ALMTF: um mecanismo TCP-friendly para controle de congestionamento em transmissões multimídia

Valter Roesler[1,2], João Marcelo Ceron[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, jmarcelo}@exatas.unisinos.br, valdeni@inf.ufrgs.br

Abstract. *The main objective of this paper is to present thoroughly ALMTF (Adaptive Layered Multicast TCP Friendly), which is one of the congestion control algorithms in use by SAM (Multimedia Adaptive System). The algorithm of ALMTF infers the network congestion level, defining the rate (or number of layers) which the receiver can afford, in a way that the amount of received bandwidth is friendly with other TCP flows, as well as fair with other flows of its own algorithm. The benefit obtained with this system is the possibility of transmission of shows and TV channels through Internet, where each receiver adapts automatically to the best possible quality at that moment.*

Keywords: congestion control, TCP-Friendly, multicast.

Resumo. *O principal objetivo deste artigo é apresentar em detalhes o ALMTF (Adaptive Layered Multicast TCP Friendly), que é um dos algoritmos de controle de congestionamento utilizados pelo SAM (Sistema Adaptativo para Multimídia). O algoritmo ALMTF infere o nível de congestionamento existente na rede, determinando a taxa (ou número de camadas) que o receptor pode suportar, de forma que a quantidade de banda recebida gere um tráfego na rede que seja eqüitativo com outros tráfegos ALMTF concorrentes, ou outros tráfegos TCP concorrentes. O benefício obtido com o sistema é a possibilidade de transmissão de shows e canais de TV via Internet, onde cada receptor se adapta automaticamente na melhor qualidade possível naquele momento.*

Palavras-chave: controle de congestionamento, TCP-Friendly, multicast.

1 Introdução

A utilização de transmissões multimídia em redes de computadores está se tornando comum nos últimos anos, pois som e vídeo se integram harmoniosamente às páginas estáticas de texto e imagens existentes na Internet, acrescentando dinamismo e tendo um papel importante para a assimilação das informações existentes. Segundo Bo Li [Li 2003], a distribuição de vídeo em tempo real está emergindo como uma das aplicações mais importantes em IP multicast, sendo um componente essencial de muitas necessidades atuais, como videoconferência e ensino a distância.

É nesse contexto que se insere o presente trabalho, que mostra um método de controle de congestionamento para transmissões multicast em camadas denominado ALMTF (*Adaptive Layered Multicast TCP Friendly*), e consiste de um bloco de um sistema maior, denominado 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 do ALMTF 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).

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

Quando não existe uma rede privativa virtual, as transmissões vão estar concorrendo com tráfego existente na Internet, que é composto 95% de TCP [Thompson 97], portanto, é importante que o ALMTF gere tráfego eqüitativo com TCP, mesmo que o TCP não seja eqüitativo com seu próprio tráfego, pois sua taxa de subida depende do RTT.

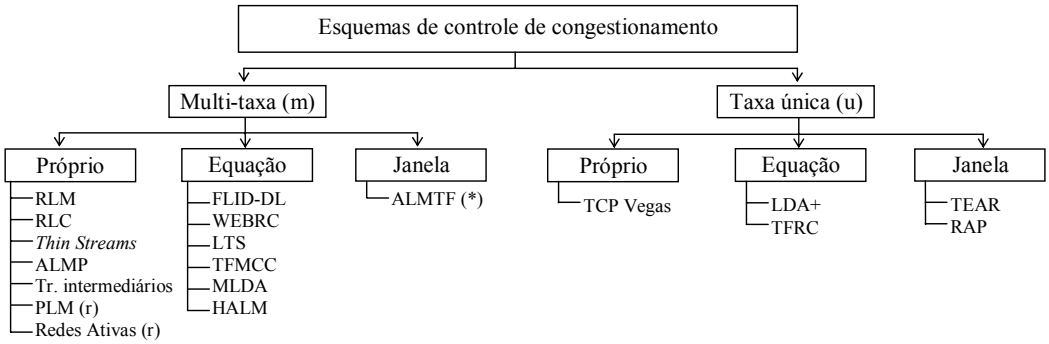
Este artigo tem por objetivo analisar o algoritmo ALMTF, detalhando a metodologia utilizada para que o mesmo tenha uma característica semelhante ao tráfego TCP, porém com maior estabilidade. Este artigo está dividido da seguinte forma: a sessão 2 mostra os trabalhos relacionados. A sessão 3 detalha o algoritmo de controle de congestionamento ALMTF. Na sessão 4, o ambiente de simulação é descrito, bem como os resultados atingidos e comparações efetuadas com outros algoritmos similares. A sessão 5 apresenta as conclusões deste artigo.

2 Trabalhos Relacionados

É importante a classificação dos algoritmos de controle de congestionamento em relação ao método que eles utilizam para conseguir eqüidade com o tráfego TCP, portanto, os algoritmos foram divididos da seguinte forma: a) multi-taxa e taxa única; b) mecanismo de adaptação de tráfego próprio, utilizando a equação do TCP ou através de adaptação por janela. Essas categorias foram privilegiadas, conforme ilustrado na figura 2. Também foi considerada importante a rápida visualização dos mecanismos que efetuam comunicação fim-a-fim ou exigem modificações nos roteadores intermediários, pois os últimos não podem ser utilizados na Internet atual, e foram marcados com a legenda (r). Todos os protocolos multi-taxa analisados são baseados em multicast e todos de taxa única são baseados em unicast.

O protocolo "transcodificadores intermediários" foi classificado como fim-a-fim, entretanto, necessita de agentes rodando dentro de domínios específicos para seu funcionamento. Ele foi considerado multi-taxa, pois utiliza vários grupos multicast, porém, cada receptor se cadastra em um único grupo multicast. Detalhes de cada um dos algoritmos na figura 2 podem ser obtidos em http://www.inf.unisinos.br/~roesler/tese_/tese.pdf. Eles não serão detalhados aqui por

motivos de espaço.



(m)=multicast; (u)=unicast; (r)=apoiado pelo roteador.
(*) O ALMTF utiliza principalmente janela, mas também utiliza a equação do TCP

Figura 2. Classificação de esquemas de controle de congestionamento

3 Detalhamento do algoritmo ALMTF

Para a implementação do algoritmo ALMTF, criou-se dois novos agentes no simulador NS2[1], em linguagem C++: o agente transmissor (*ALMTF Agent*) e o agente receptor (*ALMTFSink Agent*), além do código que implementa a lógica do algoritmo em si, feito na linguagem *tcl*. O agente transmissor gera os pacotes em multicast nas taxas especificadas em cada camada. O agente receptor recebe os pacotes, calcula o RTT, o número de perdas e outras informações, atualizando algumas variáveis do código em *tcl*, que são utilizadas pelo algoritmo.

O cabeçalho dos pacotes ALMTF tem os campos mostrados na figura 3, com um total de 16 bytes, sendo utilizado para cálculo de erros, cálculo de RTT, cálculo de intervalo entre mensagens de *feedback* e sincronismo para subir camadas.

32 bits

Numseq	Num_nodereceptor / Num_receptores	
Timestamp		
Timestamp_echo		
Timestamp_offset	sincjoin	Flags

Figura 3 – Cabeçalho dos pacotes ALMTF

O significado de cada campo é descrito brevemente a seguir, e ficará mais claro ao longo do artigo: ***Numseq***: número de seqüência do pacote; ***Num_nodereceptor***: identificação do receptor, utilizado para cálculo de RTT; ***Num_receptores***: número de receptores atuais; ***Timestamp***: instante de tempo que foi transmitido o pacote; ***Timestamp_echo***: cópia do instante de tempo no qual o transmissor recebeu um pacote de "Solicitação de Eco; ***Timestamp_offset***: diferença de tempo entre o transmissor receber uma "Solicitação de Eco" e enviar a resposta; ***Sincjoin***: aviso de sincronismo de *join*; ***Flags***: as seguintes *flags* foram definidas: a) ***FL_INIRAJADA (10000000)***: indicador de inicio de rajada; b) ***FL_MEDERTT (01000000)***: indica que este pacote é para medição de RTT.

[1]:http://www.isi.edu/ nsnam/ns/. Acesso em: nov. 2003.

A lógica do algoritmo ALMTF foi implementada em linguagem *tcl*, interagindo com o agente transmissor e receptor quando necessário. As principais variáveis do algoritmo são: a) ***bwshare***: banda máxima que o receptor pode utilizar inferida pelo mecanismo de janelas. Seu uso será detalhado no item 3.1; b) ***banda_eqn*:** banda máxima que o receptor pode utilizar inferida através da equação do TCP. Esse valor é utilizado somente como apoio à decisão, pois o *bwshare* é a variável principal na qual o ALMTF se baseia. Seu uso será detalhado no item 3.2. c) ***RTT (Round Trip Time)***: tempo entre duas execuções consecutivas do algoritmo ALMTF, sendo utilizado também para cálculo da taxa da equação do TCP. d) ***bwmax***: atualizado pelo receptor quando se utiliza transmissão com pares de pacotes. O valor máximo de *bwshare* e *banda_eqn* é determinado por essa variável, que reflete a banda máxima da rede.

Para controle de fluxo, o ALMTF utiliza uma combinação de dois métodos para inferência de banda eqüitativa: por janela (J – atualizando variável *bwshare*), e por equação (E – atualizando variável *banda_eqn*), conforme mostra a figura 4. Na figura, a linha contínua fina representa a taxa descoberta pelo mecanismo de janela, e a linha pontilhada a taxa descoberta pelo mecanismo de equação. Existem duas camadas, C1 e C2, e o receptor se baseia nos dois mecanismos para tomar uma decisão. A linha contínua cheia representa a camada efetivamente na qual o receptor efetuou *join*. As seguintes considerações podem ser feitas:

- **Se J = E ou J > E**: utiliza a taxa descoberta pelo mecanismo de janela, que é o principal do ALMTF. Entretanto, a fim de evitar diferenças muito grandes, limita *bwshare* em, no máximo, duas vezes *banda_eqn*, conforme mostra o bloco "a" na figura 4, fazendo com que a taxa de subida seja mais suave;
- **Se J < E**: Através de simulações, chegou-se à conclusão de que o mecanismo de janela diminui banda de forma muito rápida, podendo causar instabilidades. Assim, utiliza-se uma proteção que, se a variável *bwshare* descer abaixo do limite de determinada camada (C1 + C2 na figura 4), o sistema avalia se a banda obtida pela equação está acima ou abaixo deste limite. Se estiver acima, como no bloco "b" da figura, o algoritmo não desce de camada protegido pela equação.

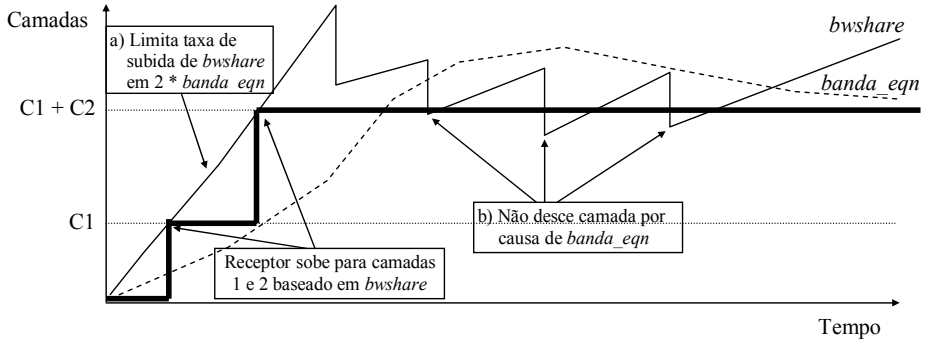


Figura 4. Controle de fluxo do ALMTF (janela e equação)

Os principais mecanismos do algoritmo ALMTF são detalhados a seguir.

3.1 Mecanismo de controle de congestionamento baseado em janela

O mecanismo de controle de congestionamento por janela é o principal no ALMTF, portanto, neste item será feita também uma análise geral do algoritmo. Para se criar um mecanismo de controle de congestionamento por janela semelhante ao TCP, utilizou-se

uma variável denominada *cwnd*, que representa o tamanho da janela de congestionamento atual do ALMTF, em pacotes. Essa variável está relacionada diretamente à quantidade de banda permitida para o receptor se cadastrar (variável *bwshare*). A relação entre *cwnd* e *bwshare* é dada pela função *win2rate()*, que efetua basicamente o seguinte cálculo:

$$bwshare = \frac{cwnd * packetsize * 8}{RTT}$$

Esse cálculo representa a banda utilizada em uma conexão TCP equivalente para um determinado tamanho de janela. Por exemplo, se o RTT é 100 ms e o tamanho de pacote é 500 bytes, tendo *cwnd* igual a 2 pacotes, a banda é igual a 80 kbit/s.

O algoritmo de controle de congestionamento do ALMTF é executado a cada RTT (cujo cálculo será explicado no item 3.3). Dessa forma, se o RTT for maior, o algoritmo é executado menos vezes, refletindo o que acontece com uma conexão TCP.

3.2 Controle de congestionamento baseado na equação do TCP

O controle de congestionamento pela equação do TCP utilizado no ALMTF é semelhante ao utilizado no TFMCC [Widmer 2001b] [Padhye 98]. A cada pacote recebido, o ALMTF chama a função *estimabanda()*, que retorna o valor estimado através da equação. A fatia média de banda utilizada por um fluxo TCP é dada por:

$$B(p) \approx \frac{packetsize}{RTT\sqrt{\frac{2p}{3}} + \left(1{,}3\sqrt{\frac{3p}{8}}\right) * 4 * RTT * p\left(1 + 32p^2\right)}$$

Como pode-se verificar a partir da equação, é necessário saber o tamanho do pacote, o RTT e as perdas (variável *p*). O ALMTF utiliza um tamanho de pacote fixo, definido no arquivo de simulação. O cálculo de RTT será descrito no item 3.3, e o cálculo de perdas será descrito a seguir.

Para o cálculo de perdas (variável *p*), o ALMTF considera os oito últimos eventos de perdas, que são calculados conforme ilustra a figura 5.

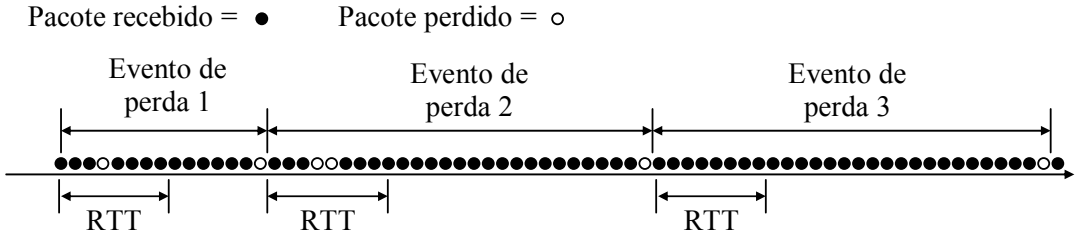


Figura 5 – Criação do vetor "evento de perda" para cálculo da variável *p*

Quando acontece uma perda, o algoritmo verifica se já passou um RTT, se passou, inicia um novo evento de perda, caso contrário, mantém o mesmo evento. Na figura, pode-se ver que durante o "evento de perda 1" ocorreram duas perdas e chegaram 13 pacotes (visto através do número de círculos preenchidos, que representam os pacotes recebidos), durante o "evento de perda 2" ocorreram três perdas e chegaram 24 pacotes, e assim por diante. Através desse método, é gerado um vetor de oito entradas contendo

o número de pacotes recebidos em cada evento de perdas, sendo que o valor inicial do vetor refere-se ao evento mais recente. No caso da figura, o vetor seria: *vet_perdas* = [27; 24; 13; 0; 0; 0; 0; 0].

No vetor de perdas, é dado um maior peso para os eventos mais recentes. O filtro utilizado é o mesmo definido em [Rhee 2000], [Widmer 2001] e [Floyd 99], que se baseiam nos 8 tempos anteriores, definindo o vetor de pesos [5; 5; 5; 5; 4; 3; 2; 1], ou, como é utilizado no ALMTF, *vet_pesos* = [1,0; 1,0; 1,0; 1,0; 0,8; 0,6; 0,4; 0,2]. O valor de *p* é o inverso da média do valor dos eventos de perdas, conforme mostra o código abaixo, sendo que "*soma*" é a soma do vetor de pesos, ou 6,0.

```
media = 0
  for (i = 0, i < 8, i++)
    media = media + vet_perdas(i)*vet_pesos(i)/soma
p = 1/media
```

3.3 Cálculo do RTT

O RTT é um dos elementos que gera o maior impacto no protocolo TCP, pois determina o aumento na janela do TCP (*cwnd*), que está diretamente ligada ao aumento da sua taxa de transmissão. Isso faz com que um algoritmo que queira gerar tráfego eqüitativo ao TCP deva se basear nessa variável, porém, em multicast e sem mensagens de *feedback* constantes, o cálculo é mais complexo que no TCP, conforme descrito a seguir.

O cálculo do RTT no ALMTF é feito no receptor, sendo ilustrado na figura 6. O transmissor envia três pacotes de dados (Dados 1 a Dados 3) contendo seu *timestamp*, representado por TT1 a TT3. Na chegada do primeiro pacote, o receptor envia um pacote de "Solicitação de Eco", contendo no cabeçalho seu *timestamp* (representado por TR1) e sua identificação, (*numrec*). Quando o transmissor recebe a solicitação de eco, ele armazena seu tempo de chegada, e calcula a diferença de tempo entre a chegada da solicitação e a transmissão do novo pacote de dados (representado por ΔT). No próximo pacote de dados, o transmissor envia, além do seu *timestamp*, o tempo do receptor (TR1), a identificação do receptor e ΔT, conforme visto no pacote "Dados 2" da figura.

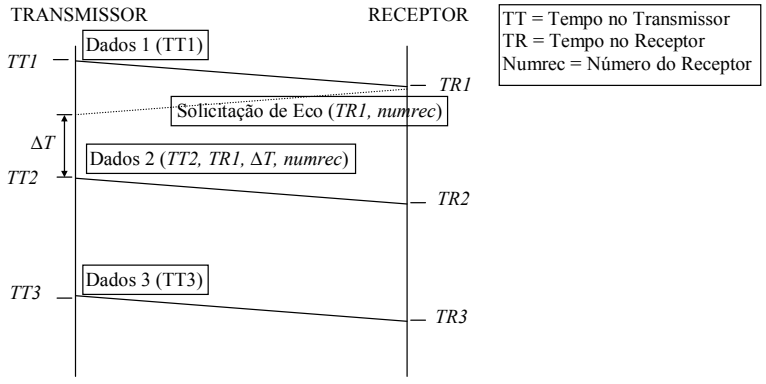


Figura 6. Cálculo do RTT do ALMTF no receptor

O cálculo do RTT é composto de duas fases: a) quando o receptor recebe uma resposta à sua solicitação de eco (como no instante TR2 da figura 6), e calcula o RTT com precisão, bem como a diferença entre os relógios do transmissor e do receptor; b) quando o receptor recebe um pacote de dados somente com *timestamp* do transmissor (como nos instantes TR1 e TR3 da figura 6), e calcula a variação do RTT. Além do cálculo descrito acima, foi implementado um filtro para aumentar a estabilidade das

medições de RTT, evitando variações bruscas.

3.4 Intervalo entre feedbacks

O receptor deve enviar *feedbacks* (pacotes de "solicitação de eco") ao transmissor a intervalos regulares, a fim de calcular com precisão o tempo de ida (T_{ida}) do pacote, conforme visto no item 3.3, entretanto, o número de pacotes não deve ultrapassar um determinado limite, pois caso contrário provocaria uma implosão de *feedbacks* quando houvesse um grande número de receptores cadastrados na sessão. No ALMTF, definiu-se que, até 60 receptores, o total de mensagens deve estar limitada a uma por segundo, assim, se existir um receptor, o mesmo vai enviar uma mensagem por segundo. Se existirem 60 receptores, cada um deles pode enviar uma mensagem a cada 60 segundos. Acima de 60 receptores, mantém-se uma atualização por minuto, a fim de evitar uma espera muito grande para cálculo do RTT com precisão. Com 6.000 receptores, tem-se uma mensagem a cada 10ms, ou 100 pacotes por segundo. Se cada pacote de *feedback* possuir 64 bytes (512 bits), tem-se um tráfego adicional de 51,2 kbit/s para 6.000 receptores. No início dos tempos, ou seja, quando o receptor recebe o primeiro pacote, também é enviada uma mensagem de "solicitação de eco", e o receptor calcula o RTT.

3.5 Cálculo de perdas

No ALMTF, a detecção das perdas é feita com base no número de seqüência existente em cada pacote transmitido (semelhante ao protocolo RTP). Caso o receptor detecte em qualquer camada uma falha na seqüência recebida, ele incrementa a variável *numloss*, que reflete a quantidade de perdas durante determinado tempo. Na verdade, foi utilizado o mesmo conceito definido em [Rhee 2000], [Widmer 2001] e [Floyd 99], onde várias perdas durante o mesmo RTT são consideradas como uma única ocorrência de perdas.

3.6 Inferência de banda máxima via pares de pacotes

O algoritmo ALMTF utiliza pares de pacotes para determinar a banda máxima na qual o receptor pode se inscrever. A descrição do método de pares de pacotes encontra-se em [Roesler 2003b]. Para evitar variações bruscas, criou-se um filtro para obter um valor com maior probabilidade de estar correto. A descrição e os resultados do filtro encontram-se em http://www.inf.unisinos.br/~roesler/tese/8_ParesPacotes/.

3.7 Sincronização entre receptores

Existe a necessidade de sincronização entre receptores da mesma sessão, caso contrário, haveria uma diminuição na estabilidade do sistema, devido às tentativas dessincronizadas de *join* por parte dos receptores. O ALMTF utiliza pontos de sincronização, como mostra a figura 7, que representa o valor de *sincjoin* numa transmissão com 5 camadas (C1 a C5). Os círculos na figura representam o limite de camada no qual o receptor pode se inscrever, que também é representado pela variável *sincjoin* (transmitida pelo transmissor no cabeçalho do pacote). No primeiro intervalo de tempo (*sincjoin = 2*) todos os receptores cadastrados na camada C1 podem efetuar tentativas de *join* para a camada C2, entretanto, um receptor cadastrado na camada C2 não poderá se inscrever na camada C3. No próximo intervalo de tempo (*sincjoin = 3*), os receptores podem se cadastrar até a camada C3, e assim por diante. Se existissem mais camadas, o número seria dividido por quatro, a fim de seguir a distribuição

mostrada na figura. Por uma característica do algoritmo, todos receptores estão sempre cadastrados na camada C1.

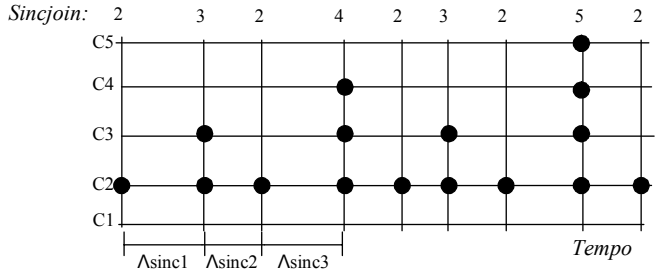


Figura 7. Pontos de sincronização no ALMTF

Como pode ser visto através da figura 7, os pontos de sincronização acontecem mais freqüentemente nas camadas inferiores, dando maior chance aos receptores com menos banda subirem de camada e aumentando a eqüidade de tráfego.

4 Simulações efetuadas

As simulações foram divididas buscando analisar o algoritmo em relação a: adaptabilidade, escalabilidade, estabilidade e eqüidade de tráfego. Efetuou-se uma comparação com os seguintes algoritmos relacionados.

- RLM (*Receiver-Driven Layered Multicast* – [McCanne 96]), utiliza transmissão em camadas com multicast e adaptação no receptor, ou seja, possui a mesma filosofia do ALMTF. Além disso, é o algoritmo que serve como base de comparação da maioria dos outros algoritmos, por ter sido o primeiro desenvolvido para esse fim;
- ALMP (*Adpative Layered Multicast for Private Networks* – [Roesler 2003]): tem o mesmo objetivo e utiliza métodos semelhantes ao ALMTF e ao RLM, porém voltado para redes privativas, sem concorrência com o TCP.
- TFMCC (*TCP-Friendly Multicast Congestion Control* – [Widmer 2001b]): utiliza a equação do TCP como método de adaptação. O algoritmo TFMCC não trabalha com transmissão em camadas, se adaptando ao receptor mais lento entre todos os receptores que estão assistindo a transmissão, entretanto, foi comparado por utilizar multicast baseado na equação do TCP. Além disso, as simulações de eqüidade são feitas com um transmissor e um receptor, tornando o algoritmo TFMCC comparável ao ALM, pois quando existe somente um receptor, este é o mais lento.

4.1 Adaptação em ambientes heterogêneos, estabilidade e escalabilidade

Para comparar os algoritmos em termos de adaptação em ambientes heterogêneos, estabilidade e escalabilidade, foi utilizada como base a topologia vista na figura 8, onde BR significa *Bloco de Receptores*.

Os parâmetros configurados foram: r = 25, totalizando 100 receptores; camadas exponenciais (30 kbit/s, 60 kbit/s, 120 kbit/s, 240 kbit/s, 480 kbit/s e 960 kbit/s) – esse dado não se aplica no TFMCC; tamanho do pacote = 500 bytes. Os algoritmos ALMP e ALMTF foram simulados sem o uso de pares de pacotes, para permitir uma melhor comparação com os outros algoritmos. A figura 9 mostra os resultados das simulações para todos os algoritmos comparados.

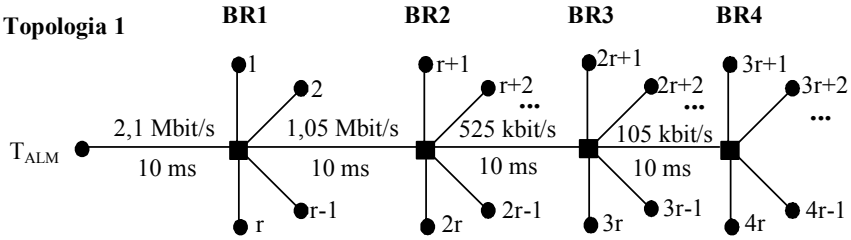
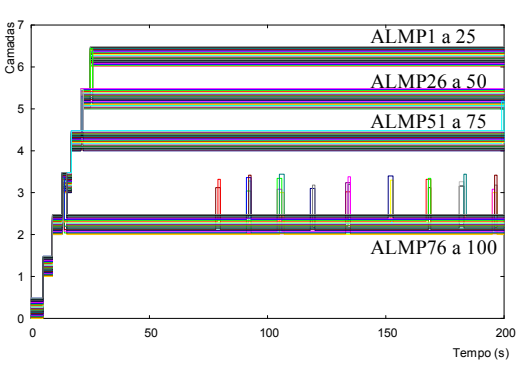
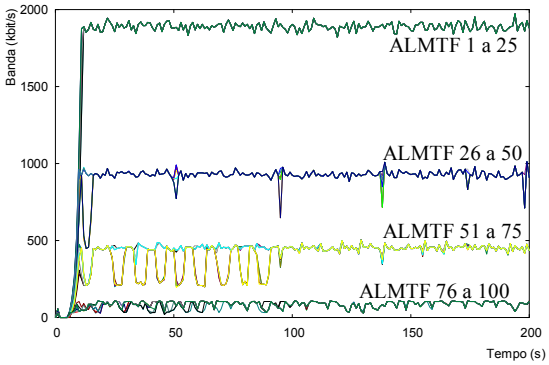


Figura 8. Topologia para simulações de adaptação e escalabilidade

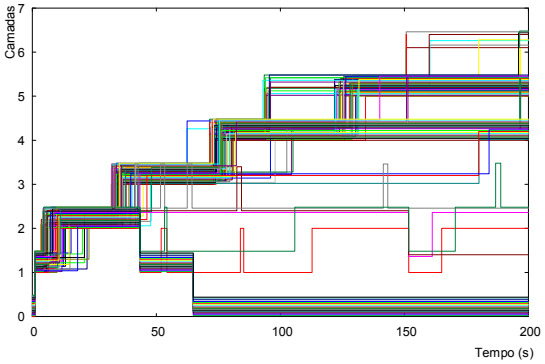
A partir da análise da figura 9, percebe-se que o ALMP (gráfico "a") e o ALMTF (gráfico "b") adaptaram-se ao correto número de camadas, permitindo que receptores localizados em enlaces de diferentes larguras de banda obtenham uma qualidade compatível com a sua possibilidade de tráfego.



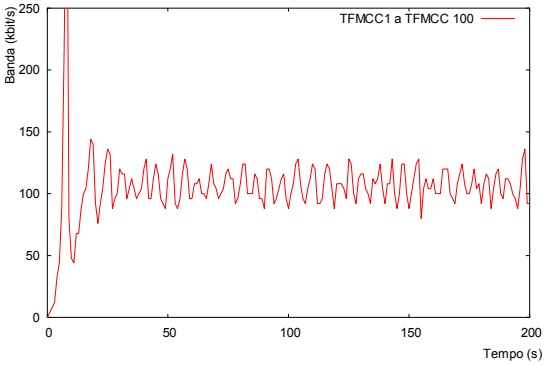
a) algoritmo ALMP



b) algoritmo ALMTF



c) algoritmo RLM



d) algoritmo TFMCC

Figura 9. Comparação de escalabilidade para 100 receptores

O ALMP (gráfico "a") obteve a melhor escalabilidade entre todos os algoritmos analisados, com poucas variações de camada. Pode-se observar pelo gráfico que todos os receptores adaptaram-se ao correto número de camadas. Os receptores ALMP1 a ALMP25 adaptaram-se na camada 6 (demandando 1.890 kbit/s, ou seja, taxas da camada 1 à camada 6, que equivale a 30 kbit/s + 60 kbit/s + 120 kbit/s + 240 kbit/s + 480 kbit/s + 960 kbit/s), sendo compatível com o enlace de 2,1 Mbit/s. Os receptores ALMP26 a ALMP50 adaptaram-se na camada 5 (total de 930 kbit/s), que é compatível com o enlace de 1,05 Mbit/s. Os receptores ALMP51 a ALMP75 adaptaram-se na camada 4 (taxa de 450 kbit/s), adequado para o enlace de 525 kbit/s, e os receptores ALMP76 a ALMP100 adaptaram-se na camada 2 (taxa de 90 kbit/s), coerente com o

enlace de 100 kbit/s.

O ALMTF (gráfico “b”), é mais agressivo, fazendo com que o número de tentativas de subir de camada seja superior ao ALMP. Assim, para facilitar a visualização, utilizou-se o gráfico por fluxo ao invés de por camadas. Como pode-se observar, todos os receptores (ALMTF1 a ALMTF100) adaptaram-se ao correto número de camadas, recebendo uma característica de tráfego muito semelhante entre si.

O RLM (gráfico “c”) mostrou certo problema na escalabilidade, pois alguns receptores não conseguiram adaptar-se e abandonaram a sessão aproximadamente no instante 60s. O TFMCC (gráfico “d”) suporta uma grande escalabilidade, porém, dentro do seu modelo, que não possui adaptabilidade em ambientes heterogêneos, ou seja, utiliza a taxa de transmissão do receptor com a menor banda. O transmissor do TFMCC transmitiu um único fluxo multicast de aproximadamente 100 kbit/s, compatível com a largura de banda disponível no enlace mais lento da topologia. Esse fluxo multicast foi destinado a todos os 100 receptores.

Em termos de estabilidade, pode-se observar que o ALMP, o RLM e o TFMCC tiveram uma ótima estabilidade. O ALMTF necessita ser mais agressivo devido à sua característica de compartilhamento de banda com o TCP. A figura 10 mostra a mesma simulação para o ALMTF, porém com a utilização de pares de pacotes. Nesse caso, como os pares de pacotes permitem a cada receptor o cálculo da banda máxima disponível na rede, praticamente não existem tentativas de subir camada, o número de perdas é zero e a adaptabilidade é consistente com a banda disponível.

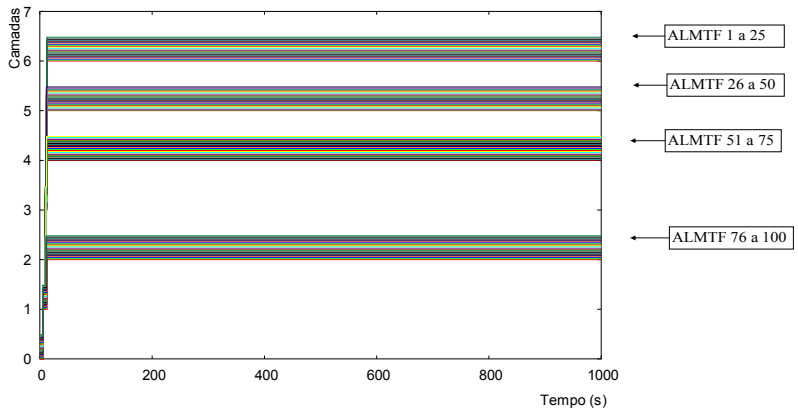


Figura 10 – ALMTF com 100 receptores utilizando pares de pacotes

4.2 Análise de eqüidade com o próprio tráfego e com TCP concorrente

As simulações de análise de eqüidade de tráfego avaliam o comportamento do algoritmo com tráfego concorrente, verificando se o mesmo tem capacidade para compartilhar tráfego igualmente com outros fluxos de mesma sessão e tráfego TCP concorrente.

A topologia simulada é ilustrada pela figura 11. Na topologia, existe um número *m* de fluxos do algoritmo em questão e um número *n* de fluxos TCP concorrendo através de um enlace único. A banda B no enlace central permite aproximadamente 500 kbit/s por receptor, de forma a acomodar 4 camadas exponenciais para cada sessão do algoritmo em camadas (que demandam um total de 450 kbit/s: 30+60+120+240).

Utilizou-se camadas exponenciais (30 kbit/s, 60 kbit/s, 120 kbit/s, 240 kbit/s, 480 kbit/s e 960 kbit/s), pacotes de 500 bytes e fila *droptail* com 20 pacotes.

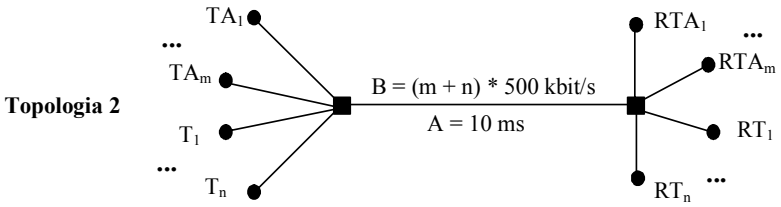


Figura 11. Topologia para simulações de eqüidade

As simulações mostradas na figura 12 serão utilizadas com o objetivo de comparar a eqüidade para dois fluxos do mesmo algoritmo iniciando em instantes diferentes e compartilhando um enlace de 1 Mbit/s. O primeiro fluxo inicia no instante zero, e o segundo no instante 100s.

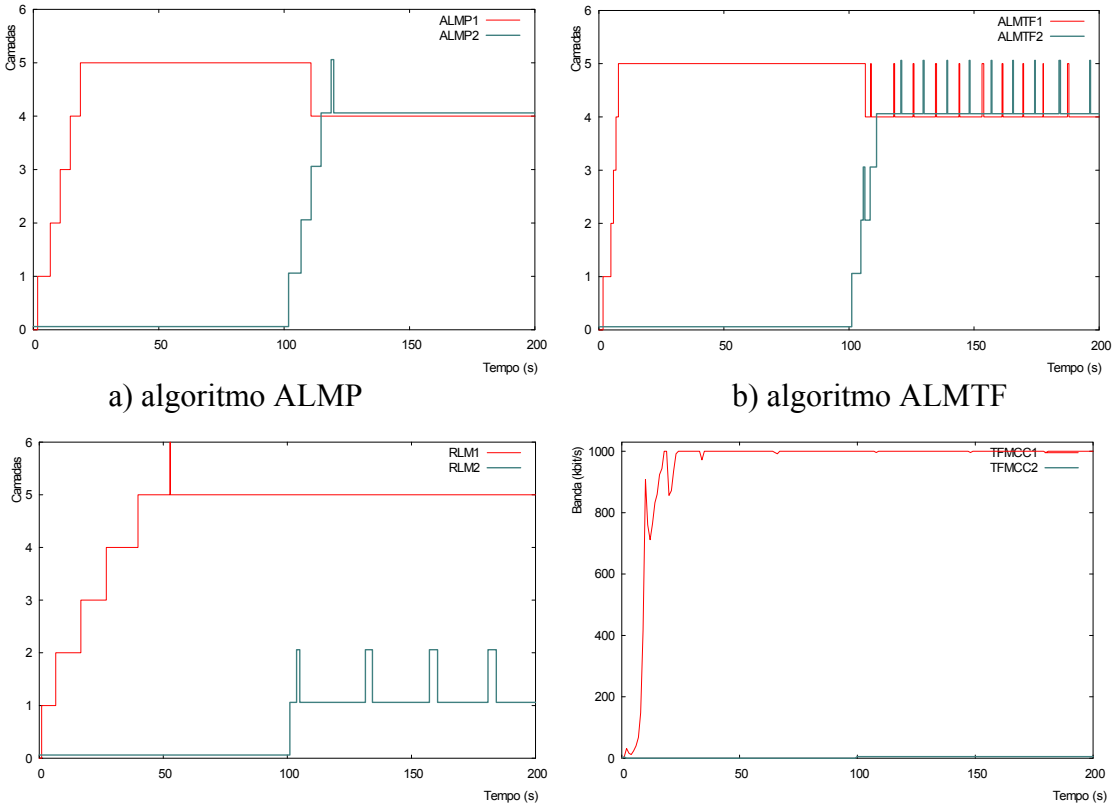


Figura 12. Eqüidade para dois fluxos do mesmo algoritmo

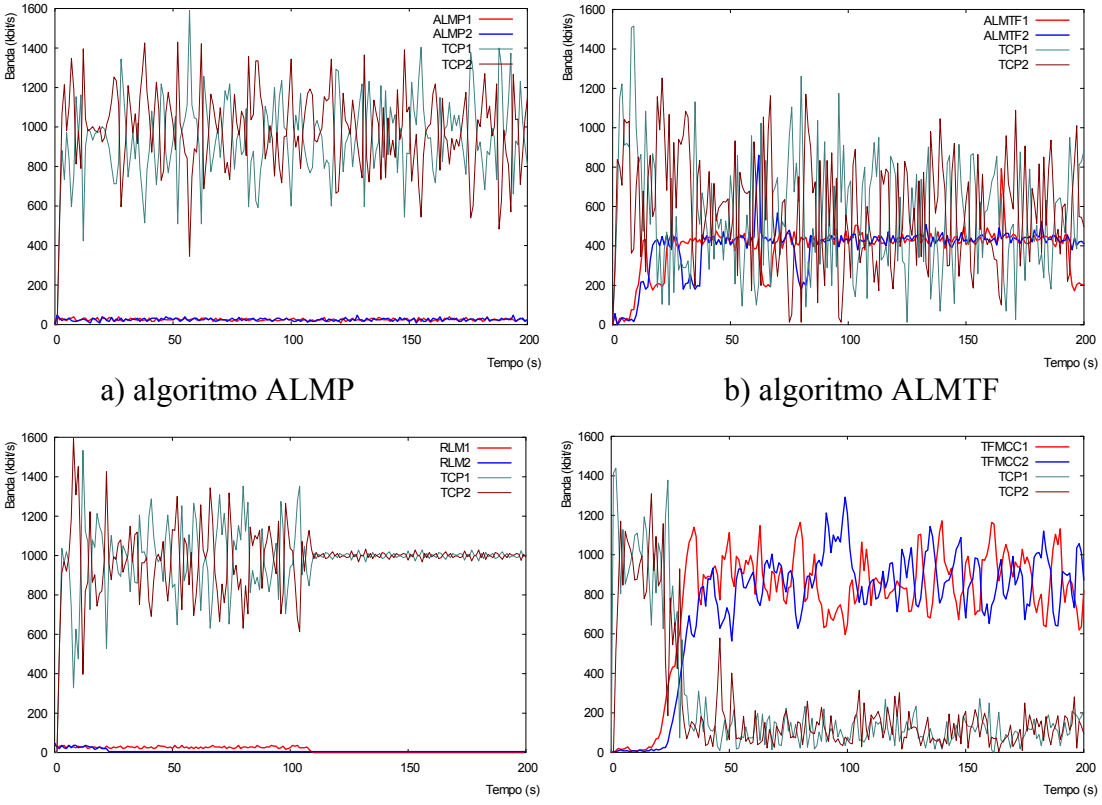
Em termos de eqüidade de tráfego, pode-se ver que o ALMP (gráfico "a") e o ALMTF (gráfico "b") adaptaram-se rapidamente à nova condição de tráfego após a entrada do segundo receptor. Além disso, após a adaptação, eles compartilharam tráfego de forma eqüitativa. O segundo fluxo do ALMP necessitou aproximadamente 25s para adaptar-se, enquanto que o segundo fluxo do ALMTF necessitou em torno de 10s.

O RLM (gráfico "c") não conseguiu adaptar-se, pois possui problemas de eqüidade com o próprio tráfego quando os fluxos iniciam em instantes diferentes. Como o receptor se baseia no tempo para efetuar nova tentativa de subir camada, ele é

extremamente conservador em termos de banda, dificultando a entrada de novos fluxos.

O algoritmo do TFMCC (gráfico “d”) utiliza a equação da banda no receptor para determinar a quantidade de tráfego que pode ser transmitida, entretanto, o uso da equação mostrou dificuldades para diminuir a banda quando entra outro fluxo semelhante no instante 100s, e não permitiu a entrada do segundo fluxo. Isso mostra que o uso da equação do TCP possui problemas de adaptabilidade em regime permanente. Outros testes foram realizados para detalhar o problema, e descobriu-se que, quando os fluxos iniciam com pouca diferença de tempo, eles conseguem adaptar-se corretamente.

Pode-se concluir que os algoritmos ALMP e ALMTF conseguem adaptar-se e compartilhar banda com outros tráfegos de mesmo tipo, mesmo quando eles iniciam em instantes diferentes de tempo. Entretanto, os algoritmos RLM, RLC e TFMCC possuem problemas de adaptação para fluxos que iniciam em instantes diferentes de tempo.



a) algoritmo ALMP

b) algoritmo ALMTF

c) algoritmo RLM

d) algoritmo TFMCC

Figura 13. Comparação com dois fluxos TCP concorrentes

A figura 13 mostra uma simulação de dois fluxos do algoritmo em questão concorrendo com dois fluxos TCP, num enlace central de 2 Mbit/s, ou seja, deixando idealmente a banda de 500 kbit/s por receptor.

O ALMP (gráfico “a”), não é adequado para competir com o TCP. Isso fica claro no gráfico, onde o tráfego ALMP ficou sempre na camada um, pois o TCP é muito mais agressivo. O ALMTF (gráfico “b”) adaptou-se corretamente. No ALMTF, a vazão ideal seria de 450 kbit/s (equivalente a 4 camadas), e a vazão ideal do TCP seria de 550 kbit/s (ocupando o restante da banda). Nesta simulação, o ALMTF utilizou aproximadamente 410 kbit/s, enquanto o TCP utilizou aproximadamente 510 kbit/s.

No caso do RLM (gráfico “c”), não aconteceu adaptação, pois os fluxos TCP tomaram conta da banda e os fluxos RLM não conseguiram adaptar-se, ficando inicialmente na primeira camada. Após certo tempo, os receptores abandonaram a sessão, e a banda ficou exclusivamente para o TCP.

O TFMCC (gráfico “d”), utilizou mais banda do que o ideal para uma divisão eqüitativa da banda, conforme pode ser visto no gráfico e na tabela. A banda eqüitativa deveria ser 500 kbit/s, e os fluxos TFMCC utilizaram aproximadamente 790 kbit/s, deixando o TCP com aproximadamente 200 kbit/s.

Pode-se concluir que o único algoritmo que se comportou de forma eqüitativa com o TCP foi o ALMTF. Todos os outros falharam em algum aspecto, ou transmitindo muito a mais que o TCP ou muito a menos que o TCP. Foram efetuadas diversas simulações para mostrar isso, além das incluídas neste artigo. A relação completa está acessível em http://www.inf.unisinos.br/~roesler/tese/4_Simulacoes.

5 Conclusões

A tabela 1 apresenta um resumo das comparações efetuadas, identificando os pontos fortes e fracos de cada algoritmo. Cada linha da tabela será analisada a seguir.

Na primeira linha da tabela 1 (Adaptação em ambientes heterogêneos) considerou-se que todos algoritmos adaptaram-se corretamente, entretanto, o TFMCC se adapta ao receptor mais lento, portanto, não suporta ambientes heterogêneos.

Na segunda linha da tabela, considerou-se que o RLM não suporta escalabilidade, pois muitos receptores não conseguiram entrar na sessão com 100 receptores, conforme ilustrado na figura 9.

A terceira linha da tabela analisa se o algoritmo permite adaptação com tráfego do mesmo tipo iniciando em instantes diferentes. Observa-se que somente os algoritmos ALMP e ALMTF conseguem adaptar-se.

A última linha da tabela analisa a eqüidade dos algoritmos. Para tanto, foi utilizado o conceito de *fairness index* (FI), que é a relação entre a taxa que o receptor está utilizando (TU) frente à taxa eqüitativa para este receptor (TE), ou seja, FI = TU / TE, e quanto mais próximo do valor “um”, mais eqüitativo é o fluxo. O índice FI foi feito para a simulação da figura 13, com dois fluxos TCP concorrendo com dois fluxos do algoritmo. Pode-se observar que o único algoritmo que conseguiu um índice de *fairness* próximo a “um” foi o ALMTF, que ficou com FI = 0,91. O TFMCC ficou com um índice FI = 1,57, significando que seus fluxos utilizaram 57% de banda a mais do que deveriam.

Em linhas gerais, pode-se concluir que o ALMP seria o algoritmo mais indicado para utilização em redes privativas (sem a presença do TCP concorrendo), devido à sua característica de adaptação em ambientes heterogêneos, aliada à sua alta estabilidade, escalabilidade e eqüidade com tráfego concorrente do seu próprio tipo, independente do mesmo iniciar em momentos diferentes. O uso do RLM fica comprometido principalmente devido à sua falta de habilidade em adaptar-se quando entram novos fluxos em instantes diferentes de tempo.

Caso seja necessário utilizar a transmissão em ambientes que necessitem concorrer com tráfego TCP, o mais indicado seria o ALMTF, principalmente devido à

sua capacidade de dividir banda de forma eqüitativa com tráfego TCP, não importando se o mesmo inicia antes ou depois. Além disso, o ALMTF mostrou possibilidade de adaptação em ambientes heterogêneos, tempo de adaptação rápido, alta escalabilidade e eqüidade com seu próprio tráfego. O uso do TFMCC é problemático pois não consegue adaptar-se quando entram novos fluxos em instantes diferentes de tempo.

Tabela 1. Resumo das comparações entre os algoritmos

Descrição	ALMP	ALMTF	RLM	TFMCC
Adaptação em ambientes heterogêneos	sim	sim	sim	não
Escalabilidade para 100 receptores	sim	sim	não	sim
Tráfego iniciando em instantes diferentes	sim	sim	não	não
FI com dois TCP concorrentes	0,05	0,91	0,03	1,57

6 Referências

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

Li, B. Liu, J. "Multirate video multicast over the Internet: an overview. **IEEE Network**, New York, v.17, n.1, p.24-29, Jan./Feb. 2003.

McCanne, S. Jacobson, V. Vetterli, M. "Receiver driven layered multicast". In: ACM SIGCOMM, 1996, Stanford, California, USA. **Proceedings…** New York: ACM, 1996. p.117-130.

Padhye, J. Firoiu, V. Towsley, D. Kurose, J. "Modeling TCP throughput: a simple model and its empirical validation". In: ACM SIGCOMM, 1998, Vancouver, Canada. **Proceedings…** New York: ACM, 1998.

Rhee, I. Ozdemir, V. Yi, Y. "**TEAR**: TCP emulation at receivers – flow control for multimedia streaming". 2000. Technical Report – NCSU.

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, fevereiro de 2003.

Roesler, V. Finzsch, P. Andrade, M. Lima, V. 2003b. "Análise do mecanismo de pares de pacotes visando estimar a band

a da rede via UDP". In: SBRC, 21., 2003, Natal, RN. **Anais…** Natal: UFRN, 2003.

Thompson, K. Miller, G. Wilder, R. "Wide-area Internet traffic patterns and characteristics". **IEEE Network**, New York, v.11. n.6, p.10-23, Nov. 1997.

Widmer, J. Denda, R. Mauve, M. "A survey on TCP-friendly congestion control". **IEEE Network**, New York, v.15, n.3, 2001.

Widmer, J. Handley, M. 2001b. "Extending equation-based congestion control to multicast applications". In: ACM SIGCOMM, 2001, San Diego, California, USA. **Proceedings…** New York: ACM, 2001.