Alocação Dinâmica de Taxa de Transmissão em Redes de Pacotes Utilizando Redes Neurais Recorrentes Treinadas com Algoritmos em Tempo Real

Flávio Henrique Teles Vieira, Rodrigo Pinto Lemos, *Member, IEEE* e Luan Ling Lee, *Member, IEEE*

*Resumo—* **Propõe-se neste artigo o uso de redes neurais recorrentes para realizar predição de tráfego em redes de computadores. Estima-se em tempo real a taxa de transmissão que deve ser disponibilizada a fim de se evitar perdas de bytes a uma eficiente utilização da rede. Para tal, redes neurais recorrentes foram treinadas com algoritmos de aprendizagem em tempo real: RTRL (Real Time Recurrent Learning) e filtro de Kalman estendido. Os algoritmos foram aplicados na alocação dinâmica de taxa de transmissão em um enlace de rede, verificando as suas eficiências na predição e controle de tráfego.**

***Palavras-chave* Predição, Controle, Taxa, Tráfego, Redes Neurais.**

1. INTRODUÇÃO

A

integração de serviços nas redes de comunicação traz a necessidade de gerenciar tráfegos com diferentes requisitos de qualidade de serviço e com características complexas. A modelagem adequada do tráfego oferecido à rede é importante para o desenvolvimento de estratégias de

alocação eficiente de recursos e de controle.

Diversos estudos apontam a importância da auto- similaridade nos dados de tráfego de redes de computadores [1]-[2]-[3], assim como suas propriedades multifractais [4]. A auto-similaridade introduz dificuldades na otimização do desempenho da rede e na garantia de qualidade de serviço por causar impactos significativos no atraso e no aumento de taxa de perda de pacotes [5]-[6]-[7]. Um efeito da auto- similaridade é que os buffers localizados nos multiplexadores e comutadores devem ser maiores, conduzindo a atrasos. Portanto, as implicações principais no desempenho da rede devido ao tráfego auto-similar são períodos concentrados de congestionamento e amplificada perda e atraso de pacotes.

Normalmente, um aumento linear no tamanho do buffer produz uma diminuição exponencial na perda de pacotes e um

Flávio Henrique Teles Vieira e Luan Ling Lee pertencem ao Departamento de Comunicações da Faculdade de Engenharia Elétrica e Computação da Universidade Estadual de Campinas (FEEC-UNICAMP), São Paulo, Brasil. E-mails: [{flavio,lee@decom.fee.unicamp.br}.](mailto:flavio%2Clee@decom.fee.unicamp.br)

Rodrigo Pinto Lemos pertence a Escola de Engenharia Elétrica da Universidade Federal de Goiás (UFG), Goiânia, GO, Brasil. Email:

[{lem](mailto:lemos@eee.ufg.br)[os@eee.ufg.br}](mailto:os@eee.ufg.br)

aumento efetivo do uso da capacidade de transmissão, porém com tráfego auto-similar essas propriedades não funcionam [1].

Tráfego com rajadas multiescala implica na existência de períodos concentrados de alta atividade em diferentes escalas de tempo, causando uma degradação linear na perda de pacotes e superlinear para o atraso de pacote. Quanto mais o tráfego apresenta distribuição *heavy tailed* mais se degradam parâmetros como perda de pacotes e taxa de retransmissão.

Uma descoberta importante é que quanto maior a carga em uma rede Ethernet, maior o grau de auto-similaridade do tráfego [8]. Como a tendência é de se encontrar redes cada vez maiores com aplicações interativas e multimídias diversas na composição do tráfego, os problemas se agravarão. Aumentando-se as rajadas auto-similares do tráfego ocorre um aumento da ocupação do buffer para vários tamanhos de buffer, resultando em atraso. Há uma relação entre perda de pacotes e atraso, se diminui um, aumenta o outro, o que implica que qualidade de serviço fica mais difícil de ser garantida para tráfego em rajadas. Como também há períodos de baixa intensidade de tráfego, a alocação de uma taxa única para o tráfego além de levar a uma sub-utilização da rede pode levar a perdas significativas para tráfego auto-similar. Sugerimos portanto, o uso de redes neurais recorrentes treinadas em tempo real para se efetuar alocação dinâmica de taxa de transmissão.

O controle de congestionamento deve então tentar contornar os problemas trazidos pela auto-similaridade. O controle de congestionamento pode ser preventivo, em que ações são tomadas baseadas principalmente no contrato de tráfego, além de ações como: policiamento de usuários, controle de prioridades e formatação de tráfego. Porém, quando o processo é de longa dependência é difícil garantir que o tráfego respeite o contrato realizado na admissão da conexão. Daí a necessidade de um monitoramento constante do tráfego e de um controle preditivo. Além do mais, o algoritmo *leaky bucket* como método de policiamento não é uma solução satisfatória para tráfego auto-similar [9].

Neste trabalho propomos o uso de algoritmos capazes de fazer predições em tráfego real. Para validação dos algoritmos foram utilizados traços de tráfego reais Ethernet retiradas das

medições da Bellcore1 que apresentam características auto- similares e multifractais. A fim de obtermos estruturas que não dependam do conhecimento do modelo mais adequado ao tráfego utilizamos redes neurais recorrentes para sua predição e controle automático.

1. ALOCAÇÃO DINÂMICA DE TAXA DE TRANSMISSÃO

Em geral, o tráfego em redes de alta velocidade é complexo, não-linear e não-estacionário. Uma proposta interessante é a da alocação dinâmica de recursos [10]-[11]. A idéia é estimar a taxa de transmissão necessária para garantir a transmissão do tráfego em intervalos regulares (Figura 1), uma vez que o uso de uma taxa fixa de serviço para tráfego com cauda pesada e altamente correlacionado pode causar excessiva perda de pacotes caso esta taxa não esteja próxima ao pico de tráfego [12]. Diferentemente de alguns trabalhos onde se muda a taxa da fonte, objetiva-se aqui adaptação de taxa da rede (enlace)[13]-[14].

Se de um lado uma maior eficiência de transmissão dos enlaces da rede pode ser atingida por adaptação freqüente da taxa de transmissão, temos que esta freqüência de adaptação está limitada pelo tempo de processamento do protocolo de controle da rede. Para tráfego de vídeo foi observado que as mudanças nas cenas estão na escala temporal de 1 segundo ou mais, sendo assim, a taxa de transmissão precisa ser adaptada dentro de um intervalo de algumas centenas de milisegundos [10].

Em uma rede ATM por exemplo, a demanda de taxa de transmissão em um circuito virtual (VC) pode ser acomodada por vários protocolos. No nível de enlace, há esquemas de particionamento de capacidade entre VCs, descarte de células com baixa prioridade. No nível de rede, esquemas de roteamento podem ser usados para obter taxa extra, sem violar a qualidade de serviço de outros VCs.

tx *G*max{xˆ(n + *l*), l D, D +1, ..., D + M} (1) onde xˆ(n + *l*) é o valor predito de tráfego a *l* instantes de tempo a frente e G será descrito posteriormente.

O intervalo de adaptação de taxa é composto pelo o tempo

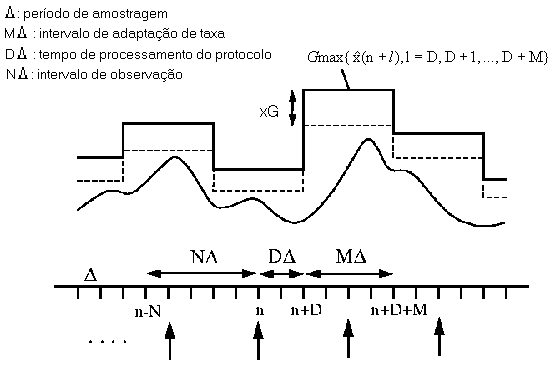
de computação do algoritmo de predição (treinamento da rede neural ou intervalo de observação) e D ' , tempo de processamento do protocolo de alocação. Para tal sistema propomos a utilização de dois algoritmos de predição: redes neurais treinadas com filtros de Kalman estendido e algoritmo RTRL (Real Time Recurrent Learning), os quais proporcionam operação em tempo real, visando um controle adaptativo e contínuo da taxa de transmissão.

1. REDES NEURAIS RECORRENTES

Redes neurais artificiais têm sido bastante empregadas na predição e identificação de séries temporais [15]-[16]-[17]. A rede neural MLP (Multilayer Perceptron) é a mais utilizada dentre as redes neurais existentes, apesar de usar um mapeamento entrada-saída estático.

O interesse em redes neurais recorrentes está no seu processamento temporal e na sua capacidade de implementar memórias adaptativas [18]. A rede neural recorrente deve aprender a extrair informação relevante do passado da seqüência contínua e variante de entrada e computar uma saída relativa a um valor futuro dos dados de entrada. As redes recorrentes possuem estados que evoluem de acordo com certas equações não-lineares podendo modelar sistemas dinâmicos. Alguns algoritmos calculam o gradiente do erro eficientemente como o TBTT (Truncated Backpropagation Through Time) e o BTT(h,h’) [19]. No algoritmo RTRL (Real-Time Recurrent Learning) [20] os pesos são atualizados de forma “on-line” a uma complexidade computacional relativamente elevada: *O(N4)*, em que *N* é o número de nós da rede. No entanto, constatamos que se consegue resultados satisfatórios com apenas dois neurônios e que um aumento no número de neurônios nem sempre leva a um melhor

desempenho.

Seja uma rede neural consistindo de *N* neurônios com *M* entradas externas, *x(n)* o vetor de entradas *M*x1 no tempo *n*, e *y(n+1)* o vetor de saídas *N*x*1* no instante de tempo *n+1*. Definiremos o vetor u(*n*) como uma concatenação dos dois vetores *x(n)* e *y(n)***.** Seja *A* o conjunto de entradas externas, e *B*

o conjunto de saídas. Se *i*  *A* então *ui* (*n*) *xi* (*n*) , se *i*  *B*

então *ui* (*n*) *yi* (*n*) . A rede possui duas camadas: camada de

processamento e camada de concatenação entrada-saída (Figura 2). A rede neural é completamente conectada com *MN*

Fig.1. Divisão do tempo para alocação de taxa

Para o esquema de alocação dinâmica de taxa proposto, seja

' o período de amostragem do sinal de tráfego, a taxa de

conexões diretas e *N* 2 conexões de realimentação e *z-1* é o retardo de uma unidade de tempo aplicado ao vetor de saída. Chamaremos de *W* a matriz de pesos tendo dimensão *N(M+N)*.

transmissão será adaptada em intervalos de M ' (Figura 1) ou seja M intervalos de tempo discreto n, para um valor igual a:

Seja a atividade interna de um neurônio *j* no tempo *n* para

*j*  *B* :

1 <http://www.acm.org/sigcomm/ITA>

*v j* (*n*)

¦*wji* (*n*)*ui* (*n*)

*i**A**B*

(2)

em que

*w ji*

representa os pesos sinápticos. A saída do

Como

'*w* (*n*) –*K*  w*H* (*n*) *K*¦*e*

(*n*)*S j* (*n*) , os

neurônio *j* no próximo instante é dada por:

*kl*

*kl*

(*n*)

*j kl*

*j**C*

w*w*

*y j* (*n* + 1) *M* (*v j* (*n*))

(3)

pesos sinápticos serão atualizados de acordo com a equação:

A função utilizada, diferentemente do algoritmo de

Williams, é linear [19]. Tem-se então, as equações que descrevem a dinâmica do sistema.

*wkl* (*n* + 1) *wkl* (*n*) + '*wkl* (*n*)

1. ALGORITMO DE KALMAN ESTENDIDO

(8)

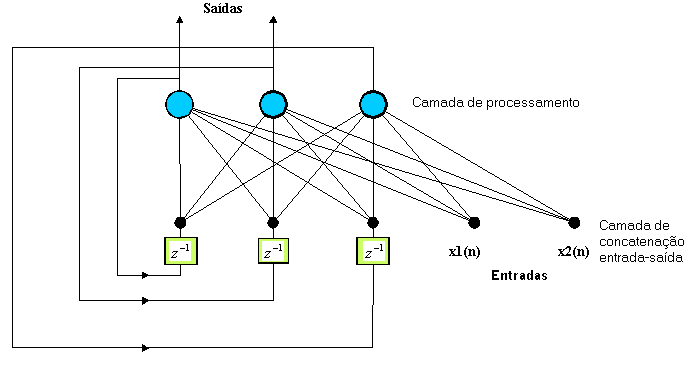


Fig. 2. Rede neural recorrente utilizada

IV. ALGORITMO RTRL

O algoritmo de treinamento com o qual trabalhamos é uma extensão do algoritmo de aprendizagem recorrente em tempo real (RTRL) descrito por Williams e Zipser em 1989 para redes recorrentes [19]. Seja *d j* (*n*) a resposta desejada para o

neurônio *j* no instante *n* e *C* o conjunto de neurônios de saída visíveis. Podemos definir um erro variante no tempo como *e j* (*n*) *d j* (*n*) – *y j* (*n*) se *j*  *C* . Definindo uma soma

instantânea de erros quadráticos no instante *n* como

O método de filtragem de Kalman consiste no uso de um conjunto de equações que provêem uma computação eficiente e recursiva para solução do método dos mínimos quadrados [22].

O filtro de Kalman estendido (EKF) pode ser usado como um algoritmo em tempo real para determinação dos pesos em uma rede neural recorrente. Considera-se então, o problema da aprendizagem como um problema de filtragem. Como a rede neural possui uma estrutura não-linear, usaremos o filtro de Kalman estendido. O filtro de Kalman estendido lineariza a parte não-linear do sistema e usa o filtro de Kalman original neste modelo linearizado.

O EKF foi inicialmente aplicado em redes MLP por Singhal e Wu [23]. Eles mostraram que o algoritmo EKF convergia mais rápido do que o ‘backpropagation’ e, às vezes, quando o ‘backpropagation’ falhava, o EKF (Extended Kalman Filter) convergia para uma boa solução [24]-[25]. Puskorius e Feldcamp aplicaram o algoritmo no treinamento de redes neurais recorrentes, assim como Mathews e Williams [24]-[26]-[27]. Williams observou que o algoritmo RTRL pode ser visto como uma simplificação feita no EKF e que o EKF converge mais rápido do que os algoritmos BPTT e

*H* (*n*) 1

¦

2

*e* 2 *j*

(*n*) , devemos minimizar a função-custo

RTRL.

Seja o vetor

*d(n)* de saídas desejadas de tamanho *s*x1. O

*j**C*

*H total*  ¦*H* (*n*) . Para isso é usada uma aproximação do

*n*

objetivo é achar os pesos *w(n)* (estados do sistema) tais que a diferença entre a resposta da rede e o desejado seja mínima em termos de erro quadrático. As equações que governam a

método do gradiente descendente: *WH total*

Para um peso particular *wkl* (*n*) :

¦*WH* (*n*) .

*n*

operação da rede neural recorrente são:

*d* (*n*) *hn* (*w*(*n*), *u*(*n*)) + *r*(*n*)

*w*(*n* + 1) *w*(*n*)

(9)

(10)

w*H* (*n*) –¦*e*

(*n*) w*y j* (*n*)

(4)

onde *d(n)* é tratado como o vetor de medição, *r(n)* vetor de

w*wkl*

(*n*)

*j*

*j**C*

w*wkl*

(*n*)

erro de medição e a função não linear *hn*

descreve a relação

Usando (2) e (3) podemos obter [21]:

entre entradas *u(n)* e os pesos *w(n)*. O algoritmo EKF aplicado no treinamento de redes neurais se baseou nas seguintes

w*y j* (*n* + 1) *M* c(*v*

(*n*))ª– ¦*w*

(*n*) w*yi* (*n*) + *G*

*u* (*n*)º (5)

equações:

w*wkl* (*n*)

*j* « *ji*

¬ *i**B*

w*wkl* (*n*)

*kl l* »

¼

## Equações de Atualização de Medições (correções):

Reescrevendo esta equação utilizando uma variável triplamente indexada:

*P*(*n* \ *n* – 1)*H T* (*n*)

*j* w*y j* (*n*)

*K* (*n*)

*n* (11)

*n*

*S kl* (*n*)

*kl*

w*w*

(*n*)

(6)

*Hn* (*n*)*P*(*n* \ *n* – 1)*H T* (*n*) + *R*(*n*)

*S j* (*n* + 1) *M* c(*v*

(*n*))ª¦*w* (*n*)*S j* (*n*) + *G*

*u* (*n*)º

(7)

*w*ˆ(*n* \ *n*) *w*ˆ(*n* \ *n* –1) + *K* (*n*)[*d* (*n*) – *hn* (*w*ˆ(*n* \ *n* –1),*u*(*n*))] (12)

*kl j*

« *ji kl*

¬*i**B*

*kl l* »

# ¼

*P*(*n* \ *n*) (*I* – *K* (*n*)*Hn* (*n*))*P*(*n* \ *n* – 1) (13)

em que:

## Equações de Atualização Temporal (predição):

*w*ˆ(*n* + 1 \ *n*) *w*ˆ(*n* \ *n*) *P*(*n* + 1 \ *n*) *P*(*n* \ *n*)

*H* (*n*) w*hn* (*w*ˆ(*n*), *u*(*n*))

*n* w*w*

(14)

(15)

(16)

dá apenas nos intervalos de tempo onde não se tem adaptação de taxa. Notou-se que o treinamento da rede neural sem a recorrência e seu uso somente em operação diminui o EQMN de predição.

Para a série Bc-Octint, o EQMN das predições a um passo dos instantes 801 a 1701 foi de 0,3850, utilizando dois neurônios, com cinco entradas equivalentes a 5 instantes em sequência do tráfego real e taxa de aprendizagem 0,1

(Tabela1).

e *hn* (.) [*h*1, *h*2 ,..., *hs* ] são as *s* saídas da rede neural, *R* é a matriz de covariância do erro de medição, *P* é a matriz de covariância do erro de estado, *w*ˆ é uma estimativa de estado (pesos) e *K* é conhecido como ganho de Kalman [19].

A diferença entre a aplicação do EKF no treinamento de redes com alimentação direta e redes recorrentes está no

*n*

Para a série Bc-Octext, utilizou-se também dois neurônios, uma entrada e taxa de aprendizagem 0,2. Foi obtido um EQMN de 0,3946 para a predição dos pontos 1000 a 2000 (Figura 4).

cálculo da matriz jacobiana transposta .

*Hn* (*n*) , onde

*HT* (*n*) é a sua

O algoritmo EKF possui complexidade computacional alta para grandes redes neurais. Seja N o número de neurônios da rede neural, então o número de operações aritméticas a cada

passo é *O*(*N* 4 ) , o mesmo gasto pelo RTRL; porém o

armazenamento é de *O*(*N* 4 ) elementos contra *O*(*N* 3 ) do

RTRL, pois é necessário armazenar a matriz *P* [28]. Entretanto, assim como no RTRL pode se usar uma rede neural recorrente com poucos neurônios.

1. SIMULAÇÕES E RESULTADOS

As predições em tempo real em uma rede de computadores podem ser utilizadas no controle de fluxo do tráfego, dado que os erros de predição obtidos com o treinamento em tempo real são comparáveis ao do treinamento “on batch” com redes neurais MLP e ou RBF [29]-[30].

Utilizando atualizações dos pesos a cada instante pode-se melhor acompanhar sinais como a série Bc-Octint (Figura 3), que apresentam variações bruscas e que se diferenciam muito, em diferentes instantes de tempo. Além do que as redes neurais recorrentes são capazes de mapear pontos *l* instantes no futuro com pontos do presente.

A rede neural treinada com o algoritmo RTRL e sua operação se mostraram insensíveis à escolha inicial dos pesos e estável, se estes forem inicializados com valores baixos

(0,0001). Para se avaliar a qualidade das predições foi usado o

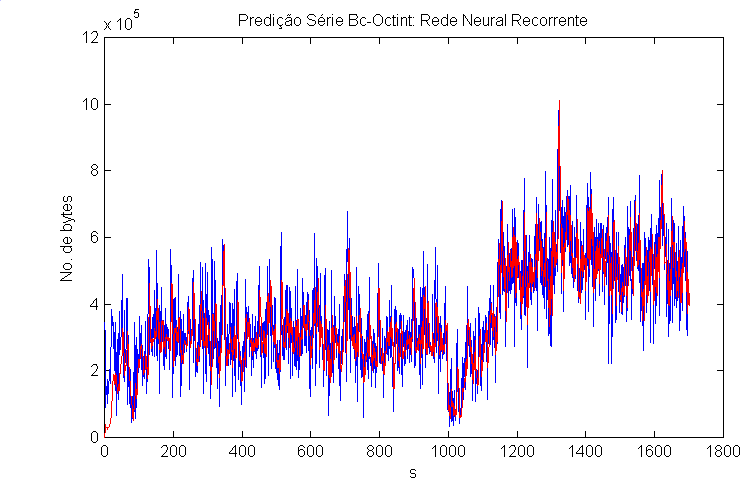
Fig. 3. Predição a um passo por rede neural recorrente treinada com algoritmo RTRL (em vermelho).

TABELA I EQMN DE PREDIÇÃO

|  |  |  |
| --- | --- | --- |
| **EQMN** | | |
|  | RTRL | EKF |
| Série Bc-octext | 0,3946 | 0,3972 |
| Série Bc-octint | 0,3850 | 0,3398 |

Utilizou-se nas simulações uma taxa de amostragem ' de 0,1s, D=1 e M=4, portanto a rede neural deve mapear diretamente pontos distantes de 4 instantes de tempo (Figura 1). A cada instante, guardamos a predição do quarto ponto à frente, sendo feitas 4 predições: xˆ(n1 + 4) , xˆ(n2 + 4) , xˆ(n3 + 4) e xˆ(n4 + 4) . Assim, a taxa de transmissão alocada

erro quadrático médio normalizado (EQMN) dado por:

*EQMN*  1 *p* [ *y*(*n*) – *y*ˆ(*n*)]2

¦

*V* 2 *p n* 1

(17)

para os M ' segundos seguintes será:

tx *G*max{xˆ(n1 + 4), xˆ(n 2 + 4), xˆ(n 3 + 4), xˆ(n 4 + 4)} (18) onde G é um parâmetro de controle adicionado para absorver

onde *y(n)* é o valor real da série, *y*ˆ(*n*) é o valor predito, *V* 2 é a variância da seqüência real sobre o intervalo de duração da predição e *p* é o número de amostras do teste.

As séries utilizadas nas simulações foram: Bc-Octext cuja

principalmente altos picos de tráfego não bem preditos [10]. Utilizou-se G=1,25.

Definiremos a eficiência do esquema de alocação dinâmica

para o sistema a cada instante como:

escala de tempo escolhida foi de 1min com 2046 pontos e a série Bc-Octint com 1759 pontos e escala de tempo 1s. Instantes diferentes da mesma série temporal são utilizados no

*U* q(n)

# b(n)

(19)

treinamento e predição da rede neural. O treinamento desta se

onde q*(n)* é o número de bytes a serem escoados e *b(n)* é a

taxa de transmissão alocada no instante *n*.

Nas simulações foi considerado um enlace único com taxa variável, ou seja, foram usados como entrada para o sistema com alocação dinâmica de taxa, 3000 pontos da série Bc- Octint, equivalente a 5 min de tráfego Ethernet chegando em um nó com buffer finito.

Inicialmente, estipulamos o número de bytes suportado pelo buffer de 70% do valor máximo da série Bc-Octint, amostrada a cada 0,1s: 62.429bytes. No algoritmo de Kalman estendido inicializamos todos pesos com valor igual a 0,1, para tirarmos a aleatoriedade da escolha inicial dos pesos. A matriz *R* foi inicializada com valor igual a 100 e a matriz *P* com 800\**I*14x14 onde *I* é a matriz identidade. O erro médio quadrático das predições em tempo real de quatro instantes à frente foi de 1,1038 (RTRL) e 1,2076 (EKF) para os pontos de 1000 a 3000. A Figura 4 mostra a taxa de transmissão alocada para escoamento do tráfego usando o algoritmo de Kalman. A eficiência média do esquema de alocação (equação 19) foi de 0,7809 (RTRL) e 0,7220 (EKF).

Utilizando EKF obteve-se uma alocação de taxa média de 4,841mx/s, onde mx=89.184 bytes. Da mesma forma que para o algoritmo RTRL, alocando uma taxa de transmissão única teríamos uma perda de 1.217.700 bytes de tráfego. Utilizando alocação dinâmica 526.450 bytes são perdidos (Figura 5).

Variando o tamanho do buffer, porém mantendo a mesma configuração da rede neural e do sistema, podemos relacionar a taxa de perda de bytes e o número de bytes máximo armazenado no buffer (Figura 6 ).

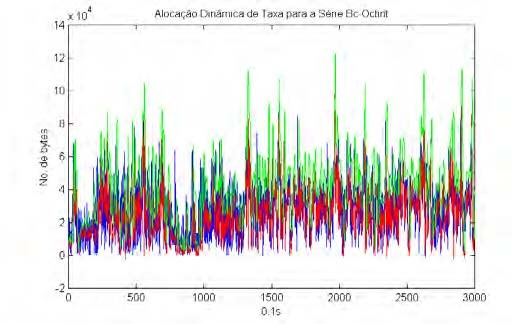


Fig.4. (Vermelho) Predição de 4 instantes à frente. (Azul) Série real. (Verde) Taxas de transmissão alocadas a cada 4 instantes

Quanto a ocupação do buffer, os seguintes dados foram obtidos:

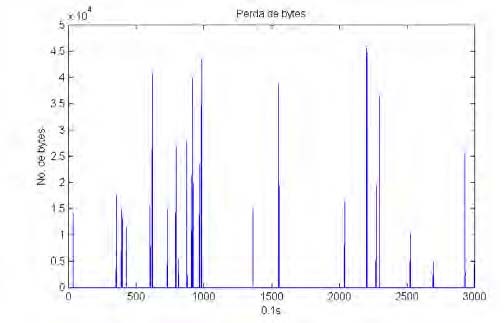
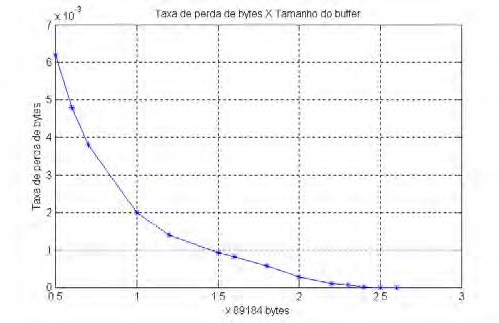
Fig. 5. Número de bytes perdidos a cada 0,1s com alocação dinâmica de taxa (EKF)

TABELA 2 TRÁFEGO NO BUFFER

|  |  |  |
| --- | --- | --- |
|  | **RTRL** | **EKF** |
| Qmax (máximo número  de bytes no buffer) | 175.570 bytes | 207.590 bytes |
| Qmed (número médio  de bytes no buffer) | 8.058 bytes | 6.851,5 bytes |
| Varq (variância do número de bytes no  buffer) | 4,0133.108 bytes | 3.7774108 bytes |

Com o método proposto, obteve-se uma alocação de taxa média de 4,7777mx/s, onde mx=89.184 bytes. Se tivéssemos alocado essa taxa de transmissão média, mas fazendo-a única e constante teríamos uma perda de 1.340.400 bytes. Utilizando alocação dinâmica 857.150 bytes são perdidos. Além de uma menor perda de bytes, há uma melhor utilização do sistema, podendo por exemplo, disponibilizar quando possível, maiores taxas para outras conexões e alocar o necessário para o escoamento do tráfego considerado.

Fig.6. Relação entre a taxa de perda de bytes e o tamanho do buffer (EKF)

1. CONCLUSÕES

Com o treinamento em tempo real de redes neurais recorrentes busca-se além de diminuir os dados de treinamento, atualizar os pesos da rede neural de forma a lidar com variações estatísticas do tráfego que possam acontecer, para as quais redes neurais com treinamento ‘on batch’ poderiam não estar preparadas.

Utilizando o esquema de controle de taxa de transmissão proposto, deve-se levar em conta o tamanho do buffer, as taxas disponíveis, aí então, configurar a rede neural de forma a ter uma menor perda de bytes possível através de um adequado dimensionamento do buffer (Figura 6).

Notou-se que a taxa de aprendizagem da rede neural treinada com algoritmo RTRL influencia na alocação de taxa de transmissão e assim na perda de bytes. Uma estimativa de taxa de transmissão mais elevada devido a uma maior taxa de

aprendizagem, leva a menores perdas de bytes, mesmo com um maior erro de predição.

Os algoritmos propostos mostraram segundo a análise realizada, serem ferramentas úteis na otimização dos recursos de uma rede de computadores, uma vez que acarretaram uma maior utilização da rede a uma menor perda de bytes.

Como trabalho futuro fica a computação em tempo real de um intervalo de confiança das predições e sua inclusão na alocação de taxa a fim de garantirmos uma margem de segurança mais precisa do que o fator G da equação 18.

1. REFERÊNCIAS
2. Z. Sahinoglu, S.Tekinay, “On Multimedia Networks: Self-similar Traffic and Network performance”, IEEE Communications Magazine, January, 1999;
3. D.Veitch, P. Abry, P, “A wavelet based joint estimator of the parameters of long-range dependence,” IEEE Trans. Inform. Theory–Special Issue on Multiscale Statistical Signal Analysis and Its Applications, vol. 45, no. 3, Apr. 1999;
4. D.Veitch, P.Abry, “Wavelet analysis of long-range dependent traffic,” IEEE Trans. Info. Theory, vol. 44, no. 1, pp. 2–15, 1998;
5. R.H.Riedi, M.S.Crouse, V.Ribeiro e R.G.Baraniuk, “A multifractal wavelet model with application to network traffic” IEE Trans. Info. Theory, vol. 45, no.3, pp.992 –1018, Abril 1999.
6. K.Park, K, G. Kim, e M.Crovella, “On the Relation between File Sizes, Transport Protocols and Self-similar Network Traffic”, Proc.IEEE Int’l. Conf. Network;
7. K.Park, G.Kim, e M.Crovella, “On the Effect of Traffic Self-similarity on Network Performance”, Proc.SPIE Int’l. Conf. Perf. And Control of Network Sys., 1997, pp.296-310;
8. T.Tuan, K.Park, “Congestion Control for Self-similar Network Traffic”, Dept.of Comp. Science, Purdue Univ., CSD-TR 98-014, Maio, 1998;
9. W.Stallings, “High Speed Networks: TCP/IP, ATM Design Principles”, Prentice Hall, 1998, pp.181-207;
10. J.W.Roberts, “Engineering for quality of service” , Self-similar network traffic and performance evaluation, John Wiley & Sons, 2000;

[10] S.Chong, e S.Li, “Predictive dynamic bandwidth allocation for efficient transport of real-time VBR vídeo over ATM”, IEEE JSAC, vol. 13, no.1, Jan. 1995, pp.12-23;

[11] S. Jagannathan e J.Talluri. “Adaptive Predictive Congestion Control of High-Speed ATM Networks”. IEEE Transactions on Broadband, Vol.48, No.2, June 2002;

[12] Bor-Sen Chen, Sen-Chueh Peng, e Ku-Chen. “Traffic Modeling, Prediction and Congestion Control for High-Speed Networks: A Fuzzy AR Approach. IEEE Transactions in Fuzzy Systems, Vol. 8, No.5, October 2000;

[13] A.Adas,“Supporting real time VBR video using dynamic reservation based on linear prediction”, Proc.IEEE INFOCOMM’96, 1996, pp.1476- 1483;

1. A. Adas, e A.Mukherjee, “On resource management and QoS guarantees for long range dependent traffic”, Proc.IEEE INFOCOMM, April 1995, pp.779-787;
2. E.A.Wan, “Time series prediction by using a connectionist network with internal delay lines” In Time Series Prediction: Forecasting the Future and Understanding the past, Addison –Wesley, 1994, pp.195- 217;
3. A.Aussem, F.Murtag, M. Sarazin, “Dynamical Recurrent Neural Networks- towards environmental time series prediction”, International Journal on Neural Systems, 6, 1995, pp.145-170,;
4. A. Aussem, F.Murtag, “Combining Neural Network Forecasts on Wavelet-Transformed Time Series” ,Connection Science 9, 1997, pp.113-121;
5. Y.Bengio, P.Frasconi, M.Gori, “Recurrent Neural Networks for Adaptive Temporal processing”, Universitá di Firenze, Julho 1993;
6. R.J.Williams, D.Zipser, “A learning algorithm for continually running fully recurrent networks” Neural Computation 1, 1989, pp.270-280;
7. R.J.Williams, D.Zipser, “An efficient gradient-based algorithm for on- line training of recurrent network trajectories” Neural Computation 2, 1989, pp. 490-501;
8. S.Haykin. “Neural Networks - A Comprehensive Foundation”. Prentice Hall. 1994;
9. R.E.Kalman, “A New Approach to Linear Filtering and Prediction Problems,” Transaction of the ASME - Journal of Basic Engineering, Março, 1960, pp.35-45;
10. S.Singhal, L.Wu, “Training multilayer perceptrons with the extended Kalman filter algorithm”. Advances in Neural Information Processing Systems, 1989, pp. 133-140;
11. R.J.Williamns, “Training Recurrent Networks Using the Extended Kalman Filter”. Technical Report . Boston: Northestern University, College of Computer Science, 1992;
12. G.V.Puskorius, e Feldkamp, L.A. “Decoupled extended Kalman filter training of feedforward layered networks”. Proceedings of the International Joint Conference on Neural Networks, 1991, pp.771-777;
13. G.V.Puskorius, e Feldkamp, L.A. “ Neurocontrol of nonlinear dynamical systems with Kalman filter trained recurrent networks”, IEEE Transactions on Neural Networks, vol 5, Março, 1994, pp.279-297;
14. M.B.Mathews, “Neural networks nonlinear adaptive filtering using the extended Kalman filtering algorithm”. Proceedings of the International Neural Networks Conference, vol.1, Paris, Julho, 1990, pp115-119;
15. R.J.Williams. “Training Recurrent Networks Using the Extended Kalman Filter”. Proceedings of the International Joint Conference on Neural Networks, Vol. IV, pp.241-246, Baltimore, 1992;
16. F.H.T.Vieira “Predição de tráfego em redes de comunicações utilizando redes neurais e análise wavelet- Alocação dinâmica de largura de faixa”. Dissertação de mestrado. Universidade Federal de Goiás, Goiânia, Goiás, Brasil.
17. F.H.T.Vieira, R.P.Lemos, L.L.Lee. “Aplicação de Redes Neurais RBF Treinadas com Algoritmo ROLS e Análise Wavelet na Predição de Tráfego em Redes Ethernet”. Proceedings of the VI Brazilian Conference on Neural Networks- pp.145-150, June 2-5, 2003 -SP- Brasil.
18. BIOGRAFIAS

**Flávio Henrique Teles Vieira** nasceu em Barra do Garças (MT), Brasil, em 1978. Recebeu o título de bacharel em Engenharia Elétrica pela Universidade Federal de Goiás (UFG) em 2000 e o título de Mestre em Engenharia Elétrica pela Escola de Engenharia Elétrica da UFG em 2002. É aluno de doutorado na Faculdade de Engenharia Elétrica e Computação da Universidade Estadual de Campinas (FEEC-UNICAMP). Atualmente atua nas seguintes áreas de pesquisa: Tráfego em redes de

computadores, Processamento de Sinais e Redes Neurais.

**Rodrigo Pinto Lemos** nasceu em Cássia-MG em 1969. Obteve os títulos de Engenheiro Eletricista pela Universidade Federal de Goiás (1992), Mestre e Doutor em Engenharia Elétrica pela UNICAMP, na área de Eletrônica e Comunicações - Processamento de Sinais (respectivamente em 1995 e 1997). Desde 1996 atua como professor da Escola de Engenharia Elétrica da Universidade Federal de Goiás, atualmente no cargo de Professor Adjunto. De 1998 a 2000 foi Subcoordenador de Pós-Graduação da EEE/UFG. Seus interesses de pesquisa envolvem Processamento Estatístico de Sinais, Previsão, Modelagem e Controle de Tráfego.

**Luan Ling Lee** obteve o título de engenheiro eletricista pela USP em 1980, mestre em Engenharia Elétrica pela UNICAMP em 1984 e PhD em Engenharia Elétrica pela Universidade de Cornell em 1991. Foi fundador e tem sido o coordenador do Laboratório de Reconhecimento de Padrões e Redes de Comunicações (LRPRC) da FEEC-UNICAMP desde 1994. Foi o chefe do Departamento de Comunicações em 2000. Desde 2002 ele tem sido Professor Titular pela FEEC-

UNICAMP. Atualmente ele atua intensamente em duas áreas de pesquisa: Reconhecimento de Padrões e Redes de Comunicações.