem levar em consideração as necessidades das aplicações. Precisa-se estudar a reorganização dos nós ao longo do tempo. Estações chegam e saem, além de criarem conexões temporárias dedicadas que se desfazem rapidamente. Pretende-se abordar, também, o escalonamento *interpiconet* e o encaminhamento de pacotes na rede. Os módulos relacionados a estas funcionalidades estão em desenvolvimento, já sendo possível, inclusive, realizar o encaminhamento de pacotes entre *piconets* com o BlueNetS.

Referências

[1] A. Kumar, e R. Gupta. Capacity evaluation of frequency hopping based ad-hoc systems. *ACM Sigmetrics/Performance*, Jun 2001.

[2] S. Zürbes, W. Stahl, e K. Matheus. Radio network performance of bluetooth. *IEEE ICC'00*, 3:1563–1567, 2000.

[3] A. Capone, M. Gerla, e R. Kapoor. Efficient polling schemes for bluetooth piconets. *IEEE ICC'01*, Jun 2001.

[4] C. M. Cordeiro, e D. P. Agrawal. Enhancing performance of the bluetooth wireless channel through dynamic segmentation. *20 SBRC*, Mai 2002.

[5] A. Das, A. Ghose, A. Razdan, H. Saran, e R. Shorey. Enhancing performance of asynchronous data traffic over the bluetooth wireless ad-hoc network. *Proc. of IEEE Infocom*, Abr 2001.

[6] The Network Simulator (ns-2). http://www.isi.edu/nsnam/ns. Jun 2002.

[7] The Bluetooth Special Interest Group. http://www.bluetooth.com. Mai 2002.

[8] J. Haartsen. The bluetooth radio system. *IEEE Personal Communications*, 7(1):28–36, Fev 2000.

[9] Z. Liu, P. Nain, e D. Towsley. On optimal polling policies. *Queueing Systems 11*, pages 59–83, 1992.

[10] M. Kalia, D. Bansal, e R. Shorey. Data scheduling and sar for bluetooth mac. *IEEE Vehicular Technology Conference (VTC)*, Mai 2000.

[11] N. Johansson, U. Korner, e P. Johansson. Performance evaluation of scheduling algorithms for bluetooth. *Broadband Communications*, pages 139–150, 2000. Kluwer Academic Publishers.

[12] N. Golmie, R. E. Van Dyck, e A. Soltanian. Interference of bluetooth and ieee 802.11: Simulation modeling and performance evaluation. *4th Int. ACM Workshop on Modeling, Analysis and Simulation of Wireless and Mobile Systems*, Jul 2001.

[13] BlueHoc: Bluetooth Performance Evaluation Tool. http://www-124.ibm.com/developerworks/opensource/bluehoc. Mar 2002.

[14] T. S. Rappaport, S. Y. Seidel, e K. Takamizawa. Statistical channel impulse response models for factory and open plan building radio communication system design. *IEEE Transactions on Communications*, 39(5):794–807, Mai 1991.

[15] P. Johansson, N. Johansson, U. Körner, J. Elgg, e G. Svennarp. Short range radio based ad hoc networking: Performance and properties. *IEEE ICC'99*, 1999.

[16] S. Chawla, H. Saran, e M. Singh. Qos based scheduling for incorporating variable rate coded voice in bluetooth. *IEEE ICC'01*, Jun 2001.

[17] M. Schwartz. Broadband integrated networks. *Prentice Hall Inc*, 1996.

[18] P. Brandy. A technique for investigating on/off patterns of speech. *Bell Labs Tech. Journal*, 44(1):1–22, Jan 1965.

[19] A. Markopoulou, F. Tobagi, e M. Karam. Assessment of voip quality over internet backbones. *IEEE Infocom*, Jun 2002.