IntelliDyLBA: Um Esquema de Balanceamento de Carga para Redes MPLS com Aprendizado Desassistido baseado em Lógica Difusa e Algoritmos Genéticos*

Eduardo Guimarães Nobre[1,2], Pablo Rocha Ximenes Ponte[1], Marcial Porto Fernandez[1], Joaquim Celestino Júnior[1]

[1]Laboratório de Redes de Comunicação e Segurança da Informação (LARCES)
Universidade Estadual do Ceará (UECE)
Av. Parajana, 1700 – Itaperi – 60.720-020 – Fortaleza – CE – Brasil

[2]Programa de Pós-Graduação em Engenharia Elétrica
Universidade Federal do Ceará (UFC)
Av. Mr. Hull, s/n – Pici – 60455-760 – Fortaleza – CE – Brasil

{eduardo,pablo,marcial,celestino}@larces.uece.br

Abstract. *This paper describes a novel traffic engineering scheme based on a load balance reactive method for congestion control in MPLS networks that makes use of genetic algorithm based learning and fuzzy logic techniques. It is an improvement of the "Fuzzified Dynamic Load Balance Algorithm" (FuDyLBA) with the addition of auto-learning techniques. Computer simulation experiments have shown a better traffic distribution over the links of a network when compared to its predecessor, while showing intelligent adaptation to traffic and structural network changes.*

Resumo. *Este artigo descreve um novo esquema de engenharia de tráfego baseado em um método reativo de balanceamento de carga para controle de congestionamento em redes MPLS utilizando técnicas de lógica difusa e aprendizado baseado em algoritmos genéticos. Trata-se de um aperfeiçoamento do "Fuzzified Dynamic Load Balance Algorithm" (FuDyLBA) com a adição de técnicas de aprendizado desassistido. Experimentos obtidos por simulações computacionais apontam que o IntelliDyLBA é capaz de uma distribuição mais equânime do tráfego pelos enlaces de uma rede comparado ao seu predecessor, ao adaptar-se de forma inteligente a variações de tráfego e estrutura de rede.*

1. lntrodução

Uma das aplicações mais interessantes do *Multi-Protocol Label Switching* (MPLS) para redes baseadas em IP é a Engenharia de Tráfego, ou *Traffic Engineering* (TE) [Awduche e Jabbari 2002]. O Principal objetivo da TE é otimizar o desempenho de uma rede através da utilização eficiente de seus recursos. No contexto do MPLS isto é operacionalizado através dos *Label Switched Paths* (LSP) que tratam-se de caminhos

** Trabalho parcialmente financiado pela Fundação Cearense de Apoio ao Desenvolvimento Científico e Tecnológico (FUNCAP) e pelo Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq).*

estabelecidos através dos enlaces de uma rede onde rótulos são utilizados como parâmetros para as decisões de comutação. A característica mais importante dos LSPs para a TE é a possibilidade do estabelecimento de rotas explícitas através de uma rede, propiciando o controle efetivo do direcionamento dos diversos fluxos componentes do tráfego de uma rede.

A principal abordagem utilizada pelo *framework* MPLS em termos de engenharia de tráfego é o chamado roteamento baseado em restrições, ou *Constraint Based Routing* (CBR) [Salvadori et al. 2002]. Neste artigo, é apresentado um esquema alternativo de engenharia de tráfego que incorpora técnicas de aprendizado desassistido baseado em algoritmos genéticos e lógica difusa, o *Intelligent Dynamic Load Balance Algorithm* (IntelliDyLBA). Trata-se de um aprimoramento de nosso trabalho anterior, o algoritmo *Fuzzified Dynamic Load Balance Algorithm* (FuDyLBA) [Celestino Jr. et al. 2004], um algoritmo de balanceamento de carga para redes MPLS baseado em lógica difusa. Enquanto que os algoritmos CBR são baseados em um mecanismo preventivo, o IntelliDyLBA age quando o congestionamento é detectado pelo sistema de controle difuso, sendo, assim, baseado em um mecanismo reativo. O congestionamento, em uma rede MPLS, pode ser detectado basicamente de duas formas: ou a capacidade de um determinado link está "próxima" de sua capacidade total, ou uma nova requisição por um LSP não pôde ser executada por falta de recursos. O IntelliDyLBA trabalha na primeira hipótese, monitorando a intensidade de utilização de cada enlace da rede através de um sistema de controle difuso. Outra característica marcante do algoritmo é a sua capacidade de aprendizado que o torna adaptativo, sempre se ajustando às condições variantes de uma rede. Isso soluciona um grande problema encontrado nos esquemas de engenharia de tráfego para redes MPLS que é a característica estática relacionada à adaptação a mudanças dinâmicas de uma rede. Essas dinâmicas se traduzem pela variação das características de diversos elementos na rede, principalmente relacionadas a tráfego, topologia e capacidade dos enlaces.

Este trabalho é organizado da seguinte forma: na seção 2, é feita uma breve discussão sobre as técnicas de engenharia de tráfego em redes MPLS; na seção 3 o contexto de nosso modelo, bem como suas motivações, é definido; na seção 4 apresentamos o algoritmo proposto e algumas considerações a seu respeito; na seção 5 demonstramos os experimentos utilizados na validação do modelo, bem como os resultados obtidos; e na seção 6 fazemos uma breve conclusão do trabalho.

2. Engenharia de Tráfego em Redes MPLS

Os provedores de serviços de rede lutam diariamente para minimizar o congestionamento em seus sistemas. Reduzir o congestionamento, para redes baseadas em comutação de pacotes, é, também, diminuir retardos de transmissão, ou *delays*, o que cria as condições básicas para uma melhor garantia de qualidade de serviços (QoS) além de diminuir a carga de tráfego nos roteadores. Já em redes baseadas em comutação de circuitos, reduzir o congestionamento significa aumentar a largura de banda disponível em cada enlace, de forma que futuras requisições por conexões possam ser aceitas [Salvadori et al. 2002].

De acordo com a RFC 3272, "*Overview and Principles of Internet Traffic Engineering*" [Awduche et al. 2002], os mecanismos de gerenciamento de

congestionamento para Engenharia de Tráfego podem ser caracterizados, de acordo com a sua política, pelos seguintes critérios: pelo tempo de resposta, pelo caráter reativo ou preventivo; e pelo fato de ser baseado na geração de fluxo ou em sua demanda.

A maioria das propostas de mecanismos de gerenciamento de TE são preventivas, ou seja, acomodam caminhos pela rede de forma a prevenir situações de congestionamento. Os dois mecanismos mais mencionados na literatura são o *Constraint Based Routing* (CBR) e o *Traffic Splitting* (partição de tráfego) [Salvadori et al. 2002]. O CBR teve suas origens no problema de roteamento para Qualidade de Serviço em redes baseadas em IP e diz respeito à alocação de LSPs baseando-se em diversos tipos de restrições como largura de banda disponível, atraso máximo, etc. O mecanismo de *Traffic Splitting* funciona distribuindo porções de um fluxo, ou particionando o tráfego por LSPs paralelos entre o par de nós ingresso-egresso.

Um dos esquemas de CBR mais mencionados é o chamado MIRA *(Minimum Interference Routing Algorithm)*, ou Algoritmo de Roteamento de Interferência Mínima [Kar et al. 2000]. Ele é baseado em um algoritmo *online* de escolha heurística dinâmica de caminho. A idéia principal é tirar vantagem do conhecimento prévio do par de nós ingresso-egresso para que seja evitado o estabelecimento de rotas que passem por enlaces que possam interferir em futuras novas alocações de caminhos. Tais enlaces críticos são identificados pelo MIRA como sendo aqueles que, se ficarem altamente utilizados, poderão impedir que futuras requisições por conexão sejam satisfeitas entre os mesmos nós ingresso-egresso. O principal ponto fraco deste mecanismo é a complexidade computacional causada pelo cálculo do fluxo máximo entre os nós comunicantes, requisito para a identificação dos enlaces críticos, e a utilização desbalanceada da rede [Salvadori et al. 2002]. Como demonstrado em [Wang et. al. 2002], o MIRA não é capaz de estimar gargalos em enlaces que sejam críticos para clusters de nós de rede.

Além disso, ele não leva em conta a carga de tráfego atual em suas decisões de roteamento. Imaginemos um cenário onde um par origem-destino é conectado por duas ou mais rotas, cada uma com a mesma largura de banda residual. Quando uma nova requisição por um LSP for gerada, uma destas rotas será escolhida para satisfazer a requisição. Após este procedimento, todos os enlaces pertencentes às rotas que não foram escolhidas tornam-se críticos, de acordo com a definição supramencionada. Isso significa que todas as futuras requisições de novos LSPs entre o mesmo par ingresso-egresso serão roteadas pelo mesmo caminho, enquanto as outras rotas continuam livres de carga, o que causa um uso desbalanceado dos recursos da rede. Ainda pior, em cenários onde LSPs são alocados e desalocados dinamicamente, este esquema pode causar a criação de rotas por caminhos ineficientes e bloqueios futuros para certas rotas. Essa gama de problemas é comum a todos os mecanismos CBR propostos na literatura [Salvadori et al. 2002]. Pode-se observar, então, a ineficiência deste esquema de engenharia de tráfego ao lidar com variações dinâmicas em redes MPLS.

Dos poucos modelos de controle de congestionamento (gerenciamento reativo) para redes MPLS propostos na literatura temos o *Fast Acting Traffic Engineering* (FATE) proposto em [Holness e Phillips 2000], uma extensão do *Constraint Routing Label Distribution Protocol* (CR-LDP), um protocolo de sinalização baseado em CBR para distribuição de rótulos em redes MPLS. O FATE incrementa o CR-LDP da

seguinte forma: quando determinados LSPs experimentam perda de pacotes, eles são re-roteados para caminhos menos congestionados, remediando, assim, a condição de congestionamento.

Outro esquema reativo encontrado na literatura foi proposto por [Jüttner et al. 2000] que é baseado no re-roteamento otimizado de um LSP já estabelecido quando um novo LSP não pôde ser alocado na rede por falta de recursos. A idéia baseia-se no fato de que em altos níveis de utilização a alocação sob demanda de LSPs baseados em CBR pode sofrer falhas. A otimização na escolha da nova rota para o LSP que mudará seu caminho utiliza técnicas de Programação Linear Inteira em uma adaptação do protocolo *Constrained Shortest Path First* (CSPF). Mais uma vez, trata-se de uma solução acessória que serve de coadjuvante às técnicas de CBR.

Podemos enumerar, também, no rol de esquemas reativos, o *First-Improve Dynamic Load Balance Algortihm* (FID) [Salvadori et al. 2002], um algoritmo baseado em uma busca local otimizada onde, para cada enlace considerado congestionado em uma determinada rede, é procurado o primeiro LSP que, se re-roteado, é capaz de melhorar a capacidade do enlace em questão sem prejudicar a capacidade geral de balanceamento da rede. O FID é um aprimoramento do *Dynamic Load Balance Algorithm* (DyLBA) [Battiti et al. 2002], onde a principal distinção está na profundidade da busca, que no caso do DyLBA só encerra quando acha o melhor LSP a ser re-roteado, ao invés de parar na primeira opção viável.

2.1. FuDyLBA

Por último, podemos elencar, como mecanismo reativo, o algoritmo FuDyLBA [Celestino Jr. et al. 2004]. Ele funciona percorrendo cada elemento do conjunto de enlaces de uma rede, classificando-os através de um controlador difuso que retorna uma intensidade de descarga para o enlace em questão. Esse valor de intensidade será utilizado por uma função de descarga. Essa função recebe como entrada um nível que serve de parâmetro para o quanto um determinado enlace deverá ser liberado de carga. Tal função examina cada LSP que percorre o enlace, calculando um caminho alternativo para esse LSP que não passe pelo enlace em questão. Depois, compara o novo caminho com o caminho atual, sem a contribuição do LSP a ser examinado. Se o novo caminho oferecer uma largura de banda residual inferior a do caminho atual, o próximo LSP é examinado. Caso a largura de banda residual do novo caminho seja maior, o LSP é re-roteado e a busca local reinicia de volta ao primeiro LSP que percorre o enlace. A função *unload* apenas terminará após ter atingido um percentual de descarga para aquele enlace, igual à intensidade requisitada, ou após ter examinado o último LSP que percorre o enlace sem que nenhum outro fosse capaz de ser re-roteado.

Para calcular a intensidade adequada de descarga para cada enlace, o FuDyLBA possui um controlador difuso que recebe, como entrada, a intensidade de utilização do enlace e retorna, como saída, a intensidade da função de descarga para aquele enlace.

Para compreender o nível de carga em cada enlace, o controlador *fuzzy* do FuDyLBA define três valores lingüísticos de entrada: pouco congestionado, medianamente congestionado e muito congestionado. Cada um desses valores é especificado por sua respectiva função de pertinência triangular ou trapezoidal. Para definir lingüisticamente a intensidade de atuação da função de descarga, encontramos no

controlado *fuzzy* do FuDyLBA três funções não contínuas para as pertinências de saída, das quais: descongestionar pouco, descongestionar medianamente e descongestionar muito.

Cada enlace da rede é medido em relação à sua proporção de utilização de largura de banda. Quando uma determinada medida se enquadra em qualquer dos valores lingüísticos de entrada, o enlace é considerado congestionado e a função de descarga é executada recebendo, como entrada, a saída do processo de defuzzificação do controlador difuso do FuDyLBA .

3. Motivações e Definição do Problema

Como discutido em [Celestino Jr. et al. 2004] as funções de pertinência do controlador difuso do algoritmo FuDyLBA desempenham um papel fundamental para o comportamento do esquema de balanceamento de carga. De fato, funções de pertinência bem projetadas implicam em um melhor algoritmo. Existe, para tanto, um elemento chave para o projeto destas funções de pertinência que é a dinâmica da rede. Um determinado conjunto de funções de pertinência, por exemplo, pode funcionar bem em certa conjuntura de rede, digamos, com tráfego de alta intensidade, mas, quando o tipo de tráfego variar, tornando-se eventualmente de menor intensidade, essas mesmas funções de pertinência deixam de ser a melhor opção. É impraticável, porém, modelar um conjunto de funções de pertinência que seja o ideal para qualquer situação de rede, visto as inúmeras possibilidades geradas pela dinâmica deste tipo de sistema.

O problema de contemplar a dinâmica de uma rede não é exclusividade de algoritmos difusos, mas apresenta-se como problema em todos os métodos de balanceamentos de carga para redes MPLS, principalmente nos esquemas preventivos. O próprio FID [Salvadori et al. 2002] possui um parâmetro de entrada que permite um ajuste de seu comportamento.

Uma forma eficiente de se resolver o problema da adaptabilidade em algoritmos de balanceamento de carga para redes MPLS é a utilização de técnicas de algoritmos genéticos. Tais técnicas podem aprimorar um algoritmo de balanceamento de carga ao longo de seu funcionamento de forma que determinadas características dele possam evoluir a fim de que convirjam a um comportamento ótimo, retratando um aprendizado baseado na dinâmica da rede.

3.1. Definição do Problema

A rede considerada consiste de *n* roteadores. Um subconjunto de roteadores que formem o par ingresso-egresso entre os quais conexões podem ser potencialmente estabelecidas é definido. Cada requisição de conexão é feita ao roteador de ingresso ou a central de gerenciamento da rede (no caso de gerência centralizada) o qual determina a rota explícita para o LSP de acordo com a topologia atual e com os recursos disponíveis da rede. Para realizar o algoritmo de balanceamento é necessário que cada roteador na rede (ou a central de gerência) saiba a topologia atual da rede e a largura de banda residual em cada enlace. Para tanto, cada roteador na rede MPLS precisa ter suporte a algum protocolo de roteamento baseado em estado do link com suporte a notificações de largura de banda residual por enlace (*residual bandwidth advertisements*). A largura de banda residual para um determinado enlace de rede deverá ser calculada pela diferença

entre a sua capacidade total e a soma do consumo de largura de banda de todos os LSPs que passem por ele.

Nosso objetivo é aumentar a largura de banda residual de cada enlace na rede a fim de distribuir sua carga o mais eqüitativamente possível por todos os enlaces, de forma que haja o mínimo possível de enlaces ociosos ou de enlaces sobrecarregados, sempre evoluindo o comportamento do algoritmo, contabilizando as variações na dinâmica da rede de forma inteligente.

4. IntelliDyLBA: Um Algoritmo de Engenharia de Tráfego Fuzificado para Balanceamento de Carga com Aprendizado Genético Desassistido

A meta principal do algoritmo é balancear o tráfego de uma rede MPLS dinamicamente adaptando-se a mudanças nas características de rede. Se reduzirmos este problema ao aumento da largura de banda residual de cada enlace da rede, teremos um objetivo secundário que é maximizar a banda residual de cada enlace da rede, mas com a restrição de que a operação a ser realizada para a maximização local não comprometa a qualidade global do balanceamento. Toda essa operação deve levar em conta mudanças dinâmicas no conjunto de características de rede de forma a melhorar-se independentemente. O algoritmo deverá, então, aferir seu próprio funcionamento, sempre valorizando ações que gerem um impacto positivo sobre o balanceamento da rede na forma de aprendizado.

4.1. Visão Geral do IntelliDyLBA

O IntelliDyLBA estende o funcionamento básico do FuDyLBA [Celestino Jr. et al. 2004], com mudanças no sistema de controle difuso e com a adição de aprendizado genético desassistido.

Inicialmente, o único controlador do FuDyLBA foi substituído por um conjunto de 12 controladores difusos. Esses controladores foram projetados tendo como base um estudo experimental da relação entre as funções de pertinência do controlador difuso do FuDyLBA e diversas características dinâmicas de rede. Foram realizados 672 experimentos, onde foi possível se apontar determinados conjuntos de funções de pertinência que possuíam desempenho superior no funcionamento do algoritmo em determinadas características de rede. Foram selecionados os 12 conjuntos (entrada/saída) de funções de pertinência com melhor desempenho, contemplando todos os cenários de rede. A partir destas funções de pertinência, foram desenvolvidos 12 controladores difusos.

O IntelliDyLBA realiza rodadas de balanceamento semelhantes a do FuDyLBA, utilizando cada um destes controladores por vez. Além disso, o IntelliDyLBA mantém um registro das variações do desvio padrão das larguras de banda residual do conjunto de enlaces da rede. Para cada utilização de um dos 12 controladores difusos, o conjunto entrada-saída do controlador difuso atual é guardado, juntamente com a variação no desvio padrão produzida pela atuação de seu ente controlado, a função de descarga (*unload*), em uma matriz, a qual chamamos de matriz de aprendizado. Depois de utilizados todos os controladores difusos, a matriz de aprendizado estará completa e o algoritmo poderá iniciar o módulo de aprendizagem genético. Esse módulo, primeiramente, seleciona, da matriz de aprendizagem, os 50 melhores exemplos. Depois

disso, ele utiliza essa relação de exemplos como base para uma otimização baseada em técnicas de algoritmos genéticos. Com isso, este módulo aproxima o funcionamento dos 12 controladores difusos ao que está disposto na lista de exemplos, fazendo com que o algoritmo convirja a um funcionamento adaptado às características de rede atuais.

4.2. O Pseudocódigo

O pseudocódigo do algoritmo proposto é demonstrado na figura 1. É possível se identificar três blocos, onde o primeiro demonstra a estrutura geral do algoritmo, o segundo representa a função de descarga de enlace que é controlada de forma difusa e o terceiro representa o módulo de aprendizagem baseado em algoritmos genéticos.

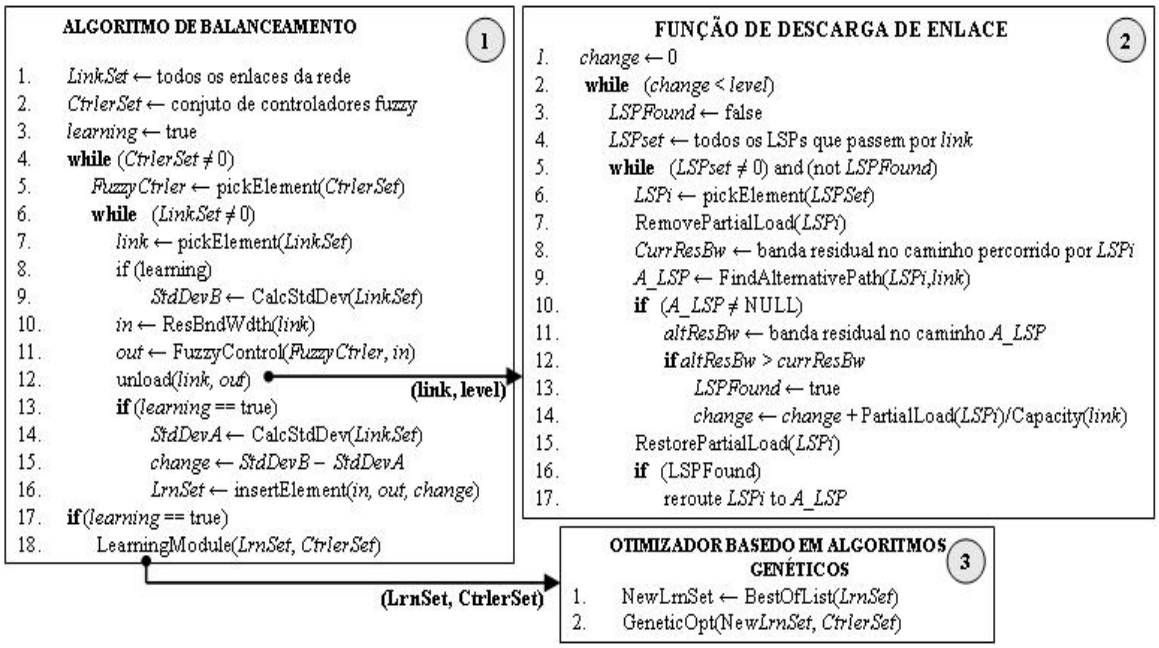

Figura 1. Pseudocódigo do IntelliDyLBA

4.3. Estrutura Geral

O primeiro bloco do pseudocódigo representa a estrutura geral do algoritmo IntelliDyLBA. Este é o principal bloco do pseudocódigo. Nele podemos observar todos os elementos do algoritmo e suas inter-relações.

Neste bloco, o algoritmo percorre todos os enlaces da rede. Cada enlace é classificado através de um dos 12 controladores difusos. Esse controlador receberá como entrada a intensidade de largura de banda residual do enlace e retornará a intensidade de descarga com a qual este enlace deverá ser descarregado pela função *unload*. É contabilizada, também, a diferença entre o desvio padrão das larguras de banda residuais de todos os enlaces da rede antes e depois da descarga. Por fim, os valores obtidos para a entrada do controlador difuso, sua saída, e a variação no desvio padrão das bandas residuais são registrados em uma nova entrada da matriz de aprendizado. Essa nova entrada significa, em última análise, um registro do impacto de um conjunto entrada/saída do controlador difuso na qualidade de balanceamento da rede (variação de desvio padrão). A seguir, o próximo enlace é analisado e descarregado pelo

mesmo processo. Após percorrer todos os enlaces da rede, o algoritmo seleciona o próximo controlador da lista de controladores difusos e novamente percorre todos os enlaces da rede, classificando-os de forma difusa, realizando a descarga através da função *unload*, contabilizando a variação no desvio padrão e registrando nova entrada na matriz de aprendizado.

Esse processo continua até que todos os controladores difusos da lista de 12 controladores tenham sido utilizados para balancear a rede, sempre registrando seus conjuntos entrada/saída com respectivos impactos na qualidade de balanceamento da rede (variação no desvio padrão) na matriz de aprendizado.

4.4. Sistema de Controle Difuso

O sistema de controle difuso do IntelliDyLBA é composto por 12 controladores difusos que são utilizados em um escalonamento cíclico no formato *round robin*. Cada controlador é utilizado em uma rodada de balanceamento que percorre todos os enlaces da rede.

As funções de pertinência de entrada (fuzificação) para os 12 controladores representam os valores da variável lingüística “intensidade de banda residual”, enquanto que as funções de pertinência de saída (defuzificação) representam os valores da variável lingüística “intensidade de descarga”.

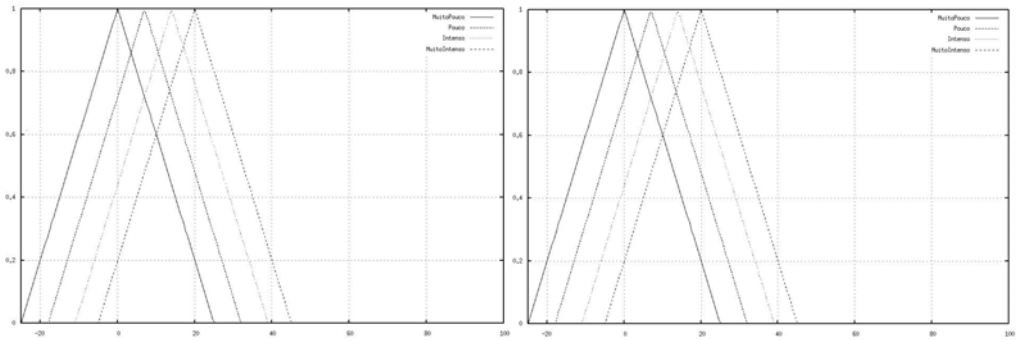

Figura 2. Funções de pertinência de um dos controladores do IntelliDyLBA

Os controladores se dividem em três grupos no que diz respeito às funções de pertinência. O primeiro grupo de controladores utiliza uma relação de três funções triangulares para representar as pertinências de entrada e outras três funções triangulares para representar as pertinências de saída. Estas funções de pertinência de entrada e de saída representam os valores lingüísticos “pouco”, “mediano” e “intenso”. O segundo grupo de controladores utiliza um conjunto de quatro funções de pertinência de entrada triangulares e quatro funções triangulares para as pertinências de saída. Elas representam os valores lingüísticos “muito pouco”, “pouco”, “intenso” e “muito intenso”. Por fim, o último grupo de controladores possui um conjunto de cinco funções triangulares para as pertinências de entrada e cinco funções triangulares para as pertinências de saída, ambos representando os valores lingüísticos “muito pouco”, “pouco”, “mediano”, “intenso” e “muito intenso”. Cada um destes três grupos de controladores possui quatro controladores que se distinguem entre si pela distribuição das funções de pertinência, variando de funções mais próximas a funções mais afastadas umas das outras. A figura

2 mostra as funções de pertinência de entrada e de saída, respectivamente, para um dos controladores do IntelliDyLBA.

É importante notar que as características das funções de pertinência demonstradas aqui são apenas os valores iniciais com que os 12 controladores difusos irão funcionar. Com o funcionamento do algoritmo, essas funções irão evoluir geneticamente de forma que seus valores também irão se modificar.

As regras de inferência do sistema de controle difuso do algoritmo proposto seguem, também, uma regra em sua formação para cada um dos 12 controladores difusos. Podemos observar o conjunto de regras de inferência para cada um dos três grupos de controladores na figura 3.

Figura 3. Regras de inferência para os grupos de controladores do IntelliDyLBA

4.5. A Função *unload*

Esta função tem como objetivo descarregar o enlace indicado pela entrada *link* de uma porcentagem de carga indicada pela entrada *level*. Seu pseudocódigo está representado no segundo bloco da figura 1. O funcionamento desta função é semelhante ao da função *unload* do algoritmo *FuDyLBA* que é descrito detalhadamente em [Celestino Jr. et al. 2004].

Basicamente, a função *unload* inspeciona todos os LSPs que percorrem o enlace indicado por *link* em busca daquele que, se re-roteado para um caminho alternativo que não inclua o enlace em questão, melhore a qualidade de balanceamento da rede. Ao ser encontrado, este LSP é re-roteado para o caminho alternativo. Em seguida, é verificado se a diminuição de carga do enlace atinge o nível indicado por *level*. Caso negativo, incicia-se uma nova busca por um outro LSP que satisfaça o mesmo critério. Esse processo terminará apenas quando a intensidade de descarga for atingida ou quando todos os LSPs forem inspecionados.

4.6. Pseudocódigo – Módulo de Aprendizado por Algoritmo Genético.

O terceiro bloco representa o módulo de aprendizado por algoritmo genético. Nele, ocorre a otimização dos 12 controladores difusos do modelo através de técnicas de algoritmos genéticos.

Neste bloco podemos visualizar apenas dois passos. O primeiro cria a lista de exemplos que serão usados para a aprendizagem, escolhendo os 50 melhores elementos (conjunto entrada/saída) da matriz de aprendizado criada anteriormente. Para isso, o valor da variação do desvio padrão (*change*), que é o terceiro elemento de cada linha da matriz de aprendizagem, é utilizado como parâmetro classificador, onde o melhor elemento será aquele com menor valor. Com isso, criamos uma lista de exemplos que

representam o funcionamento da função objetivo, à qual os 12 controladores difusos deverão se aproximar. No segundo passo, chamamos a rotina de otimização por algoritmo genético que recebe o conjunto dos 12 controladores difusos e a lista de exemplos como parâmetros. Com isso, ela aproximará o funcionamento dos controladores ao conjunto de exemplos, fazendo com que convirja para um comportamento mais próximo da lista de exemplos, semelhante ao da função objetivo.

4.7. Função de Otimização por Algoritmo Genético.

O uso de algoritmos genéticos para aperfeiçoar controladores difusos foi mostrado por [Kim et al. 1995], [Velascos e Magdalena 1995] e [Herrera e Lozano 1995]. A otimização é um processo que busca encontrar uma melhor combinação de parâmetros de um controlador fuzzy para atingir o melhor resultado possível. Podemos considerar um controlador fuzzy como uma caixa com vários botões de ajuste dos parâmetros, e o processo de otimização consiste em escolher uma combinação de valores de ajuste dos botões (parâmetros) que resulta em melhores resultados, ou seja, que transforme o comportamento do controlador de forma que a relação de suas entradas e respectivas saídas se aproxime do conjunto de exemplos (entrada/saída) que representam o funcionamento ideal (função objetivo). Os valores dos parâmetros do algoritmo genético utilizados na otimização dos 12 controladores difusos foram escolhidos dentro das faixas usuais [Fernandez et al. 2003].

5. Simulações e Resultados

Essa seção apresenta o resultado do funcionamento do algoritmo IntelliDyLBA em contraste com seus predecessores FuDyLBA [Celestino Jr. et al. 2004] e FID [Salvadori et al. 2002]. Considerando que o FID já foi extensivamente comparado com outras técnicas de engenharia de tráfego para redes MPLS em [Salvadori et al. 2002] e devido às restrições óbvias de espaço, nos restringimos a demonstração da relação destes três algoritmos apenas.

Todas as simulações foram realizadas com o MNSv2 (*MPLS Network Simulator version 2*) [Ahn e Chun 1999], uma extensão do NS-2 (*Network Simulator 2*) [McCanne e Floyd 1998]. O sistema de controle difuso, bem como o módulo de aprendizado por algoritmo genético, foram simulados utilizando-se a ferramenta *JFS Fuzzy System* [Mortensen 1998].

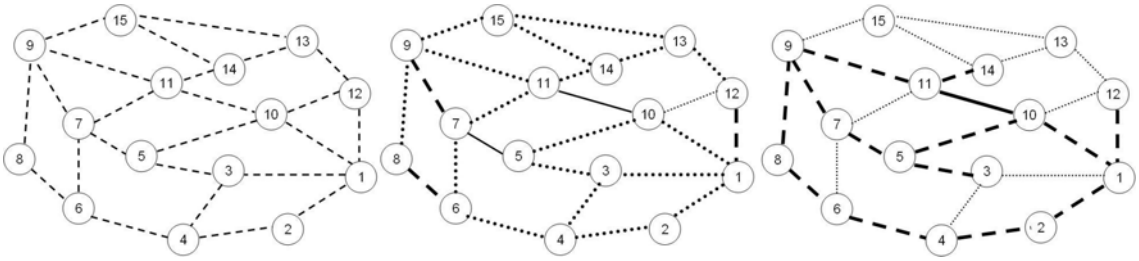

Figura 4. Topologias utilizadas nos experimentos.

5.1. Experimentos

Todas as simulações utilizaram as topologias indicadas na figura 4 onde linhas pontilhadas claras representam enlaces com capacidade de 1024 unidades, linhas pontilhadas escuras representam enlaces de 1536 unidades, linhas tracejadas claras representam enlaces de 1664 unidades, linhas tracejadas escuras representam enlaces de 2048 unidades, linhas cheias claras representam enlaces de 2560 unidades e linhas cheias escuras representam enlaces com capacidade de 3072 unidades. A principal distinção entre as topologias é a distribuição da capacidade de seus respectivos enlaces. A primeira topologia possui desvio padrão da capacidade de banda dos enlaces igual a zero, a segunda possui esse desvio igual a 338 e a terceira possui o desvio padrão igual a 577.

Em todas as simulações foram injetados 600 LSPs ao longo de 50 segundos. A injeção de LSPs durante as simulações seguiu uma distribuição pseudo-randômica para o intervalo de injeção entre cada LSP, para o tempo de permanência de cada LSP e para o consumo de banda de cada LSP. O intervalo de injeção variou de 0.05 a 0.1 segundo, o tempo de permanência variou de 3 a 8 segundos e o consumo de banda variou de 8 a 32 unidades. Foram realizadas 30 rodadas de simulações finitas independentes [Pawlikowski et al. 2002] com 30 sementes distintas para as distribuições pseudo-randômicas. Durante as simulações houve a variação da topologia, de forma que cada grupo de 10 simulações (10 das 30 sementes) usasse apenas uma das topologias indicadas na figura 4. Dessa forma, além de garantir uma variação independente na injeção de tráfego, os experimentos garantiram, também, uma variação topológica que permitiu melhor aferição da adaptabilidade dos algoritmos.

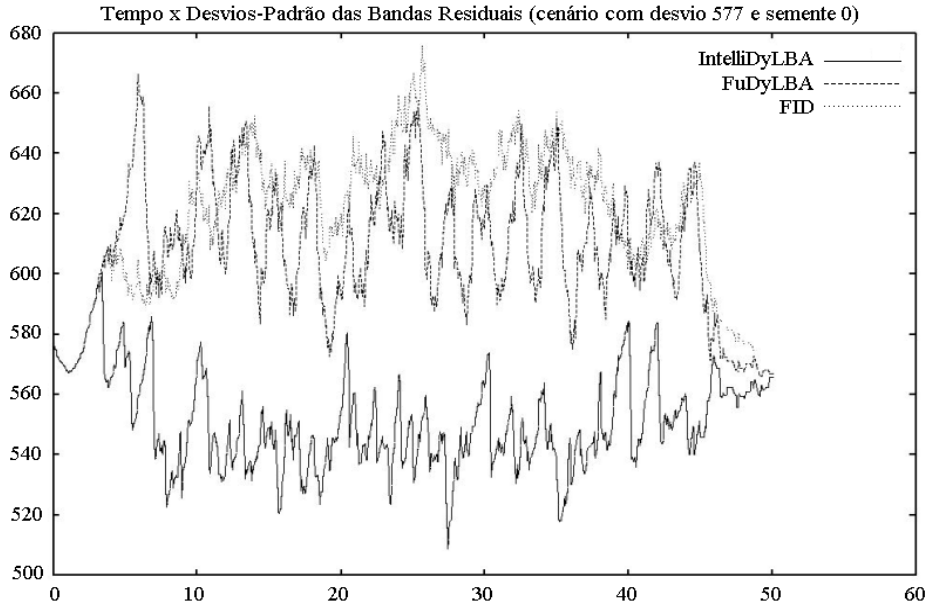

Figura 5. Desvio padrão das bandas residuais dos enlaces ao longo do tempo para um dos 30 experimentos para os três algoritmos.

5.2. Métrica de Avaliação

Cada enlace de uma rede possui uma medida de largura de banda residual. Se agruparmos todos eles, verificaremos que as suas respectivas medidas de banda residual tendem a se aproximarem umas das outras ou a se dispersarem umas das outras conforme a rede recebe carga (tráfego). Para aferir a dispersão de um determinado conjunto de amostras, utiliza-se o cálculo do desvio padrão. Ele irá indicar, através de um valor finito, o quanto as amostras estão dispersas umas das outras. Um desvio padrão igual a zero indica que não existe dispersão, ou seja, que o valor de todas as amostras é o mesmo. Interpretando este método na situação onde as amostras indicam a intensidade de banda residual dos enlaces de uma rede, o desvio padrão pode ser traduzido como uma medida de balanceamento de carga. Um desvio padrão maior indica enlaces com utilização mais dispersa, o que significa um balanceamento de carga de má qualidade.

O objetivo primordial do algoritmo proposto é balancear dinamicamente a carga de uma rede. Portanto, utilizou-se o desvio padrão das larguras de banda residuais dos enlaces de rede para avaliação do desempenho dos três algoritmos simulados.

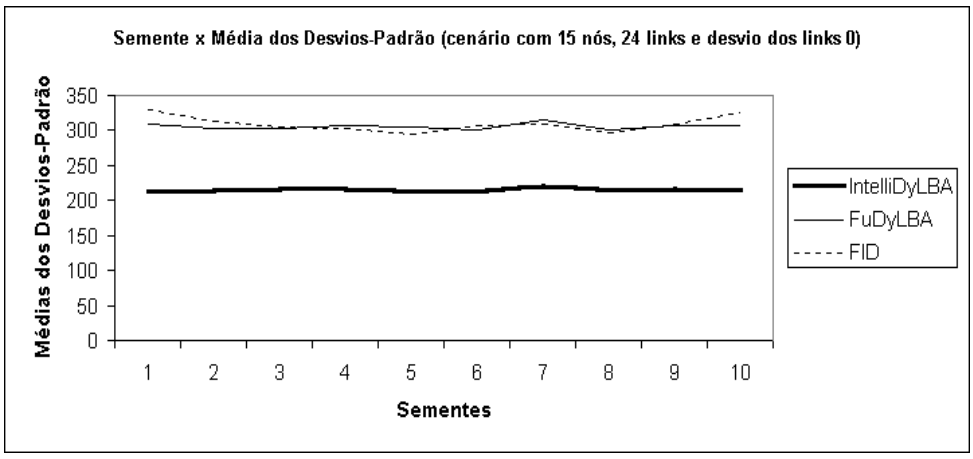

Figura 6. Média dos desvios padrão das bandas residuais dos enlaces ao longo dos experimentos da topologia 1.

5.3. Resultados

Em cada uma das 30 simulações foi mensurado o desvio padrão da banda residual dos enlaces ao longo do tempo. Podemos ver o resultado de uma das 30 simulações na figura 5. Como se pode observar, o IntelliDyLBA realiza o balanceamento mais rapidamente que seus pares, além de manter a rede em um nível de balanceamento melhor por mais tempo.

Esse comportamento se repetiu ao longo dos 30 experimentos, mesmo quando houve variação de topologia.

Tabela 1. Intervalos de confiança para os experimentos da primeira topologia.

Algoritmo	Desvio Padrão Médio	Nível de Confiança	Intervalo de Confiança
FID	309.642	95%	+/- 6.89435
FuDyLBA	305.915	95%	+/- 2.6985
IntelliDyLBA	215.557	95%	+/- 1.40914

Para representar pontualmente o balanceamento de carga em cada simulação, optou-se por calcular a média dos desvios padrão das larguras de banda residuais dos enlaces de cada simulação. Com isso, pôde-se ter uma noção mais exata da qualidade do balanceamento ao longo de toda a simulação. Dessa forma, a figura 6 representa a distribuição dessas médias, para os três algoritmos, nas simulações que utilizaram a primeira topologia. A partir da análise dessas amostras, pode-se averiguar que o IntelliDyLBA possuiu um desvio padrão médio sempre bem inferior aos seus pares ao longo de todas as simulações. Na tabela 1 podemos observar uma comparação entre estes valores incluindo seus respectivos intervalos de confiança.

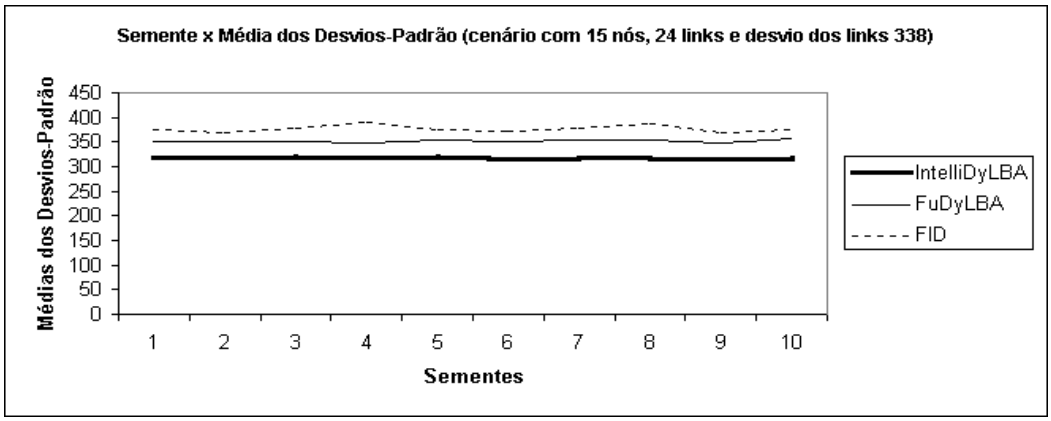

Figura 7. Média dos desvios padrão das bandas residuais dos enlaces ao longo dos experimentos da topologia 2.

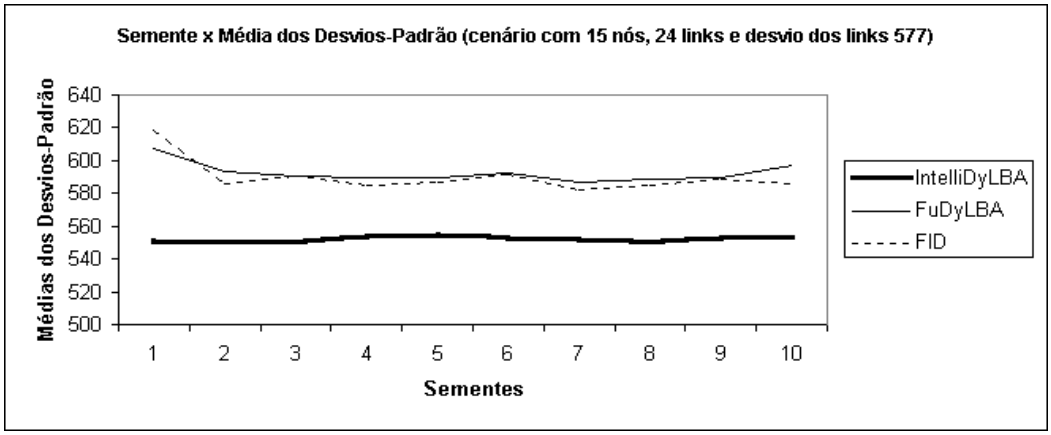

Figura 8. Média dos desvios padrão das bandas residuais dos enlaces ao longo dos experimentos da topologia 3.

A segunda e terceira topologias apresentaram resultados semelhantes ao da primeira topologia para o IntelliDyLBA, corroborando o seu desempenho superior na capacidade de balancear a carga de uma rede MPLS. Podemos observar a distribuição de médias do desvio padrão da banda resídua do enlaces ao logo dos experimentos que utilizaram a segunda topologia na figura 7, enquanto que a figura 8 mostra a mesma distribuição para os experimentos que utilizaram a terceira topologia.

Analisando-se os resultados das três topologias, podemos observar que o FuDyLBA, por vezes, tem um comportamento pior que o do FID. Contrastando essa observação com as características topológicas, podemos observar que isso ocorre

principalmente com a terceira topologia, onde o desvio padrão das capacidades totais dos enlaces é maior. Essa observação é importante para ressaltar a capacidade de adaptação do IntelliDyLBA em relação ao FuDyLBA pois, como podemos ver, em todos os casos, o IntelliDyLBA consegue se destacar. A tabela 2 esclarece os resultados do gráfico da figura 7, apresentando os intervalos de confiança para as médias dos desvios padrão e a tabela 3 apresenta o mesmo para o gráfico da figura 8.

Tabela 2. Intervalos de confiança para os experimentos da segunda topologia.

Algoritmo	Desvio Padrão Médio	Nível de Confiança	Intervalo de Confiança
FID	376.777	95%	+/- 4.42517
FuDyLBA	351.515	95%	+/- 1.71377
IntelliDyLBA	318.11	95%	+/- 0.864944

Tabela 3. Intervalos de confiança para os experimentos da terceira topologia.

Algoritmo	Desvio Padrão Médio	Nível de Confiança	Intervalo de Confiança
FID	589.977	95%	+/- 6.53577
FuDyLBA	592.536	95%	+/- 3.70995
IntelliDyLBA	552.242	95%	+/- 1.15648

6. Conclusões e Trabalhos Futuros

Demonstramos o funcionamento do IntelliDyLBA, um algoritmo reativo de balanceamento de carga que incorpora técnicas de lógica difusa e aprendizado desassistido baseado em algoritmos genéticos. Comparamos seu funcionamento com o de seus pares (FuDyLBA e FID), estabelecendo seu desempenho superior, mesmo em situações onde seu predecessor (FuDyLBA) demonstra um decaimento de desempenho o que ressalta a grande capacidade de adaptabilidade do algoritmo proposto.

Estudos futuros deverão se concentrar no aperfeiçoamento do módulo de aprendizagem, possivelmente reconstruindo-o com técnicas de redes neurais. Outro ponto a ser atacado é a construção da lista de exemplos da função objetivo, tarefa que pode se tornar um grande desafio.

Referências

Ahn, G. e Chun, W. (1999). Overview of MPLS Network Simulator: Design and implementation. http:// flower.ce.cnu.ac.kr/~fog1/mns/.

Awduche, D.; Chiu, A.; Elwalid, A.; Widjaja, I. e Xiao, X.. (2002) “Overview and Principles of Internet Traffic Engineering.”, IETF RFC 3272, http://www.faqs.org/rfcs/rfc3272.html, Maio.

Awduche, D. O. e Jabbari, B. (2002) "Internet Traffic Engineering using Multi-Protocol Label Switching (MPLS)", Computer Networks, (40):p. 111–129, Setembro.

Battiti, R. e Salvadori, E. (2002) “A Load Balancing Scheme for Congestion Control in MPLS Networks. Technical report”, Universitá di Trento, Dipartimento di Informatica e Telecomunicazioni, Novembro.

Celestino Jr., J., Ponte, P. R. X., Tomaz, A. C. F., Diniz, A. L. B. P. B. (2004) "FuDyLBA: Um Esquema de Engenharia de Tráfego para Balanceamento de Carga

em Redes MPLS Baseado em Lógica Difusa", Anais do 22º Simpósio Brasileiro de Redes de Computadores, Gramado – RS, Maio.

Chen, C.H. (1996) “Fuzzy Logic and Neural Network Handbook/the Handbook of Software for Engineers and Scientists”, IEEE, Setembro.

Fernandez, M. P.; Pedrosa, A. C. P.; Resende, J. F. (2003) “Implementação de Políticas de Gerenciamento com lógica Fuzzy e Algoritmo Genético visando à melhoria da Qualidade de Serviço (QoS)”, Revista da SBrT Vol 18-2, outubro.

Herrera, F.; Lozano, M.; and Verdegay, J. (1995); “Tuning fuzzy logic controllers by genetic algorithms,”International Journal of Approximate Reasoning, vol. 12, pp. 299–315, June.

Holness, F. e Phillips, C. (2000) “Dynamic Congestion Control Mechanism for MPLS Networks.”, In: SPIE’s International Symposium on Voice, Video and Data Communications. Internet, Performance and Control Network ystems, p. 1001–1005, Boston - MA, Novembro.

Jüttner, A.; Szviatovszki, B.; Szentesi, A.; Orincsay, D. e Harmatos, J. (2000) “On-demand Optimization of Label Switched Paths in MPLS Networks.”, In: Proceedings of IEEE International Conference on Computer Communications and Networks, p. 107–113, Las Vegas - Nevada, Outubro.

Kar, K.; Kodialam, M. e Lakshman, T.V. (2000) "Minimum Interference Routing of Bandwidth Guaranteed Tunnels with MPLS Traffic Engineering Applications", IEEE Journal on Selected Areas in Communications, 18(12):p. 2566–2579, Dezembro.

Kim, J.; Moon, Y. and Zeigler, B.P. (1994); “Designing fuzzy net controllers using GA optimization,” in Proceedings IEEE/IFAC Joint Symposium on Computer-Aided Control System Design, (Tucson (AZ), USA), pp. 83–88, Mar.

McCanne, S. e Floyd, S. (1998); “NS – Network Simulator - Version 2”; http://www.isi.edu/nsnam/ns/

Mortensen, J. E. (1998); “JFS Fuzzy System”. http://inet.uni2.dk/~jemor/jfs.htm.

Pawlikowski, K.; Jeong, H. -D. J. e Lee, J. S. R. (2002); “On Credibility of Simulation Studies of Telecommunication Networks” IEEE Communications Magazine, pp. 132–139, Jan.

Salvadori, E;.Sabel, M e.Battiti, R. (2002) “A Reactive Scheme For Traffic Engineering In Mpls Networks. Technical report”, Universitá di Trento, Dipartimento di Informatica e Telecomunicazioni, Dezembro.

Velasco, J. e Magdalena, L. (1995); “Genetic algorithms in fuzzy control systems,” in Genetic Algorithms in Engineering and Computer Science (G. Winter, J. Periaux, M. Galan, and P. Cuesta, eds.), pp. 141–165, John Wiley & Sons.

Wang, B.; Su, X. e Chen, C.P.. (2002) “A New Bandwidth Guaranteed Routing Algorithm for MPLS Traffic Engineering”, In: Proceedings of ICC, volume 2, p.1001-1005, New York – USA.