Proposta e Avaliação de uma Política de Preempção de LSPs Implementada com Lógica *Fuzzy* e Algoritmos Genéticos em um Ambiente de Rede DiffServ/MPLS

Rodrigo Campos Vieira , Paulo Roberto Guardieiro

Laboratório de Redes de Computadores
Faculdade de Engenharia Elétrica
Universidade Federal de Uberlândia
Av. João Naves de Ávila, 2121
Tel: (34) 3239-4165 Fax: (34) 3239-4246
Uberlândia, Minas Gerais, Brasil

{rodrigo, guardieiro}@lrc.eletrica.ufu.br

Resumo. *O crescente uso de aplicações de multimídia e de tempo real na Internet requer soluções baseadas em reservas de recursos para prover qualidade de serviço. Porém, o uso ineficaz dos recursos prejudica tais soluções. Assim, torna-se necessário o estabelecimento de estratégias de engenharia de tráfego integradas com propostas de qualidade de serviço, visando resolver este problema. Uma proposta de solução a tal problema emprega a preempção para evitar recursos ociosos no domínio buscando atender outros tráfegos de maior prioridade. Neste artigo, propõe-se e avalia-se uma política de preempção de LSPs implementada com um controlador fuzzy. A validade da proposta é demonstrada a partir de resultados obtidos em um estudo experimental.*

Abstract. *The increasing use of multimedia and real time applications in the Internet requires solutions based on resource reservation to provide quality of service. However, the ineffective use of these resources can reduce the performance of such solutions. Thus, in order to solve this problem, it would be necessary to establish traffic engineering strategies integrated with proposals of quality of service. A proposal of solution to such problem adopts the preemption approach to prevent idle resources in the domain, searching to take care of other traffics of higher priority. In this paper, it is proposed and evaluated a LSP preemption policy implemented with a fuzzy controller. The validity of the proposal is demonstrated by means of the results obtained in an experimental study.*

1. Introdução

Nos últimos anos, a Internet tem criado espaço para um conjunto de serviços, os quais visam o perfil e as necessidades de seus usuários e dão suporte a um certo número de aplicações interativas e de tempo real. A diferenciação de serviços (*DiffServ - Differentiated Service*) [Blake., 1998] é uma proposta que visa oferecer garantias de qualidade de serviço (QoS - *Quality of Service*), as quais são extremamente importantes para estas exigentes aplicações. O *DiffServ* possui a capacidade de agregar os fluxos de dados em diferentes classes de serviço, realizando isto nas extremidades do domínio e produzindo uma solução escalável. Apesar disso, podem ocorrer situações de congestionamento que comprometem a QoS oferecida.

Analisando sob o ponto de vista da utilização de recursos da rede, torna-se necessário o desenvolvimento de técnicas empregando Engenharia de Tráfego (TE - *Traffic Engineering*), as quais assegurem que os recursos sejam utilizados otimizadamente. Neste sentido, surge a arquitetura MPLS (*Multiprotocol Label Switching*) [Awduche., 1999, Rosen et al., 2001], com o intuito de simplificar e prover melhorias no processo de roteamento. Nesta arquitetura, insere-se um rótulo (*label*) em cada pacote que ingressa no domínio, a fim de que este seja analisado e o roteamento baseado neste rótulo seja realizado. Neste processo de encaminhamento de pacotes através do domínio são criados os LSPs (*Label Switched Paths*), os quais têm funcionamento semelhante aos circuitos virtuais do ATM (*Asynchronous Transfer Mode*) e podem ser utilizados para inferir a respeito dos requisitos de qualidade de determinado fluxo em um LSP. Tal fato só é possível caso exista um mecanismo de provimento de QoS integrado, pois o MPLS, por si só, não é capaz de assegurar a QoS. Dentre as integrações, uma possibilidade é aquela que une as arquiteturas *DiffServ* e MPLS [le Faucher et al., 2002], a qual tem se mostrado bastante eficaz, conforme demonstra a literatura. Assim como o congestionamento influi no *DiffServ*, uma quantidade excessiva de LSPs criados pode ocasionar, conforme o algoritmo utilizado, a escassez de recursos em um determinado enlace enquanto existem recursos ociosos nos demais. Tal fato evidencia uma das limitações encontradas neste tipo de integração e que tem motivado estudos a respeito. Dentre estes, encontramos estudos analíticos buscando a criação de modelos e estudos baseados em simulação com o intuito de propor soluções. Outra possibilidade que vem surgindo são os estudos baseados em emulação [Leu and Casellas, 2004], que visam a implementação de um ambiente de testes e a realização de experimentos com dados reais.

Na pesquisa bibliográfica realizada a respeito da limitação anteriormente citada, encontrou-se a proposta de [Fernandez et al., 2003], onde apresenta-se a arquitetura de um sistema de gerenciamento baseado em políticas que coordena o provimento dinâmico dos nós de um domínio *DiffServ*. Outra proposta, o projeto Tequila [Trimintzios, 2001], utiliza um protocolo de sinalização que muda o controle de admissão do domínio conforme a entrada de tráfego, de acordo com um levantamento estatístico feito anteriormente para modelar este tráfego. Além disso, em [de Oliveira, 2003] desenvolve-se uma nova política de preempção ponderada com um esquema adaptativo que minimiza o re-roteamento. Em [Rosenbaum et al., 2003] ilustra-se a construção de um *testbed* MPLS usando computadores pessoais (PCs) e também o sistema operacional Linux, bem como apresenta-se resultados experimentais quanto à separação de fluxos obtida. Em [Heuven et al., 2004] apresenta-se o desenvolvimento de um módulo MPLS para instalação no *kernel* do sistema operacional, a fim de permitir a criação de um domínio MPLS experimental.

Neste trabalho, propõe-se e avalia-se a implementação de uma política de preempção de LSPs. Aplica-se tal política em um ambiente experimental de emulação capaz de prover diferenciação de serviços e Engenharia de Tráfego (DS-TE - *DiffServ-aware MPLS Traffic Engineering*), o qual é implementado utilizando-se PCs convencionais e OS Linux. Neste cenário, é especificada uma função que modela como o sistema pode ser afetado por políticas de preempção presentes na literatura. A etapa seguinte consiste em otimizá-la através da aplicação de uma técnica de busca heurística, tal como os Algoritmos Genéticos (GA - *Genetic Algorithms*). Realiza-se, então, a aplicação das regras definidas pela otimização das políticas de preempção ao domínio DS-TE alterando-se os parâmetros de configuração do domínio, modificando o processo de criação de LSPs, por meio de um controlador lógico fuzzy (FLC - *Fuzzy Logic Controller*) que atua de acordo com o comportamento do tráfego.

Este artigo está estruturado da forma descrita a seguir. Na seção 2 apresenta-se de

forma sucinta uma descrição do problema abordado e a solução proposta neste trabalho; na seção 3 apresentam-se as ferramentas empregadas neste trabalho a fim de otimizar a qualidade de serviço num ambiente *DiffServ*/MPLS; na seção 4 ilustra-se a topologia de rede em estudo e descreve-se o seu funcionamento; na seção 5 apresentam-se análises acerca dos resultados obtidos; finalmente, na seção 6 relatam-se as observações finais e as conclusões deste trabalho.

2. Motivação

2.1. Descrição do Problema

Dentro de um domínio MPLS, cada LSP passa pelo estágio de inicialização de forma independente, pois nenhum LSP tem conhecimento dos recursos disponíveis aos demais LSPs. Assim, quando ocorre a inicialização de novos LSPs em um domínio MPLS, pode ocorrer a subutilização de recursos nos troncos de tráfego, caso não estejam disponíveis recursos suficientes para atender aos requisitos de QoS dos LSPs inicializados, independente do nível de prioridade atribuído. Isto ocorre porque ao se estabelecer um tronco de tráfego é definida também uma largura de banda (BW - *Bandwidth*) máxima a ser utilizada por este tronco e esse limite nem sempre é condizente com as necessidades de um determinado LSP a todo instante. Isto significa que a garantia aos requisitos de QoS das aplicações de multimídia e de tempo real não será atingida devido à utilização inadequada da BW dentro de um domínio MPLS.

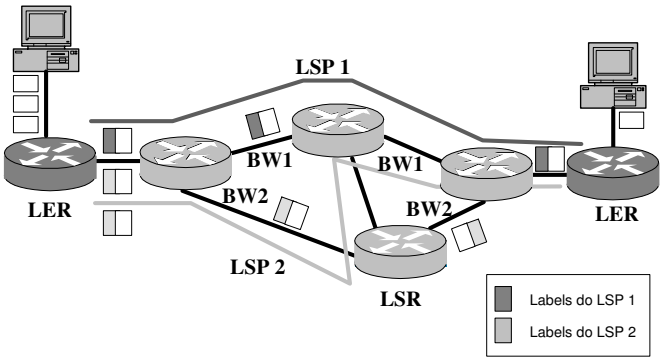

Figura 1: Ilustração da criação arbitrária de LSPs

Na Figura 1 considera-se um domínio *DiffServ*/MPLS, representando um AS (*Autonomous System*), onde os pacotes são rotulados no LER (*Label Edge Router*). As BWs dos enlaces estão apresentadas como BW1 e BW2. Foram criados os LSPs 1 e 2. Caso cheguem ao domínio novos fluxos exigindo a criação de um novo LSP com requisitos de QoS que não possam ser atendidos pelos enlaces, eles acabarão sendo descartados ou aguardarão pelo atendimento, mesmo que possuam níveis de prioridade maior do que os existentes. Assim, torna-se necessário um mecanismo capaz de gerenciar tais recursos eficientemente.

2.2. Solução Proposta

A questão da criação de LSPs de forma arbitrária podendo afetar o desempenho global no AS pela inexistência de recursos disponíveis é, na verdade, um problema pertencente à classe do controle. Ou seja, devido à ausência de algum mecanismo de controle condizente com o comportamento do sistema, não se obtém o resultado esperado. Neste caso, seria suficiente aplicar a metodologia de controle clássico sobre um modelo do sistema e a resposta adequada poderia ser obtida ao se ajustar o controlador utilizado.

Apesar de parecer viável, tal solução teria como obstáculos a característica de incerteza e imprecisão oferecida ao se tentar modelar os fluxos de dados da Internet e a necessidade de um profissional qualificado no ajuste do controlador. O primeiro obstáculo justifica a utilização do FLC, devido à não linearidade e à ausência de um modelo matemático preciso para estimar o tráfego da Internet. Ainda assim, continua existindo o segundo obstáculo, pois o FLC também precisa ser ajustado por um especialista. Sendo assim, tornou-se viável a utilização de uma ferramenta de busca heurística, tal como o GA, tendo em vista que ela é capaz de configurar o FLC com parâmetros ótimos. Assim, com essa proposta tem-se uma política de priorização de LSPs, de acordo com o comportamento dos fluxos de dados no AS.

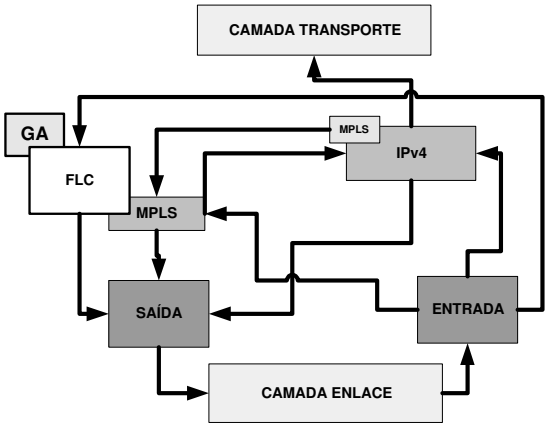

Figura 2: Estrutura de implementação da proposta

A Figura 2 apresenta a estrutura implementada nos computadores do ambiente experimental, ou seja, a proposta de solução. Essa Figura representa os módulos presentes no *kernel* do OS Linux que permitem que os serviços de rede sejam oferecidos, sendo eles a camada de enlace, camada de rede e a camada de transporte da arquitetura TCP/IP. Ilustram-se as camadas de enlace e transporte de forma simplificada, pois as mesmas não pertencem ao escopo do trabalho, sendo a camada de transporte representada pelo protocolo TCP. Além disso, descreve-se a camada de rede de forma a evidenciar os elementos que a compõem dentro do *kernel* do OS Linux, tais como o protocolo IPv4, o módulo MPLS instalado e as portas de entrada e saída. A proposta consiste no módulo FLC agindo como um configurador do módulo MPLS, sendo este último quem determina a política adotada na inicialização de LSPs. Apresenta-se também o módulo GA que aparece como uma entidade que interfere programando o FLC, mas que devido a problemas de escalabilidade não pode ser implementada em cada nó da rede.

Em um ambiente de rede que possui suporte a MPLS, existem dois tipos de datagramas que podem chegar à camada de rede: datagramas IP tradicionais ou datagramas com rótulo MPLS. Quando tratam-se de datagramas IP tradicionais, o processo lógico dentro da camada de rede inicia-se com a sua passagem pelo módulo de entrada, o qual o encaminha para o módulo IPv4. Neste módulo IPv4, retira-se o cabeçalho e avalia-se o endereço de destino. Faz-se, então, a determinação da rota a ser seguida e remonta-se o datagrama, enviando-o para o módulo de saída. Por outro lado, se o datagrama possui o rótulo MPLS, encaminha-se diretamente do módulo de entrada para o módulo MPLS, o qual faz a extração do cabeçalho. O módulo MPLS encontra-se sobre a supervisão do FLC, o qual configura os parâmetros para a inicialização e preempção dos LSPs de acordo com o tráfego no ambiente de rede. Assim sendo, o módulo MPLS faz o encapsulamento do datagrama com o novo rótulo e cria um LSP, de acordo com as políticas configuradas, para transportar este datagrama. Por fim, o datagrama é enviado para o módulo de saída,

o qual devolve-o para a camada de enlace do ambiente de rede.

3. Técnicas Empregadas no Desenvolvimento da Proposta

3.1. Preempção

A preempção é uma técnica que permite o gerenciamento otimizado de recursos em ambientes com pouca disponibilidade destes. Em sistemas preemptivos, um processo pode ser retirado da CPU (*Control Process Unit*) devido à chegada de um processo de maior prioridade, mesmo que não tenha terminado a sua execução [Baker, 1993]. O objetivo, na proposta apresentada, é maximizar a utilização dos recursos da rede enquanto minimiza-se o número de conexões que teriam seu acesso à rede negado devido à insuficiência de recursos.

Em [Awduche., 1999], os autores apresentam os requisitos para utilização de TE em um domínio MPLS e destacam a necessidade de parâmetros de prioridade e preempção como atributos de TE de um tronco de tráfego. Para isto define-se aqui a Classe de Tráfego (CT - *Class of Traffic*), que representa o conjunto de troncos de tráfegos atravessando um enlace, onde um conjunto específico de restrições de largura de banda é definido. A integração DS-TE pode suportar até 8 CT ($CT_c, c = 0, \ldots, 7$) e, por definição, cada CT é atribuído a uma restrição de largura de banda (BC - *Bandwidth Constraint*) ou a um conjunto de tais restrições ($BC_b, b = 0, \ldots, 7$). Segundo o RDM (*Russian Doll Model*), em padronização pelo IETF [le Faucher et al., 2003], tem-se definido que o número máximo de BC é igual ao de CT e todos LSPs pertencentes a CT_c não devem usar mais do que BC_b (sendo $b \leq c \leq 7$ e $BC_b \leq BC_{b-1}$, para $b = 1. \ldots, 7$). Ainda em [Awduche., 1999], foram definidos os dois atributos de preempção, s (*setup preemption priority*) e h (*holding preemption priority*), os quais podem assumir diferentes valores e controlar a dinâmica da criação dos LSPs. Dessa forma, utilizam-se estas definições para criar uma classe de TE (TE-Class) e estabelecer políticas de preempção para atuarem. Tais políticas podem realizar:

- preempção nas conexões de menor prioridade, o que determina que a QoS dos tráfegos de maior prioridade será satisfeita.
- preempção do menor número de LSPs, o que determina um número menor de LSPs re-roteados.
- preempção da menor quantidade de largura de banda que satisfaça a requisição, o que determina melhor utilização dos recursos.

A escolha de uma dada política determinará a composição da função objetiva a ser otimizada para que a preempção ofereça resultados satisfatórios. Sendo assim, é definida em [de Oliveira, 2003] a seguinte função objetiva:

$$F(\mathbf{z}) = \alpha(\mathbf{z} \cdot \mathbf{y^T}) + \beta(\mathbf{z} \cdot \mathbf{1^T}) + \gamma(\mathbf{z} \cdot \mathbf{b^T}) \quad (1)$$

onde:
$\mathbf{b^T}$ vetor transposto com as larguras de banda reservadas pelos LSP ativos.
$\mathbf{y^T}$ vetor transposto do inverso da prioridade de *holding* de cada LSP.
$\mathbf{z}$ vetor binário que determina qual LSP sofrerá preempção.
α , β e γ pesos que ajustam qual das políticas anteriores será utilizada.

Desta forma, torna-se necessário o emprego de ferramentas capazes de otimizar tal função para a utilização eficiente dos recursos do domínio. Neste processo, necessita-se encontrar o mínimo global da função, de acordo com os critérios das políticas de preempção estabelecidas.

3.2. Lógica *Fuzzy*

A lógica difusa (*fuzzy logic*) permite classificar as respostas às experiências humanas que não podem ser classificadas simplesmente como verdadeiras ou falsas em diferentes níveis de verdade baseada numa linguagem natural [de Caluwe, 1997]. Por isso, a lógica difusa, com base na teoria dos conjuntos nebulosos (*fuzzy set*), tem se mostrado adequada para tratar problemas de controle que possuam imperfeições da informação como estas apresentadas. Sendo assim, trata-se de uma ferramenta capaz de capturar informações vagas e convertê-las para um formato numérico, de fácil manipulação pelos computadores. Alguns estudos vêm sendo realizados a respeito da incerteza do estado de uma rede para garantir a QoS das conexões. Trabalhos como [Lorenz and Orda, 1998] demonstram que quando a conexão exige apenas largura de banda, a possibilidade de se prever o comportamento da rede é maior do que quando são necessárias garantias de limites de atraso fim a fim e de *jitter*, pois o efeito da incerteza da rede é mais perceptível. No entanto, apresenta-se como solução a decomposição dos requisitos de atraso em restrições locais e a definição de uma classe de distribuição de probabilidade, tornando a solução exata e eficiente. Problemas dessa natureza, os quais envolvem incerteza e imprecisão, acabam por permitirem uma solução baseada na lógica *fuzzy*, através da implementação de um FLC [Cheng and Chang, 1996].

Em vista disso, implementa-se um FLC no ambiente de rede em estudo cujas funções de pertinência estabelecem sobre quais LSPs deve-se realizar a preempção de acordo com a equação 1, definida anteriormente. Tal processo poderia ser implementado diretamente no FLC, mas devido à necessidade de escalabilidade e de resposta à dinâmica do comportamento dos fluxos no AS, tal trabalho é destinado ao GA, uma ferramenta de pesquisa heurística.

3.3. Algoritmos Genéticos

Os algoritmos genéticos propõem a modelagem de soluções baseada na teoria da evolução e na genética. Segundo as observações de Darwin, os indivíduos mais aptos a sobreviverem em um certo ambiente tem a característica de incorporar as adequações utilizadas para este fim à sua informação genética e transmiti-la através das diversas gerações de uma população. Assim, os algoritmos genéticos propõem a criação de uma população, a qual representa um conjunto de candidatas à solução do problema avaliado. Esses indivíduos são selecionados de acordo com sua aptidão e geram uma nova população, através de operações genéticas [Beasley et al., 1993]. O processo é iterativo até que se determine os indivíduos habilitados para fornecerem uma solução ótima de acordo com sua aptidão. Existem diversas aplicações usando algoritmos genéticos, mas uma que tem merecido destaque é a integração dessa técnica para o treinamento de sistemas nebulosos (*fuzzy*) através da busca heurística de soluções ótimas.

A primeira etapa num algoritmo genético é a determinação da representação do problema em estudo, aquele que se deseja otimizar. A etapa seguinte é a seleção, a qual pode ser um sorteio aleatório e, logo após isso, são aplicados os operadores genéticos. O cruzamento é operador genético mais predominante, o qual permite que as características de diferentes elementos de uma população sejam transferidos para a seguinte, através da recombinação destas características. O valor típico de sua probabilidade de ocorrência é de 60% e este foi o valor adotado neste trabalho.Em contrapartida, a mutação é operador genético que ocorre com menos freqüência, mas que desempenha um papel importante, não deixando que a busca pare em ótimos locais, ou seja, melhorando a diversidade da população. Os valores usuais de sua probabilidade de ocorrência encontram-se entre 1% e 5%.

Como foi apresentada anteriormente, uma função custo $F(\mathbf{z})$ é responsável por ajustar a minimização da variável $\mathbf{z}$, a qual é modelada aqui como um cromossomo da população e que pode sofrer variações. Desta forma, o cromossomo é composto pelo vetor z, pelo custo global ajustado calculado através da função, pela aptidão média e pela aptidão geral do indivíduo em relação à população. O algoritmo é controlado pelo número de iterações, sendo observado que a marca de cinqüenta iterações tem sido suficiente para a estabilidade dos resultados.

4. Implementação de um Modelo Experimental *DiffServ*/MPLS

Ao se propor uma abordagem visando solucionar algum problema encontrado, torna-se necessária estabelecer qual alternativa adotar. Existe uma grande diferença entre realizar-se um estudo baseado em simulações e baseado em modelos experimentais. No primeiro, elabora-se um modelo matemático capaz de descrever o comportamento do sistema com algumas restrições e são coletados resultados através do uso de alguma ferramenta, quer seja ela matemática ou computacional. No segundo caso, elabora-se um modelo físico em escala reduzida do sistema a ser analisado para que o mesmo possa ser submetido a experimentos reais e possa ser inferido o comportamento do sistema completo. Neste trabalho, optou-se pelo modelo experimental para ser implementado a fim de realizar tais experiências. Tal escolha justifica-se devido a trabalhos existentes, conforme descrito em [Rosenbaum et al., 2003].

4.1. Modelo Experimental

Considerando-se as metodologias propostas para o provimento de QoS em Redes IP, optou-se pela utilização de uma integração *DiffServ*/MPLS, visando obter uma diferenciação de serviços de forma eficiente e uma redução do atraso fim a fim experimentado em Redes IP convencionais. Para tanto, utilizou-se um ambiente de rede simplificado (Figura 3), o qual permite modelar de forma simplificada um AS da Internet. Para a construção deste ambiente de rede utilizaram-se PCs convencionais e o OS Linux, com a finalidade de implementar roteadores. A escolha destes elementos permite um estudo genérico, bem como a minimização de custos.

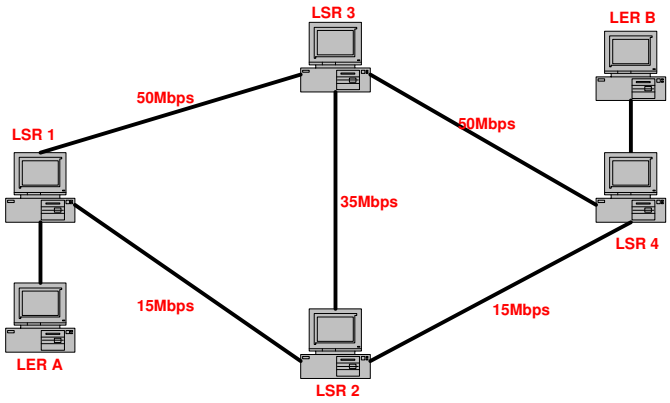

Figura 3: Ambiente de rede

Como pode ser observado na Figura 3, os enlaces foram configurados para apresentar larguras de banda distintas, de forma que o roteamento padrão privilegie aquele com maior capacidade, independentemente de haver ou não congestionamento. Assim, tem-se alocados 50 Mbps no ramo superior e 15 Mbps no ramo inferior.

4.2. Dinâmica do Funcionamento do Modelo

O funcionamento da rede apresentada é baseado na classificação do *DiffServ* utilizando os PHB (*Per-Hop Behavior*) BE, AF e EF [Baker, 1999, Benson, 2002]. Além disso, nesta rede está habilitada a capacidade de preempção de LSPs. A tomada de decisão a respeito da preempção é feita utilizando-se um FLC. Este controlador foi definido utilizando funções de pertinência capazes de determinar a prioridade de cada LSP criado dentro da rede e com uma base de regras estabelecida a partir do conhecimento semântico, o que garante a definição de um controlador correto, mas sem garantia de eficiência. A fim de otimizar os resultados produzidos pelo controlador, empregou-se o GA, visando escolher os melhores parâmetros para o controlador. Interessante notar que, apesar de ser difícil determinar com precisão quais parâmetros ao serem otimizados produzirão o melhor resultado, o algoritmo genético torna-se um processo contínuo de otimização e permite a atualização dinâmica dos parâmetros, de acordo com as mudanças de topologias e padrões de tráfego. A partir dos resultados do GA, o FLC trabalha analisando os pacotes que estão entrando na rede, assim como o estado de utilização que a mesma experimenta. Através destas variáveis e da prioridade dada a cada LSP e por meio dos parâmetros configurados dinamicamente, é possível decidir se a preempção será realizada. Desta forma, tem-se o emprego das ferramentas de inteligência artificial juntamente com a preempção a fim de prover QoS fim a fim no ambiente de rede considerado.

5. Apresentação e Análise dos Resultados

A fim de validar o ambiente experimental proposto foram inicialmente realizados testes para comprovar a capacidade no oferecimento de QoS caso os diversos fluxos pudessem ser separados em troncos de tráfego com características distintas. A metodologia desenvolvida nesse trabalho foi submeter o ambiente a três fontes de tráfego, uma baseada no UDP e duas baseadas no TCP, compará-las em termos da vazão obtida pelos serviços e a associação das mesmas a troncos de tráfego. É importante ressaltar que a fonte com tráfego UDP compartilha o enlace superior do ambiente de rede (50 Mbps) com a fonte com tráfego TCP1, enquanto a fonte com tráfego TCP2 utiliza-se dos recursos do enlace inferior do ambiente de rede (15 Mbps).

Para esta avaliação, foram criados troncos de tráfegos para cada tipo de aplicação e os fluxos de dados foram transportados através destes. Ao contrário de um caminho compartilhado comum, o tronco de tráfego age como um circuito virtual dedicado à conexão. Pode ser observado na Figura 4 que os fluxos TCP conseguem manter-se estáveis de acordo com os valores definidos para cada LSP, enquanto o fluxo UDP trafega em outro tronco de tráfego conseguindo alcançar 60% da largura de banda máxima permitida. Assim, a fonte UDP aumenta sua taxa de geração apenas até atingir este valor e é impedida de experimentar novos aumentos, porque não pode retirar recursos dos outros troncos de tráfego TCP, mesmo que compartilhe o mesmo enlace.

Analisa-se, agora, o ambiente experimental de rede sob diferentes condições de tráfego e, posteriormente, aplicam-se as metodologias de inteligência artificial para alcançar a priorização de LSPs obtida com a preempção. Num primeiro momento é necessário observar o comportamento do ambiente quando sujeito a fontes de tráfego comuns usando os protocolos TCP e UDP, destacando-se a assimilação, em média, elevada da capacidade do enlace apresentada pela aplicação utilizando do protocolo UDP. Comprova-se, pela Figura 5 , que aplicações que utilizam este protocolo, geralmente modeladas como fontes CBR e representando tráfego de tempo real, necessitam de um atendimento bem planejado, pois, caso contrário, podem comprometer as demais aplicações. Nesse cenário, torna-se necessário a aplicação de mecanismos de diferenciação de fluxos,

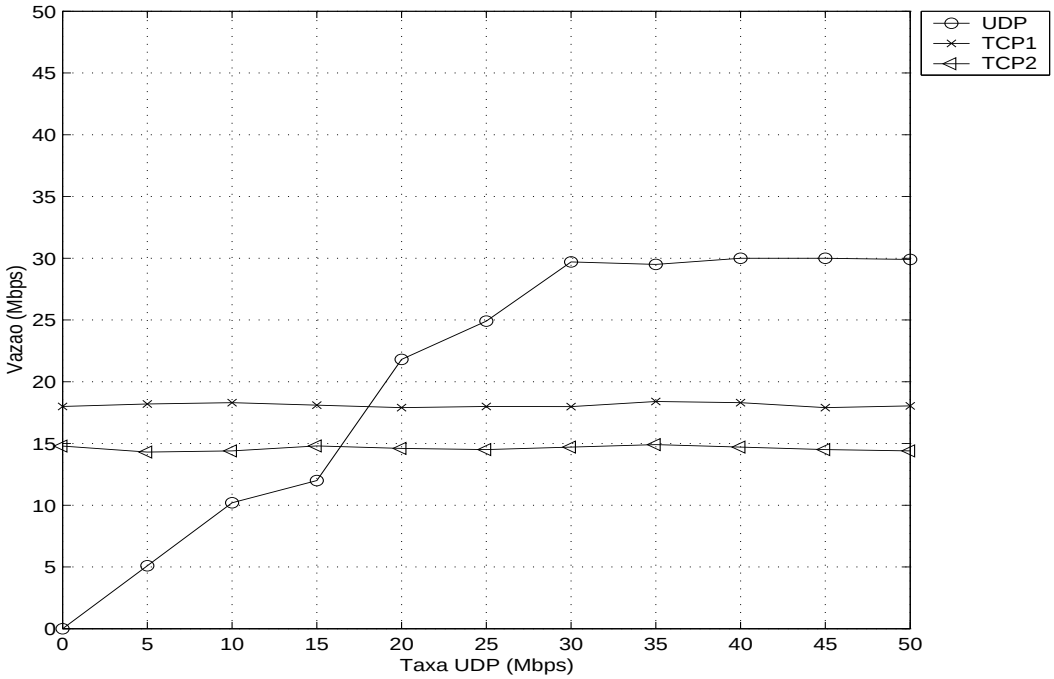

Figura 4: Vazão usando três troncos de tráfego com LSPs distintos

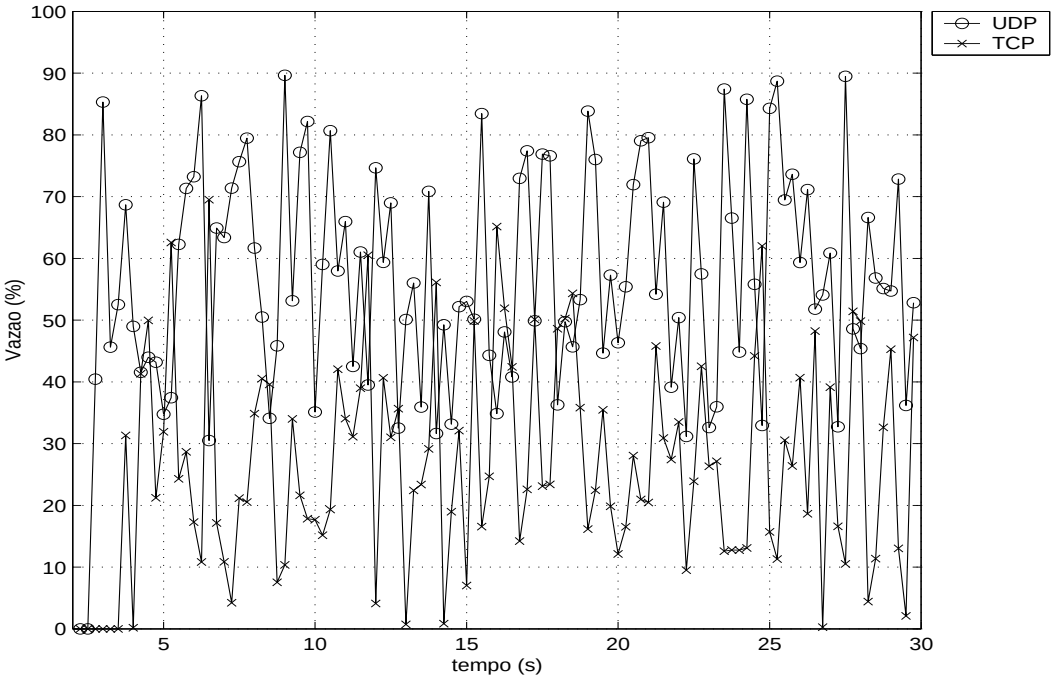

Figura 5: Vazão para tráfego *Best-Effort*

tal como o *DiffServ*, para realizar esta tarefa.

Observa-se, nas Figuras 6 e 7, a não diferenciação de serviços provida por este ambiente, uma vez que o mesmo não possui mecanismos que possibilitem este procedimento ativados. Deve-se considerar, então, que, de acordo com o tipo de aplicação, este comportamento de atraso torna-se indesejável, comprometendo o correto atendimento aos serviços. A Figura 7 (*jitter*, ou variação do atraso) ilustra também este problema evidenciando as possíveis adversidades que aplicações de tempo real enfrentam num AS comum.

A não diferenciação de serviços observada na Figura 5 tem motivado diversos pes-

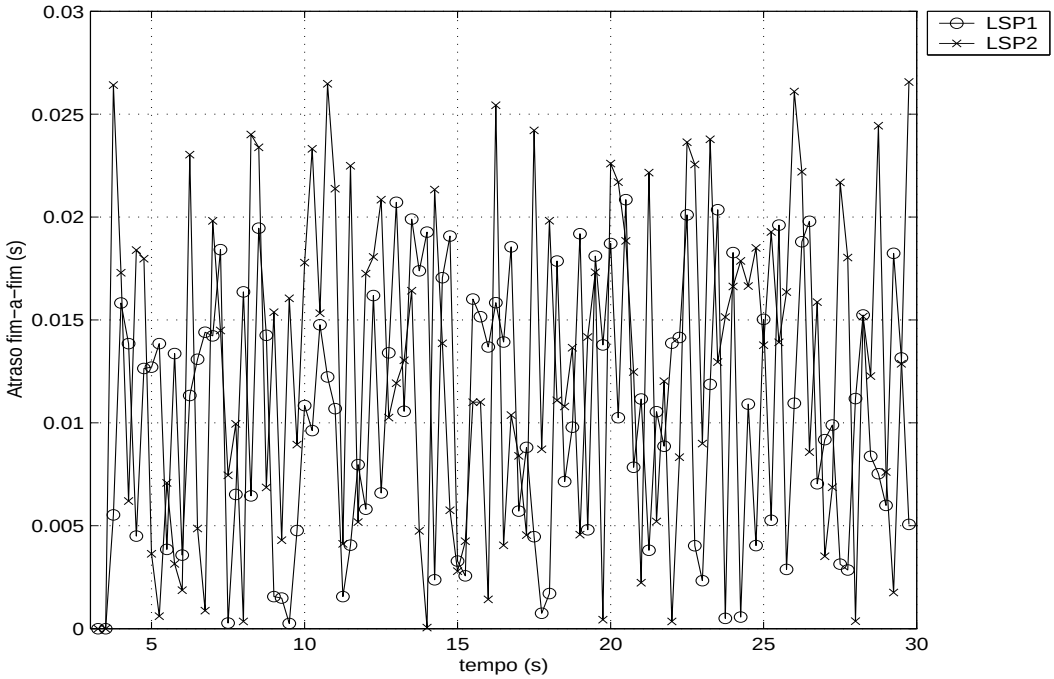

Figura 6: Atraso fim a fim para tráfego *Best-Effort*

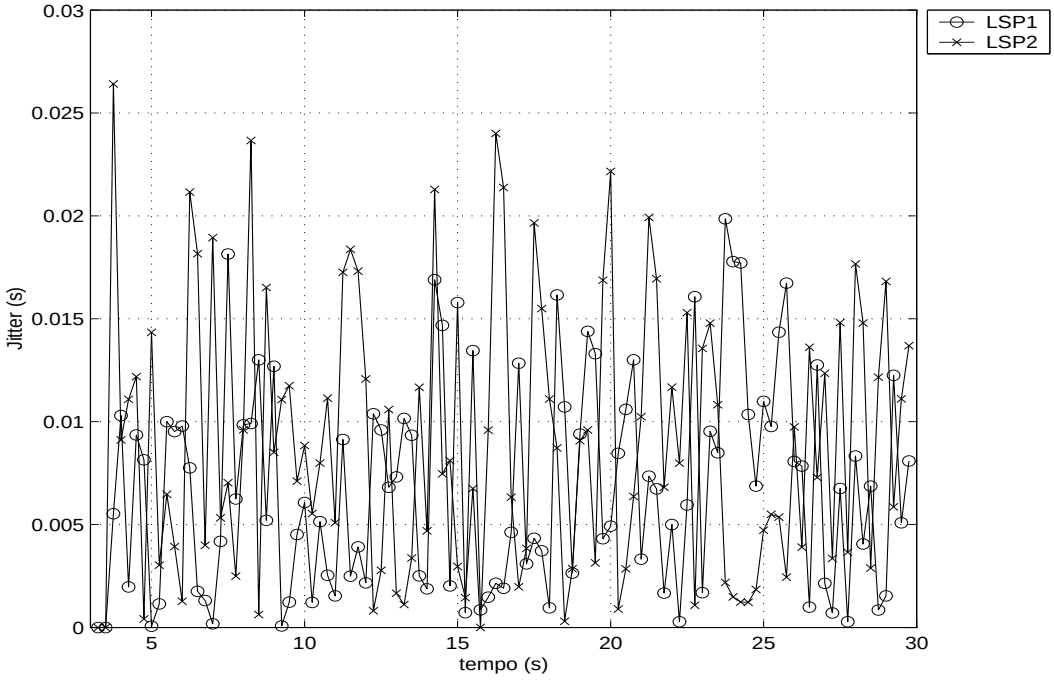

Figura 7: *Jitter* para tráfego *Best-Effort*

quisadores a proporem soluções, tais como o *DiffServ*. Além disso, observou-se também a existência de algumas limitações quanto ao atendimento de novas conexões, fato este que motivou o desenvolvimento de políticas de preempção em um ambiente MPLS. A seguir, considera-se a união da diferenciação de serviços e da engenharia de tráfego, além da implementação de políticas de preempção. Todavia, para a aplicação de tal política é útil um mecanismo que faça o controle de sua atuação, o que conduz ao controle *fuzzy*, pois este se adequa bem a situações onde não há um modelo matemático preciso do processo a ser controlado. Isso levou ao desenvolvimento de um algoritmo genético baseado na função custo de prioridade de LSP (equação 1), permitindo que se determine dinami-

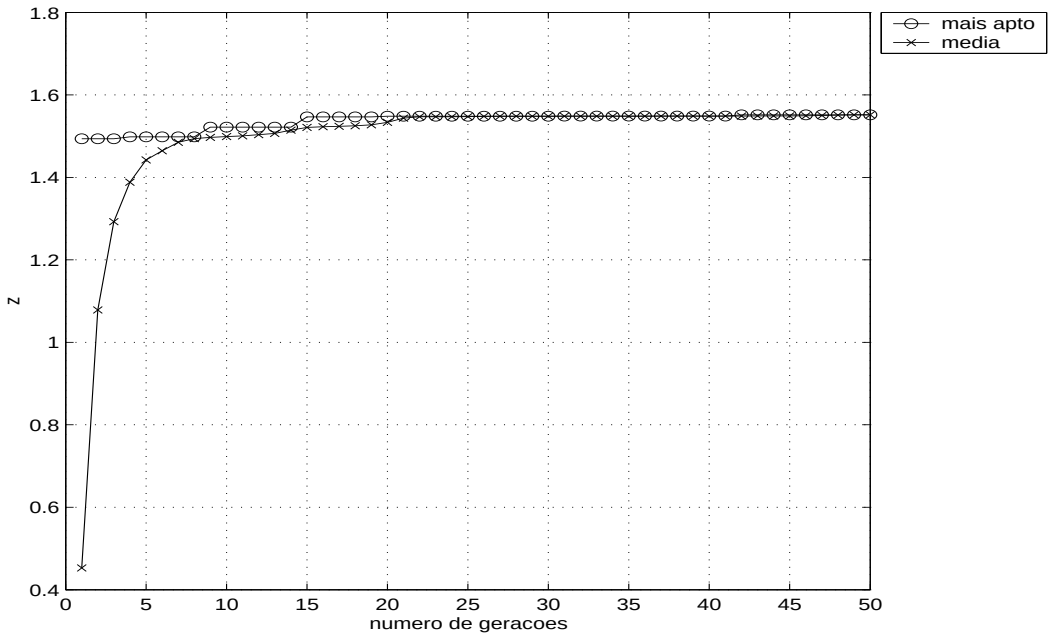

Figura 8: Curva de desempenho do GA

camente quais parâmetros devem ser alterados na função de pertinência do controlador *fuzzy*, permitindo-se alcançar o melhor desempenho no sistema.

A figura 8 ilustra o desempenho do algoritmo genético para uma simulação com parâmetros do ambiente de estudo. Observa-se que o "cromossomo" mais apto para minimizar a função foi obtido num número relativamente pequeno de gerações, considerando-se que o tempo de execução está intimamente ligado à corretude do algoritmo e à escolha da linguagem apropriada. A curva de desempenho ilustra o comportamento dos parâmetros ótimos (x) e da média da população de parâmetros (o), onde os valores do eixo vertical representam a função objetiva escolhida para a configuração da função de pertinência do controlador *fuzzy* com um ajuste de escala. Como esse controlador utiliza uma função de pertinência trapezoidal, devido a maior facilidade na implementação, determinou-se através da execução desse algoritmo a largura da base dessa função de pertinência conforme as condições de tráfego ilustradas na Figura 5 e a maior prioridade dada ao tráfego gerado pela fonte presente no LSP3 (Figura 9). É necessário visualizar-se o resultado produzido no ambiente experimental para que seja possível concluir a respeito da corretude do método.

A Figura 9 ilustra o resultado obtido ao se atribuir ao fluxo LSP3 maior prioridade, inclusive sobre o LSP1, levando-o, teoricamente, a se apropriar de maior quantidade de LSPs e a utilizar a largura de banda que antes era utilizada pelo tráfego LSP2. Deve-se notar que, praticamente no instante que a fonte consegue recursos e inicia a transmissão de informações no AS, LSP3 se apropria da largura de banda do tráfego LSP2, o qual ficou impossibilitado de enviar pacotes e experimentou altos índices de perda. Portanto, o tráfego LSP3 utilizou da melhor forma a capacidade do enlace e acabou obtendo vazão maior do que o tráfego pertencente ao LSP1 neste intervalo de tempo. Esse cenário é interessante do ponto de vista de aplicações que necessitam o atendimento imediato, não podendo sofrer da falta de recursos. A solução é perfeitamente viável para casos de aplicações que ocorram por um curto intervalo de tempo, mas mesmo assim, outros tráfegos sempre serão prejudicados. O nível de QoS contratado pelo usuário é quem ditará de quem é a preferência.

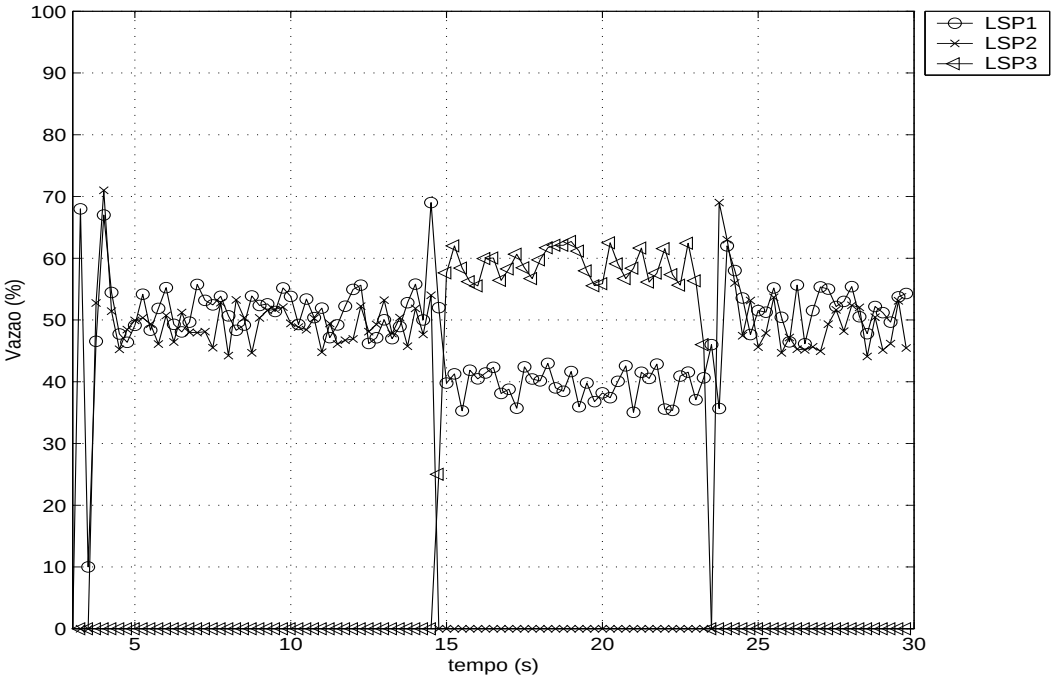

Figura 9: Vazão considerando o ambiente de rede preemptivo

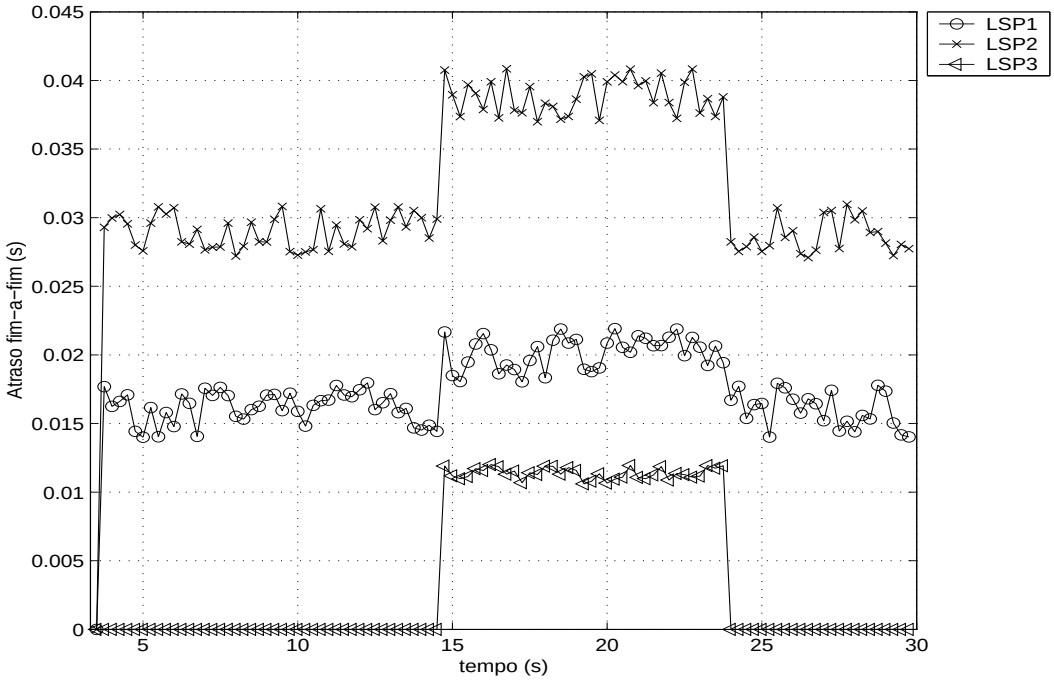

Figura 10: Atraso fim a fim considerando o ambiente de rede preemptivo

Além disso, percebe-se que o atraso fim a fim e o *jitter*, parâmetros essencialmente importantes para aplicações de tempo real, conseguem uma redução em suas amplitudes e passam a ter um comportamento menos sujeito a variações, como observa-se nas Figuras 10 e 11. Isto permite concluir que o método de preempção possui aplicabilidade, mas é necessário conhecimento do comportamento do tráfego no AS para determinar em que situações deve-se empregar as políticas de preempção.

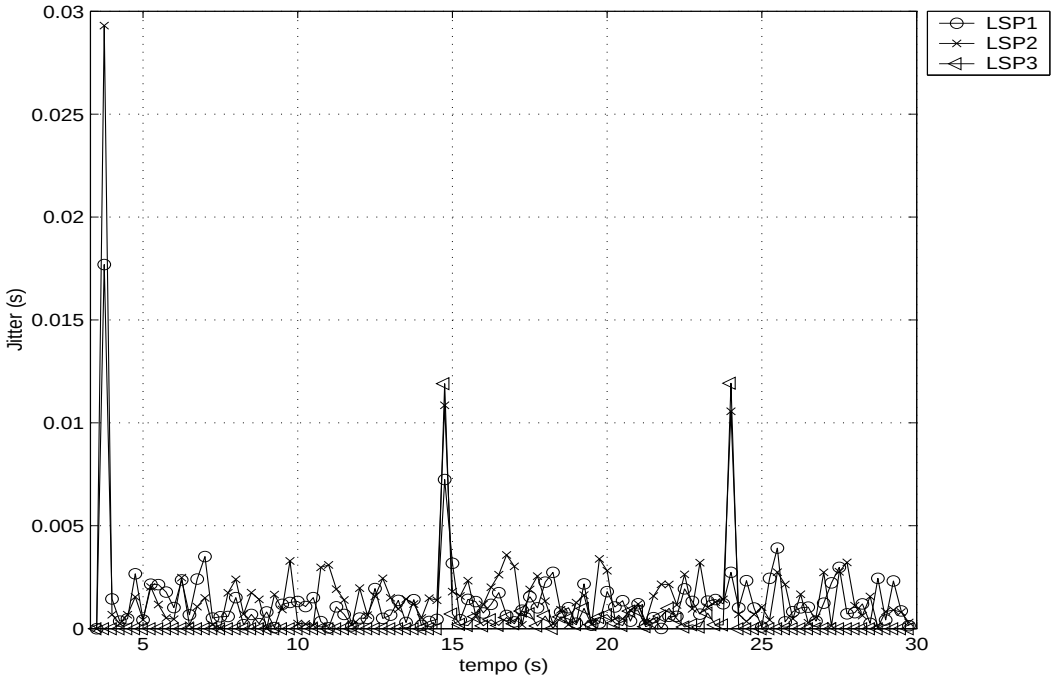

Figura 11: *Jitter* considerando o ambiente de rede preemptivo

6. Conclusões

Neste artigo apresenta-se a proposta e a avaliação de uma política de preempção de LSPs usando um FLC e um algoritmo genético como suporte. Isto é feito num ambiente de rede *DiffServ*/MPLS, o qual é implementado com o uso de uma arquitetura aberta (*software* livre e *hardware* convencional). Inicialmente, este ambiente experimental de rede, representando um AS, foi utilizado para a realização de experimentos considerando o modelo de rede *Best Effort*. Os resultados obtidos em termos de vazão, atraso fim a fim e *jitter* comprovaram a inexistência de diferenciação de serviços. Posteriormente, o ambiente de rede experimental foi configurado para realizar a integração *DiffServ*/MPLS. Neste caso, utilizou-se também no AS as políticas de preempção configuradas segundo os critérios de menor quantidade de LSPs ou de menor quantidade de largura de banda alocada por LSP. Estes critérios foram modelados como uma função que serviu como objeto de busca para o algoritmo genético proposto procurar um valor ótimo de forma a minimizá-la.

Os resultados obtidos demonstraram que ao se atribuir maior prioridade a um fluxo em um LSP do que a atribuída aos demais, faz-se com que o mesmo aproprie-se da quantidade de largura de banda necessária para atender aos seus requisitos. Notou-se também que, no instante do surgimento do fluxo desse novo LSP, ocorreu um aumento no atraso fim a fim experimentado pelos demais LSPs, devido à prioridade dada a este novo LSP. Portanto, esse cenário torna-se interessante do ponto de vista de aplicações que necessitam de atendimento com restrições temporais. Assim sendo, a implementação de políticas preemptivas auxiliadas por mecanismos de controle inteligente (FLC), proposta e avaliada neste artigo, torna-se perfeitamente viável. Porém, recomenda-se sua utilização para aplicações que necessitem dos recursos por um curto intervalo de tempo. Isto se deve ao fato de que os demais fluxos de dados sempre serão prejudicados para obter-se o atendimento dos requisitos de QoS de aplicações de multimídia interativas e de tempo real.

Agradecimentos

Os autores agradecem o apoio financeiro da FAPEMIG através do Projeto TEC-1027/02.

Referências

Awduche., O. D. (1999) "Requirements for Traffic Engineering over MPLS", Internet RFC 2702.

Baker, F. (1999) "Assured forwarding PHB group", Internet RFC 2597.

Baker, T. P. (1993) "Stack-Based Scheduling of Real-Time Process", IEEE Computer Society Press.

Beasley, D., Bull, D. R., and Martin, R. R. (1993) "An Overview of Genetic Algorithms: part 1, fundamentals", University Computing, pp. 58–69.

Benson, K. (2002) "An Expedicted Forwarding PHB (Per-Hop Behaviour)", Internet RFC 3246.

Blake., S. (1998) "An Architecture for Differentiated Services", Internet RFC 2475.

Cheng, R. and Chang, C. (1996) "Design of a Fuzzy Traffic Controller for ATM Networks", IEEE/ACM Transactions on Networking, Vol 4, pp. 460–469.

de Caluwe, R. (1997) "Fuzzy and Uncertain Object-Oriented Databases: Concepts and Models", British Library, Singapore.

de Oliveira, J. C. (2003) "New Techniques for End-to-End Quality of Service Provisioning in DiffServ/MPLS Networks", PhD Thesis, Georgia Tech.

Fernandez, M., Pedroza, A., and Rezende, J. (2003) "Implementação de Políticas de Gerenciamento Através de Lógica Fuzzy Visando a Melhoria da Qualidade de Serviço", Simpósio Brasileiro de Redes de Computadores (SBRC 2003), pp. 185-200.

Heuven, P., Den-Berghe, S., and Coppens, J. (2004) "RSVP-TE Daemon for DiffServ over MPLS under Linux", Atlantis Project.

Leu, J. R. and Casellas, R. (2004) "Project: MPLS for Linux", Sourceforge Project, http://sourceforge.net/projects/mpls-linux/, September.

le Faucher et al., F. (2002) "Multiprotocol Label Switching (MPLS) Support of Differentiated Services", Internet RFC 3270.

le Faucher et al., F. (2004) "Russian dolls bandwidth constraints model for Diffserv-aware MPLS traffic engineering", Internet draft, draft-ietf-tewg-diff-te-russian-07.txt.

Lorenz, D. and Orda, G. (1998) "QoS Routing in Networks with Uncertain Parameters", IEEE/ACM Transactions on Networking, Vol 6, pp. 768-778, December.

Rosenbaum, G., Jha, S., and Hassan, M. (2003) "Empirical study of traffic trunking in Linux-based MPLS test-bed", Intern. Journal of Network Management, pp. 277-288.

Rosen, E., Viswanathan, A., and Callon, R. (2001) "Multiprotocol Label Switching Architecture", Internet RFC 3031.

Trimintzios, P. (2001) "An architectural framework for providing QoS in IP differentiated services networks", IFIP/IEEE International Symposium on Integrated Network Management (IM 2001), Seattle, USA, pp. 17-34.