USO DE TÉCNICAS CLÁSSICAS, HMM E REDES NEURAIS NA PREDIÇÃO DE ATRASO DE EXECUÇÃO EM REDES VOIP: UMA AVALIAÇÃO EMPÍRICA

JOSÉ B. ARAGÃO JÚNIOR, MAIQUEL S. MELO, RÔMULO S. ARAÚJO, DANIELO G. GOMES, GUILHERME A. BARRETO E JOSÉ NEUMAN DE SOUZA

Departamento de Engenharia de Teleinformática, Universidade Federal do Ceará
Campus do Pici, Bloco 925, 60455-760, Fortaleza, CE, Brasil
E-mails: {jr_aragao, maiquel, romulo, danielo, guilherme}@deti.ufc.br, neuman@ufc.br

Abstract— VoIP applications use buffers to soften the effects caused by the jitter. To do this, a delay in the execution is applied to the voice talkspurt to minimize the loss of packets due to the late arrival. To determine the best value for the delay is fundamental to guarantee an acceptable quality to the audio conference. Several algorithms were developed to predict the best execution delay. We evaluated seven algorithms, some classic, and other innovative for such purpose, like HMM and Neural Networks, and compared their performance analysing some metric, like the average lost packets.

Keywords— VoIP, playout delay, predictive algorithms, HMM, Neural Networks.

Resumo— As aplicações VoIP utilizam *buffers* para suavizar os efeitos causados pelo *jitter*. Para atingir tal objetivo, um atraso na execução é aplicado ao jato de voz para minimizar a perda de pacotes devido à chegada tardia. Determinar o melhor valor para o atraso é fundamental para garantir que a ligação possua uma qualidade aceitável. Vários algoritmos foram desenvolvidos com o objetivo de prever o melhor atraso de execução. Nós avaliamos sete algoritmos, alguns clássicos, outros inovadores para tal propósito, como os que utilizam HMM e Redes Neurais, e comparamos seus desempenhos levando em conta algumas métricas de desempenho, como a média de pacotes perdidos.

Palavras-chave— VoIP, atraso de execução, algoritmos de predição, HMM, Redes Neurais

1 Introdução

A utilização de serviços em tempo real, como a telefonia, em redes de comutação de pacotes, impõe muitos obstáculos devido, principalmente, às técnicas de roteamento dinâmicas empregadas nessas redes.

Em redes de comutação por pacotes, como a Internet, a voz gerada por uma fonte é digitalizada e carregada em pacotes IP que são emitidos a intervalos regulares e periódicos. Esses pacotes de voz podem trafegar por caminhos diferentes na rede e sofrer atrasos de enfileiramento variados em diferentes roteadores. Isto causa uma variação do atraso (*jitter*) que faz com que pacotes de voz não cheguem ao receptor com a mesma seqüência temporal com a qual foram gerados.

A utilização de um *buffer* é necessária a fim de suavizar os efeitos do *jitter* e simular um fluxo contínuo do emissor ao receptor. Este *buffer* armazena os pacotes, atrasando suas execuções por determinado período de tempo, permitindo desta maneira que pacotes mais atrasados cheguem a tempo de serem executados nos seus respectivos tempos programados de execução.

A esse atraso imposto pelo *buffer* é conhecido como excesso de atraso de execução. A escolha de um excesso de atraso adequado minimiza a perda de pacotes por chegada tardia – pacotes que chegam após o seu tempo programado de execução. Por outro lado, um excesso de atraso muito grande pode afetar a qualidade da conversação. Em geral, a escolha de um excesso de atraso de execução é um equilíbrio entre a introdução de atrasos adicionais e a taxa de pacotes perdidos por chegada tardia [Ranganathan, 2005].

Com a avaliação apresentada neste artigo é possível determinar qual a melhor técnica de predição de atraso de execução para uma aplicação VoIP, levando em conta os seguintes fatores: eficiência versus custo computacional.

2 Algoritmos de Atraso de Execução

Nesta seção serão apresentados sete algoritmos de execução de atraso, onde serão abordadas as vantagens e desvantagens de cada um deles. A seguir será descrita a terminologia necessária para a apresentação desses algoritmos.

2.1 Notação

A notação utilizada neste artigo, adaptada de [Rabiner, 1989], não exige a necessidade de sincronização entre os relógios do emissor e do receptor, no entanto, precisamos assumir que a diferença entre os relógios não varia com o tempo, ou seja, a duração de um segundo em ambos relógios deve ser sempre a mesma.

Os atrasos de rede dos pacotes são então normalizados pelo menor valor de atraso de rede do trace para assim, gerar valores de atraso sempre positivos, mesmo que a diferença entre os relógios do receptor e do emissor seja negativa.

A Tabela 1 mostra o conjunto de símbolos da notação utilizada em todo o artigo.

TABELA 1: NOTAÇÃO

Notação	Descrição
M	número de jatos de voz em um dado trace.
t_k^i	tempo em que o emissor gera o pacote i do k-ésimo jato de voz.
a_k^i	tempo em que o receptor recebe o pacote i do k-ésimo jato de voz.
n_k	número de pacotes de áudio em um jato de voz.
$\hat{d}_k^i$	atraso entre a geração do i-ésimo pacote do k-ésimo jato no emissor e sua recepção no receptor: $\hat{d}_k^i = a_k^i - t_k^i$.
$\hat{d}$	$\hat{d} = \min_{1 \le k \le M, 1 \le i \le n_k} \{\hat{d}_k^i\}$
d_k^i	atraso normalizado da rede sofrido pelo pacote i no k-ésimo jato de voz: $d_k^i = \hat{d}_k^i - \hat{d}$
$d_k^{(i)}$	i-ésimo menor atraso normalizado do k-ésimo jato.
p_k^i	tempo em que o receptor executa o pacote i do k-ésimo jato de voz.
$\hat{p}_k$	atraso de execução do k-ésimo jato.
$\hat{e}_k$	excesso de atraso de execução do jato k.

A Figura 1 mostra um cenário de sincronização de uma conversação entre emissor e receptor. A notação definida anteriormente é utilizada para descrever os elementos da figura.

Os pacotes são gerados no emissor a intervalos constantes (por exemplo, a cada 20ms). Devido às características das redes comutadas por pacotes, como o enfileiramento dos roteadores, os pacotes sofrem atrasos distintos na rede, podendo chegar ao receptor com intervalos de tempos variados.

Desse modo, a ordem de chegada dos pacotes não segue necessariamente a ordem de emissão, assim, um pacote *i+1* pode chegar ao receptor antes do pacote *i*. O receptor atrasa os pacotes com o objetivo de reordenar a seqüência de pacotes emitidos e minimizar os efeitos do *jitter*, reconstruindo a seqüência temporal com a qual os pacotes foram gerados.

Duas métricas serão utilizadas para comparar o desempenho dos algoritmos: *alp* (média de pacotes perdidos, do inglês *average lost packets*) e *ted* (atraso fim-a-fim, do inglês *the end-to-end delay*). O primeiro indica a porcentagem dos pacotes descartados por chegada tardia, já o segundo está relacionado com o atraso total sofrido pelos pacotes de voz.

2.2 Algoritmo de Atraso de Execução Offline

Algoritmos de atraso de execução *offline* servem como medida de comparação para os algoritmos de atraso de execução preditivos.

2.2.1 Algoritmo 1: Algoritmo Ótimo para um Jato de Voz Isolado

Com este algoritmo podemos calcular, para cada jato de voz, um atraso de execução, $\hat{p}_k$, que nos garanta que i pacotes sejam executados. Este atraso de execução é calculado como segue:

$$\hat{p}_k = d_k^{(i)} \quad (1)$$

$$i = n_k (1 - alp) \quad (2)$$

As equações (1) e (2) mostram que o atraso ótimo para qualquer jato k é igual ao maior dos atrasos normalizados de seus i pacotes menos atrasados, considerando a *alp*.

Considere o seguinte cenário, um jato de voz k com 6 pacotes, com *alp* = 0, tempos de emissão em ms representados pelo vetor $\vec{t}$ = *[0 20 40 60 80 100]* e atrasos de rede dados por $\vec{d}$ = *[75 70 50 60 50 80]*. Dessa forma, o vetor dos tempos de recepção $\vec{a}$ é dado pela soma de $\vec{t}$ e $\vec{d}$, logo $\vec{a}$ = *[75 90 90 120 130 180]*.

O pacote mais atrasado do jato k deverá ser executado tão logo chegue ao receptor, pois como é o último a chegar, seu tempo de execução será igual ao seu tempo de recepção. Sendo $\vec{p}$ o vetor que representa os tempos de execução dos pacotes e *i = 6* o pacote mais atrasado considerando uma *alp = 0*, p(6) = 180ms. Como os pacotes devem ser executados com o mesmo intervalo de tempo com o qual foram gerados (no exemplo a cada 20ms), então p(5) = 160ms, p(4) = 140ms, p(3) = 120ms, p(2) = 100ms e p(1) = 80ms, onde p(1) = d(6) é o próprio atraso de execução do jato de voz e d(6) é o maior atraso de rede do jato k.

Com esse algoritmo o excesso de atraso de execução para o jato de voz é calculado como segue:

$$\hat{e}_k = \hat{p}_k - d_k^f \quad (3)$$

Onde d_k^f é o atraso de rede do primeiro pacote a chegar ao receptor e $\hat{e}_k = e_k^1$, isto é, o excesso de atraso do primeiro pacote do jato. Utiliza-se (3) em todos os outros algoritmos no cálculo do número de pacotes perdidos.

O atraso de execução do jato de voz será sempre o maior atraso de rede dos pacotes do jato, mesmo que este não tenha sido o atraso de rede do último pacote do jato.

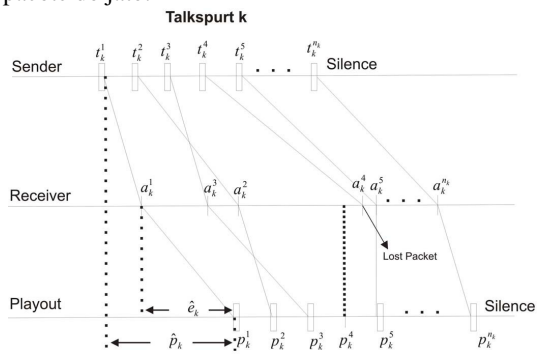

Figura 1: Tempos de emissão/recepção e execução dos pacotes

O atraso de execução ótimo é calculado logo após a chegada de todos os pacotes do jato, por isso ele é não causal, sendo utilizado principalmente para comparação com algoritmos preditivos.

2.3 Algoritmos de Predição de Atraso de Execução

Os algoritmos de atraso de execução preditivos realizam a predição do atraso de execução de um jato de voz durante o período silencioso imediatamente anterior a ele.

2.3.1 Algoritmo 2: Versão Causal do Algoritmo Ótimo para um Jato de Voz Isolado

É a versão causal ou *on-line* do Algoritmo 1. Nele o atraso de execução para um jato de voz *k* é dado pela seguinte fórmula:

$$\hat{p}_k = d_{k-1}^{(i)}, 1 \le k \le M \qquad (4)$$

A equação (4) mostra que o atraso de execução ótimo para um jato de voz é aplicado ao que jato que está chegando.

2.3.2 Algoritmo 3: Versão Causal Suavizada do Algoritmo Ótimo

Este algoritmo é uma versão suavizada do algoritmo anterior. A principal diferença é o uso de um simples filtro IIR (Resposta ao Impulso Infinito, do inglês *Infinite Impulse Response*), como mostra a equação (5):

$$\hat{p}_k = \rho \hat{p}_{k-1} + (1-\rho) d_{k-1}^{(i)}, \; 1 \le k \le M \qquad (5)$$

2.3.3 Algoritmo 4: Algoritmo Estatístico

Este algoritmo, apresentado por [Moon, 1998] e [Ramjee, 1994], funciona como um filtro recursivo linear, com um fator de peso α. A estimativa do atraso, para um certo pacote *i* no jato *k*, é dada por:

$$\hat{d}_k^{\,1} = \alpha \hat{d}_{k-1}^{\,n_{k-1}} + (1-\alpha) d_k^{\,1}, \qquad 1 \le k \le M \qquad (6)$$

$$\hat{d}_k^{\,i} = \alpha \hat{d}_k^{\,i-1} + (1-\alpha) d_k^{\,i}, 2 \le i \le n_k, 1 \le k \le M \qquad (7)$$

Onde $\hat{d}_k^{\,i}$ é o atraso estimado para o *i*-ésimo pacote do *k*-ésimo jato, e $\hat{d}_k^{\,n_k}$ é o atraso estimado para o último pacote do jato de voz *k*, que representa o atraso médio estimado para o jato de voz *k*. A estimativa da variância do atraso, para um certo pacote *i* no jato *k*, é dada por:

$$\hat{v}_k^{\,i} = \alpha \hat{v}_k^{\,i-1} + (1-\alpha) \left| \hat{d}_k^{\,i} - d_k^{\,i} \right|, 1 \le i \le n_k, 1 \le k \le M \qquad (8)$$

O atraso de execução predito para o *k*-ésimo jato é calculado como segue:

$$\hat{p}_k = \hat{d}_k^{\,n_k} + \beta \hat{v}_k^{\,n_k}, \qquad 1 \le k \le M \qquad (9)$$

Em nossas implementações utilizamos $\alpha = 0.998002$ e $\beta = 4$, que são os valores adotados por [Ramjee, 1994].

2.3.4 Algoritmo 5: Algoritmo de Predição utilizando HMM

Esta técnica foi proposta por [Yensen, 2003], nela o excesso de atraso é dado por:

$$\hat{e}_k = \hat{p}_k - \hat{d}_k^{\,n_k} \qquad (10)$$

onde $\hat{p}_k$ é o atraso de execução ótimo calculado de acordo com o Algoritmo 1, e $\hat{d}_k^{\,n_k}$ é o atraso estimado do último pacote do jato k calculado por (6) e (7).

Os valores de $\hat{e}_k$ são quantizados em intervalos fixos I em um número finito de níveis, *N*, os quais representam os estados do modelo. Esta quantização é necessária, pois é inviável trabalhar com espaço de estados contínuo [Yensen, 2003]. A quantização é definida pela equação a seguir:

$$Q[\hat{e}_k] = Q[\hat{p}_k - \hat{d}_k^{\,n_k}] \qquad (11)$$

Desta forma cada jato de voz possuirá um atraso de execução favorável, representado pelo seu estado $S_k = j$, onde $Q[\hat{e}_k] = j \cdot I$. A seqüência desses estados nos dá um histórico do comportamento da rede e é utilizada no treinamento do HMM para a geração do vetor de probabilidades iniciais, π, e da matriz de probabilidades de transições de estados, A.

Para cada jato de voz num trace é definido um vetor $O_k = [e_k^1 \; e_k^2 \cdots e_k^{n_k}]$, onde O_k é a observação para o jato *k* e e_k^i é o excesso de atraso para o pacote *i* do jato *k* em relação ao atraso médio estimado para o jato, sendo:

$$e_k^i = \begin{cases} d_k^i - \hat{d}_k^{\,n_k}, & if \quad d_k^i \ge \hat{d}_k^{\,n_k}, 1 \le i \le n_k \\ 0, & if \quad d_k^i < \hat{d}_k^{\,n_k}, 1 \le i \le n_k \end{cases} \qquad (12)$$

Da seqüência de jatos de voz extrai-se a seqüência $O_1, O_2, O_3, \ldots, O_k$ de observações.

Para cada jato de voz é calculada a média (μ_k) e a variância (σ_k^2) dos e_k^i. A média das médias dos jatos pertencentes ao estado *j* define a média do estado *j*, (μ_j), e a média das variâncias destes mesmos jatos define a variância do estado *j*, (V_j).

A cada jato *k*, é calculada a densidade de observação em relação a cada um dos estados como proposto em [Yensen, 2003], formando a matriz de densidades de observações, B, e finalizando assim a fase de treinamento do HMM.

De posse dos parâmetros do modelo, π, A, B, utiliza-se o algoritmo de Viterbi para predizer o estado mais provável em relação ao jato. O excesso de atraso previsto para um jato *offline* é aplicado ao próximo jato que está chegando.

Pode ser necessária a normalização dos valores de B e a utilização de logaritmos no algoritmo de Viterbi a fim de se evitar o *overflow* gerado pelas sucessivas multiplicações de números fracionários.

2.3.5 Algoritmo 6: Algoritmo Auto-Regressivo

Neste algoritmo os atrasos normalizados de cada jato de voz foram usados na predição do atraso de rede normalizado para o jato de voz seguinte. Dentro do jato de voz *k*, o atraso do pacote *i+4* foi estimado com base nos atrasos dos quatro pacotes antecedentes, isto é, foi utilizada uma janela de 4 pacotes. Desta forma:

$$d_k^{(i+4)} = \alpha_0 + \alpha_1 d_k^{(i+3)} + \alpha_2 d_k^{(i+2)} + \alpha_3 d_k^{(i+1)} + \alpha_4 d_k^{i} \quad (13)$$

onde *k*, é o número do jato de voz no trace, $1 \leq k \leq M$, e *i* é o número do pacote de voz do jato.

O vetor $\alpha = [\alpha_0\ \alpha_1\ \alpha_2\ \alpha_3\ \alpha_4]$ é obtido da resolução do sistema linear montado a partir da equação (13).

Após obtenção do vetor alfa, calcula-se o atraso de rede estimado do ultimo pacote que é então utilizado como atraso de rede do primeiro pacote do próximo jato. O atraso de execução do jato atual, $\hat{p}_k$, será utilizado como atraso de execução do próximo jato de voz, assim como no Algoritmo 2, daí:

$$\hat{e}_{k+1} = \hat{p}_k - d_k^{n_k} \quad (14)$$

6) Algoritmo 7: Algoritmo MLP

Este algoritmo foi proposto por [Ranganathan, 2005]. Em sua implementação o atraso de execução do jato de voz, $\hat{p}_k$, é calculado como segue:

$$\hat{p}_k = h\mu(d_k) + M\sigma(d_k) \quad (15)$$

onde $\mu(d_k)$ e $\sigma(d_k)$ são, respectivamente, a média e o desvio padrão dos atrasos de rede do jato de voz *k*.

A média e o desvio padrão são obtidos submetendo as médias e os desvios padrões dos atrasos de rede dos pacotes de *x* jatos de voz precedentes a uma rede neural MLP com quatro neurônios na camada escondida, dois neurônios na camada de saída, passo de aprendizagem igual a 0.05 e fator de momento igual a 0.75. Neste artigo adotou-se $x = 4$, $h = 0{,}2$ e *M*, que geralmente é ajustado de acordo com as condições da rede, foi fixado em *2*.

O excesso de atraso estimado para o jato de voz é então calculado pela diferença entre o valor de $\hat{p}_k$ predito e o valor do atraso de rede de seu primeiro pacote a chegar. Desta forma:

$$\hat{e}_k = \hat{p}_k - d_k^1 \quad (16)$$

Na implementação e execução desta rede neural, não houve a realização de etapa de treinamento. O aprendizado é *on-line*, ou seja, à medida que os dados (média e variância de atraso de rede de jatos passados) são fornecidos à rede, a mesma ajusta seus pesos sinápticos.

Este tipo de implementação, sem treinamento *off-line* da rede, penaliza os primeiros jatos de voz com valores ruins de excesso de atraso, mais devido a rápida convergência, ela se adequa bem às condições da rede.

3 Implementação

Os algoritmos vistos neste artigo foram simulados utilizando traces obtidos de [Moon, 2007]. As Figuras 2 e 3 ilustram os atrasos de rede normalizados para os pacotes de voz contidos nestes traces.

A avaliação do desempenho foi realizada com as seguintes medidas: estatísticas acerca do atraso fim a fim, como média e desvio padrão, taxa média de perda de pacotes e o número de pacotes perdidos por jato.

O mínimo, a média, o máximo e o desvio padrão do atraso total fim a fim foram calculados com base nos valores de $\hat{p}_k$ calculados para cada jato, de cada trace, para cada um dos algoritmos implementados.

A taxa média de perda de pacotes e o número de pacotes perdidos por jato, foram calculados da seguinte forma: para os algoritmos 1, 2, 3 e 4, os valores de $\hat{p}_k$ são encontrados para cada jato e a partir dele são encontrados os tempos de execução programados para cada pacote de cada jato. Em seguida, comparam-se os tempos de chegada de cada pacote com seus respectivos tempos programados de execução.

Nos Algoritmos 5, 6 e 7, $\hat{e}_k$ é calculado para cada jato de voz. Estes valores são então somados aos tempos de chegada do primeiro pacote de seus respectivos jatos, permitindo assim encontrar o tempo de execução de todos os pacotes do jato.

A técnica proposta pelo Algoritmo 5 exige um alto custo computacional. Para reduzir esse custo, assumimos uma matriz homogênea de probabilidades de transição de estados.

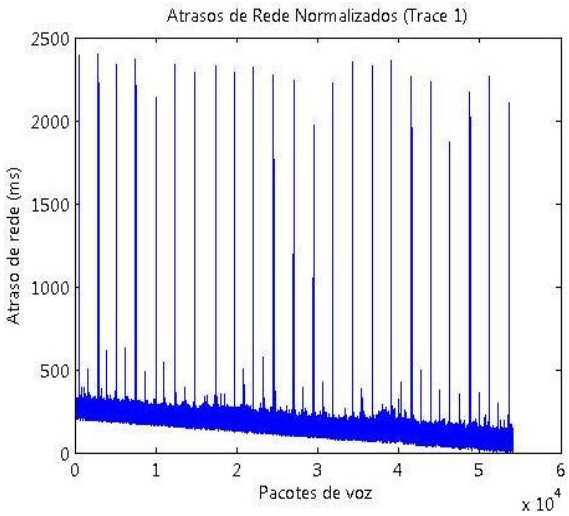

Figura 2: Atraso de Rede Normalizado para o Trace 1

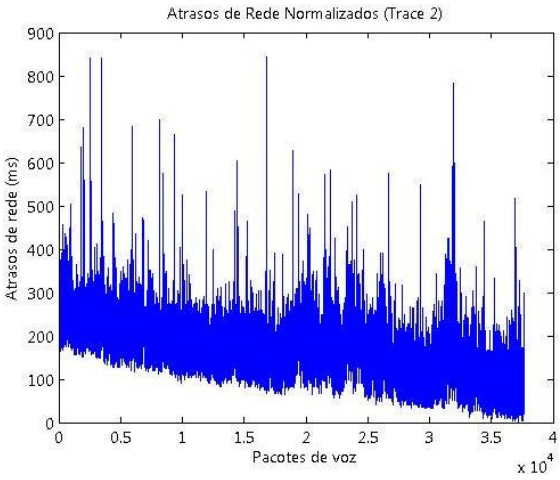

Figura 3: Atraso de Rede Normalizado para o Trace 2

Após a realização de muitas simulações, adotamos um modelo HMM com 15 estados, distribuídos em intervalos constantes de 15ms a 225ms. A matriz densidade de probabilidade de observação e a matriz gerada pelo algoritmo de Viterbi foram normalizadas com o objetivo de evitar *underflow*, causado pelas sucessivas multiplicações de números pequenos.

Em todas as implementações, um pacote será descartado e dado como perdido se seu tempo de chegada for maior que seu tempo programado de execução. Deste modo, calculamos a *alp* e o número de pacotes perdidos por jato de voz.

4 Conclusão

Os resultados obtidos da avaliação dos algoritmos estão expostos nas tabelas 2 e 3.

O Algoritmo 1 foi configurado com uma *alp* de 5% e os resultados de atraso total fim a fim obtidos com sua implementação servem apenas como base de comparação para os outros algoritmos.

Dos algoritmos preditivos, o Algoritmo 2 é o que possui a implementação mais simples e menor custo computacional. Porém, mesmo sendo simples, apresentou bons resultados.

Após testes empíricos percebeu-se que o Algoritmo 3 obteve melhores resultados com a utilização de um parâmetro de suavização $\rho = 0.5$.

TABELA 2: RESULTADOS OBTIDOS COM O TRACE 1.

Método	ted (ms)				alp(%)
	Min.	Média	Max.	Std	
ALG. 1	116	334.75	2404	406.85	5.00
ALG. 2	128	363.5	2464	417.63	3.67
ALG. 3	144	363.65	2336	345.38	3.44
ALG. 4	232.88	519.6	1163.1	213.14	1.93
ALG. 5	95	397.91	2430	204.82	2.93
ALG. 6	75.96	381.43	2350	441.57	4.91
ALG. 7	208.16	405.5	2033.5	292.74	1.95

TABELA 3: RESULTADOS OBTIDOS COM O TRACE 2.

Método	ted (ms)				alp(%)
	Min.	Média	Max.	Std	
ALG. 1	140	261.27	800	77.95	5.00
ALG. 2	148	297.64	844	101.87	5.23
ALG. 3	150	297.59	694	82.74	4.71
ALG. 4	236.79	360.57	637.9	62.69	3.28
ALG. 5	95	404.91	669	87.06	2.90
ALG. 6	38.57	317.37	1334.7	103.67	4.17
ALG. 7	260.91	398.92	1206.2	178.19	1.98

A técnica proposta pelo Algoritmo 4 obteve bons resultados de *alp*. Este algoritmo mostrou ter baixo custo computacional e este algoritmo é o que se adapta melhor às situações de atraso na rede.

O Algoritmo 5, que utiliza HMM alcançou bons resultados para os traces utilizados neste artigo. Porém ele requer um alto custo computacional tanto para a fase de treinamento como para a predição do atraso de execução. Além disso, a construção de vários modelos, cada um apropriado para determinada situação da rede, se faz necessária para que esta técnica funcione corretamente.

O Algoritmo 6 não obteve melhorias significativas em relação aos algoritmos mais simples, e sua implementação aumenta em muito o custo computacional, pois requer muito espaço em memória e muito processamento nos cálculos de multiplicações de matrizes e de matrizes inversas.

O Algoritmo 7, que utiliza uma rede neural MLP como preditor, obteve bons resultados na predição de valores para médias e desvios padrões de atrasos de rede. Com estes resultados, foi possível obter ótimos valores de excesso de atraso estimado com o uso da equação (15). Com os valores de excesso de atraso obtidos, registraram-se baixos valores médios de perda de pacotes (*alp*).

Agradecimentos

Os autores agradecem a CAPES pelo suporte financeiro através de bolsas de mestrado e PRODOC.

Referências Bibliográficas

Moon, S.B., Kurose, J., Towsley, D. (1998). *Packet audio playout delay adjustment: Performance bounds and algorithms*. ACM/Springer Multimedia Systems 6, pp. 17–28.

Moon, S. B. (2007) Sue Moon's Publications, Disponível em: <http://an.kaist.ac.kr/~sbmoon/publication.html>

Rabiner, L. (1989). "A tutorial on hidden Markov models and selected applications in speech recognition," Proc. IEEE, Vol. 77, No. 2, pp. 257–286.

Ramjee, R., Kurose, J., Towsley, D., Schulzrinne, H. (1994). *Adaptive playout mechanisms for*

packetized audio applications in wide-area networks. In: Proceedings of the IEEE Infocom, pp. 680–688.

Ranganathan, M. K., Kilmartin, L. (2005). *Neural and Fuzzy Computation Techniques for Playout Delay Adaptation in VoIP Networks*. IEEE Transaction on Neural Networks. Vol 16 no.5, September

Yensen T., Lariviere J., Lambadaris I., and Gaubran A. R. (2003). *HMM Delay Prediction technique for VOIP*. IEEE Transaction on Multimedia. Vol. 5 no.3, September.