Análise do mecanismo de pares de pacotes visando estimar a banda da rede via UDP

Valter Roesler[1,2], Peter Finzsch[1], Maiko de Andrade[1], José Valdeni de Lima[2]

[1]UNISINOS – Universidade do Vale do Rio dos Sinos, Centro de Ciências Exatas e Tecnológicas. Av. Unisinos, 950, CEP 93022-000. São Leopoldo – RS

[2]Instituto de Informática – Universidade Federal do Rio Grande do Sul (UFRGS) Caixa Postal 15064 - Av. Bento Gonçalves, 9500 Bloco IV– Porto Alegre – RS

{roesler, peteman, maiko}@exatas.unisinos.br, valdeni@inf.ufrgs.br

Abstract. *This paper intends to analyze in depth the packet-pair and packet-train method to estimate the network bottleneck. The results can be useful to any algorithm that needs network bandwidth inference, but is intended in particular to a new layered congestion control algorithm [Roesler 2003]. This work uses low rate UDP packets, and the receiver estimates the bandwidth without sending ACKs to transmitter, which is helpful in multicast communications. Many simulations were performed. Besides that, the system was implemented and a lot of tests were executed in a laboratory test bed and also in the Internet, with measurements via ADSL, intranet and the Internet2.*

Keywords: packet-pair, congestion control, friendliness.

Resumo. *O presente artigo tem como objetivo analisar em detalhes o funcionamento do mecanismo de pares de pacotes e trens de pacotes, utilizado para descoberta de banda da rede, permitindo a inferência de sua velocidade máxima. Espera-se utilizar os resultados em um novo algoritmo de controle de congestionamento em camadas [Roesler 2003], mas a análise pode ser útil para qualquer sistema que necessite inferência de banda entre dois pontos da rede. Uma diferença entre este trabalho e outros relacionados é que este utiliza somente UDP, ou seja, o receptor infere a banda sem necessidade de envio de ACKs ao transmissor, o que é útil quando se pensa em comunicações multicast. Além disso, os pares de pacotes são enviados a cada 0,5 segundos ou mais, evitando rajadas na rede. Para validar o mecanismo, diversas simulações foram efetuadas, bem como a implementação do sistema. Diversos testes foram realizados, tanto em um ambiente controlado de laboratório como na Internet, com medições via ADSL, Intranet e Internet2.*

Palavras-chave: pares de pacotes, controle de congestionamento, friendliness.

1 Introdução

Para que novos algoritmos baseados em UDP sejam *TCP-friendly*, ou seja, "sua quantidade de dados transferida em longo prazo não exceda a quantidade de dados transferida por um fluxo TCP sob as mesmas circunstâncias" [Floyd 1999], os mesmos devem inferir sua fatia de banda e efetuar controle de congestionamento de forma

amigável com os outros tráfegos. Uma informação muitas vezes útil nesse caso é a banda máxima entre transmissor e receptor, bem como a banda eqüitativa da rede. É nesse ponto que o mecanismo de pares de pacotes pode auxiliar, pois através dessa técnica é possível inferir a largura de banda tanto em unicast como em multicast.

No caso do unicast, o próprio transmissor pode inferir a banda, esperando o ACK dos pacotes do par e se baseando no espaçamento entre eles para calcular a velocidade máxima da rede. Neste caso, deve-se ter cuidado para o receptor não retardar o envio do ACK (técnica de *delayed ACK* [Braden 1989]). O uso dos ACKs possui duas características negativas [Legout 2000], [LAI 2000]: a) não há como saber se o congestionamento é no sentido transmissor-receptor ou vice-versa, pois o espaçamento pode ter sido causado no canal reverso; b) a reação ao congestionamento é mais lenta, já que o transmissor deve esperar os ACKs para poder reduzir sua taxa de transmissão.

No caso do multicast, normalmente é o receptor que infere a largura de banda da rede, baseado no espaçamento do par de pacotes. Nesse caso, a banda máxima medida é no sentido transmissor-receptor, e como o receptor não utiliza ACKs, diminui a necessidade de geração de tráfego, entretanto, exige um cliente e um servidor.

Entre outros objetivos, este trabalho pretende analisar o mecanismo e verificar viabilidade de integração no SAM (Sistema Adaptativo para Multimídia) [Roesler 2003], cujo diagrama em blocos pode ser visto na figura 1. O transmissor gera a codificação do sinal multimídia em camadas e transmite cada camada como um grupo multicast diferente. No lado do receptor, o módulo de controle de congestionamento controla a adaptação, que é baseada em detecção de perdas, estimativa de eqüidade de banda e estimativa de banda máxima (através da técnica de pares de pacotes).

No SAM, o transmissor envia os pacotes de conteúdo multimídia em pares de pacotes, e o receptor não envia ACKs, portanto, não é gerado qualquer tráfego acima do que seria gerado sem o uso da técnica. Outra possibilidade seria enviar os pares de pacotes separadamente, a cada 0,5 segundos ou mais, gerando pouco tráfego adicional.

Figura 1. Visão geral do SAM (Sistema Adaptativo para Multimídia).

A maior parte dos trabalhos relacionados busca a implementação de uma ferramenta que obtenha rapidamente uma resposta para a banda máxima entre dois pontos, gerando tráfego na rede. Esse não é o caso para a utilização que se pretende.

A principal contribuição deste artigo é efetuar uma análise da viabilidade do mecanismo de pares de pacotes orientado a receptor, via UDP, sem a utilização de ACKs e com geração espaçada de pares de pacotes, ou seja, adequado para uso em multicast e sem gerar rajadas de tráfego na rede. Os resultados podem ser úteis para qualquer algoritmo que necessite inferência de banda entre dois pontos da rede.

O resultado da análise deve dar uma idéia da possibilidade que tem o receptor para inferir a banda máxima na rede a partir do espaçamento entre os pacotes recebidos, verificando o nível de exatidão desta inferência. A análise é complementada com

simulações e também com uma implementação real. Várias situações são analisadas visando validar o método de pares de pacotes, como o seu uso em enlaces de diferentes velocidades, com vários tamanhos de pacotes e com diferentes RTTs.

O restante deste documento é organizado da seguinte forma: a sessão 2 mostra os trabalhos relacionados. A sessão 3 detalha o funcionamento do método de pares de pacotes. Na sessão 4, o ambiente de simulação é descrito, bem como os resultados atingidos. O mesmo acontece na sessão 5, porém, numa implementação testada em um ambiente controlado e no ambiente da Internet. A sessão 6 apresenta as conclusões, bem como a intenção de trabalhos futuros.

2 Trabalhos Relacionados

Um dos primeiros mecanismos de controle de fluxo através de pares de pacotes foi proposto em [Keshav 1991]. Nele o autor assume que todos os roteadores utilizam filas do tipo "*round-robin*", e todos os fluxos são servidos igualmente. Essa disciplina foi definida em detalhes e denominada "*Rate Allocating Servers*". Utilizando essa infra-estrutura, o nível de transporte do transmissor envia os pacotes em pares, medindo o espaçamento dos ACKs na chegada e calculando a banda eqüitativa na rede. Essa informação é utilizada em seu esquema de controle de fluxo baseado em taxa.

Outro trabalho relacionado é o proposto em [Carter 1996], onde o método de pares de pacotes é utilizado com o objetivo de descobrir a maior largura de banda entre um conjunto de servidores alternativos, permitindo a escolha do servidor com a maior taxa de transmissão. Para validar os argumentos, Carter desenvolveu uma ferramenta chamada *bprobe*, que tem como objetivo estimar a largura de banda máxima no gargalo da rede, e outra chamada *cprobe*, cujo objetivo é estimar a largura de banda eqüitativa entre dois pontos. Na ferramenta *bprobe*, são transmitidos aproximadamente 50 pares de pacotes ICMP do tipo *echo*, com vários tamanhos. O espaçamento entre os ACKs de retorno é medido, e o resultado final da banda é a União dos resultados próximos. Na ferramenta *cprobe*, são transmitidos 4 conjuntos de 8 pacotes, e a banda é calculada para cada conjunto, prevenindo perda de pacotes em alguma medição. O resultado é a média da banda obtida, ou a banda eqüitativa da transmissão.

Kevin Lai [LAI 2000] utilizou um mecanismo denominado "*packet tailgating*", onde o primeiro pacote do par é ICMP com TTL reduzido, sendo encaminhado juntamente com outro de tamanho pequeno, fazendo com que o segundo pacote seja sempre enfileirado junto com o primeiro, até o descarte do primeiro pelo roteador (quando seu TTL expira). Com base nas mensagens de retorno dos roteadores e ACKs do pacote de *tailgating*, Lai propôs a ferramenta *nettimer*.

Posteriormente, o protocolo TCP Westwood (TCPW) [Gerla 2001] utilizou a taxa de retorno dos ACKs e o espaçamento entre eles para determinar a banda eqüitativa da transmissão. Com base nessa taxa, o TCPW utiliza um mecanismo modificado, o AIAD (*Additive Increase Adaptive Decrease*) para não precisar diminuir pela metade a taxa de transmissão quando ocorre uma perda, já que nem sempre uma perda significa congestionamento, não precisando então diminuir a taxa de envio de dados. Simulações mostram uma melhora de vazão de até 578% em enlaces *wireless* sem prejuízo das conexões TCP Reno em andamento. Em testes reais, o protocolo mostrou um aumento de vazão entre 31% e 185% numa rede de alta velocidade da NASA, e de 47% na

Internet quando a conexão TCP inclui um enlace de satélite.

3 Descrição do método de pares de pacotes

Um efeito bem conhecido nas redes de computadores é o espaçamento em pacotes adjacentes causado pelo limite físico de banda no enlace de menor velocidade entre origem e destino, ou seja, no gargalo da rede [Keshav 1991], [Jacobson 1988]. Esse efeito é utilizado para estimativa de banda por diversos protocolos, como o TCP Westwood [Gerla 2001] e o PLM [Legout 2000], além de algumas ferramentas de medição de banda, como o *bprobe* [Carter 1996] e o *nettimer* [LAI 2000].

A essência do mecanismo pode ser ilustrada através da Figura 2, onde se pode ver que, quando passam por um enlace de menor velocidade (1 Mbit/s na topologia da Figura 2) na ausência de tráfego concorrente, os pacotes sofrem um espaçamento que é preservado nos enlaces de maior velocidade. Através desse espaçamento e do tamanho dos pacotes, é possível inferir a largura de banda entre origem e destino.

Para verificar o funcionamento numa topologia multicast simples, foi modificado o código fonte do simulador NS[1] a fim de que ele pudesse transmitir em pares de pacotes, e foi efetuada uma simulação, conforme topologia ilustrada na Figura 2. Como pode-se ver, os pacotes são transmitidos em pares através de um enlace de 2 Mbit/s, e sofrem um espaçamento no gargalo da rede, representado pelo roteador 1, que possui uma largura de banda de 1 Mbit/s. Esse espaçamento permanece até o receptor R1, e pode ser visto claramente no enlace de 10 Mbit/s.

O tamanho dos pacotes transmitidos foi de 500 bytes, e o espaçamento medido em R1, do início da chegada do primeiro pacote até o início da chegada do segundo pacote foi de 4 ms (visto através do simulador, tempos de 2,738691 s - 2,734691 s = 4 ms). Com esses dados, o receptor infere que o máximo que a rede consegue transmitir é 500 bytes (4.000 bits) em 4 ms, ou seja, 1 Mbit/s, que é equivalente ao enlace de menor velocidade, ou o gargalo da rede.

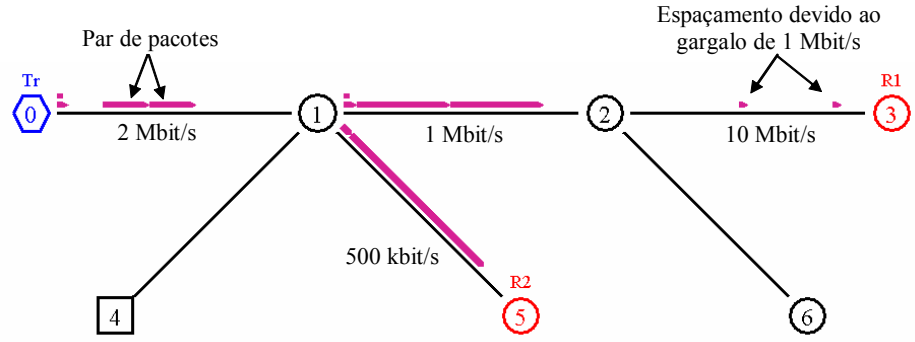

Figura 2. Exemplo de inferência de banda por pares de pacotes.

Entretanto, numa rede real, o tráfego concorrente dificulta a utilização de pares de pacotes de forma precisa, especialmente quando se utilizam filas do tipo FIFO, que é a mais utilizada na Internet [Croll 2000]. Alguns erros de inferência no receptor são

[1] Projeto VINT – *Virtual InterNetwork Testbed*. Em http://www.isi.edu/nsnam/vint (03/01/2003)

descritos a seguir:

- **Inferência de banda maior que a real**: o par de pacotes sofre um espaçamento no gargalo da rede, conforme mostra a Figura 3 (no item (a) e enlace de 1 Mbit/s). Entretanto, caso exista tráfego concorrente (representado pelos pacotes brancos) após o gargalo da rede, o espaçamento pode ser eliminado na fila do roteador 2. Isso faz com que a transmissão do par de pacotes no enlace de 10 Mbit/s aconteça sem espaçamento algum, fazendo o receptor R1 inferir uma largura de banda de até 10 Mbit/s ao invés de 1 Mbit/s.
- **Inferência de banda menor que a real**: os roteadores recebem tráfego de várias interfaces, podendo acontecer de inserir um pacote entre o par de pacotes transmitido, provocando um "escorregamento" no segundo pacote do par, levando a um espaçamento maior do que aconteceria devido simplesmente ao tráfego concorrente. Caso isso aconteça no gargalo, como, por exemplo, no roteador 1 da Figura 3 (item b), o receptor R1 vai inferir uma largura de banda menor do que o real, pois o espaçamento é maior do que deveria. De acordo com a figura, com pacotes de 500 bytes, o espaçamento para inferir a banda máxima deveria ser 4 ms, gerando uma inferência de banda de 1 Mbit/s, entretanto, o espaçamento é de 12 ms, e a banda inferida é de 333 kbit/s, o que é menor do que o real. Outra causa de inferência de banda menor pode ser devido aos atrasos gerados pelo Sistema Operacional (os pares de pacotes podem não ser transmitidos um após o outro, no caso da geração de uma chamada de sistema entre eles) ou pela concorrência da rede local (a taxa de transferência do Ethernet compartilhado não é determinística).

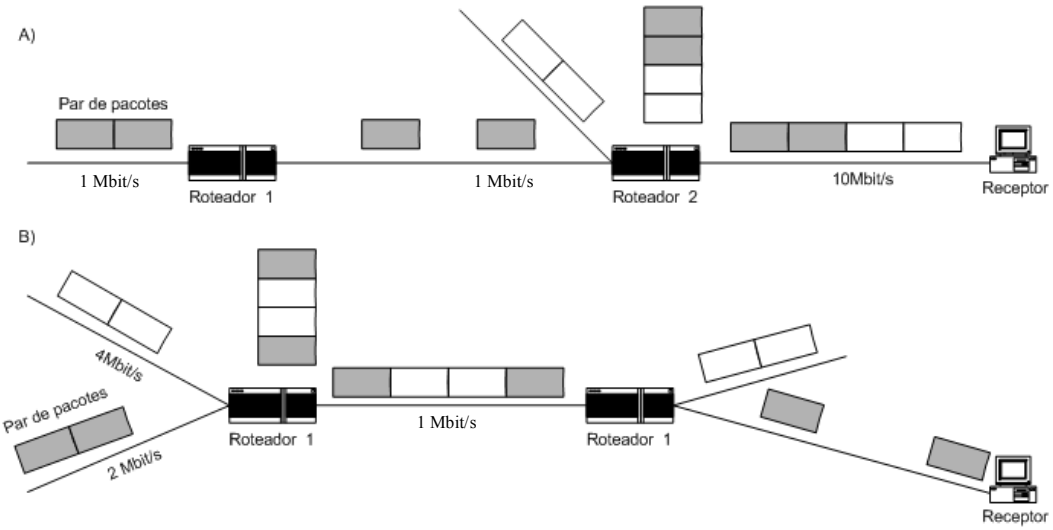

Figura 3. Patologias no uso de pares de pacotes em redes reais.

Caso o roteador utilize uma política de filas que aproxime o conceito de GPS (*Generic Processor Sharing*) [Huston 2000], a distribuição da transmissão de dados será mais precisa entre os diversos fluxos que passam pelo roteador, permitindo maior precisão na inferência de banda através de pares de pacotes. O ideal seria a utilização de uma política de filas do tipo WF2Q (*Worst Case Fair Weighted Fair Queuing*), conforme proposto por Legout no desenvolvimento do protocolo PLM (*Packet Pair Receiver Driven Layered Multicast*) [Legout 2000]. Entretanto, essa não é a realidade da Internet atual, portanto, tal alternativa não foi simulada neste artigo.

4 Simulações efetuadas

O simulador de redes utilizado foi o NS2, e os objetivos das simulações em relação ao método de pares de pacotes foram os seguintes:

1. Analisar o funcionamento do método através de simulação;
2. Analisar o impacto da variação no tamanho do pacote;
3. Verificar o funcionamento do método em enlaces de diferentes velocidades;
4. Analisar as conseqüências da variação do RTT.

A fim de atingir os objetivos propostos para as simulações, utilizou-se a topologia mostrada na figura 4. Na figura, estão representados os transmissores e receptores de pares de pacotes (PP) e de tráfego concorrente (TCP).

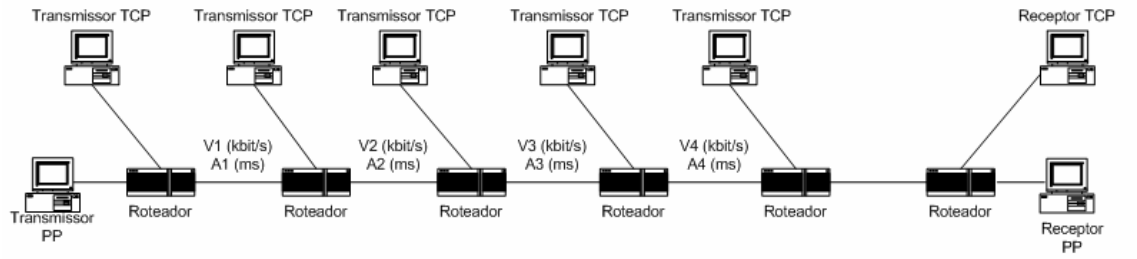

Figura 4: Topologia da rede simulada.

Os enlaces entre os roteadores possuem velocidades e atrasos variáveis, sendo representado por V (em kbit/s) e A (em ms). Todos outros enlaces possuem uma largura de banda de 10 Mbit/s e atraso de 1 ms.

O código do simulador foi modificado de forma que o transmissor enviasse um par de pacotes aproximadamente a cada 500 ms. Cada par de pacotes contém um número de seqüência, que avisa ao receptor se o pacote é o primeiro ou o segundo do par. Quando chega o primeiro pacote, o receptor armazena o tempo de chegada. Quando chega o segundo, ele calcula a diferença de tempo e faz a inferência da banda, gravando o resultado num arquivo de *log*. Caso chegue um pacote com o número de seqüência errado, o receptor assume que houve erro na transmissão, ignorando a medida para esse par de pacotes.

Diversas simulações foram efetuadas a fim de responder cada um dos objetivos listados acima. Cada item a seguir trata da metodologia e resultados obtidos para cada um dos objetivos.

4.1 Analisar o funcionamento do método através de simulação

A simulação a seguir visa simplesmente verificar a funcionalidade do método de pares de pacotes. Para isso, foram iniciadas 20 sessões FTP (quatro em cada transmissor TCP) concorrentemente com o transmissor de pares de pacotes, numa simulação de 50 segundos. Nesse conjunto de simulações, a velocidade dos enlaces intermediários (V1 a V4) foi de, respectivamente, 1Mbit/s, 2Mbit/s, 4Mbit/s e 8Mbit/s. Os valores foram escolhidos de forma a gerar as patologias descritas através da figura 3. Posteriormente, as velocidades foram invertidas (8M, 4M, 2M, 1M). O atraso (A1 a A4) foi fixo em 1 ms, e o tamanho de todos pacotes foi de 1000 bytes.

Pode-se observar no primeiro gráfico da figura 5 os dois tipos de patologia

descritos anteriormente.O receptor inferiu banda erroneamente, chegando a ter picos de 8Mbit/s, quando o espaçamento entre os pares de pacotes desaparecia na fila do roteador 4 (anterior ao enlace de 8Mbit/s). Entretanto, pode-se ver que a média ficou próxima ao resultado correto, que é de 1 Mbit/s, levando a crer que seja possível criar um algoritmo para desprezar os picos e inferir a banda de forma aproximadamente correta.

O segundo gráfico teve as velocidades invertidas, portanto, o espaçamento dos pares de pacotes nunca diminuía. Esse é o motivo pelo qual nenhuma inferência deu acima de 1Mbit/s. Entretanto, quando pacotes TCP entravam no meio do par, causavam atraso e provocavam uma inferência de banda menor do que o gargalo da rede. Da mesma forma que o primeiro gráfico, a média ficou próxima da realidade. Alguns pacotes são perdidos, e nem todos gráficos atingem as 100 amostras nos 50 segundos.

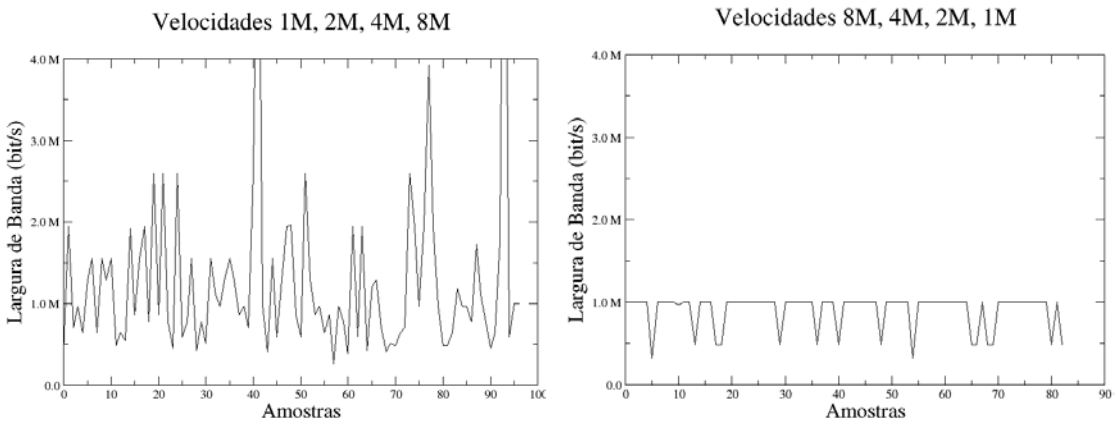

Figura 5: Inferência da banda do receptor de pares de pacotes.

4.2 Analisar o impacto da variação no tamanho do pacote

Para analisar o impacto da variação do tamanho do pacote na precisão das medições, utilizou-se pacotes de 64 bytes, 500 bytes, 1000 bytes e 1500 bytes, com e sem tráfego concorrente. O tráfego concorrente foi composto de 5, 10, 20 e 40 sessões FTP divididas igualmente nos transmissores TCP da figura 4, com tamanho de pacote de 1000 bytes. As velocidades V1 a V4 foram, respectivamente, 1M, 2M, 4M e 8Mbit/s, pois geram as duas patologias descritas na figura 3. O atraso A dos enlaces foi mantido em 1ms.

Sem tráfego concorrente a inferência de banda é perfeita, e todas as medidas são de 1Mbit/s. Com tráfego concorrente, as variações acontecem. A figura 6 mostra 20 sessões TCP concorrendo com o par de pacotes. Pode-se ver que o tamanho do pacote influencia na medição, e pacotes pequenos são menos estáveis. Com pacotes de 64 bytes, houve vários picos de 8Mbit/s (o pacote era muito pequeno frente ao do TCP, e perdia o espaçamento na fila do roteador). Com pacotes de 500 bytes o pico máximo foi de 4Mbit/s, e a média das inferências já ficou próxima do gargalo de 1Mbit/s. Com pacotes de 1000 bytes, ou seja, de tamanho igual aos pacotes TCP, a inferência foi a mais estável e correta de todas. Essa característica se manteve em inúmeros outros testes realizados, mostrando que pacotes de tamanho maior geram inferência de banda mais precisa.

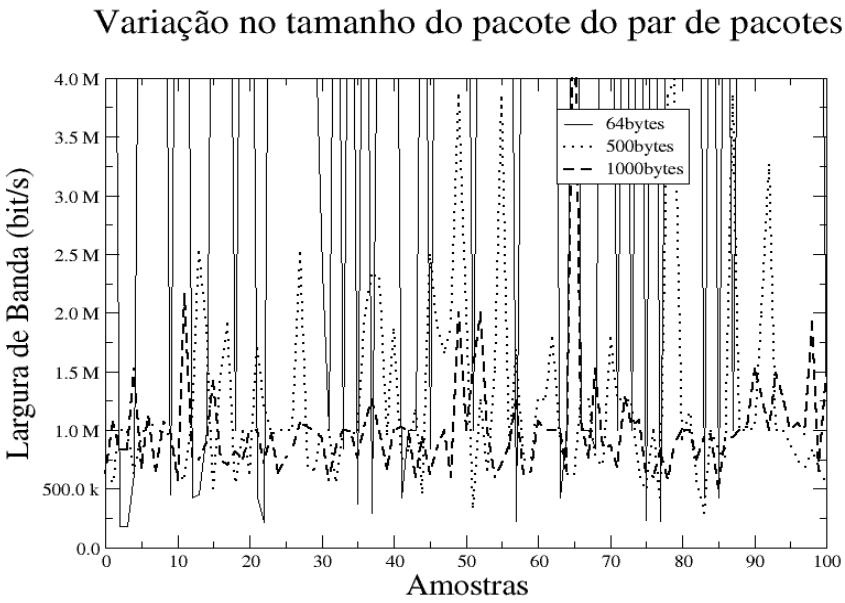

Figura 6: Pares de pacote com 64 bytes, 500 e 1000 bytes.

4.3 Verificar o funcionamento do método em enlaces de diferentes velocidades

Para analisar se existe alguma variação na precisão do método para redes de alta e baixa velocidade, variou-se a velocidade V1 a V4 dos enlaces da seguinte forma: a) 100kbit/s, 200kbit/s, 400kbit/s e 800kbit/s; b) 1Mbit/s, 2Mbit/s, 4Mbit/s e 8Mbit/s; c) 10Mbit/s, 20Mbit/s, 40Mbit/s e 80Mbit/s (nesse caso se aumentou a velocidades dos enlaces de 10Mbit/s para 100Mbit/s). O tráfego concorrente variou de 5, 10, 20 e 40 sessões FTP, com tamanhos de pacote de 1000bytes. O tamanho dos pacotes do par também foi de 1000 bytes. O atraso A foi mantido em 1ms.

A figura 7 mostra os resultados para os gargalos de 100kbit/s e 10Mbit/s, pois o de 1Mbit/s já foi visto anteriormente. Pode-se ver que, tanto para 20 como 40 sessões FTP concorrentes, o sistema inferiu aproximadamente a banda correta, na média.

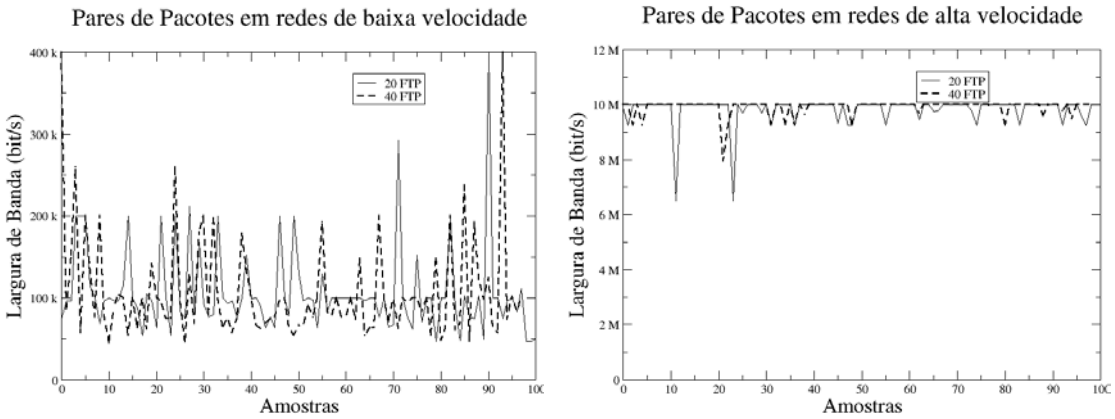

Figura 7: Gargalo de 100 kbit/s e 10Mbit/s.

4.4 Analisar as conseqüências da variação do RTT

Nesse conjunto de simulações, o atraso A de cada enlace foi variado a fim de analisar seu impacto sobre a precisão da inferência de banda. Assim, as variáveis A1 a A4 variaram de 1ms, 10ms e 100ms em todos enlaces simultaneamente. Com atraso de 1ms, o RTT do primeiro transmissor TCP é de 10ms, e com atraso de 100ms, o RTT passa para mais de 800ms. As velocidades dos enlaces foram mantidas em 1Mbit/s, 2Mbit/s, 4Mbit/s e 8Mbit/s, e os testes foram efetuados com 5, 10, 20 e 40 sessões FTP concorrentes. O tamanho dos pacotes TCP e dos pares de pacotes foi fixo em 1000 bytes.

A figura 8 mostra o resultado para 20 sessões FTP concorrentes. Pode-se ver que, com RTT grande, a precisão do método é maior. Isso se deve à demora de ajustes do tráfego TCP, que vai ter um menor número de pacotes circulando na rede a cada momento.

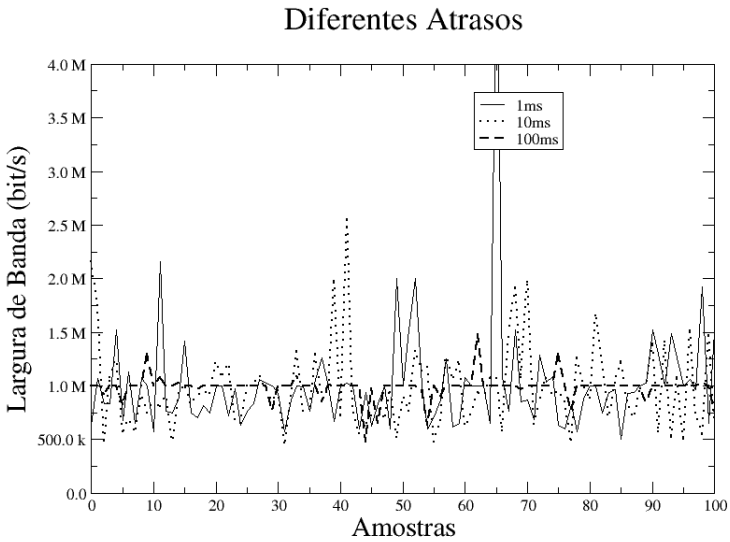

Figura 8: Impacto da variação do RTT na precisão de inferência de banda.

5 Implementação do sistema

O objetivo da implementação foi comprovar os resultados das simulações numa rede real, com tráfego concorrente e diferentes velocidades de enlace. O código está disponível em http://prav.unisinos.br/pp. Na implementação real, outras variáveis afetam o método de pares de pacotes, como o atraso no sistema operacional, a concorrência no *hub* de saída de rede, e a variabilidade de tráfego na Internet. Assim, os objetivos totais da implementação foram os mesmos das simulações acrescidos dessas outras variáveis mencionadas, e estão descritos a seguir:

1. Analisar o erro gerado pelo Sistema Operacional e concorrência na rede local;
2. Analisar o impacto da variação no tamanho do pacote;
3. Verificar o funcionamento do método em enlaces de diferentes velocidades.

O sistema é composto de um módulo transmissor e um módulo receptor. A implementação do receptor foi efetuada em Unix, pois esse sistema permite uma maior precisão na leitura de tempo. Já o transmissor foi implementado para Unix e Windows,

e a análise do espaçamento entre os pacotes mostrou que ambos são precisos, conforme descrição adiante.

O módulo transmissor envia, de tempos em tempos, uma rajada de pacotes com determinado tamanho. Essa rajada pode ser configurada para dois pacotes, caracterizando o par de pacotes (*packet-pair*), ou para mais pacotes, sendo chamada de "trem de pacotes" (*packet-train*). O protocolo utilizado é o UDP e a linha de comando do transmissor tem as seguintes opções:

```
pptrans -h IPdest -p porta -s tam -d atraso -n tamtrem

-h IPdest: IP do destino;
-p porta: porta a ser utilizada no destino;
-s tam: tamanho dos pacotes a serem transmitidos;
-d atraso: atraso entre cada trem de pacotes, em ms;
-n tamtrem: número de pacotes na rajada.
```

O parâmetro "*-n*" foi criado para verificar se a precisão do método aumenta caso forem transmitidos mais de dois pacotes em seqüência, ou seja, se em vez de um par de pacotes, for transmitido um trem de pacotes.

O módulo receptor recebe o trem de pacotes e calcula a largura de banda com o transmissor baseando-se no espaçamento entre eles. Os resultados são vistos na tela do micro e também gravados em um arquivo de *log*, a fim de que se faça uma análise gráfica. A linha de comando do receptor tem as seguintes opções:

```
pprec -p porta -n tamtrem -o arquivo

-p porta: porta a ser utilizada no receptor;
-n tamtrem: numero de pacotes em seqüência;
-o arquivo: nome do arquivo de log de saída.
```

Todo pacote que circula na rede é composto de cabeçalho e dados. Nos testes efetuados, havia um cabeçalho Ethernet produzindo um *overhead* de 18 bytes, um cabeçalho IP de 20 bytes e um cabeçalho UDP de 8 bytes. Em função desses cabeçalhos, o parâmetro de tamanho de pacote escolhido pelo usuário no transmissor (opção *–s*) era reduzido em 46 bytes a fim de deixar o tamanho final do pacote conforme solicitado. Entretanto, ainda foram desprezados o preâmbulo do pacote Ethernet (8 bytes) e o "*Inter Frame Gap*" (IFG), equivalente a 12 bytes ou 96 bits, que é utilizado para permitir compartilhamento nas redes locais [Spurgeon 2000]. Em linhas gerais, o algoritmo do transmissor efetua as seguintes operações:

```
Lê parâmetros de entrada(); (1)
seqno=0;
while () {
    for(i=0; i<tamtrem; i++) {
      buffpp[0]=seqno++;      (2)
      sendto(...)
    }
    usleep (ppdelay*1000);    (3)
}
```

Inicialmente, os parâmetros (*-h*, *-p*, *-s*, *-d* e *-n*) são lidos e armazenados (1). Em seguida, o programa entra no laço principal, ficando nele até o término do programa. A posição *buffpp[0]* contém o indicativo do número de seqüência do pacote no trem de pacotes transmitido (2). Para que o algoritmo funcione, o tamanho do trem de pacotes deve ser sempre 2^n, onde *n* varia de 1 (par de pacotes) até 8 (rajadas de 256 pacotes).

Após transmitir o trem de pacotes, o programa entra em espera pelo tempo (em milisegundos) solicitado pelo usuário (3).

No lado do receptor, o algoritmo deve descobrir se houve perda de pacotes na rede, incrementando um indicador de perdas caso positivo. A seguir pode ser visto resumidamente o algoritmo executado no receptor, que é explicado em seguida.

```
while(1) {
    numrec=recvfrom(...);
    gettimeofday(...);                                    (1)

    SE (PRIMEIRO_PACOTE_DO_TREM)
        Armazena_Tempo(); flag=1;                         (2)
    ELSE // outros pacote da mesma rajada
        SE (ERRO_SEQÜÊNCIA)
            IncPerdas(); Continua_laço;
        SE (ULTIMO_PACOTE_DO_TREM)
            flag=0;
            diftempo = tempoultimo - tempoprimeiro;
           // Cálculo da banda                             (3)
            BW=((tamtrem-1)*((numrec+46)*8)*1000/diftempo);
            fprintf(flog...); // grava no arquivo de log
}
```

Dentro do laço principal, o receptor lê o pacote da placa de rede, armazenando imediatamente o número de bytes que vieram (*numrec*) e o tempo que o pacote chegou (1). Em seguida, o receptor descobre se o pacote é o primeiro do trem de pacotes, lendo a posição inicial do buffer de recepção (se múltiplo de *tamtrem*, é o primeiro). Caso seja o primeiro pacote do trem, o receptor simplesmente armazena o tempo e liga uma *flag* indicando que o primeiro pacote chegou (2). Caso seja algum outro pacote da rajada, o receptor verifica se houve perda de pacotes (número de seqüência inválido ou *flag* em zero), incrementando um contador de perdas. Caso não haja perdas e o pacote seja o último do trem de pacotes (segundo no caso do par de pacotes), o receptor calcula a diferença de tempo na chegada dos pacotes e efetua o cálculo da banda, em kbits/s (3). O cálculo da banda deve levar em consideração os 46 bytes de cabeçalho, conforme descrito anteriormente. Após obter o resultado, atualiza um *log* para posterior análise e geração dos gráficos.

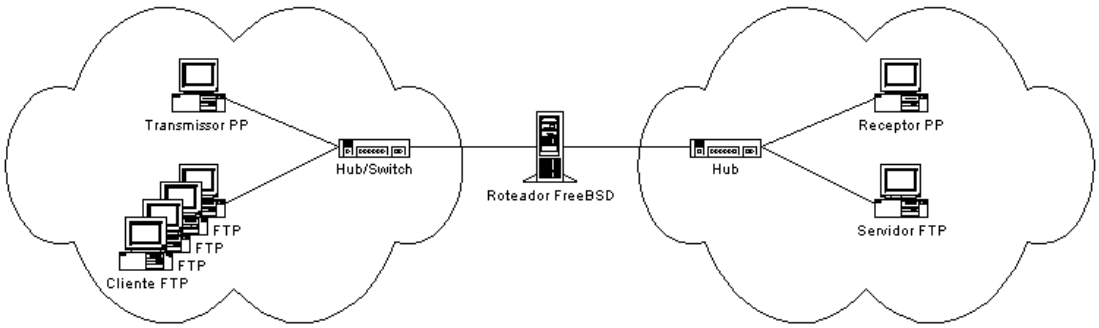

Figura 9. Ambiente controlado de teste.

Para efetuar os testes, se utilizou um ambiente controlado e também a Internet, com acesso via ADSL, Intranet, Internet tradicional e Internet2.

O ambiente controlado de teste pode ser visto na figura 9. O transmissor envia pares de pacotes configurados com as opções de acordo com cada objetivo. O *hub* é utilizado para verificar o erro devido ao sistema operacional e concorrência na rede. Para gerar tráfego concorrente, se utilizaram conexões FTP em paralelo. O roteador é uma máquina FreeBSD, e é possível configurar a velocidade do enlace via comando *ipfw*. Em alguns testes o transmissor e o receptor PP ficavam na mesma subrede.

5.1 Atraso devido ao Sistema Operacional e concorrência na rede

Para verificar o atraso de saída dos pares de pacotes causados pelo sistema operacional e concorrência na rede, os pacotes foram capturados via *hub* através do software *tcpdump*, e os resultados foram analisados com a interface do Ethereal[1], onde o espaçamento entre os pares foi medido. Caso o atraso fosse superior ao esperado, seria considerado uma indicativa de erro na geração dos pares de pacotes.

A figura 10 mostra a tela do analisador dos pacotes, que foram capturados sem tráfego concorrente. Como cada pacote foi configurado com um tamanho de 1000 bytes, ou 8 kbits, e a rede é de 10 Mbit/s, pode-se calcular que o intervalo *i* de saída entre os pacotes deve ser, teoricamente: $i = 8\ kbits\ /\ 10Mbit/s$, ou seja, $i = 0{,}8ms$. O espaçamento medido entre os primeiros quatro pares de pacotes foi de 0,793ms, 0,813ms, 0,806ms e 0,808ms, gerando um erro de aproximadamente 8 µs, ou 1%. Foi considerado que o resultado medido está coerente com a teoria. O mesmo se repete entre outros tamanhos de pares de pacotes, bem como no sistema operacional tipo Unix, comprovando que o atraso no sistema operacional é desprezível para pacotes de 1000 bytes. Entretanto, vale lembrar que, para pacotes de 100 bytes, o atraso normal é de 0,08ms, ou 80µs, e um erro de 8µs corresponde a 10%, que é inserido já na saída dos pacotes.

No.	Time	Source	Destination	Protocol
1	0.000000	192.168.2.2	192.168.2.3	UDP
2	0.000793	192.168.2.2	192.168.2.3	UDP
3	0.504387	192.168.2.2	192.168.2.3	UDP
4	0.505200	192.168.2.2	192.168.2.3	UDP
5	1.009245	192.168.2.2	192.168.2.3	UDP
6	1.010051	192.168.2.2	192.168.2.3	UDP
7	1.509196	192.168.2.2	192.168.2.3	UDP
8	1.510004	192.168.2.2	192.168.2.3	UDP

Figura 10. Espaçamento devido ao sistema operacional.

As medições foram realizadas com e sem processos rodando em paralelo. Sem processos em paralelo, as medições ficaram bastante estáveis. Com processos em paralelo (abertos aleatoriamente durante todo o teste), alguns pares de pacotes ficaram mais espaçados que o normal. No caso mais extremo, o par de pacotes número 46 sofreu um espaçamento de 2,554ms, e o correto seria de 1,2ms (pacotes de 1500 bytes). A conseqüência é que a banda medida ficou em 4,6 Mbit/s, e não 10 Mbit/s. Entretanto, foram erros esporádicos e não chegaram a inviabilizar as medições.

Com tráfego concorrente no mesmo *hub*, algumas vezes entra um pacote TCP no meio do par de pacotes UDP, gerando um erro grande já na saída da rede, como era de se esperar. No item 5.3 esse resultado será analisado.

[1] www.ethereal.com

5.2 Variação no tamanho do pacote

Para analisar se o método é afetado pela variação no tamanho do pacote, o programa foi rodado repetidamente em ambiente controlado e na Internet, e o tamanho dos pacotes foi modificado, assumindo os valores de 64 bytes, 500 bytes, 1000 bytes e 1518 bytes. Foram efetuadas 500 medições para cada teste, e o espaçamento entre cada par de pacotes ficou em 0,5s. O teste foi repetido para um par de pacotes (*tamtrem*=2) e uma rajada de 8 pacotes (*tamtrem*=8), com o objetivo de ver se a precisão do método é afetada pelo tamanho da rajada. Vale lembrar que alguns roteadores fragmentam pacotes IP com mais de 576 bytes, invalidando as medições. Uma possível alternativa é efetuar descoberta de MTU antes de enviar os pacotes, entretanto, isso não foi implementado na presente versão.

A figura 11 mostra uma medição realizada a partir da sala do PRAV (Pesquisa em Redes de Alta Velocidade), passando pela Intranet da Unisinos e por um enlace de 2Mbit/s com o provedor Terra. A partir daí o sinal entra na Universidade Federal do Rio Grande do Sul e volta à sala do PRAV através do enlace de Internet2 (155 Mbit/s). A topologia é conhecida, e apesar da Unisinos possuir 4 Mbit/s com a Internet comercial, o enlace é composto por dois E1, de 2Mbit/s cada.

O primeiro gráfico da figura 11 mostra o resultado para pacotes de 64bytes, com rajadas de 2 e 8 pacotes. Pode-se ver que a instabilidade é muito grande com pares de pacotes (2pp), porém, melhora um pouco com rajadas de 8 pacotes. No segundo gráfico, o tamanho do pacote foi de 1000 bytes, sendo bem mais estável que o primeiro. Com pares de pacotes, o sistema apresentou alguns picos, mas a média se mostrou bem próxima do valor real da rede. Com rajadas de 8 pacotes o sistema ficou bastante estável, inferindo a banda correta da rede na grande maioria das amostras. Outros tamanhos de pacotes foram utilizados, e chegou-se à conclusão que pacotes de 500 bytes já resultam numa boa precisão de medida.

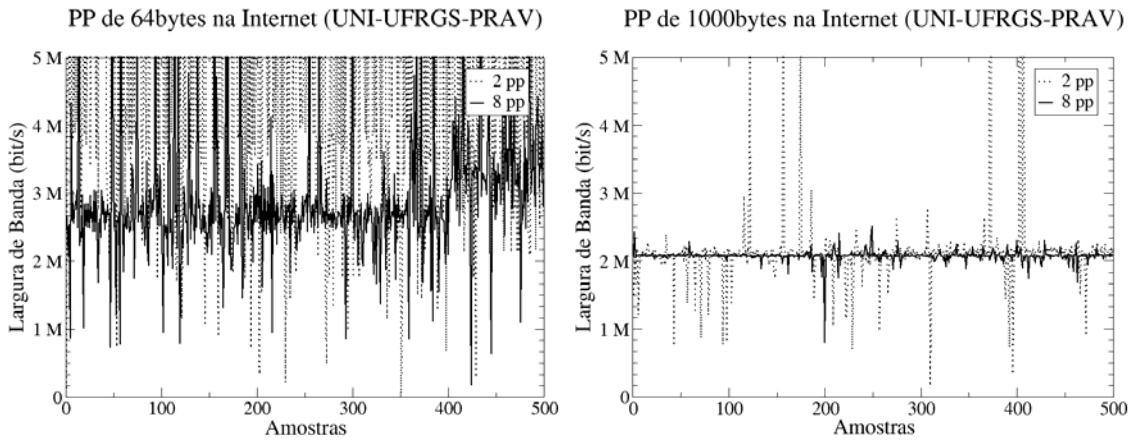

Figura 11. Pacotes de 64 bytes e 1000 bytes, rajadas de 2 e 8 pacotes.

Os mesmos testes foram repetidos em ambiente controlado, com e sem tráfego concorrente. Os resultados foram similares, porém, com uma dificuldade adicional: a limitação de tráfego utilizada no roteador (comando *ipfw*) sofria de certas instabilidades, provavelmente causado pelo algoritmo conformador de tráfego. A conseqüência é que cerca de 10% das vezes o roteador enviava dois pacotes com o máximo de sua velocidade (10 Mbit/s), causando erro naquela medição. Apesar disso, o restante das

medições dava uma boa idéia do funcionamento do algoritmo, e os resultados no ambiente controlado foram similares ao ambiente real, ou seja, mostraram uma maior instabilidade em pacotes com tamanho pequeno.

5.3 Variação na velocidade dos enlaces

Para analisar se o método funciona e detecta a banda em enlaces de várias velocidades, o programa foi rodado repetidamente em ambiente controlado e na Internet. O tamanho dos pacotes foi mantido em 1000 bytes, pois dá uma boa precisão na medida, conforme visto no item anterior. Foram efetuadas 500 medições para cada teste, e o espaçamento entre cada par de pacotes ficou em 0,5s.

Em ambiente controlado, a velocidade do roteador foi configurada para 100 kbit/s, 500 kbit/s e 1 Mbit/s. Além disso, utilizou-se diretamente duas estações se comunicando na mesma subrede através de um *hub* de 10 Mbit/s, e posteriormente através de um *switch* de 100 Mbit/s. Colocou-se uma, duas e quatro sessões FTP concorrentes a fim de verificar as diferenças nos resultados.

O primeiro gráfico da figura 12 mostra o resultado para uma banda de 100 kbit/s, com e sem tráfego concorrente. Pode-se ver que o sistema detectou o resultado correto com uma boa precisão na maior parte das amostras, havendo certos erros devido ao software de roteamento (*ipfw*) e ao tráfego concorrente. As bandas de 500kbit/s e 1Mbit/s produziram resultados semelhantes.

O segundo gráfico da figura 12 mostra o resultado para uma banda de 10 Mbit/s, com o transmissor, o receptor e as sessões FTP no mesmo *hub*. A qualidade das medições aumentou, principalmente devido à eliminação do roteamento *ipfw*.

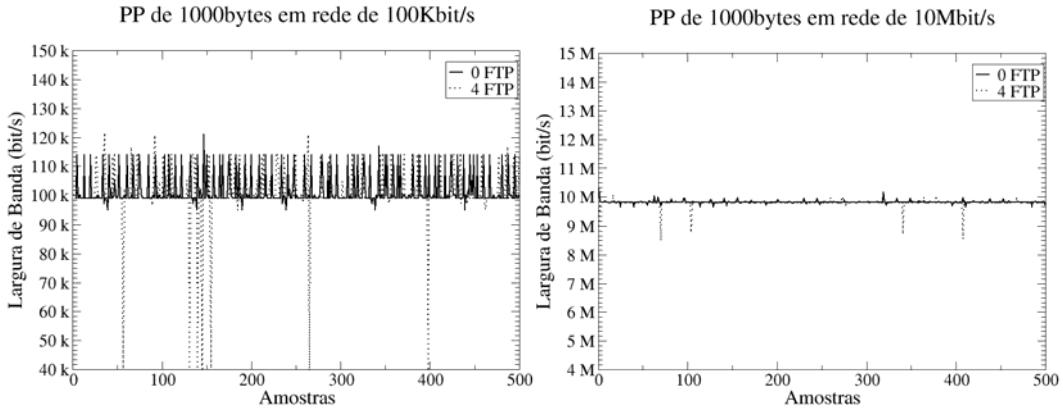

Figura 12. Ambiente controlado, 100 kbit/s e 10 Mbit/s.

O mesmo teste foi repetido para um switch de 100Mbit/s, com vários tamanhos de pacote. Para pacotes de 1000 bytes a média ficou em 101 Mbit/s, com um máximo de 160 e mínimo de 72 Mbit/s. O resultado foi muito similar com tráfego FTP concorrente.

Na Internet, foram realizados diversos testes, como, por exemplo, o mostrado na figura 11, onde o sinal passava pela intranet da Unisinos, provedor comercial, UFRGS e voltava, via Internet2, para o PRAV, na Unisinos. Além disso, testou-se entre duas máquinas na Intranet da Unisinos (gargalo de 10 Mbit/s), obtendo-se um resultado bastante estável, com média de 10,8 Mbit/s. Outro teste realizado foi entre a Unisinos e uma residência com ADSL (banda de saída de 160 kbit/s e entrada de 320kbit/s). A

figura 13 mostra o resultado obtido para pacotes de 1000 bytes. Como pode ser visto, o tráfego medido ficou um pouco abaixo do valor oficial configurado no modem, entretanto, bastante estável.

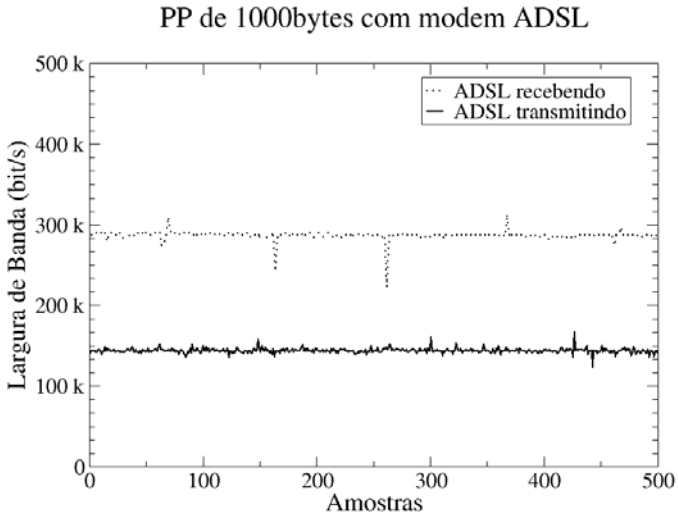

Figura 13. Conexão Internet via ADSL.

6 Conclusões e trabalhos futuros

O presente artigo apresentou uma análise detalhada do método de pares de pacotes sobre UDP, que permite a inferência de banda no receptor sem necessidade de enviar ACKs ao transmissor. Além disso, os pares de pacotes eram enviados aproximadamente a cada 0,5s, não gerando rajadas nem excessivo tráfego na rede. Um dos objetivos foi a análise da viabilidade de integração no SAM, que utiliza o próprio conteúdo multimídia em pares de pacotes, e os receptores não enviam ACKs, não gerando tráfego adicional. Através das simulações e implementação, concluiu-se que é viável a utilização do método para que o receptor descubra a banda máxima, mas não a banda eqüitativa. Além disso, o método pode ser útil para inferir a banda máxima fim-a-fim de quaisquer novos mecanismos de controle de congestionamento, principalmente os baseados em multicast.

Para analisar o método, utilizou-se simulação e também uma implementação real em linguagem ‘C’. Através do ambiente de simulação, verificou-se que o método de pares de pacotes funciona, porém, algumas vezes infere banda acima da verdadeira, e algumas vezes abaixo. O motivo dessas inferências errôneas foi verificado através das simulações, e acontecem devido à diminuição do espaçamento dos pacotes nos roteadores de maior velocidade localizados após o gargalo (infere banda maior), ou entrada de tráfego concorrente no meio do par de pacotes (infere banda menor). Entretanto, mesmo com esses problemas, na média, o método funciona adequadamente.

Tanto as simulações como os resultados da implementação mostraram que o método é menos estável com pacotes pequenos, gerando variações bruscas de inferência. Esse fato poderia ser utilizado por outros algoritmos de inferência de banda baseados em pares de pacotes existentes na literatura, que enviam rajadas de pacotes pequenos e grandes. De acordo com as medidas efetuadas, não vale a pena utilizar pacotes de tamanho pequeno. O motivo é que o pacote pequeno ocupa menos tempo na

rede que o pacote grande, e uma pequena variação no espaçamento gera um grande impacto.

Uma alternativa para melhorar a estabilidade e a precisão da medida, com qualquer tamanho de pacote, é utilizar uma rajada maior (trem de pacotes). Nos testes, foram testadas rajadas de 8, 16 e 32 pacotes. Com 8 pacotes já se percebe uma diferença grande nos resultados, como pode ser visto na figura 11.

Alguns trabalhos da literatura afirmam que a utilização de trem de pacotes permite a inferência da banda eqüitativa da rede, e não a banda máxima, porém, os resultados obtidos mostraram que a banda obtida é a banda máxima.

Como continuidade dos trabalhos, pretende-se criar um algoritmo para filtrar medições errôneas, permitindo o cálculo adaptativo da banda, à medida que os pares de pacotes chegam a intervalos regulares.

Referências

Braden, R. “Requirements for Internet Hosts -- Communication Layers”. RFC 1122: California: IETF. October 1989.

Carter, R., Crovella, M. “Measuring Bottleneck link speed in packet switched networks”. March, 1996.

Croll, A.; Packman, E. “Managing Bandwidth – Deploying QoS in Enterprise Networks”. New Jersey: Prentice Hall. 2000.

Floyd, S. Fall, K. “Promoting the use of end-to-end congestion control in the Internet,” IEEE/ACM Transactions on Networking, v. 7, n. 4, Aug. 1999.

Gerla, M., Sanadidi, M.Y., Wang, R., Zanella, A. “TCP Westwood: Congestion Window Control Using Bandwidth Estimation”. In: Proceedings of IEEE Globecom 2001, v. 3, pp 1698-1702, San Antonio, Texas, USA, Nov. 25-29, 2001.

Huston, G. “Internet performance survival guide”. New York: John Wiley & Sons Inc, 2000.

Jacobson, V. “Congestion avoidance and control”. In Proceedings of SIGCOMM 1988.

Keshav, S. Morgan, S.P. “A Control-Theoretic Approach to Flow Control”. ACM SIGCOMM 91, Zurich, p.3-15, Set. 1991.

Lai, K. Baker, M. “Measuring link bandwidths using a deterministic model of packet delay”. In proceedings of ACM SIGCOMM 2000, August 2000.

Legout, A., Biersack, E. "PLM: Fast Convergence for Cumulative Layered Multicast Transmission Schemes". In Proceedings of ACM SIGMETRICS 2000, Santa Clara, California, June 2000.

Roesler, V. Bruno, G. Lima, V. “A new receiver adaptation method for congestion control in layered multicast transmissions”. In: ICT’2003. Proceedings… Tahiti, French Polynesia: IEEE, February 2003.

Spurgeon, C. E. “Ethernet, the definitive guide”. Sebastopol: O’reilly. February, 2000. 500p.