UNIVERSIDADE FEDERAL DE MINAS GERAIS
ESCOLA DE ENGENHARIA
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA
DOUTORADO EM ENGENHARIA ELÉTRICA

ENGENHARIA DE TRÁFEGO EM DOMÍNIO MPLS UTILIZANDO TÉCNICAS DE INTELIGÊNCIA COMPUTACIONAL

Nilton Alves Maia

Orientador: Prof. Luciano de Errico
Co-orientador: Prof. Walmir Matos Caminhas

Tese apresentada à Banca Examinadora designada pelo Colegiado do Programa de Pós-Graduação em Engenharia Elétrica da Universidade Federal de Minas Gerais, como requisito parcial à obtenção do título de Doutor em Engenharia Elétrica.

Belo Horizonte, Novembro de 2006

À minha esposa **Sônia Beatriz** e a minha filha **Débora**,
aos meus pais **José** (in memoriam) e **Íris**
e aos meus irmãos e irmãs.

AGRADECIMENTOS

Inicialmente, à **Deus** por ter iluminado o meu caminho, e permitido que eu chegasse até aqui.

Aos meus pais **José** (in memoriam) e **Íris**, aos meus irmãos e irmãs **Sônia Maria**, **César**, **Solange**, **Marcos**, **Marcelo**, **Marcílio**, **Simone** e **Cibele**, aos meus sogros **Bráz** e **Nélcia**, ao meu cunhado **Gilberto**, pelo apoio e incentivo em todas as minhas decisões.

Aos meus orientadores, Prof. **Luciano de Errico** e Prof. **Walmir Matos Caminhas**, pela amizade, incentivo, dedicação, orientação, valiosas contribuições ao trabalho e pela oportunidade proporcionada.

Aos professores **Dalton Soares Arantes** e **Linnyer Beatriz Ruiz** pelas valiosas sugestões no Exame de Qualificação e pela presença na banca. Aos professores **Gustavo Guimarães Parma** e **Joaquim Celestino Júnior** pela presença na banca e contribuições à tese.

Aos professores do Programa de Pós-graduação em Engenharia Elétrica da Universidade Federal de Minas Gerais pelo apoio, amizade e ensinamentos proporcionados.

Aos colegas professores da Universidade Estadual de Montes Claros e das Faculdades Santo Agostinho pela torcida.

Ao Professor **Afonso Guimarães** (in memoriam) pelos ensinamentos e exemplo de vida.

Aos amigos **Luiz Llamas Font**, **Silvio Mendes Pinto, Antonio Borges e Guiomar Clara Machado Fonseca** pelos incentivos.

Finalmente, agradeço à minha esposa **Sônia Beatriz** e a minha filha **Débora**, pelo amor, paciência, compreensão e apoio constante.

RESUMO

A necessidade atual de transmissão simultânea de dados, voz e vídeo modificaram o alvo das tecnologias de redes de telecomunicações. Em vez de prover um único tipo de serviço, as redes devem oferecer variados níveis de Qualidade de Serviço (QoS), exigindo novas técnicas para o seu gerenciamento. Uma dessas técnicas é a Engenharia de Tráfego, que permite dinamicamente arranjar a distribuição de tráfego na rede, minimizando congestionamentos, instabilidades ou comprometimento da QoS já acordada. Por outro lado, a crescente complexidade das redes atuais torna a intervenção humana um ponto fraco no processo de gerenciamento e controle. Isto leva à necessidade das redes apresentarem algum tipo de comportamento inteligente, possuindo características como adaptabilidade, tolerância a falhas e robustez a variações ambientais. Este trabalho propõe e desenvolve um sistema de Engenharia de Tráfego, capaz de sustentar tráfego misto (dados, voz e vídeo) com diversos níveis de QoS na rede, utilizando MPLS, princípios de Computação Autonômica e técnicas de Inteligência Computacional. Enquanto o MPLS permite uma implementação efetiva para controle de tráfego e QoS, a Computação Autonômica permite que a rede reaja de forma automática a mudanças de condições que ocorram durante o seu funcionamento, apresentando um comportamento de auto-gerenciamento. Em complemento, as informações de monitoramento são processadas usando heurísticas de Inteligência Computacional, como Lógica Nebulosa, Redes Neurais Artificiais e Algoritmos Genéticos, que fornecem subsídios para a tomada de decisões de forma reativa e proativa. A funcionalidade do sistema foi simulada utilizando o simulador de redes ns2, com diversas situações de tráfego de dados, voz e vídeo cruzando um domínio MPLS. Os resultados mostraram que o sistema foi capaz de implementar novas rotas, levando em conta as necessidades de QoS das aplicações e os recursos disponíveis na rede, e bloquear a admissão em caso de impossibilidade de atendimento aos requisitos especificados. Também foi demonstrada a capacidade de adaptação, com o sistema provendo rotas alternativas em caso de piora ou queda da rota atual, mudança das demandas das aplicações ou surgimento de melhores rotas na rede. Em comparação a cenários sem o uso das técnicas implementadas, a utilização do sistema desenvolvido permitiu, nos piores casos, melhorias de 44% no percentual de utilização média da rede e 52% na relação entre o tráfego oferecido e escoado, além de reduzir a perda média de pacotes a valores próximos de zero.

ABSTRACT

The current need of simultaneous transmission of data, voice, and video has changed the target of telecommunication network technologies. Instead of providing only one type of service, networks must offer different levels of Quality of Service (QoS), demanding new techniques for its management. One of these techniques is Traffic Engineering, which dynamically manages the traffic distribution in the network, minimizing congestions, instabilities or loss of the agreed QoS. On the other hand, the growing complexity of present-day networks turns the human intervention into a weak point of the process of management and control. It requires that networks be able to perform some kind of intelligent behavior, exhibiting characteristics like adaptability, fault tolerance and robustness to environment variations. This work proposes and develops a Traffic Engineering system, capable of supporting mixed traffic (data, voice, and video) with different levels of QoS in the network, using MPLS, principles of Autonomic Computing and techniques of Computational Intelligence. While MPLS permits an effective implementation of traffic control and QoS, Autonomic Computing allows the network to automatically react to environment changes during its operation, producing a self-management behavior. In complement, monitoring data are processed using Computational Intelligence heuristics, like Fuzzy Logic, Artificial Neural Networks, and Genetic Algorithms, which help in taking the necessary decisions for reactive and proactive actions. The system was tested using the network simulator ns2, with different scenarios of data, voice, and video traffic crossing a MPLS domain. The results have shown that the system is capable of implementing new routes, considering the application QoS requirements and the available network resources, and blocking new admissions that could not be attended. Adaptation has also been observed, as replacement routes were provided in case of decay or loss of present routes, change in application requirements, or availability of better routes. In comparison to situations without Traffic Engineering, the implemented system has shown improvements of 44% in the average network utilization and 52% in the offered and delivered load ratio, for the worst cases. The average packet loss was reduced to near zero.

SUMÁRIO

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE QUADROS

LISTA DE ABREVIATURAS E SIGLAS

AC	*Autonomic Computing*
AG	*Algoritmos Genéticos*
AIEC	*Algoritmo de identificação de enlaces críticos*
ARMA	*Autoregressive Moving Average*
ARIMA	*Autoregressive Integrated Moving Average*
ATM	*Asynchronous Transfer Mode*
BGP	*Border Gateway Protocol*
CAC	*Connection Admission Control*
CAL	*Módulo responsável pela criação e ativação das LSP´s*
CARQ	*Coletor das aplicações e requisitos de QoS*
CBR	*Constant Bit Rate (dependendo do contexto)*
CBR	*Constraint Based Routing (dependendo do contexto)*
CCF	*Classificador dos caminhos em famílias*
CIR	*Coletor de informações da rede*
CLPC	*Construtor de lista preliminar de caminhos mais curtos para as LSP´s*
CPMT	*Coletor e processador de medições de tráfego*
CI	*Computational Intelligence*
CR-LDP	*Constraint Routing LDP*
CSPF	*Constrained Shortest Path First*
Custo_cam	*Custo do caminho*
DAV	*Detector de aumento da vazão das aplicações*
DiffServ	*Differenciated Services*
ECL	*Módulo responsável pela escolha dos caminhos para as LSP´s*
EI-MATE	*Elwalid MATE*
EABS	*Erro Absoluto*
EMA	*Erro Médio Absoluto*
ETSI	*European Telecommunications Standards Institute*
FEC	*Forwarding Equivalence Classes*
FIFO	*First In, First Out*
HTTP	*Hypertext Transfer Protocol*
HAGA	*Heuristic Adaptive Genetic Algorithm*
OTCL	*Object Tool Control Language*
IANL	*Identificador de necessidade de LSP´s*
IETF	*Internet Engineering Task Force*
IGP	*Interior Gateway Protocol*
IP	*Internet Protocol*
IntServ	*Integrated Services*
ISP	*Internet Service Provider*
ITU-T	*International Telecommunication Union – Telecom. Standardization Sector*
LDP	*Label Distribution Protocol*
LER	*Label Edge Router*
LSP	*Label Switched Path*
LSR	*Label Switching Router*
LER	*Label Edge Router*
MATE	*MPLS Adaptive Traffic Engineering*

MLP	*MultiLayer Perceptron*
MPLS	*Multiprotocol Label Switching*
NS	*Network Simulator*
NP	*Network Performance*
OAG	*Módulo de otimização utilizando algoritmos genéticos*
OTCL	*Object Tool Control Language*
OSPF	*Open Shortest Path First*
QoS	*Quality of Service*
QoSR	*QoS Routing*
QTA	*Queue Tuning Algorithm*
RelBV	*Relação folga / vazão*
RelAT	*Relação atraso máximo / atraso representativo do caminho*
RCPT	*Reconhecimento e classificação dos perfis de comportamento de tráfego das Aplicações*
RTeo	*Relação entre o tráfego escoado e o oferecido ao domínio MPLS*
RSVP	*Resource ReSerVation Protocol*
RSVP-TE	*Resource ReSerVation Protocol for Traffic Engineering*
RNA	*Redes Neurais Artificiais*
SDP	*Shortest Distance Path*
SGA	*Simple Genetic Algorithm*
SIF	*Sistema de Inferência Nebulosa*
SLS	*Service Level Specification*
TCL	*Tool Control Language*
TCP/IP	*Transfer Control Protocol/Internet Protocol*
TE	*Traffic Engineering*
TI	*Tecnologia da Informação*
UDP	*User Datagram Protocol*
VBR	*Variable Bit Rate*
WFQ	*Weighted Fair Queueing*
WSP	*Widest Shortest Path*

CAPÍTULO 1

INTRODUÇÃO

As telecomunicações têm apresentado um grande crescimento nos últimos anos e indubitavelmente continuarão no mesmo ritmo. Há uma expectativa de integração das redes de voz e dados em uma única rede de alta velocidade na qual aplicações vão se utilizar intensivamente de recursos multimídia. Dentre estas novas aplicações, podem-se citar o comércio eletrônico, videoconferência, voz sobre IP, educação a distância, entre outras. Essas aplicações demandam monitoração e controle de Qualidade de Serviços (QoS, *Quality of Services*) e uma crescente capacidade de tráfego, implicando em consumo excessivo da largura de banda, recurso considerado escasso que deve ser compartilhado por todas essas aplicações. A necessidade de transmissão de voz, dados e multimídia, de certa forma, modifica o alvo das tecnologias de redes. Assim, ao invés de fornecer um único tipo de serviço, as redes agora têm que se confrontar com a integração de variados tipos de serviços, além de também prover a Qualidade de Serviço.

Para prover serviços com garantia de qualidade, há necessidade de novos mecanismos e protocolos, que atendam aos novos requisitos dessas aplicações. Atualmente, existem algumas propostas para prover um conjunto de extensões ao tradicional modelo de melhor-esforço, na tentativa de fornecer uma melhor QoS. Algumas destas propostas são os Serviços Integrados/RSVP, Serviços Diferenciados, MPLS *(Multi Protocol Label Switchin*g), Roteamento baseado em QoS e Engenharia de Tráfego (TE, *Traffic Engineering*). Por outro lado, a proliferação de tecnologias de internet, serviços e dispositivos, tem tornado as ferramentas atuais de projeto e gerenciamento incapazes de projetar de forma confiável e segura os sistemas e serviços de rede. De fato, têm-se alcançado um nível de complexidade, heterogeneidade, e rápida taxa de mudanças que a informação sobre a infra-estrutura está se tornando intratável e insegura (DONG et al.,

2003). Isto tem levado os pesquisadores a considerar técnicas alternativas de projeto e gerenciamento que são baseadas em estratégias usadas pelos sistemas biológicos para tratar a complexidade, heterogeneidade e incerteza. Esta abordagem é referenciada como Computação Autonômica (AC, *Autonomic Computing*). Este trabalho utiliza princípios de Computação Autonômica, MPLS, TE e técnicas de Inteligência Computacional (CI, *Computational Intelligence*), visando oferecer QoS às aplicações exigentes.

1.1 Objetivos

1.1.1 Geral

Desenvolver um sistema de Engenharia de Tráfego, capaz de sustentar tráfego misto (dados, voz, vídeo) com QoS na rede, utilizando MPLS, princípios de Computação Autonômica e técnicas de Inteligência Computacional.

1.1.2 Específicos

Os objetivos específicos do trabalho são:

a) Desenvolver um algoritmo para descoberta dos caminhos, criação e ativação das LSP´s **(**Label Switching Paths**)**.

A descoberta dos enlaces mais adequados para a interligação de um nó origem da aplicação a um nó destino, através de uma LSP é realizada levando-se em conta a prioridade, as necessidades de QoS da aplicação (vazão e atraso fim-a-fim) e os resultados das medições de tráfego efetuadas nos enlaces. A construção das LSP`s é implementada utilizando lógica nebulosa. O algoritmo proposto para descoberta dos caminhos, criação e ativação das LSP´s utiliza o principio de Computação Autonômica denominado Auto-configuração.

b) Desenvolver um algoritmo para monitoração e re-roteamento de LSP´s.

O re-roteamento de LSP´s pode ocorrer por diferentes razões, tais como a necessidade de proporcionar QoS para as aplicações prioritárias, interrupção de um enlace que faz parte da LSP principal, otimização dos recursos da rede, etc.

O sistema de TE aqui proposto tem como objetivo manter a QoS das aplicações. Caso a LSP corrente não esteja mantendo os níveis de QoS desejados, deve ser providenciado o estabelecimento de uma nova. Por outro lado, caso as aplicações com maior prioridade aumentem a sua vazão de tráfego e ultrapassem o valor combinado, se possível o sistema deve procurar atende-las procurando um caminho mais adequado. O principio de Computação Autonômica utilizado neste caso é a Auto-configuração. Além disso, após a ocorrência de interrupção em um enlace que atende a LSP principal, os pacotes deverão ser encaminhados para uma LSP reserva (alternativa). A LSP reserva é criada e ativada automaticamente. Para esta situação, o principio de Computação Autonômica utilizado é a Auto-cura.

c) Desenvolver um algoritmo para controle de admissão de conexão (*CAC – Connection Admission Control*) utilizando reconhecimento e classificação dos perfis de comportamento de tráfego das aplicações.

O sistema de TE deve controlar a admissão do tráfego das aplicações no domínio MPLS. Levando-se em conta a prioridade, o sistema de TE deve tentar o estabelecimento de LSP´s para todas as aplicações. Caso isto não seja possível, o tráfego das aplicações que não conseguiram o estabelecimento de LSP´s é bloqueado. Além disso, as aplicações que mantêm as suas taxas de vazão de tráfego dentro dos limites acordados devem ser atendidas com uma maior prioridade. As aplicações que não cumprem o acordo são atendidas, mas devem ser “penalizadas”. A idéia é alterar a prioridade de admissão das aplicações levando em conta a análise dos seus comportamentos registrados em arquivo de histórico. O valor da prioridade de atendimento de uma aplicação é alterado, subtraído-se um valor proporcional ao perfil de comportamento de tráfego identificado. O algoritmo proposto utiliza em parte o principio de Computação Autonômica denominado Auto-proteção.

d) Desenvolver um algoritmo para otimização dos recursos da rede.

A otimização dos recursos da rede deve ser feita periodicamente. A meta da otimização é fazer com que as LSP´s que atendem as aplicações, utilizem os melhores caminhos possíveis. Assim, um caminho que num determinado momento é escolhido como a melhor opção para o estabelecimento de uma LSP, pode deixar de sê-lo dependendo das novas condições da rede. A otimização dos recursos da rede é implementada utilizando Algoritmos Genéticos. O algoritmo proposto utiliza o principio de Computação Autonômica denominado Auto-otimização.

e) Desenvolver um algoritmo para realizar a identificação dos enlaces críticos e sinalizar a necessidade de ampliação de suas capacidades de forma proativa.

Numa rede real o número de aplicações estabelecidas pode crescer com o tempo. Se a infra-estrutura da rede permanece a mesma, é lógico esperar que em um determinado momento futuro, alguns enlaces da rede apresentarão congestionamento. Portanto, deve haver uma forma de identificar os enlaces mais críticos e depois prever o comportamento do tráfego, para que periodicamente seja providenciada a ampliação de suas capacidades. A previsão da vazão de tráfego nos enlaces da rede é feita utilizando Redes Neurais Artificiais (RNA). O algoritmo proposto utiliza o principio de Computação Autonômica denominado Auto-cura.

f) Avaliar os resultados obtidos com a implementação da TE.

O sistema de TE é implementado em um domínio MPLS simulado com a utilização do ns2 (VINT, 2003). A avaliação do sistema de TE é feita analisando-se a QoS oferecida às aplicações que cruzam o domínio MPLS e também os indicadores de desempenho da rede. O domínio MPLS é alimentado com aplicações do tipo dados, voz e vídeo. Os parâmetros utilizados para a avaliação da QoS oferecida às aplicações são atraso, jitter, vazão e perda de pacotes. As medições dos parâmetros de QoS são realizadas fim-a-fim. Os indicadores utilizados para avaliar o desempenho da rede são a utilização média dos enlaces e a relação entre o tráfego oferecido e o escoado (RTeo). Além disso, são avaliadas

também a perda média de pacotes obtida para o conjunto de aplicações com e sem a implementação da TE.

1.2 Justificativa

A proliferação de tecnologias de internet, serviços e dispositivos, tem tornado as ferramentas atuais de projeto e gerenciamento de redes incapazes de projetar de forma confiável e segura os sistemas e serviços de rede. De fato, têm-se alcançado um nível de complexidade, heterogeneidade, e rápida taxa de mudanças que a informação sobre a infra-estrutura está se tornando intratável e insegura (DONG et al., 2003). Com o crescimento na complexidade, a habilidade dos sistemas de comunicações em se adaptar às mudanças do tipo e intensidade do tráfego escoado na rede, torna-se cada vez mais importante. A capacidade de adaptação é, portanto, uma característica desejada quando se pensa em reusar, gerenciar e manter soluções existentes, ou introduzir novas soluções. A capacidade de adaptação leva à necessidade dos sistemas apresentarem algum tipo de comportamento inteligente, similar ao exibido por muitos sistemas biológicos. Estes apresentam características como adaptabilidade, tolerância a falhas e robustez a variações ambientais. Tais características são bastante desejáveis, levando ao surgimento de diversas pesquisas que procuram emular alguns dos aspectos observados em sistemas biológicos naturais. Duas áreas surgidas a partir destas pesquisas são Inteligência Computacional e Computação Autonômica.

A CI compreende paradigmas computacionais que procuram desenvolver sistemas que apresentam alguma forma de inteligência similar à exibida por determinados sistemas biológicos. Alguns dos paradigmas que compõem a CI foram de fato inspirados em sistemas biológicos (como as Redes Neurais Artificiais), enquanto que outros, apesar de não terem inspiração biológica, tentam gerar sistemas que produzam algum tipo de comportamento próximo ao observado em sistemas naturais (como por exemplo, o raciocínio aproximado dos sistemas nebulosos).

A crescente complexidade de gerenciamento das redes alcança níveis além da habilidade humana. Como conseqüência, a intervenção humana se torna um ponto fraco no processo de gerenciamento e controle. Isto tem levado os pesquisadores a considerar técnicas alternativas de projeto e gerenciamento que são baseadas em estratégias usadas

pelos sistemas biológicos para tratar a complexidade, heterogeneidade e incerteza. Uma abordagem que surgiu a partir destas pesquisas é referenciada como Computação Autonômica.

Numa rede IP comum, quando um roteador recebe um pacote, ele faz uma busca na sua tabela de roteamento e então, baseado no endereço IP do pacote, decide para onde enviá-lo. Essa busca pode levar bastante tempo, dependendo do tamanho da tabela de cada roteador. O MPLS (Multiprotocol Label Switching) rompe com esse paradigma, usando um rótulo de tamanho fixo a partir do qual o roteador decide por onde enviar os pacotes. No caso do MPLS, são estabelecidas conexões virtuais através de enlaces e roteadores do domínio**.** Estes caminhos que os pacotes percorrem a partir do roteador de entrada até o roteador de saída são chamados de LSPs (Label Switching Paths). A possibilidade de estabelecimento destes caminhos facilita a implementação da TE.

A TE por sua vez é o processo de arranjar como o tráfego flui através da rede para que congestionamentos causados pela utilização desigual dos enlaces possam ser evitados. A TE pode ser feita manualmente, ou usando algum tipo de técnica automatizada para descobrir e fixar os caminhos mais adequados a determinadas agregações de fluxos dentro da rede, como é proposto neste trabalho. Alem disso, uma característica importante da TE deve ser a capacidade de adaptação às mudanças nas condições da rede. Este fato leva a necessidade da TE apresentar algum tipo de comportamento inteligente, utilizando técnicas de auto-gerenciamento baseadas em estratégias usadas pelos sistemas biológicos. Portanto, o sistema de TE deve então utilizar técnicas de CI e apresentar características de um sistema autonômico, ou seja, aquele que tem a capacidade de se auto-proteger, auto-curar, auto-configurar e auto-otimizar. Estas características fundamentais que o sistema de TE deve apresentar, influenciaram decisivamente na opção pelo uso da CI e princípios de Computação Autonômica neste trabalho.

1.3 Contribuições

A contribuição do presente trabalho é o desenvolvimento de um sistema de TE, capaz de sustentar tráfego misto (dados, voz, vídeo) com diversos níveis de QoS na rede, utilizando MPLS, princípios de Computação Autonômica e técnicas de Inteligência Computacional. A decisão de criar as LSP´s para as aplicações prioritárias no domínio

MPLS, a escolha dos caminhos por onde elas serão estabelecidas, o re-encaminhamento automático de tráfego para LSP reserva em caso de falhas em enlaces, o re-encaminhamento devido ao aumento da vazão das aplicações, e também a otimização periódica do uso dos enlaces da rede, são executados pelo próprio sistema de forma automática. A meta da otimização periódica é fazer com que as LSP´s que atendem as aplicações, utilizem os melhores caminhos possíveis. É proposta também a implementação de um algoritmo para controle de admissão de conexões utilizando o reconhecimento e a classificação dos perfis de comportamento de tráfego das aplicações. Levando-se em conta a prioridade, o sistema de TE deve tentar o estabelecimento de LSP´s para todas as aplicações. Caso isto não seja possível, os tráfegos das aplicações que não conseguiram o estabelecimento de LSP´s são bloqueados. Além disso, as aplicações que mantêm as suas taxas de vazão de tráfego dentro dos limites acordados devem ser atendidas com uma maior prioridade. As aplicações que não cumprem o acordo são atendidas, mas devem ser "penalizadas". A prioridade de admissão das aplicações são alteradas a partir da análise dos seus comportamentos registrados num histórico. O valor da prioridade é alterado, subtraído-se um valor proporcional ao perfil de tráfego identificado. É proposta também a implementação de um algoritmo de identificação de enlaces críticos e sinalização da necessidade de ampliação de suas capacidades de forma proativa. O algoritmo utiliza Redes Neurais Artificiais para a previsão da vazão de tráfego nos enlaces do domínio MPLS. A metodologia utilizada para implementação do sistema de TE é apresentada no Capítulo 3.

1.4 Estrutura do trabalho

Esta tese está organizada da seguinte forma. O capítulo 2 apresenta os fundamentos conceituais mostrando um resumo sobre sistemas multimídia, abordagens para o oferecimento de QoS em redes IP, conceitos de CI, conceitos de Computação Autonômica e resumos de alguns trabalhos relacionados ao tema desta tese; o capítulo 3 apresenta a descrição da metodologia utilizada para a implementação do sistema de TE; o capítulo 4 apresenta as descrições das implementações do sistema de TE em um domínio MPLS; o capítulo 5 apresenta os resultados obtidos com as simulações; e finalmente, o capítulo 6 apresenta as conclusões e trabalhos futuros.

CAPÍTULO 2

FUNDAMENTOS CONCEITUAIS

É apresentada neste capítulo, uma introdução aos conceitos teóricos utilizados neste trabalho. Inicialmente, na seção 2.1 é feita uma introdução aos sistemas multimídia. A seção 2.2 trata sobre QoS. A seção 2.3 descreve algumas abordagens para prover QoS em redes IP, tais como os Serviços Integrados, Serviços Diferenciados, MPLS, Roteamento baseado em QoS e Engenharia de Tráfego. A seção 2.4 trata de Computação Autonômica. A seção 2.5 trata de Inteligência Computacional. A seguir, na seção 2.6 são apresentados os resumos de alguns trabalhos relacionados ao tema desta tese. Finalmente, na seção 2.7 são feitas considerações finais sobre os itens tratados neste capítulo.

2.1 Sistemas Multimídia

Os sistemas multimídia têm revolucionado os atuais estilos de vida, especialmente aqueles aspectos relacionados com a comunicação humana. Esses sistemas podem criar um mundo eletrônico em que as pessoas podem comprar, trabalhar, ou aprender em casa, assistir programas de vídeo sob demanda, acessar livrarias digitais on-line de um *desktop*, e assim por diante. Os avanços tecnológicos nos computadores, redes de alta velocidade, compressão de dados estão acelerando a realização de tais sistemas e também atraindo a atenção da sociedade como um todo. Integrar diferentes tipos de mídias como texto, áudio e vídeo em um vasto domínio de aplicações é uma das principais tendências de nossa época.

Multimídia é apenas duas ou mais mídias (TANEMBAUM, 1996). Por exemplo, um livro contém duas mídias: texto e imagens (figuras e gráficos). Quando nos referimos a multimídia, ela geralmente significa a combinação de duas ou mais mídias contínuas,

também chamadas mídias dinâmicas ou isócronas. Na prática, as duas mídias são geralmente áudio e vídeo, que nada mais é do que som mais o movimento de figuras. Assim, multimídia denota a ação de tratamento integrado de alguma informação representada como dados de mídia contínuos (tais como áudio e vídeo) bem como alguma informação codificada como dados de mídia discretos (tais como texto e imagens) (WOLF, 1997).

Os dados de mídia contínuos ou dependentes do tempo (áudio e vídeo) são aqueles em que o tempo faz parte da semântica da apresentação, ou seja, seus significados dependem da taxa em que são apresentados. Por exemplo, um vídeo possui uma seqüência de quadros que deve ser apresentado, cada um deles, com uma certa duração que deve ser preservada para obter uma apresentação com qualidade. Os dados de mídia discretos não possuem esta relação no tempo, sendo que, tradicionalmente, são utilizados em documentos impressos.

Por outro lado, existe uma expectativa de integração das redes de voz e dados em uma única rede de alta velocidade, na qual aplicações vão se utilizar intensivamente de recursos multimídia. Essas aplicações demandam controle de QoS e uma crescente capacidade de tráfego, implicando em consumo excessivo da largura de banda. A tecnologia de QoS de "melhor esforço" (BE, *Best Effort*), da Internet, transmite cada um dos pacotes de forma independente, onde a qualidade no envio do pacote depende da carga momentânea da rede. A rede faz o seu "melhor esforço" para encaminhar todos os pacotes, mas, às vezes é obrigada a descartar pacotes devido a congestionamentos. Desta forma, para oferecer garantias de desempenho às aplicações multimídia, comércio eletrônico e educação à distância na Internet, surgiu uma nova demanda por Qualidade de Serviço. O item seguinte apresenta algumas considerações sobre QoS em redes IP.

2.2 Qualidade de Serviço

A Qualidade de Serviço em redes de comunicação é um aspecto operacional fundamental para o desempenho fim-a-fim das novas aplicações (como Voz sobre IP e multimídia em geral). Na visão da ETSI (*European Telecommunications Standards Institute*), ITU-T e IETF, existem três noções definidas de QoS: a intrínseca, a percebida e a avaliada, que constituem o modelo geral mostrado na Figura 1 (GOZDECKI et al, 2003).

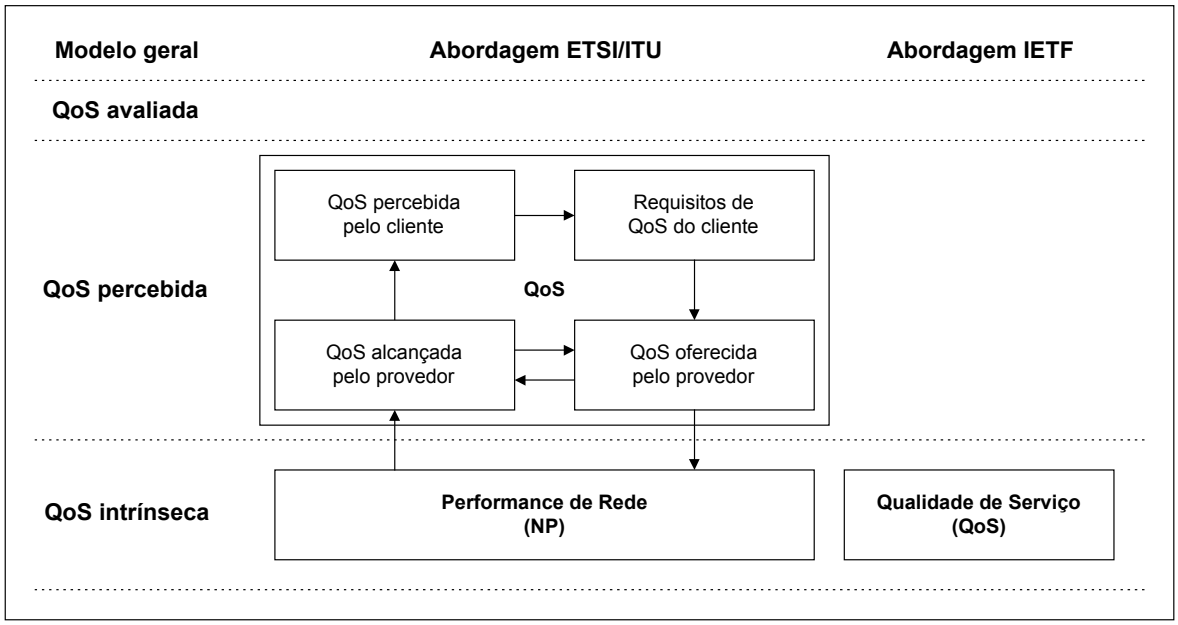

Figura 1 - O modelo geral de QoS com as abordagens da ETSI/ITU e da IETF (GOZDECKI et al, 2003).

As características do serviço descritas sob a forma de aspectos técnicos pertencem à noção de QoS intrínseca, que é determinada pelo projeto da rede de transporte e pelo provisionamento da rede de acesso, terminações e conexões (HARDY, 2001). A QoS intrínseca é obtida pela comparação das características de desempenho medida e esperada. A percepção do usuário do serviço não influencia no *rating* da QoS intrínseca. A QoS requerida é alcançada principalmente pela seleção apropriada dos protocolos de transporte, dos mecanismos de garantia de QoS e dos valores relativos dos parâmetros.

A QoS percebida reflete a experiência do usuário empregando um serviço particular. Ela é influenciada pelas expectativas do usuário quando ele compara o desempenho observado do serviço. Por sua vez, as expectativas pessoais são usualmente afetadas pela experiência do usuário com um serviço similar prestado por outro provedor e pela opinião de outros usuários. Assim, a QoS com as mesmas características intrínsecas pode ser percebida diferentemente por vários usuários.

A QoS avaliada começa a ser notada quando o usuário decide se continua utilizando o serviço ou não. Essa decisão depende da qualidade percebida, preço do serviço e respostas do provedor a queixas e problemas submetidos.

A declaração de um nível satisfatório para QoS intrínseca, para QoS percebida e para QoS avaliada pode ser considerada separadamente. A primeira declaração é de responsabilidade do provedor de rede e depende da arquitetura da rede, do planejamento e do gerenciamento.

O uso apropriado das capacitações da QoS intrínseca ajustadas para um serviço particular oferecido, juntamente com análises de mercado, são necessárias para assegurar um alto nível na QoS percebida. Esta obrigação é do provedor de serviço e esforços de propaganda e de marketing têm impacto positivo na QoS percebida (GOZDECKI et al, 2003).

A ETSI e a ITU-T (ITU-T, 1993; ITU-T, 2001) abordam a terminologia relacionada com QoS de forma semelhante. A definição de QoS dada na Recomendação E.800 (ITU-T, 1993) é "*o efeito coletivo do desempenho do serviço que determina o grau de satisfação de um usuário do serviço"*". Portanto, a abordagem da ETSI/ITU é mais aderente ao conceito da QoS percebida do que ao conceito da QoS intrínseca. Diante disso, a ITU-T introduziu a noção de desempenho de rede (NP*, Network performance*) para cobrir os aspectos técnicos. Então, é feita uma clara distinção entre QoS, entendida como algo focado nos efeitos percebíveis pelo usuário, e NP, englobando todas as funções de rede essenciais para prover o serviço.

A NP corresponde à QoS intrínseca. Esse aspecto está definido na Recomendação E.800 (ITU-T, 1993) como *a habilidade de uma rede ou porção de rede em prover as funções relacionadas às comunicações entre usuários*. A NP é definida e tratada em termos de parâmetros de elementos de rede particulares envolvidos no fornecimento de um serviço. Esses parâmetros são a chave para a eficiência e a eficácia da rede no fornecimento de um serviço (GOZDECKI et al, 2003).

Para cobrir os vários pontos de vista de QoS por parte do usuário e do provedor, a ITU-T e a ETSI distinguem quatro definições particulares, que são (GOZDECKI et al, 2003): requisitos de QoS do usuário; QoS oferecida pelo provedor; QoS alcançada pelo provedor e QoS percebida pelo usuário.

Por outro lado, a IETF focaliza a QoS intrínseca e não trabalha com a QoS percebida. O conceito de QoS é entendido pela IETF como sendo *um conjunto de requisitos de serviço a serem satisfeitos pela rede enquanto transportando um fluxo* (CRAWLEY, 1998). Esta definição é quase equivalente à noção de NP definida pela ITU-T/ETSI, sendo

definida em termos de parâmetros. O presente trabalho enfoca o conceito de QoS conforme a visão da IETF, ou seja, uma vez acordado com o usuário (ou cliente) um conjunto de requisitos de serviços, a rede (provedor) deve atender tais requisitos enquanto estiver transportando os fluxos objeto desse acordo.

A QoS intrínseca em redes é expressa ao menos pelo seguinte conjunto de parâmetros que são significativos para a maioria dos serviços baseados em IP ou seja, vazão, atraso fim-a-fim, jitter e perda de pacotes (GOZDECKI et al, 2003).

O atraso fim-a-fim é o tempo necessário para um pacote percorrer a rede, medido do momento em que é transmitido pelo emissor até ser recebido pelo receptor.

O jitter é a variação no intervalo entre chegadas de pacotes, introduzida pelo comportamento aleatório do atraso na rede. A variação do atraso pode provocar uma distorção na informação recebida.

A perda de pacotes representa o percentual de pacotes que foram transmitidos na rede, mas não alcançaram seu destino em um determinado período de tempo.

A vazão é o volume de dados movidos de um nó de rede para outro em um dado período de tempo, também expressa em Kbps ou Mbps (OLIVEIRA, 2001).

O Quadro 1 apresenta os requisitos desejáveis de QoS para algumas aplicações de dados, áudio e vídeo (ITU-T, 2002).

QUADRO 1

Requisitos desejáveis de QoS para algumas aplicações de dados, áudio e vídeo.

Tipo	Aplicação	vazão típica	atraso fim-a-fim	variação de atraso	perda de pacotes
áudio	Conversação de voz	4-64 kbit/s	< 150 ms preferido < 400 ms limite	< 1 ms	< 3%
áudio	Mensagem de voz	4-32 kbit/s	< 1 s para reprodução < 2 s para gravação	< 1 ms	< 3%
áudio	Fluxo de áudio de alta qualidade	16-128 kbit/s	< 10 s	<< 1 ms	< 1%
vídeo	Videofone	16-384 kbit/s	< 150 ms Preferido <400 ms limite	-	< 1%
Dados	Web-browsing – HTML	~10 KB	Preferido < 2 s /pagina Aceitável < 4 s/pagina	N.A.	Zero
dados	Transferência /Recuperação de arquivos	10 KB-10 MB	Preferido < 15 s Aceitável < 60 s	N.A.	Zero
Dados	Jogos interativos	< 1 KB	< 200 ms	N.A.	Zero
Dados	Telnet	< 1 KB	< 200 ms	N.A.	Zero
Dados	E-mail (acesso a servidor)	< 10 KB	Preferido < 2 s Aceitável < 4 s	N.A.	Zero
Dados	E-mail (transferência servidor-servidor)	< 10 KB	Pode ser vários minutos	N.A.	Zero

Fonte: Recomendação G.1010 (ITU-T, 2002).

2.3 Abordagens para o oferecimento de QoS em redes IP

Neste item são apresentadas as principais abordagens para o oferecimento de QoS em redes baseadas no protocolo IP.

2.3.1 A Arquitetura de Serviços Integrados (*IntServ*)

Na Internet de hoje, todos os pacotes recebem a mesmo tratamento, ou seja, é oferecido um único modelo de serviço, chamado de "melhor esforço". O modelo de "melhor esforço" apresenta um desempenho razoável para aplicações elásticas, como por exemplo, correio eletrônico, transferência de arquivos, consultas interativas a informações e aplicações cliente/servidor tradicionais. Entretanto, para as aplicações de tempo real que possuem grande sensibilidade com relação ao atraso fim-a-fim, o modelo de "melhor esforço" é inadequado. Desta forma, a IETF criou o grupo de trabalho IntServ para viabilizar o surgimento de uma rede de serviços integrados. O termo serviços integrados é empregado para designar um modelo de serviços para a Internet que inclui o serviço de melhor esforço, serviços de tempo real e serviços de compartilhamento controlado de enlace (BRADEN; CLARK; SHENKER, 1994).

O modelo de Serviços Integrados é composto por quatro componentes, o escalonador de pacotes, o classificador, o controle de admissão e o protocolo de reserva de recursos.

O Escalonador de pacotes gerencia o encaminhamento dos vários fluxos de pacotes. Outro componente importante que pode ser considerado parte do escalonador de pacotes é o avaliador, que mede características de tráfego dos fluxos para auxiliar o escalonamento de pacotes e o controle de admissão.

O Classificador mapeia os pacotes que chegam em determinadas classes, onde todos os pacotes em uma classe recebem o mesmo tratamento.

O Controle de admissão implementa o algoritmo que possibilita a um roteador determinar se um novo fluxo pode ter seu pedido de QoS atendido sem interferir nas garantias feitas anteriormente. Alguns fluxos podem ter seus pedidos de recursos rejeitados por falta de recursos em algum dos roteadores.

O modelo *IntServ* é baseado na reserva de recursos, ou seja, antes dos dados serem transmitidos as aplicações devem primeiro configurar os caminhos e reservar recursos na

rede. Em princípio, a reserva de recursos pode ser executada por qualquer protocolo que seja compatível com o modelo de serviços integrados, mas na prática o protocolo RSVP (Resource Reservation Protocol) (BRADEN, 1997) é o padrão de fato. O RSVP é utilizado pelos roteadores para repassar as requisições de QoS para todos os outros roteadores que estiverem no caminho entre fonte e destino e para estabelecer e manter informações de estado que possibilitam oferecer o serviço desejado.

O modelo de Serviços Integrados propõe duas classes de serviço em adição ao Serviço de "melhor esforço", o Serviço Garantido e o Serviço de Carga Controlada.

O Serviço Garantido (SHENKER; PARTRIDGE; GUERIN, 1997) fornece limites em termos de atrasos de enfileiramento que os pacotes sofrerão nos roteadores. Ele garante tanto o atraso quanto a taxa de bits. Uma sessão requerendo o Serviço de Carga Controlada (WROCLAWSKI, 1997) receberá uma qualidade de serviço muito próxima da qualidade que um fluxo poderia receber de uma rede não sobrecarregada.

A arquitetura de Serviços Integrados é baseada no conceito de que todas as informações de estado relacionadas aos fluxos deveriam estar nos sistemas finais. Neste sentido, existem alguns problemas com a arquitetura Serviços Integrados (FERGUSON; HUSTON, 1998; XIAO; NI, 1999):

- O montante de informações de estado aumenta proporcionalmente ao número de fluxos. Isto causa uma sobrecarga de armazenamento e processamento nos roteadores. Portanto esta arquitetura não é escalável;
- Os requisitos nos roteadores são altos: todos os roteadores devem implementar RSVP, controle de admissão, classificação e escalonamento de pacotes;
- Para Serviço Garantido, toda a rede deve suportar *IntServ*;
- *IntServ*/RSVP não é muito adequado às aplicações do tipo navegadores WWW, onde a duração de um fluxo típico é apenas de poucos pacotes. A sobrecarga causada pela sinalização RSVP poderia facilmente deteriorar o desempenho da rede percebida pela aplicação.

2.3.2 A Arquitetura de Serviços Diferenciados

Devido às dificuldades de implementar e utilizar *IntServ*/RSVP, os Serviços Diferenciados (*DiffServ*) foram introduzidos. O tratamento oferecido por essa abordagem é

efetuado sobre uma agregação de fluxos e não sobre um fluxo individual. Os pacotes que não pertencem a nenhum fluxo contratado recebem o serviço padrão, o qual é equivalente ao serviço de melhor esforço atualmente oferecido pela Internet.

Na arquitetura DiffServ os roteadores são agrupados em domínios, sendo que a posição de um roteador no domínio determina se ele é de núcleo ou borda. Um domínio DiffServ normalmente consiste de uma ou mais redes sob uma mesma administração, como por exemplo, uma intranet de organização ou um provedor de facilidades de comunicações (*ISP, Internet Service Provider*) (BLAKE et all., 1998). A classificação e marcação de pacotes ocorrem nas bordas do domínio. Os pacotes são marcados diferentemente nos roteadores da borda para criar várias classes ou agregações de fluxos. O encaminhamento das agregações é feito segundo uma política de Comportamento por Nó (*Per-Hop Behavior*, PHB). Os PHBs definem o encaminhamento de pacotes em cada nó DiffServ. PHBs são identificados através da marcação efetuada em um campo do pacote chamado DSCP (*Differentiated Services Code Point)*. O campo DSCP é obtido pela renomeação do campo TOS (*Type of Service*), no caso do IPv4, ou do campo *Traffic Class*, no caso do IPv6. A identificação das agregações de fluxos (PHBs) no interior de um domínio DiffServ é efetuada analisando-se o campo DSCP marcado no nó de borda

Existem duas propostas de PHB para DiffServ: o PHB EF (*Expedited Forwarding*) e o PHB AF (*Assured Forwarding*). O PHB EF pode ser utilizado para a obtenção de um serviço fim-a-fim com baixa perda, baixo atraso, baixo *jitter* e largura de banda suficiente. Já o PHB AF garante um desempenho melhor que o de um tráfego tipo “melhor esforço”. Além dos PHB EF e PHB AF existe também o PHB default, que serve para escoar o tráfego de “melhor esforço”, assegurando compatibilidade com o encaminhamento padrão em todos os roteadores.

2.3.3 MPLS

Na Internet, ao receber um pacote, um roteador faz uma busca em sua tabela de roteamento e, baseado no endereço IP do pacote, decide para onde enviá-lo. Essa busca pode levar bastante tempo, dependendo do tamanho da tabela de cada roteador. O MPLS (*Multiprotocol Label Switching*) consiste em encaminhamento de pacotes IP baseado em rótulos (labels) ao invés de endereços. Neste sentido, o MPLS acrescenta a noção de

encaminhamento de pacotes orientado a conexão nas redes IP. Com isso, permite-se às redes IP o estabelecimento e uso de caminhos de tráfego, ou seja, as rede MPLS possuem a flexibilidade das redes sem conexão e as vantagens inerentes às redes orientadas a conexão (GIRISH et al., 2000). Os caminhos que os pacotes percorrem de um roteador a outro são chamados de LSPs (Label Switching Paths). O MPLS desvincula as funções de controle e encaminhamento, atribuindo um pacote a uma Classe de Equivalência de Encaminhamento (FEC, *Forwarding Equivalence Classe*) específica apenas uma vez, quando o pacote entra na rede. A informação sobre a FEC à qual um pacote pertence é então codificada como um rótulo que é inserido no pacote. Nos roteadores subseqüentes o cabeçalho do pacote não é mais analisado e não há mais busca na tabela de roteamento. O rótulo é utilizado como um índice em uma tabela que especifica o próximo salto e um novo rótulo. O roteador seguinte troca o rótulo antigo pelo rótulo novo e encaminha o pacote para o próximo salto.

Existem dois tipos de LSP dependendo do método usado para determinar o caminho (LEMMA, 2003): LSP com roteamento "hop-by-hop" e LSP com roteamento explícito. No caso do roteamento "hop-by-hop" cada LSR escolhe de forma independente o próximo "hop" para cada FEC. No roteamento explícito, um único LSR especifica por quais LSR´s vai passar a LSP. Além disso, no caso da LSP com roteamento explícito o caminho escolhido deve levar em conta algumas restrições como largura de banda, requisitos de QoS e políticas administrativas. O roteamento explicito utiliza o CR-LDP (*Constraint Routed Label Distribution Protocol*) (JAMOUSSI et al., 2002) ou o RSVP-TE (*Resource Reservation Protocol for Traffic Engineering Extensions*) (AWDUCHE et al., 2001) como protocolo de sinalização.

O LSR (*Label Switch Routers*) ou dispositivo da borda de entrada do domínio é também denominado Roteador de Borda (LER, *Label Edge Router*) MPLS. O LSR de entrada ou LER é responsável pela inserção do rótulo no pacote e atribuição dos pacotes a uma FEC. Este processo de ligação de pacotes a uma FEC é efetuado somente na admissão do pacote como foi descrito. Quando um LER está na saída do domínio MPLS, ele é responsável pela retirada do rótulo, mantendo a semântica normal de um pacote IP, a fim de ser entregue a uma rede não MPLS. A Figura 2 mostra os diversos dispositivos de uma rede de MPLS.

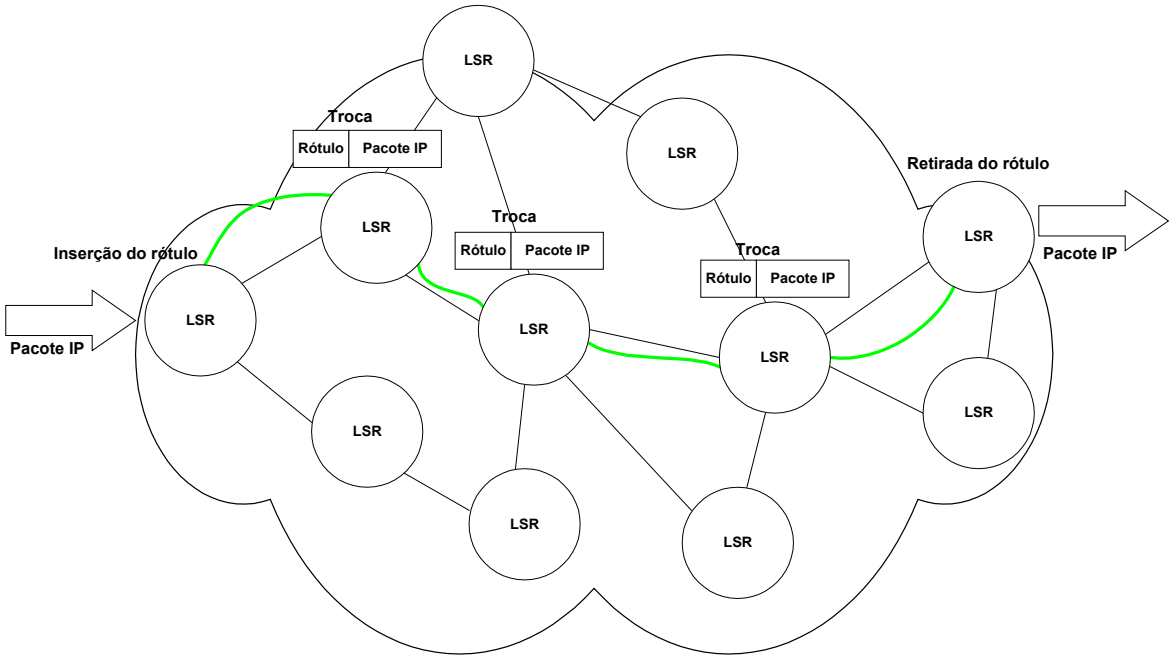

Figura 2 – Dispositivos de um domínio MPLS.

Os LSR´s do interior do domínio fazem as trocas dos rótulos possibilitando o encaminhamento do pacote para o roteador MPLS seguinte. Os LSRs de um domínio MPLS comunicam-se através de um protocolo adequado, a fim de manter atualizadas as tabelas de encaminhamento do domínio.

À medida que o pacote identificado pelo MPLS atravessa os LSRs, ele finalmente atinge a saída da nuvem MPLS, ou seja o LSR da borda de saída ou LER. Qualquer dispositivo que esteja fora deste LER não será capaz de receber os dados identificados pelo MPLS. Este último dispositivo de MPLS remove o rótulo, ou identificador do MPLS do pacote, e entrega um pacote de IP bruto.

Uma das principais aplicações para MPLS hoje é a Engenharia de Tráfego (TE). A TE permite alterar o caminho normal que alguns pacotes seguiriam caso fossem encaminhados pelo esquema convencional, ou seja, pelo caminho mais curto, escolhido pelo protocolo de roteamento. O MPLS consegue forçar pacotes a seguirem certas rotas preestabelecidas, o que é impossível no esquema convencional.

2.3.4 Roteamento baseado em QoS (QoSR)

O roteamento baseado em QoS (ASH, 2001) é um mecanismo de roteamento que seleciona o caminho percorrido pelos pacotes de um fluxo baseado no conhecimento da

disponibilidade de recursos da rede, bem como nos requisitos de QoS dos fluxos, tais como largura de banda e atraso.

O Roteamento Baseado em Restrições (CBR, *Constraint Based Routing*) é o processo de computar rotas que são sujeitas a múltiplas restrições (WROCLAWSKI, 1997). As restrições utilizadas podem ser custo monetário, políticas de segurança e requisitos de QoS. O QoSR pode ser considerado uma variação de Roteamento Baseado em Restrições, onde as restrições são requisitos de QoS.

Uma das grandes diferenças entre QoSR e o roteamento convencional é a manutenção de estado sobre a capacidade dos recursos da rede em atender requisitos de QoS. Os principais objetivos do QoSR são determinação dinâmica de possíveis caminhos, otimização da utilização dos recursos e degradação graciosa de desempenho (CRAWLEY, 1998).

No caso da *determinação dinâmica de possíveis caminhos*, embora o QoSR possa encontrar um caminho que atenda os requisitos de QoS de um fluxo, o direcionamento do tráfego para ele pode depender de outras restrições (Roteamento Baseado em Restrições).

A *Otimização* visa a implementação de um esquema para utilização eficiente dos recursos, aumentando em conseqüência a vazão total alcançada pela rede. Os Caminhos ociosos podem ser aproveitados para satisfazer demandas por requisitos específicos de QoS, o que é impossível com o roteamento convencional. Isso pode ser usado para realizar a Engenharia de Tráfego.

No caso da *degradação graciosa de desempenho*, o roteamento dependente de estado pode compensar problemas transientes na rede, escolhendo caminhos alternativos, que permitem que as aplicações melhor se adaptem as condições momentâneas da rede.

O roteamento baseado em QoS é obviamente diferente do tradicional roteamento denominado “melhor esforço”. O QoSR pode encontrar um caminho mais longo, mas muito menos sobrecarregado que o caminho mais curto, que geralmente é o mais congestionado. O QoSR é normalmente orientado a conexão, além de utilizar reserva de recursos para fornecer garantia de QoS. As técnicas QoSR e reserva de recursos são complementares e geralmente são implementadas em conjunto. Os protocolos de reserva de recursos, como o RSVP, oferecem um método para requisitar e reservar recursos da rede, mas eles não proporcionam nenhum mecanismo para encontrar um caminho que tenha recursos suficientes para satisfazer os níveis de QoS requisitados. Por outro lado,

QoSR permite a determinação de um caminho com uma grande chance de acomodar a requisição de QoS, mas não inclui um mecanismo para reservar os recursos necessários. A função de Controle de Admissão de Conexão determina se a conexão deve ser aceita ou rejeitada. Se a reserva de recursos através do caminho selecionado foi efetuada com sucesso, então a conexão deve ser aceita, em caso contrário, deve ser rejeitada (SHIGANG; NAHRSTEDT, 1998).

2.3.5 Engenharia de Tráfego

A TE é a utilização de princípios tecnológicos e científicos para a medição, caracterização, modelagem e controle do tráfego com o objetivo de avaliação e otimização do desempenho das redes IP (AWDUCHE et al., 2000). A melhora na QoS prestada pela rede devido a aplicação da TE pode ser observada através de parâmetros de tráfego tais como atraso fim-a-fim, variação do atraso ou perda de pacotes, bem como pela percepção humana, como por exemplo, em aplicações multimídia. As atividades básicas da TE são o controle e a otimização do roteamento com o objetivo de distribuir o tráfego uniformemente pela rede. A TE pode ser feita manualmente, ou usando algum tipo de técnica automatizada para descobrir e fixar os caminhos mais adequados a determinadas agregações de fluxos dentro da rede. A Figura 3 ilustra o encaminhamento de pacotes sem e com TE. Em um domínio que não implementa TE, todos os pacotes devem ser encaminhados pelo caminho mais curto, no caso, o caminho do meio, e quase sempre resulta em congestionamento. A existência de congestionamento pode provocar a degradação da QoS.

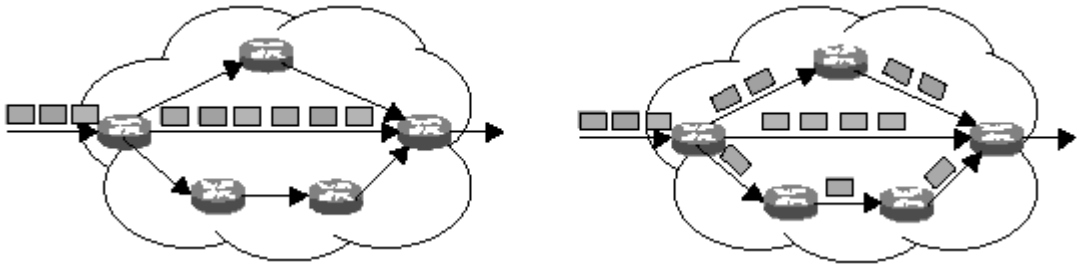

Figura 3 – Encaminhamento de pacotes sem e com TE.

Para restringir o acesso a recursos da rede que estejam congestionados e/ou regular a demanda para diminuir a situação de sobrecarga, devem ser definidas certas políticas para

a aceitação, estabelecimento e manutenção de conexões (AWDUCHE; REKHTER, 2001). Esse gerenciamento baseado em políticas traz a vantagem de prover a habilidade de controlar a rede como um todo, ao invés de controlar dispositivos de rede, interfaces e filas de maneira independente.

A TE engloba quatro problemas básicos, ou seja, o controle de admissão de novas conexões; roteamento de pacotes, dadas algumas restrições; re-roteamento de conexões já estabelecidas; e planejamento dos recursos da rede (GIRISH et al., 2000).

O *controle de admissão de novas conexões* determina se uma requisição pode ser ou não admitida e, em caso afirmativo, seleciona uma rota para essa conexão ao longo da rede.

O *roteamento baseado em restrições* é responsável pela seleção de caminhos considerados de alguma forma ótimos, e que satisfarão a um dado conjunto de restrições e requisitos. Métricas utilizadas no roteamento baseado em restrições incluem o custo em dinheiro, número de nós intermediários, taxa de transmissão, confiabilidade, atraso fim-a-fim e variação de atraso fim-a-fim (XIAO; NI, 1999). O processo é realizado em duas etapas. Na primeira são excluídos todos os enlaces que não tenham largura de banda suficiente para transportar a nova conexão ou que, de alguma outra forma, não atendam as exigências na nova demanda; na segunda é realizada a escolha do caminho a ser utilizado sobre a rede residual (SPRAGGS, 2000). Quando a escolha do caminho a ser utilizado é feita através de um algoritmo de menor caminho, o método é chamado CSPF (*Constraint Shortest Path First*). A rota escolhida pelo roteamento baseado em restrições não é necessariamente o "menor caminho" calculado por um algoritmo como o de Dijkstra, mas o caminho de algum sentido ótimo que atende a todos os requisitos exigidos (AWDUCHE et al., 1999).

O terceiro problema, *re-roteamento das conexões já estabelecidas* é o processo pelo qual alguns fluxos de tráfego são novamente roteados, para melhorar a eficiência da rede. Periodicamente, pode ser necessário que o operador de uma rede modifique o roteamento dos fluxos de tráfego, porque algumas condições da rede se alteraram. Por exemplo, um fluxo de tráfego pode ser re-roteado para um caminho secundário, em caso de falhas no caminho principal, ou no caso em que um tráfego de maior prioridade esteja utilizando todo o recurso do caminho principal. Quando a situação voltar ao normal, o fluxo de tráfego pode ser novamente ser re-roteado para o caminho principal original. Sem a

possibilidade de re-otimização, a rede poderia se tornar progressivamente distante do ponto ótimo, cada vez que uma falha ocorresse (SPRAGGS, 2000). Contudo, embora a possibilidade de re-roteamento dos fluxos seja interessante para a TE, deve ser usada com cautela (diariamente ou semanalmente, por exemplo), porque inclui troca de mensagens entre os diversos elementos de rede, o que pode levar a situações de sobrecarga, caso ocorra com muita freqüência.

O último problema, *o planejamento dos recursos da rede*, inclui o planejamento das capacidades dos enlaces e tamanhos dos buffers e objetiva proporcionar à rede a capacidade de atender novas demandas de tráfego. O planejamento dos recursos da rede deve considerar a possibilidade de demandas futuras, o que pode ser estimado a partir do histórico do tráfego na rede (GIRISH et al., 2000).

Para resolver os problemas anteriormente citados, é necessário que o processo de TE contenha alguns componentes chaves. Os componentes chaves são medição, modelagem, análise, e otimização (AWDUCHE et al., 2002).

A *medição* é crucial para a TE. A situação de uma rede somente pode ser conclusivamente determinada através de medição. A medição também é fundamental para a função de otimização uma vez que provê dados de realimentação os quais são usados pelos subsistemas de controle da TE. Estes dados são usados para de forma adaptativa otimizar a performance da rede. A Medição é também necessária para determinar a QoS de rede e também para avaliar a efetividade das políticas de TE. A experiência sugere que a medição é mais efetiva quando adquirida e aplicada sistematicamente.

A *modelagem* envolve a construção de um modelo que descreve características de tráfego relevantes e atributos da rede. Um modelo é uma representação abstrata da rede que captura suas características relevantes, atributos, e outras características como restrições e atributos de nós e enlaces. Um modelo de rede pode facilitar análise e/ou simulação. A simulação pode ser usada para prever a performance da rede sob várias condições como também para orientar os planos de expansão (AWDUCHE et al., 2002). Os modelos sobre o comportamento das fontes de tráfego são particularmente úteis para análise. O desenvolvimento de modelos sobre o comportamento das fontes de tráfego que sejam consistentes com os dados empíricos obtidos a partir da rede em operação é um tópico de pesquisa importante na TE.

O terceiro componente da TE é responsável pela *análise* do estado da rede e caracterização da carga de tráfego representativa. A *análise* pode envolver investigação sobre a concentração e distribuição de tráfego através da rede, identificando as características da carga de tráfego oferecida, potenciais gargalos, enlace sub-utilizados, pontos de falhas, etc (AWDUCHE et al., 2002). A *análise* pode ser reativa de e/ou proativa. A análise do tipo proativa identifica problemas potenciais que não existem atualmente, mas que poderão se manifestar no futuro. A análise do tipo reativa identifica problemas existentes, determina a causa através de diagnóstico, e avalia alternativas para resolver o problema, se necessário.

A *otimização de desempenho da rede* pode ser corretiva ou preventiva (AWDUCHE et al., 2002). Na otimização corretiva, a meta é resolver um problema real. Na otimização preventiva, a meta é melhorar o desempenho da rede até mesmo quando os problemas ainda não existam e/ou não são previstos. A otimização de desempenho da rede é um processo ininterrupto. Ela pode ser efetuada em tempo real ou não. A otimização que ocorre durante o planejamento da rede não é realizada em tempo real (XIAO et al., 2000).

2.4 Computação Autonômica

A crescente complexidade de gerenciamento das redes alcança níveis além da habilidade humana. Como conseqüência, a intervenção humana se torna um ponto fraco no processo de gerenciamento e controle. Isto tem levado os pesquisadores a considerar técnicas alternativas de projeto e gerenciamento que são baseadas em estratégias usadas pelos sistemas biológicos para tratar a complexidade, heterogeneidade e incerteza. Uma destas alternativas é a Computação Autonômica.

A Computação Autonômica foi inspirada no funcionamento do sistema nervoso humano e visa o desenho e a construção de sistemas auto-gerenciados. Assim, um sistema de software que opera por conta própria ou com um mínimo de interferência humana, de acordo com um conjunto de políticas é chamado Autonômico. Mais especificamente, um sistema autonômico é um ambiente computacional auto-gerenciado, autônomo e ubíquo, ou seja, aquele que oferece serviços o tempo todo e em todo lugar. Além disso, o sistema esconde a sua complexidade, fornecendo ao usuário uma interface na exata medida de suas necessidades. O sistema sempre decidirá sobre si próprio usando políticas de alto nível

definidas por humanos. Ele constantemente checará e otimizará o seu estado atual, adaptando-se automaticamente às mudanças nas condições ambientais (STERRITT et al, 2005). As subseções seguintes apresentam alguns conceitos básicos sobre Computação Autonômica.

2.4.1 Propriedades da Computação Autonômica

A Computação Autonômica foi proposta pela IBM como uma abordagem para reduzir o custo e a complexidade do gerenciamento de uma estrutura de Tecnologia da Informação (TI). Ela não é um novo campo, mas algo como uma combinação de teorias e práticas selecionadas a partir de várias áreas existentes, incluindo teoria de controle, algoritmos adaptativos, agentes de software, robótica, sistemas distribuídos e de tempo real, inteligência computacional, etc. A abordagem básica da Computação Autonômica é construir sistemas capazes de gerenciar a si mesmos, que possam antecipar suas cargas de trabalho e adaptar seus recursos para otimizar a performance. A essência da Computação Autonômica é o auto-gerenciamento. O auto-gerenciamento tem como objetivo liberar os administradores da preocupação com detalhes relacionados a operação e manutenção dos sistemas (KEPHART; CHESS, 2003). Para implementar o auto-gerenciamento, o sistema deve ao mesmo tempo estar atento a si próprio e ao seu ambiente. Desta forma, o sistema deve conhecer com precisão a sua própria situação e ter consciência do ambiente operacional em que atua. Um sistema autonômico tem quatro propriedades principais, auto-configuração, auto-cura, auto-otimização e auto-proteção (KEPHART; CHESS, 2003).

A auto-configuração é a capacidade do sistema de se adaptar automática e dinamicamente em resposta a mudanças ocorridas no ambiente (SALEHIE; TAHVILDARI, 2005). Por exemplo, quando uma solicitação de chamada é recebida por uma rede, o sistema autonômico deve escolher o melhor caminho para encaminhar os seus pacotes. Se o caminho escolhido deixar de oferecer uma QoS adequada às necessidades da aplicação, o sistema deve procurar um caminho alternativo e providenciar o re-roteamento sem a necessidade de intervenção manual.

A auto-cura é a capacidade de descoberta, diagnóstico e reação às interrupções (KEPHART; CHESS, 2003). Cada sistema deve ser capaz de se recuperar através da

detecção do componente com falha, bloqueá-lo e substituí-lo. Isto deve ser realizado sem que o sistema aparente ter tido uma interrupção ou falha. Cada sistema deve prever os problemas e executar ações para prever a falha. O objetivo principal da auto-cura é maximizar a disponibilidade, a sobrevivência, a manutenibilidade e a confiabilidade do sistema (GANEK; CORBI, 2003).

A auto-otimização é a capacidade de maximizar eficientemente a alocação e a utilização de recursos visando a atender os requisitos dos diferentes usuários (SALEHIE; TAHVILDARI, 2005). O sistema autonômico deve continuamente buscar maneiras de melhorar sua operação, identificando e aproveitando oportunidades para se tornar mais eficiente em performance e também em custo.

A auto-proteção é a capacidade de estabelecer efetivamente a confiabilidade, antecipando, detectando e recuperando-se dos efeitos de ataques (KEPHART; CHESS, 2003). A defesa deve ser feita no sentido de resolver os problemas originados de ataques maliciosos ou de falhas em cascata que não foram sanadas pela auto-cura. Pode ser realizada também uma antecipação da resolução de eventuais problemas futuros, baseando-se em avisos emitidos por sensores. A ação neste caso, deve ser no sentido de evitar os problemas ou diminuir os seus efeitos.

Além das citadas, um sistema autonômico possui outras quatro propriedades secundárias, ou seja, autoconsciência, consciência ambiental, auto-monitoramento e auto-ajuste (STERRITT et al, 2005). Neste caso o sistema deve estar atento a sua situação interna e as condições atuais de operação do ambiente externo. As eventuais mudanças ocorridas são detectadas através do auto-monitoramento e como conseqüência as adaptações necessárias são realizadas. O sistema deve ter um conhecimento dos seus recursos disponíveis, seus componentes, requisitos de desempenho desejado, estado atual, estados das interconexões com outros sistemas, além de regras e políticas de como eles podem ser ajustados. Deve-se observar que desde o lançamento da Computação Autonômica em 2001, a lista auto_* de propriedades tem crescido substancialmente. A lista agora inclui também características como auto-antecipação, auto-adaptação, auto-definição, auto-destruição, auto-diagnose, auto-governo, auto-organização, auto-recuperação, auto-reflexão e auto-simulação (STERRITT et al, 2005).

2.4.2 Elementos Autonômicos

Um sistema autonômico é formado por um conjunto de elementos autonômicos. Um elemento autonômico contêm um único gerente que representa e monitora um ou mais elementos gerenciados (KEPHART; CHESS, 2003). Cada componente de um sistema autonômico possui um elemento autonômico embutido responsável por implementar serviços e funções de gerenciamento. Cada elemento autonômico atua como um gerente, responsável por promover a produtividade dos recursos e a qualidade dos serviços providos pelo componente do sistema no qual está instalado. No núcleo do elemento autonômico está o laço de controle que integra o gerente com o elemento gerenciado. Um elemento autonômico é constituído pelas funções de monitoração, análise, planejamento, execução e base de conhecimento (KEPHART; CHESS, 2003). Conforme é mostrado na Figura 4, a interação do gerente com o elemento gerenciado é realizada através dos sensores e dos atuadores.

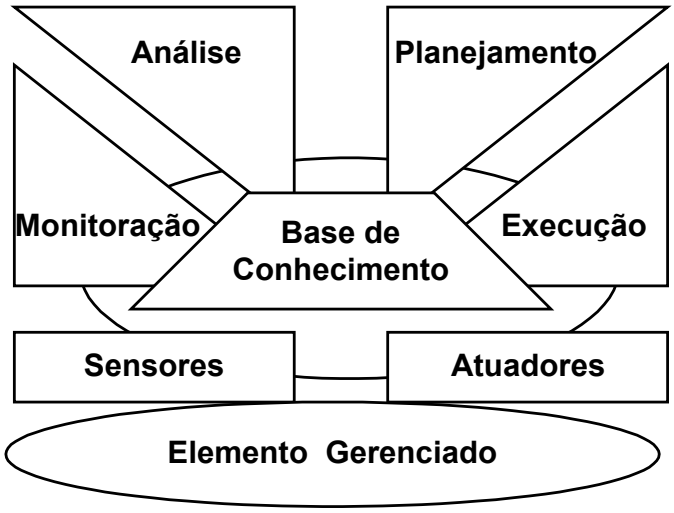

Figura 4 – Elemento Autonômico.

A função de monitoração filtra os dados recebidos do sensor e armazena-os na base de conhecimento. O serviço realizado pela função de monitoração possibilita que o elemento autonômico conheça a si mesmo e a seus componentes. Alem disso, possibilita conhecer também o ambiente que o cerca. A função de análise compara os dados coletados com os valores desejados, ambos armazenados na base de conhecimento. As informações

obtidas a partir das comparações realizadas são utilizadas para verificar a ocorrência de problemas, como por exemplo, a queda no desempenho do sistema em função da ocorrência de falhas. O dispositivo de planejamento define estratégias para corrigir se necessário a tendência identificada pelo dispositivo de análise. Finalmente, o dispositivo de execução ajusta os parâmetros do elemento gerenciado através dos atuadores e armazena os valores ajustados na base de conhecimento (MÜLLER et al, 2006).

2.4.3 Políticas

É de fundamental importância para o comportamento do sistema autonômico, a habilidade de em alto nível utilizar diretivas de escopo amplo que possam ser traduzidas em ações especificas a serem tomadas pelos elementos. Isto é obtido através do uso de políticas (WHITE et. al, 2004). Uma política é a representação numa forma padrão, dos comportamentos desejados ou das restrições de comportamento do sistema. Em Computação Autonômica, podem ser utilizados pelo menos três tipos de políticas ou seja, ação, meta e função de utilidade (WHITE et. al, 2004).

As *políticas de ação* estão no mais baixo nível de especificação. Uma política de ação define a ação que dever ser executada para que o sistema mude de estado. Ela é tipicamente da forma Se (condição) então (ação). Como exemplo, Se (tempo de resposta < 2s) então (aumenta uso da CPU em 5%). Um elemento autonômico empregando políticas de ação deve medir e/ou sintetizar as quantidades declaradas na condição e também deve executar a ação sempre que a condição é satisfeita.

No próximo nível estão as *políticas de metas*. As políticas de metas descrevem as condições a serem atingidas sem especificar como obtê-las. Um exemplo de uma política de meta é "O tempo de resposta não deve exceder 2 segundos". As políticas de metas são mais poderosas que as de ação porque um elemento humano ou autonômico pode passar uma instrução a outro, sem a necessidade de conhecer detalhadamente como este realiza internamente a ordem recebida. Os elementos autonômicos que empregam políticas de metas devem possuir capacidades de planejamento e modelagem suficientes para traduzir metas em ações.

No alto nível estão as *políticas com função de utilidade*. Uma política com função de utilidade é uma função objetivo que expressa um valor para cada possível estado. As

funções de utilidade são mais poderosas que as políticas de metas porque elas determinam automaticamente a meta mais adequada para uma dada situação. Os elementos autonômicos que empregam políticas com funções de utilidade devem possuir capacidades de otimização e modelagem suficientes sofisticadas para traduzir funções de utilidade em ações.

2.4.4 Níveis de maturidade

A maioria dos sistemas existentes não pode ser simplesmente redesenhada e desenvolvida a partir do nada para inclusão das capacidades autonômicas. Em lugar disso, as capacidades de autogerenciamento devem ser adicionadas gradualmente de forma incremental (MÜLLER et al, 2006). Assim, o processo de implantação de sistemas de software com tecnologia autonômica será do tipo evolucionário, ao invés de revolucionário. A IBM definiu cinco níveis de maturidade para caracterizar a gradual injeção de autonomia aos sistemas de software. Os níveis de maturidade propostos são básico, gerenciado, preditivo, adaptativo e autonômico (MÜLLER et al, 2006).

No *nível básico*, a habilidade reside na mente humana, necessitando de consultas freqüentes a procedimentos. A análise e a solução do problema são manuais.

No *nível gerenciado*, ferramentas de escrita e registro automatizam rotinas de execução e geração de relatórios. Especialistas utilizam a informação fornecida pelas ferramentas para efetuar planos e tomar decisões.

No *nível preditivo*, são emitidos avisos prévios para evitar que determinados patamares prefixados sejam atingidos. A base de conhecimento recomenda ações apropriadas. A resolução dos problemas é alavancada por uma base de dados centralizada contendo experiências e ocorrências comuns.

No caso do *nível adaptativo*, o sistema adaptativo atua por si mesmo para resolver o problema baseando-se nos avisos prévios proporcionados pelas capacidades preditivas.

No *nível autonômico*, as políticas dirigem as atividades do sistema. Neste caso, o sistema realiza dinamicamente o auto-gerenciamento baseando-se em políticas. Os usuários e administradores apenas monitoram o processo e/ou alteram os objetivos.

Apesar de atualmente existir exemplos de sistemas preditivos e adaptativos, de um modo geral a indústria permanece nos níveis de maturidade básico e gerenciado.

2.5 Inteligência Computacional

A CI compreende paradigmas computacionais que procuram desenvolver sistemas que apresentem alguma forma de inteligência similar à exibida por determinados sistemas biológicos. BEZDEK (1994) sugere que um sistema é *computacionalmente inteligente* quando trabalha apenas com dados numéricos, tem um componente de reconhecimento de padrões e não usa conhecimento no sentido da inteligência artificial clássica (conhecimento simbólico, não-numérico); e, adicionalmente, quando ele exibe (ou começa a exibir):

- adaptabilidade computacional;
- tolerância computacional a falhas;
- velocidade de processamento comparável à de processos cognitivos humanos;
- taxas de erro que se aproximam do desempenho humano.

A área de CI engloba diversos paradigmas computacionais diferentes. Os principais paradigmas da CI são Redes Neurais Artificiais (RNA), Sistemas Nebulosos e Computação Evolutiva (IYODA, 2000).

2.5.1 Redes Neurais Artificiais

As Redes Neurais Artificiais são sistemas de processamento de informação formados pela interconexão maciça entre unidades simples de processamento, denominados neurônios artificiais. Os neurônios artificiais recebem essa denominação porque foram originados a partir de um modelo matemático de um neurônio biológico. Várias arquiteturas de redes neurais já foram propostas na literatura, sendo a arquitetura multicamadas a mais popular.

As Redes neurais multicamadas contêm um conjunto de neurônios de entrada, uma camada de saída e uma ou mais camadas escondidas, conforme mostra a Figura 5. A estrutura consiste em camadas de neurônios na qual a saída de um neurônio numa camada alimenta todos os neurônios da camada seguinte. Conforme pode ser observado na Figura 5, não existem laços de realimentação. A rede *"feedforward"* é referida também como MultiLayer Perceptron (MLP).

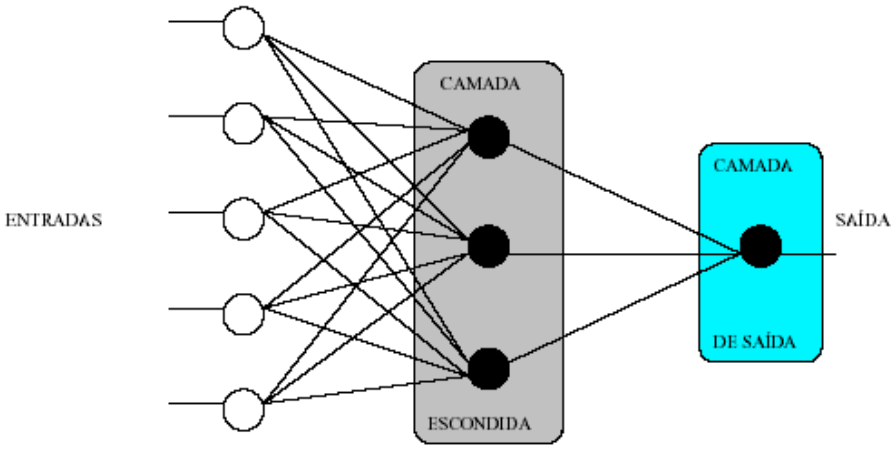

Figura 5 – Estrutura de uma rede MLP.

A rede MLP, treinada com o algoritmo *backpropagation*, ganhou muita popularidade como modelo adaptativo não linear, pois provou ser uma ferramenta altamente flexível para aplicações em problemas de reconhecimento de padrões, classificação e outros de natureza estática ou estacionária.

Em problemas dinâmicos ou temporais, como processamento digital de sinais e previsão de séries temporais, a rede MLP é muito utilizada com um "artifício" na camada de entrada da rede - o mecanismo da janela. A janela é um mecanismo através do qual se submete à rede os valores prévios da série nos quais é baseada a previsão (HAYKIN, 1994; ZEBULUM,1995). Esta janela constitui em realidade a incorporação de "memória" na entrada da rede, permitindo assim à rede MLP o aprendizado do comportamento das séries temporais estacionárias. Com este método, as séries não-estacionárias têm que ser previamente transformadas em estacionárias através de algum método estatístico, como a diferenciação. Esta maneira de utilizar a rede neural dá bons resultados na solução de problemas de previsão de séries temporais, mostrando em muitos casos um melhor rendimento que os modelos puramente estatísticos (ZEBULUM, 1995).

Em geral, para que uma rede neural se torne dinâmica, esta deve ter memória. Incorporando memória à estrutura da rede estática, a saída da rede passa a ser função do tempo. Este enfoque de construção de um sistema dinâmico não linear é muito utilizado pois mostra uma clara separação de responsabilidades. A rede estática toma conta da não linearidade e a memória toma conta do tempo.

O mecanismo da janela, utilizado intuitivamente, pode ser formalizado matematicamente, auxiliado pela Teoria de Processamento Digital de Sinais, e generalizado para incorporar memória também nas camadas escondidas, fazendo assim com que a mesma rede MLP seja capaz de aprender o comportamento de sinais temporais não estacionários diretamente. Estes tipos de redes são as denominadas "redes neurais com atrasos no tempo".

Outro método de incorporar memória à rede MLP, é agregando laços de realimentação das saídas das unidades processadoras até suas entradas, arquitetura conhecida como rede recorrente (HAYKIN, 1994). A vantagem das redes recorrentes em relação à rede MLP sem realimentação é a memória interna de estados passados, introduzida pelas conexões de realimentação. Essas duas maneiras de incorporar memória à rede neural, com atrasos no tempo e com recorrências ou realimentação, constituem os dois tipos básicos de RNA Temporais. Assim, as previsões de séries temporais podem ser classificadas como "multi-step" e "single-step" (REFENES et al, 1993).

As *previsões "multi-step"* são aquelas que se caracterizam por possuir realimentação das saídas das *RNA*s para as suas entradas. Neste tipo de previsão, o sistema neural usa um conjunto de valores correntes da série para prever os valores futuros por um período fixo. Em seguida, esta previsão é realimentada na entrada do sistema para prever o próximo período. Estas previsões são muito usadas para identificar tendências e pontos de mudanças preponderantes nas séries. Devido ao erro que é inserido a cada nova previsão, as previsões "multi-step" são usadas para prever apenas alguns passos a frente, quando o erro ainda é aceitável.

Nas *previsões "single-step"* não existe realimentação. As *RNA*s utilizam apenas os valores anteriores da série para prever um passo à frente. Todavia, este passo tanto pode ser para previsões de curto prazo como para previsões de médio e longo prazo, bastando que se tenha dados suficientes para treinar a rede. A previsão "single-step" também serve para avaliar a adaptabilidade e a robustez do sistema, mostrando que mesmo quando as *RNA*s fazem previsões erradas, elas são capazes de se auto corrigirem e fazer as próximas previsões corretamente. Além disso, as previsões "single-step" podem ser usadas como alarme.

2.5.2 Sistemas Nebulosos

Os sistemas *nebulosos* são baseados no conceito de conjuntos *nebulosos*. Os conjuntos *nebulosos* surgiram como uma nova forma de representação de conceitos como imprecisão e incerteza. Os conjuntos *nebulosos* são especialmente adequados na descrição de sistemas de processamento de informação complexos, não-lineares ou não claramente definidos. Observe que, além de trabalhar com dados numéricos, os sistemas *nebulosos* também são capazes de realizar processamento simbólico, através de uma base de regras.

A lógica *nebulosa* é uma metodologia muito eficiente quando se necessita trabalhar com informações inexatas, imprecisas, incompletas através de uma sistemática rigorosa. Os seres humanos são capazes de trabalhar de maneira efetiva com este tipo de lógica. É uma lógica bastante utilizada quando existe a necessidade de se trabalhar com sistemas onde há uma difícil representação matemática do problema.

Através da lógica *nebulosa* é possível manipular dados numéricos e conhecimentos lingüísticos simultaneamente. Com ela, geramos um mapeamento não linear de um conjunto de dados de entrada em uma saída escalar, isto é, mapeia-se números em números. As teorias dos conjuntos e da lógica *nebulosa* estabelecem a maneira como é feito este mapeamento (HENRIQUES, 1999).

Uma *variável nebulosa* é uma variável cujos valores são rótulos (*labels*) de conjuntos *nebulosos*. Por exemplo, a temperatura do processo representado pela Figura 6 é uma variável *nebulosa* que assume os *valores baixa*, *média*, e *alta*. Nesse contexto, a variável temperatura é uma variável lingüística. Uma variável lingüística é uma variável cujos valores não são números mas sim palavras ou expressões numa linguagem natural. A principal função das variáveis lingüísticas é fornecer uma maneira sistemática para uma caracterização aproximada de fenômenos complexos ou mal definidos. Na Figura 6, existem três funções de pertinência representando os valores ***baixa***, ***média***, e ***alta*** temperatura, respectivamente. Deve-se observar que as temperaturas baixa e alta são funções de pertinência do tipo trapezoidal, e a temperatura média é do tipo triangular.

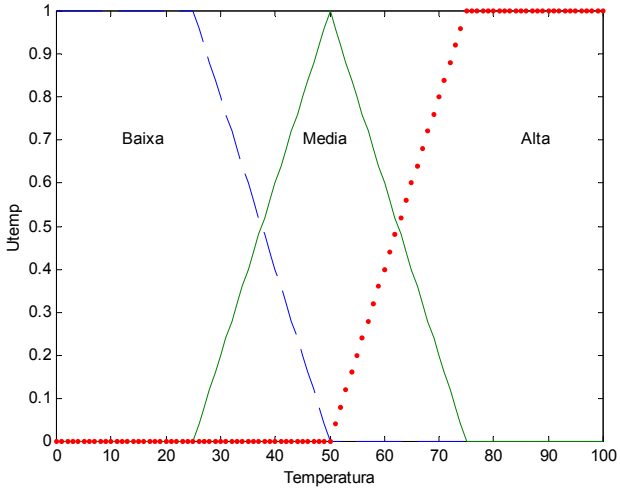

Figura 6 - Variável lingüística temperatura.

A modelagem *nebulosa* descreve o comportamento do sistema usando linguagem natural baseada na lógica *nebulosa* utilizando conjuntos *nebulosos* considerando os conceitos humanos. Um modelo *nebuloso* é caracterizado pelo conjunto de regras que expressam a relação entre as variáveis do sistema. Cada regra, representaria a descrição local da dinâmica do sistema e é composta por uma parte antecedente (condição da regra) e uma parte conseqüente (conclusão da regra). Um exemplo de uma regra *nebulosa* é mostrado a seguir:

SE pressão é alta **ENTÃO** volume é pequeno

Onde:

pressão e ***volume*** são variáveis lingüísticas, *alta* é o antecedente (premissa) e *volume* é o conseqüente.

Os sistemas *nebulosos* estimam funções com descrição parcial do comportamento do sistema, onde especialistas podem prover o conhecimento heurístico, ou esse conhecimento pode ser inferido a partir de dados de entrada-saída do sistema. Pode-se dizer que os sistemas *nebulosos* são sistemas baseados em regras que utilizam variáveis lingüísticas *nebulosas* para executar um processo de tomada de decisão (ADILÉA, 2003). O esquema básico de um Sistema de Inferência Nebulosa (SIF) é composto de cinco blocos, conforme é mostrado na Figura 7.

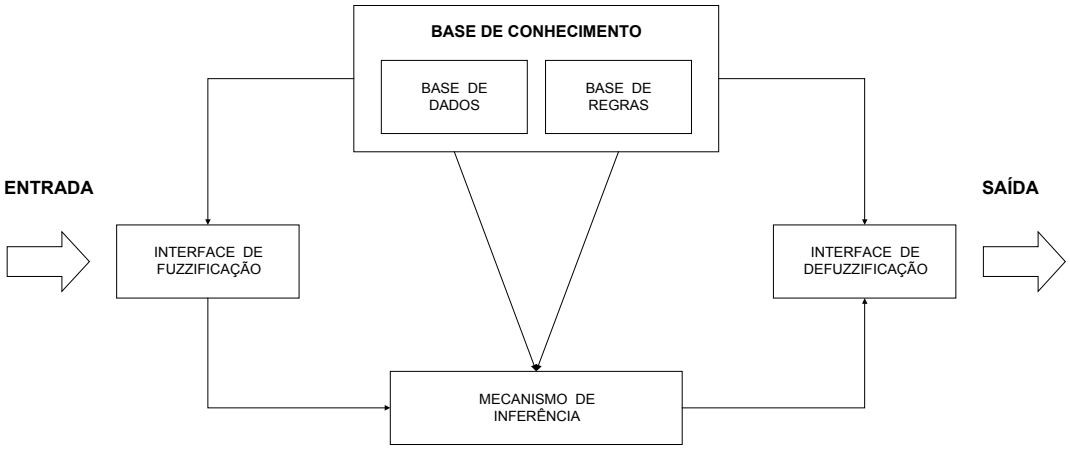

Figura 7 - Sistema de Inferência Nebulosa.

A Base de Regras contém um conjunto de regras/proposições *nebulosas* onde as variáveis antecedentes/conseqüentes são variáveis lingüísticas e os possíveis valores de uma variável lingüística são representados por conjuntos *nebulosos*.

A **Base de Dados** define as funções de pertinência do conjunto nebuloso nas regras *nebulosa*s.

O **Mecanismo de Inferência** realiza operações de inferência, para obter, a partir da avaliação dos níveis de compatibilidade das entradas com as condições impostas pela base de regras, uma ação a ser realizada pelo sistema.

A **Interface de Fuzzificação** utilizando as funções de pertinência pré-estabelecidas, mapeia cada variável de entrada do sistema em graus de pertinência de algum conjunto nebuloso que representa a variável em questão.

A **Interface de Defuzificação** transforma os resultados nebulosos da inferência em valores de saída. Calcula a saída com base na inferência obtida no **Mecanismo de Inferência**, com as funções de pertinência das variáveis lingüísticas da parte conseqüente das regras para obter uma saída não nebulosa. Nessa etapa as regiões resultantes são convertidas em valores de saída do sistema.

As regras *nebulosas* formam a parte fundamental da estrutura de conhecimento em um SIF. Os formatos de regras *nebulosas* podem ser divididos em quatro grupos principais: *Mamdani*, *Takagi-Sugeno, Tsukamoto* e de *Classificação*. Os três primeiros - *Mamdani*, *Takagi-Sugeno* e *Tsukamoto* - correspondem ao modelo de inferência *nebulosa*. A diferença básica entre esses três primeiros modelos recai no tipo de conseqüente e no procedimento de defuzificação.

2.5.3 Computação Evolutiva

A computação evolutiva é formada por algoritmos inspirados na teoria da evolução natural de Darwin. De acordo com a teoria de Darwin, o princípio de seleção privilegia os indivíduos mais aptos com maior longevidade e, portanto, com maior probabilidade de reprodução. Os indivíduos com mais descendentes têm mais chance de perpetuarem seus códigos genéticos nas próximas gerações. Os códigos genéticos constituem a identidade de cada indivíduo e estão representados nos cromossomos. A computação evolutiva tem sido muito aplicada em problemas de otimização, em especial naqueles em que técnicas tradicionais de otimização não são aplicáveis (ou apresentam desempenho insatisfatório). Um dos ramos da computação evolutiva são os Algoritmos Genéticos (AGs).

Os AGs são algoritmos de otimização global, baseados nos mecanismos de genética e seleção natural (GOLDBERG, 1989). Enquanto os métodos de otimização e busca convencionais trabalham geralmente de forma seqüencial, avaliando a cada momento uma possível solução, os AGs trabalham com um conjunto de soluções simultaneamente. As técnicas de busca e otimização geralmente apresentam um espaço de busca e uma função de avaliação. O espaço de busca contêm todas as soluções para o problema. A função de avaliação avalia, geralmente através de uma nota, cada possível solução presente no espaço de busca.

De acordo com GOLDBERG (1989), os AGs diferem dos métodos tradicionais de técnicas de busca e otimização em quatro aspectos principais. Em primeiro lugar, os AGs trabalham com uma codificação do conjunto de parâmetros e não com os próprios parâmetros. Em segundo lugar, os AGs trabalham com uma população de soluções candidatas simultaneamente, e não com uma única solução. Além disso, os AGs utilizam informações de custo ou recompensa, e não derivadas de funções. Finalmente, os AGs utilizam regras de transição probabilísticas e não determinísticas.

Na terminologia de AGs, uma solução candidata é chamada de individuo ou cromossomo. Ao conjunto de indivíduos simultaneamente avaliados é dado o nome de população. A cada individuo é associado um grau de adaptação ou *aptidão,* que mede a capacidade da solução, representada pelo indivíduo para resolver um dado problema. Os AGs não garantem uma solução ótima, mas geralmente encontram soluções quase ótimas em um tempo aceitável. Esta técnica emprega uma estratégia de busca paralela e

estruturada, embora com um forte componente aleatório, que é voltada em direção à busca de pontos de "maior aptidão", ou seja, pontos nos quais a função a ser minimizada (ou maximizada) tem valores relativamente baixos (ou altos) (BRAGA et al., 2000).

Para a utilização de AGs em problemas de otimização e busca, uma seqüência de passos deve ser observada. Segundo BRAGA et al. (2000) um algoritmo genético deve executar os passos apresentados a seguir:

1. Escolher um conjunto de cromossomos iniciais.
2. Repetir
   2.1 Definir a nota de cada cromossomo.
   2.2 Selecionar os cromossomos mais aptos.
   2.3 Aplicar operadores de reprodução sobre os cromossomos selecionados.
3. Até cromossomo adequado ser obtido ou serem realizadas N gerações.

O primeiro passo é a geração da população inicial. Em seguida deve ser realizada a codificação ou representação, ou seja, deve ser determinado como os indivíduos serão codificados. Durante o processo evolutivo, a população é avaliada, sendo que para cada indivíduo é dada uma nota, ou índice, refletindo sua habilidade de adaptação a um determinado ambiente.

O processo de seleção determina quais indivíduos da população podem participar da fase de reprodução. Os indivíduos são selecionados de acordo com uma probabilidade dada pelos índices ou notas de aptidão. Um método de seleção muito utilizado é o **método da roleta**, em que cada indivíduo ocupa, em uma roleta, uma área proporcional ao seu índice de aptidão. Desta forma, aos indivíduos com maior aptidão é associada uma fatia maior da roleta e vice-versa. Durante a seleção a roleta é girada N vezes, selecionando N indivíduos para participarem da fase de reprodução. Indivíduos com maiores notas, por possuírem áreas maiores, têm maiores probabilidades de serem selecionados. Entretanto, a utilização da função de avaliação para escolher os tamanhos das fatias da roleta nem sempre é a mais adequada. Se a nota de um individuo for muito alta, este pode monopolizar a seleção, levando a uma seleção prematura do algoritmo genético. Se por outro lado, os valores de aptidão dos indivíduos forem muito próximos, as suas fatias nas roletas serão aproximadamente iguais. Isto leva a possibilidade da seleção não favorecer os indivíduos mais aptos. Uma solução para estes problemas é a utilização da técnica de *ranking*. Nesta

técnica, a fatia é definida não pela nota relativa de cada individuo, mas pela posição que eles ocupam no *ranking* de todas as notas (BRAGA et al., 2000). Na técnica de *ranking,* a fatia de cada indivíduo é definida por meio da escolha de um valor no intervalo entre 0.0 e 1.0. Desta forma, em ordem decrescente de *ranking*, cada indivíduo ocupa, da área total da roleta que ainda não foi ocupada, uma fatia proporcional ao valor escolhido. Supondo por exemplo, que o valor escolhido foi 0.7, o indivíduo de maior *ranking* ocupará 70% da roleta. O segundo indivíduo ocupará 70% dos 30% que sobraram, ou seja, 21%. O terceiro indivíduo ocupará 70% dos 9% que sobraram, que é igual a 6.3%. O indivíduo de menor *ranking* ocupará a última área que sobrou da roleta.

Os indivíduos escolhidos na fase de seleção participam da fase de reprodução, em que podem ser combinados ou modificados, produzindo os indivíduos da próxima geração. Estas combinações e modificações são realizadas por um conjunto de operadores chamados operadores genéticos. Os principais operadores genéticos são *crossover* (ou cruzamento) e mutação. O *crossover* é o operador responsável pela recombinação de características genéticas dos pais durante a reprodução, permitindo que elas sejam herdadas pelas próximas gerações. Ele é considerado o operador genético predominante por isso é aplicado com uma probabilidade, chamada de taxa de *crossover*, maior que a taxa de *mutação*. O operador de *mutação* é responsável pela introdução e manutenção da diversidade genética na população, alterando arbitrariamente um ou mais genes de um cromossomo escolhido aleatoriamente. A *mutação* assegura que a probabilidade de se chegar a qualquer ponto do espaço de busca nunca será zero, além de contornar o problemas dos mínimos locais, por permitir a alteração da direção de busca. O operador de *mutação* é aplicado aos indivíduos com uma probabilidade dada pela taxa de mutação, que é geralmente pequena.

Ao utilizar os algoritmos genéticos para a solução de um problema, é importante escolher os parâmetros adequados às necessidades do problema e aos recursos disponíveis. Segundo BRAGA et al. (2000), os principais parâmetros são tamanho da população, taxa de cruzamento, taxa de mutação e intervalo de geração.

Com uma população pequena o desempenho pode cair. Por outro lado, uma população grande exige maiores recursos computacionais, mas geralmente fornece uma cobertura representativa do problema.

Uma **taxa de cruzamento** muito alta faz com que indivíduos com bons índices de aptidão possam ser retirados a uma velocidade que supere a capacidade de gerar melhores indivíduos. Por outro lado, se essa taxa for muito baixa, a busca pode estagnar.

Uma baixa **taxa de mutação** previne que uma dada posição fique estagnada em um valor, além de possibilitar que se chegue em qualquer ponto do espaço de busca. Com uma taxa muito alta, a busca se torna essencialmente aleatória.

O **intervalo de geração** controla a porcentagem da população que será substituída para a próxima geração. Com um valor alto, a maior parte da população é substituída, o que pode levar à perda de indivíduos de alta aptidão. Com um valor baixo, o algoritmo pode se tornar muito lento, pois o número de gerações necessárias pode ser muito grande.

2.6 Trabalhos relacionados

Recentemente, têm sido apresentadas propostas onde se discute a implementação de TE reativa em redes IP/MPLS. Algumas destas propostas utilizam CI para a implementação de funções específicas da TE. Por outro lado, existem também propostas que tratam de TE reativa mas sem a utilização de IC. As descrições sucintas de algumas destas propostas são apresentadas a seguir.

RESENDE et al. (2003) propõem um Controle de Admissão de Conexão e Roteamento num ambiente IP/MPLS usando lógica nebulosa com o propósito de oferecer QoS aos usuários. O algoritmo de controle de admissão é um instrumento que decide se uma chamada de entrada pode ser aceita ou rejeitada. O controlador proposto visa ajustar o numero de conexões estabelecidas, minimizando o congestionamento.

DIN & FISAL (2003) propõem um dispositivo de alocação dinâmica de largura de banda em enlaces num ambiente DiffServ-MPLS. O dispositivo é baseado em lógica nebulosa e num esquema de máxima alocação de largura de banda. O trabalho proposto é focado na diferenciação de serviços implementada através de um marcador localizado em um roteador de borda DiffServ-MPLS.

KHAN & ALNUWEIRI (2003) propõem um algoritmo utilizando lógica nebulosa para tentar resolver o problema do roteamento em TE. Assume-se que o protocolo de roteamento OSPF-TE está ativo na rede. O protocolo OSPF-TE possibilita que informações sobre a largura de banda disponível em cada enlace da rede sejam trocadas

dinamicamente entre os nós do domínio. Considera-se também que o MPLS está disponível para suportar rotas (caminhos) explícitos. No algoritmo proposto a Lógica nebulosa é utilizada para determinar um conjunto de possíveis caminhos. Um conjunto de funções de pertinência de cada caminho é determinado sendo que os valores destas funções são usados para rotear a demanda (solicitação). Quanto mais alto é o valor associado a função de pertinência do enlace do conjunto enlaces possíveis para estabelecimento do caminho, maior é a chance de ser incluído no caminho escolhido para estabelecimento da LSP.

LIU et al. (2003) fazem a modelagem do problema da otimização do roteamento explicito com múltiplas restrições, com o objetivo de minimizar o congestionamento global da rede. Em outras palavras o problema então é selecionar os caminhos ótimos para transportar as LSP´s através da rede MPLS, seguindo as restrições impostas. Um algoritmo Adaptativo Genético Heurístico (HAGA) é proposto. O HAGA busca melhorar o Algoritmo Genético Simples (SGA) na eficiência e habilidade de busca local, através da probabilidade adaptativa de cruzamento e mutação.

ELWALID et al. (2001) descrevem o Sistema MPLS TE adaptativo (MATE). O MATE é baseado numa abordagem distribuída com múltiplos caminhos. Nesta abordagem, várias LSP´s explicitas são estabelecidas entre um determinado nó de ingresso e outro de egresso usando protocolos padrões como CR-LDP (JAMOUSSI et al., 2002) ou RSVP-TE (AWDUCHE et al., 2001), ou configuradas manualmente. O mecanismo adaptativo proposto utiliza pacotes de prova, enviados a partir do nó de entrada, para obter estatísticas da LSP como Atraso e Perda de pacotes. Os resultados das medições são utilizados para comutar o tráfego entre as LSP´s. O mecanismo presente no MATE não é capaz de modificar a Largura de Banda das LSP´s para acomodar demandas adicionais quando todas as LSP´s estabelecidas alcançam suas máximas Larguras de Banda reservadas. Ao mesmo tempo, o MATE continua reservando o mesmo valor de Largura de Banda máxima mesmo que o tráfego diminua drasticamente em todas as LSP´s (LEMMA, 2003).

SVALOW (1999) sugere duas estratégias de otimização. A primeira usa várias LSP´s para cada destino para balancear a carga, como no MATE. Entretanto, ao invés de enviar pacotes de prova para monitorar a utilização de cada LSP, é usada a informação da

utilização de enlace proporcionada pelo OSPF ou IS-IS extensível. A segunda estratégia tenta auto-ajustar a largura de Banda baseando-se no uso real da LSP.

BUTENWEG (2003) propõe um sistema de TE reativo em redes MPLS que visa otimizar a vazão do tráfego de melhor esforço. O re-balanceamento da carga é realizado pelo re-roteamento ou através do uso de vários caminhos.

DIAS et al. (2004) propõem uma solução para o problema de TE dinâmica em redes MPLS. O problema consiste na maximização da vazão dos fluxos de dados injetados na rede e encaminhados nas LSPs. Ele foi modelado como um problema de programação matemática e resolvido pelo uso de heurísticas.

CUI et al. (2004) apresentam um algoritmo de ajuste de fila (QTA) visando fornecer QoS para TE em redes MPLS. O algoritmo visa a distribuição dos tráfegos de melhor esforço e com requisitos de QoS, para filas diferentes.

CELESTINO JR et al. (2004) propõem um esquema de TE com MPLS, usando Lógica Nebulosa, que efetua o balanceamento reativo da carga de tráfego visando o controle do congestionamento.

ANJALI et al. (2005) combinam DiffServ e MPLS com o objetivo de prover QoS para tráfego multimídia em redes IP. O tráfego das diferentes classes de serviço DiffServ é encaminhado através de LSP´s distintas.

DIN et al. (2005) propõem um regulador nebuloso de LSP´s para controle da preempção utilizando DiffServ e MPLS. O regulador utilizando lógica nebulosa controla o tráfego entrante associado a cada classe através de re-roteamento, bloqueio do tráfego de mais baixa prioridade existente, ou rejeição da admissão requerida.

Segundo SALVADORI et al. (2002), o principal mecanismo utilizado pelo *framework* MPLS, na Engenharia de Tráfego, é o chamado roteamento baseado em restrições (CBR, *Constraint Based Routing*). NOBRE et al. (2005) propõe um mecanismo alternativo de Engenharia de Tráfego que incorpora técnicas de aprendizado desassistido baseado em algoritmos genéticos e lógica nebulosa, o *Intelligent Dynamic Load Balance Algorithm* (IntelliDyLBA). Trata-se de um aprimoramento de um trabalho anterior (CELESTINO JR et al., 2004), no caso, o algoritmo *Fuzzified Dynamic Load Balance Algorithm* (FuDyLBA), um algoritmo de balanceamento de carga para redes MPLS baseado em lógica nebulosa. Enquanto os algoritmos CBR são baseados em um

mecanismo preventivo, o IntelliDyLBA é um mecanismo reativo, isto é, age quando o congestionamento é detectado pelo sistema de controle nebuloso.

ABDENNOUR (2005) propõe um preditor utilizando RNA e Lógica Nebulosa para predição de tráfego MPEG-4. O preditor de um passo a frente é baseado no ANFIS (Adaptative Network Fuzzy Inference System).

Algumas das propostas descritas tentam resolver o problema do congestionamento, enquanto outras tentam se adaptar à demanda de largura de banda adicional. Contudo em sua grande maioria, elas não cobrem o re-roteamento da LSP previamente estabelecida após a ocorrência de mudanças significativas na rede como redução ou aumento do tráfego em uma LSP, ou mesmo a sua liberação.

2.7 Considerações finais

Foi apresentada neste capítulo, uma introdução aos conceitos teóricos utilizados nesta tese. Foi feita uma introdução aos sistemas multimídia e descritas algumas abordagens para prover QoS em redes IP, tais como, Serviços Integrados, Serviços Diferenciados, MPLS, Roteamento baseado em QoS e Engenharia de Tráfego. Foram apresentados conceitos básicos de Computação Autonômica e Inteligência Computacional. Além disso, foram também apresentados os resumos de alguns trabalhos relacionados ao tema desta tese.

Após a apresentação dos fundamentos conceituais que dão suporte a este trabalho, é descrita no capítulo seguinte a metodologia proposta para a sua realização.

CAPÍTULO 3

MÉTODO

Neste capitulo é apresentado o método utilizado para o desenvolvimento e avaliação do sistema de TE. Inicialmente é apresentada na seção 3.1 a proposta de trabalho. Em seguida na seção 3.2, é apresentada a arquitetura do sistema proposto. Na seção 3.3, é feita a descrição da metodologia para implementação da TE**.** Na seção 3.4, são apresentadas as etapas do trabalho. Na seção 3.5, são apresentadas as ferramentas utilizadas para desenvolvimento da pesquisa. A seção 3.6 finaliza o capítulo realizando alguns comentários.

3.1 Proposta de trabalho

Este trabalho se propõe a desenvolver um sistema de TE, capaz de sustentar tráfego misto (dados, voz, vídeo) com diversos níveis de QoS na rede, utilizando MPLS, princípios de Computação Autonômica e técnicas de Inteligência Computacional.

É proposto o desenvolvimento de um algoritmo que possibilita a descoberta dos caminhos no domínio MPLS para as LSP´s, a criação, a ativação, e o re-roteamento destas mesmas LSP´s de acordo com as condições do ambiente. A descoberta dos enlaces mais adequados para as LSP´s será feita levando em conta as necessidades das aplicações em termos de vazão e atraso fim-a-fim, vazões de tráfego nos enlaces do domínio, e também medições de tráfego efetuadas entre os nós origem e destino das aplicações. A criação e ativação de LSP´s serão implementadas com o auxílio de lógica nebulosa. O algoritmo proposto utiliza o principio de Computação Autonômica denominado Auto-configuração.

É proposto o desenvolvimento de um algoritmo monitoração e re-roteamento de LSP´s. O re-roteamento tem como objetivo manter a QoS das aplicações. Caso a LSP

corrente não esteja mantendo os níveis de QoS desejados, deve ser providenciado o estabelecimento de uma nova. Por outro lado, caso as aplicações com maior prioridade aumentem a sua vazão de tráfego e ultrapassem o valor combinado, se possível o sistema deve procurar atende-las procurando um caminho mais adequado. Além disso, após a ocorrência de interrupção em um enlace que atende a LSP principal, os pacotes deverão ser encaminhados para uma LSP reserva. A LSP reserva é criada e ativada automaticamente. Os princípios de Computação Autonômica utilizados são Auto-configuração e Auto-cura.

O sistema de TE deve controlar a admissão do tráfego das aplicações no domínio MPLS. O controle de admissão das conexões leva em conta as prioridades das aplicações. As aplicações que mantêm as suas taxas de tráfego dentro dos valores acordados devem ser atendidas com uma maior prioridade. As aplicações que não cumprem o acordo são atendidas, mas devem ser "penalizadas". A idéia é alterar a prioridade de admissão das aplicações no domínio MPLS levando em conta os seus comportamentos observados após a análise do histórico do tráfego. O valor da prioridade de atendimento de uma aplicação será alterado de acordo com o seu perfil de tráfego. Desta forma, é proposta também o desenvolvimento de um algoritmo para controle de admissão de conexões e reconhecimento e classificação dos perfis de comportamento de tráfego das aplicações. O principio de Computação Autonômica utilizado é Auto-proteção.

O Planejamento da rede visa assegurar que os nós e as capacidades dos enlaces suportarão o crescimento previsto do tráfego. A rede deve ser projetada para o atendimento da demanda de tráfego com qualidade e menor custo possível. Portanto, há necessidade de constantes medições nos enlaces da rede para que periodicamente seja providenciada a ampliação de suas capacidades. Portanto, após a identificação dos enlaces mais críticos pode-se prever o comportamento futuro do tráfego neste enlaces, para que periodicamente seja providenciada a ampliação de suas capacidades de forma proativa. A previsão da vazão de tráfego nos enlaces críticos da rede é feita utilizando RNA´s. O principio de Computação Autonômica utilizado é Auto-cura.

A otimização dos recursos da rede deve ser feita periodicamente. A meta da otimização é fazer com que as LSP´s que atendem as aplicações, utilizem os melhores caminhos possíveis. Assim, um caminho que num determinado momento é escolhido como a melhor opção para o estabelecimento de uma LSP, pode deixar de sê-lo dependendo das novas condições da rede. A otimização dos recursos da rede será

implementada utilizando Algoritmos Genéticos. O principio de Computação Autonômica utilizado é Auto-otimização.

3.2 Arquitetura do sistema proposto

O sistema de TE proposto é distribuído e adaptativo. É distribuído no sentido de que cada nó de ingresso toma as suas próprias decisões, e é adaptativo por se ajustar à mudanças no estado da rede. A Figura 8 mostra a arquitetura do sistema proposto. A função ou unidade de TE está presente em cada um dos roteadores (LER, *Label Edge Router*) de ingresso do domínio MPLS. As funções de TE são responsáveis pela identificação da necessidade de criação, cálculo do caminho, ativação, monitoramento, e re-roteamento de LSP´s. No caso, as LSP´s são estabelecidas levando em conta as necessidades de QoS da aplicação (vazão e atraso fim-a-fim), e também da sua prioridade. Quanto ao re-roteamento de LSP´s, ele pode ocorrer em função da necessidade de proporcionar QoS para as aplicações prioritárias, interrupção de um enlace o qual faz parte a LSP principal e otimização dos recursos da rede, a qual é realizada periodicamente.

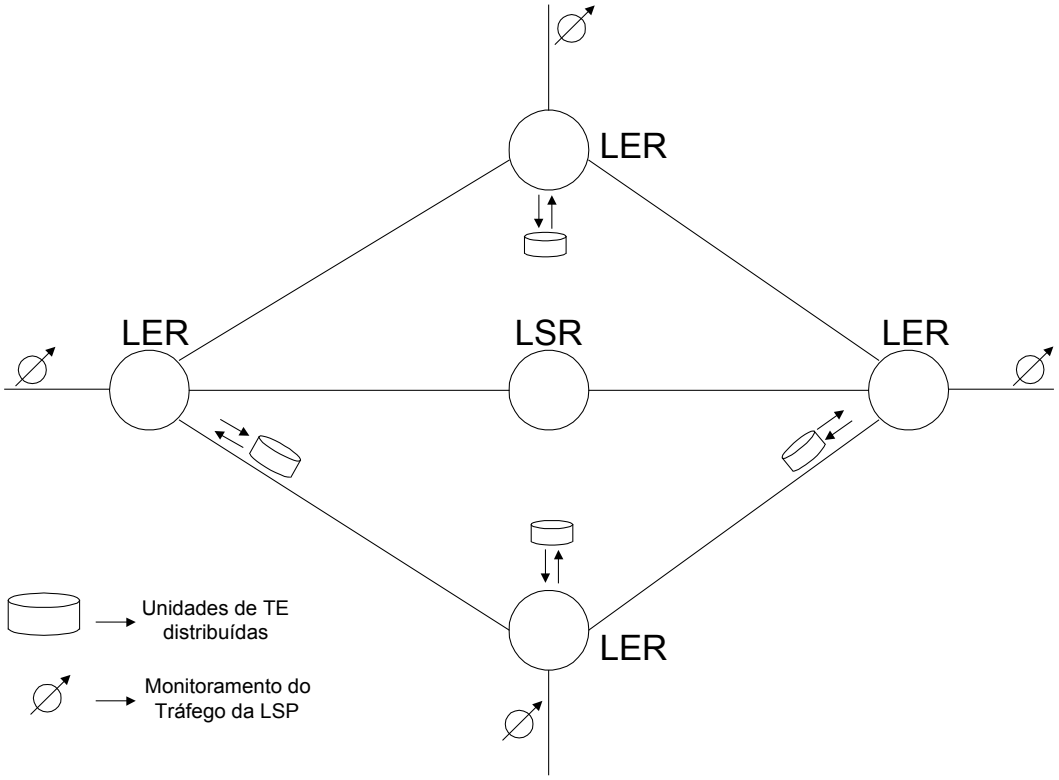

Figura 8 - Arquitetura do sistema proposto.

Um requisito importante para a implantação de TE em um domínio é o conhecimento do tráfego que é escoado através dele. Desta forma, é necessário que constantemente, sejam efetuadas medições de tráfego nos enlaces, e também entre os nós origem e destino das aplicações. Estas medições devem ser constantemente trocadas entre si pelos LSR´s do domínio MPLS utilizando protocolos como por exemplo, o OSPF-TE. O protocolo OSPF-TE permite o fornecimento de informação dinâmica sobre a rede, de um LSR para outro (KHAN; ALNUWEIRI, 2003). Esta informação pode ser, por exemplo, a vazão nos enlaces.

3.3 Metodologia para implementação da Engenharia de Tráfego

A Figura 9 apresenta a metodologia para a implementação da TE no domínio MPLS. O ponto de partida é a etapa (a), onde são feitas as definições da topologia do domínio MPLS e das aplicações de dados, voz e vídeo com os respectivos requisitos de QoS. São também definidas nesta etapa as aplicações de Background que concorrerão com as aplicações com maior prioridade. Na etapa (b) é feita a ativação da simulação do domínio MPLS.

As medições de tráfego são fundamentais para a implementação do sistema. Na etapa (c) é ativado o módulo de medição de tráfego. Nesta etapa, é realizado o recolhimento das medições de tráfego nos enlaces do domínio e também entre os nós origem e destino das aplicações (fim-a-fim). São realizados também o processamento, a classificação, e o armazenamento das medições de tráfego por enlaces e fluxos, além da escolha das medições representativas por enlaces.

Na etapa (d) é realizada a identificação e classificação do perfil de tráfego das aplicações. Em seguida, na etapa (e) é ativada a função para descoberta dos caminhos e criação das LSP´s. Na etapa (f) é realizada a ativação das LSP´s criadas. O tráfego das aplicações que não obtiveram LSP´s é bloqueado na etapa (g).

Na etapa (h) é realizada a avaliação das medições de tráfego. Neste ponto é feita a verificação se as necessidades de QoS das aplicações que possuem prioridades estão sendo atendidas. Caso os requisitos de QoS estejam abaixo das necessidades, deverão ser tomadas providências. Por outro lado, antes de efetivamente serem tomadas providencias,

é necessário verificar se a queda da qualidade não ocorreu em função da ocorrência de parada em enlaces da rede. Isto é feito na etapa (i).

Caso não tenha ocorrido queda em enlaces, é ativada na etapa (l), a função para criação e re-roteamento de LSP´s.

Caso tenha havido falha, é efetuado o registro no arquivo de histórico de quedas, mostrado na etapa (j). Na seqüência é verificado se existe LSP reservada, conforme mostra a etapa (k). Se existir LSP já reservada anteriormente, ou se a etapa (l) já foi realizada, então deve ser providenciada a sua ativação, conforme mostra a etapa (m).

Voltando a etapa (h), após a verificação feita, caso se chegue a conclusão que as necessidades de QoS das aplicações estão sendo atendidas, deve-se então passar para a etapa (n). Nesta etapa, é verificado se já existe um histórico de medições de tráfego adequado. Caso não exista histórico de medições de tráfego suficiente, então a simulação continua normalmente.

Caso já exista um histórico de medições de tráfego adequado, é ativado o módulo de otimização do uso dos enlaces do domínio MPLS, conforme mostra a etapa (o). Em seguida, é verificado na etapa (p) a necessidade de re-roteamento de LSP´s.

Caso haja necessidade de re-roteamento de LSP´s, elas são ativadas na etapa (m), sendo que a simulação continua normalmente a partir deste ponto. Em caso contrário, na etapa (q), é ativado o módulo de identificação de enlaces críticos.

Em seguida na etapa (r), é verificado se existem enlaces críticos. Caso existam enlaces críticos, é ativado na etapa (s), o módulo responsável pela previsão da vazão de tráfego.

Na etapa (t), se necessário, é sinalizada a necessidade de ampliação da largura de banda do enlace crítico.

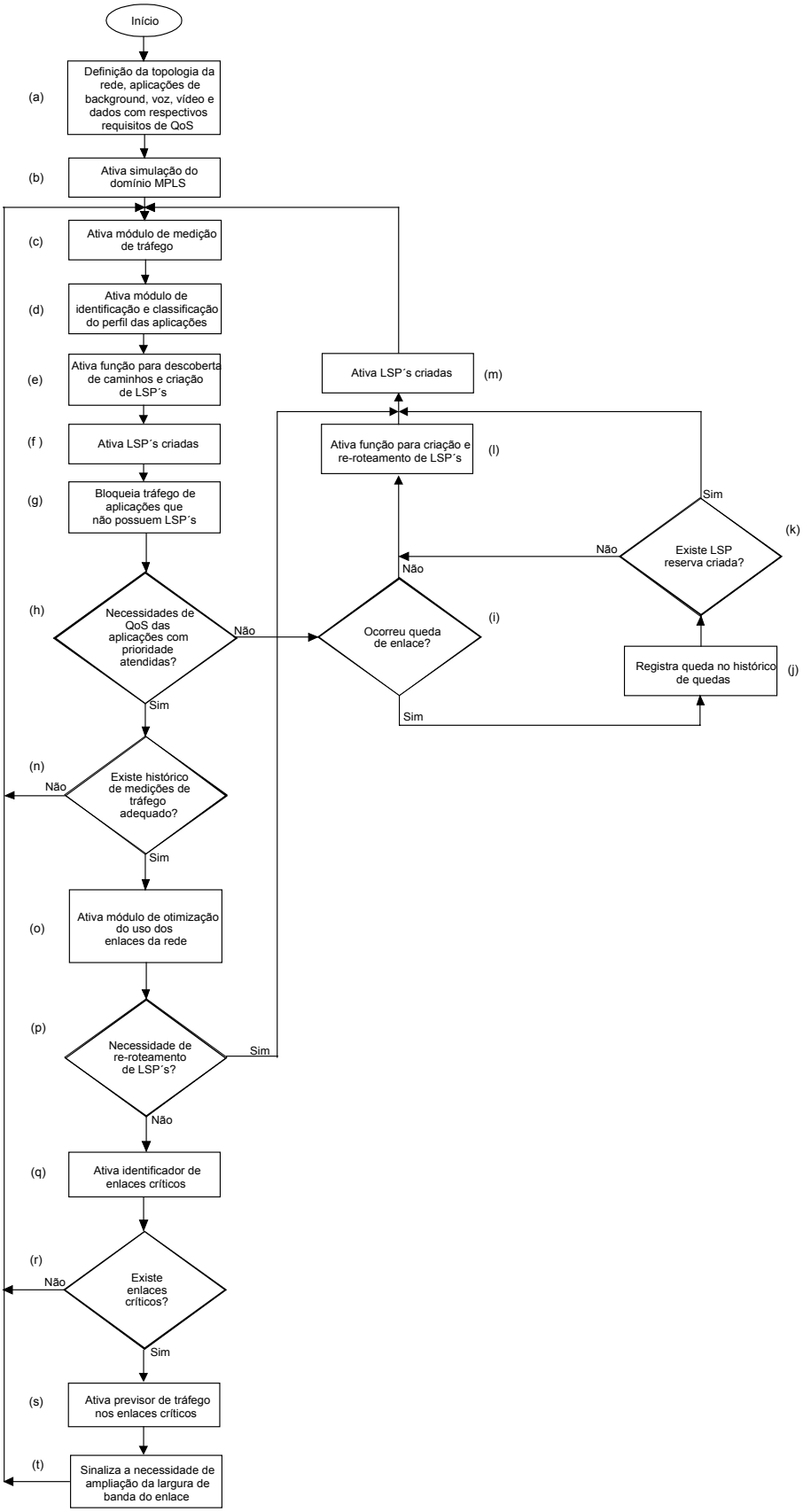

Figura 9 - Metodologia para implementação da TE.

3.4 Etapas do trabalho e avaliação

O sistema de TE proposto é implementado em um domínio MPLS simulado com a utilização do ns2 (VINT, 2003). O trabalho de desenvolvimento do sistema de TE é composto das etapas relacionadas nos objetivos específicos apresentados no capítulo 1, ou seja:

a) Desenvolver um algoritmo para descoberta dos caminhos, criação e ativação das LSP´s.
b) Desenvolver um algoritmo para monitoração e re-roteamento de LSP´s;
c) Desenvolver um algoritmo para controle de admissão de conexões utilizando reconhecimento e classificação dos perfis de comportamento de tráfego das aplicações;
d) Desenvolver um algoritmo para otimização dos recursos da rede;
e) Desenvolver um algoritmo para a identificação dos enlaces críticos e sinalização da necessidade de ampliação de suas capacidades de forma proativa;
f) Avaliação dos resultados obtidos com a implementação da TE.

Após o termino do desenvolvimento de cada algoritmo, o sistema de TE é avaliado com a realização de dez experimentos mantendo-se as mesmas condições em cada um deles.. A avaliação é feita inicialmente utilizando uma topologia de simulação simples (I). O objetivo desta simulação é verificar o comportamento do sistema em uma topologia conhecida, de forma a poder avaliar o seu comportamento sob condições controladas. Numa segunda fase de avaliação é utilizada uma topologia um pouco mais complexa (II), visando verificar o comportamento da TE em uma situação semelhante às encontradas no mundo real. Esta topologia é uma adaptação da utilizada em LIU et al. (2003), com o acréscimo de dois LSR´s, trinta fontes de tráfego de origem e trinta fontes de destino adicionais no domínio. Alem disso, foram realizadas modificações na forma de interligação entre os LSR´s.

Durante a avaliação do sistema de TE é analisada a QoS oferecida as aplicações que cruzam o domínio MPLS e também os indicadores de desempenho da rede. O domínio é alimentado com aplicações do tipo dados, voz e vídeo. Para gerar os tráfegos das aplicações de voz são utilizadas fontes de dados constantes (CBR). Os tráfegos das aplicações de vídeo foram obtidos com a utilização de traces reais de filmes (Jurassic Park I, Silence of The Lambs e Star Wars IV). No caso das aplicações de dados foram utilizadas fontes FTP e

traces reais HTTP. Os parâmetros utilizados para a avaliação da QoS oferecida às aplicações são atraso, jitter, vazão e perda de pacotes. As medições dos parâmetros de QoS são realizadas fim-a-fim. Os indicadores utilizados para avaliar o desempenho da rede são a utilização média dos enlaces e a relação entre o tráfego oferecido e escoado (RTeo). Além disso, são avaliadas também a perda média de pacotes obtida para o conjunto de aplicações com e sem a implementação da TE.

A relação entre o tráfego escoado e o tráfego oferecido ao domínio MPLS é calculada pela equação (1).

$$RTeo = \frac{TTe}{TTo} \tag{1}$$

Onde:

TTe = Intensidade total de tráfego escoado ou que foi entregue pelos LSR´s de saída do domínio MPLS aos nós de destino num determinado período de tempo;

TTo = Intensidade total de tráfego oferecido ou gerado pelos nós de origem e entregue aos LSR´s de entrada do domínio MPLS num determinado período de tempo.

A intensidade total de tráfego oferecido pelos nós de origem e entregue aos LSR´s de entrada do domínio MPLS é calculada pela equação (2).

$$TTo = \sum_{i=1}^{n} \sum_{j=1}^{m} To_{ij} \tag{2}$$

onde:

To_{ij} = Intensidade de tráfego medida no enlace que interliga o LSR de entrada i. com o nó de origem j num determinado período de tempo t;

n = Número total de LSR´s de entrada do domínio MPLS;

m = Número de enlaces interligando cada LSR de entrada com os seus respectivos nós de origem.

A intensidade total de tráfego escoado ou que foi entregue pelos LSR´s de saída do domínio MPLS aos nós de destino é calculada pela equação (3).

$$TTe = \sum_{i=1}^{o} \sum_{j=1}^{p} Te_{ij} \tag{3}$$

onde:

Te_{ij} = Intensidade de tráfego medida no enlace que interliga o LSR de saída *i* com o nó de destino *j* num determinado período de tempo *t*.

o = Número total de LSR´s de saída do domínio MPLS;

p = Número de enlaces interligando cada LSR de saída com os seus respectivos nós de destino.

3.5 Ferramentas

Esta seção relaciona as ferramentas utilizadas para o desenvolvimento do trabalho.

3.5.1 Simulador de redes ns2

Para a simulação do sistema de TE foi utilizado o simulador de redes Network Simulator (ns2), versão 2.1b8a. Esta versão do Network Simulator também foi utilizada em FERNANDEZ (2002). As simulações foram realizadas em computadores com o sistema operacional Windows 2000. O *ns2* é um simulador baseado em eventos discretos e orientado a objetos para a simulação de redes. O objetivo do *ns2* é proporcionar um ambiente para o desenvolvimento de pesquisas em torno dos protocolos que constituem a Internet, isto é, que utilizam a pilha TCP/IP, tanto no contexto das redes fixas quanto móveis, com fio e sem fio. O *ns2* é um dos simuladores mais utilizados atualmente pela comunidade científica para pesquisas em rede de computadores. Ele está sendo desenvolvido dentro do projeto VINT (2003), por algumas universidade e centros de pesquisas americanas. O *ns2* é uma ferramenta poderosa para configurar simulações complexas e também para comparação de resultados de pesquisas. Estão implementados no *ns2* os protocolos IP, TCP, UDP, FTP, HTTP, além dos protocolos de Roteamento. O *ns2* executa a simulação e pode gerar vários arquivos como resultados, que podem ser utilizados para construir tabelas e gráficos. Junto com o *ns2* é distribuído um software para animação da simulação, o nam (network animator), que pode ser executado após o término da simulação para a sua visualização. O nam utiliza um arquivo de *trace*, gerado pelo *ns2*. Em termos de QoS, a distribuição básica do ns somente inclui funcionalidade para IntServ. Entretanto, é possível obter contribuições que incluem DiffServ, MPLS e RSVP.

A biblioteca de protocolos e mecanismos implementados no *ns2* é bastante vasta, abrangendo implementação dos protocolos TCP, UDP, IP além de disciplinas de serviços, como, WFQ (*Weighted Fair Queueing*), protocolos para redes móveis, como o IP móvel, tecnologias de redes sem fio locais e de longa distância.

O *ns2* fornece também bibliotecas de funções para a geração de alguns tipos de tráfego como: CBR (*Constant Bit Rate*) utilizado para simular tráfego constante e voz, ON-OFF para tráfego em rajada e voz comprimida, FTP para gerar tráfego correspondente a aplicações de transferência de arquivos e VBR (*Variable Bit Rate*) para tráfego com taxa de dados variável.

O ns2 adota duas linguagens de programação: C++ para o núcleo do simulador e Otcl para construção de *scripts* e modelagem da simulação.

Os *scripts* que contém os comandos e o sistema a ser simulado são construídos em Otcl (Object Tool Control Language), uma versão orientada a objetos da linguagem de script tcl (Tool Control Language). Otcl é uma linguagem interpretada, e um dos motivos porque foi escolhida, é que os scripts são tarefas interativas e freqüentemente refinadas (alteradas) no programa de simulação, sem necessidade de recompilação.

Quando for necessária a inclusão de novas características em protocolos implementados na distribuição do *ns2* ou, ainda, implementar novos protocolos, é aconselhável a construção direta utilizando C++. Pode-se implementar um novo protocolo através de herança de classes existentes em C++ e depois ligar o novo objeto C++ a um objeto Otcl correspondente. O novo objeto Otcl, poderá então ser invocado a partir de scripts Otcl. Assim, será possível executar as novas funções implementadas em C++ através do desenvolvimento de scripts de simulação em Otcl. Estes scripts conterão uma chamada a um objeto Otcl ligado a um objeto C++ incluído pelo usuário no núcleo do *ns2*. Além da possibilidade de chamar funções em C++ a partir do Otcl pode-se também chamar funções definidas em Otcl no C++.

3.5.2 JFS

É utilizada a ferramenta JFS (MORTENNSEN, 2004) para desenvolver o Sistema de Inferência Nebulosa utilizado neste trabalho. Esta ferramenta oferece um ambiente para o desenvolvimento de protótipo de Sistemas de Inferência Nebulosa (SIF), além de permitir

verificação inicial do modelo especificado. Além disso, dispõe de várias ferramentas de otimização, dentre as quais o algoritmo de Wang-Mendel, algoritmos genéticos e redes neurais. Após o desenvolvimento do SIF é gerada uma biblioteca em código C++, que implementa o controlador, podendo então ser integrada ao simulador *ns2*.

3.5.3 MatLab

MATLAB é um "software" interativo de alta performance voltado para o cálculo numérico. O MatLab integra análise numérica, cálculo com matrizes, processamento de sinais e construção de gráficos. Ele permite a interação do usuário através de uma janela, denotada por *Janela de Comando*, onde os comandos devem ser fornecidos pelos usuários para que os cálculos e resultados, sejam exibidos. O MatLab também pode executar um seqüência de comandos que está armazenada em um arquivo. Além disso, ele também possui bibliotecas para cálculos específicos (*Toolboxes*). Existem *Toolboxes* para diversas áreas tais como, Algoritmos Genéticos, Redes Neurais Artificiais, Lógica Nebulosa, processamento de Sinais, Otimização, etc. Para construir os programas que implementam as Redes Neurais Artificiais e Algoritmos Genéticos utilizados no trabalho, são utilizados *Toolboxes* do MatLab.

3.5.4 Linguagens de Programação

As linguagens de programação utilizadas para desenvolvimento dos módulos de softwares necessários para a realização deste trabalho são C/C++ e Otcl.

3.6 Considerações finais

Neste capítulo foi apresentado o método utilizado para a implementação e avaliação do sistema de TE no domínio MPLS. Foram apresentados a proposta de trabalho, a arquitetura do sistema de TE, e a metodologia para implementação da TE. Em seguida foram apresentadas as etapas do trabalho, e as ferramentas utilizadas para o seu desenvolvimento. Após a descrição da metodologia, o capítulo seguinte trata das implementações do sistema de TE no domínio MPLS.

CAPÍTULO 4

IMPLEMENTAÇÕES DO SISTEMA DE ENGENHARIA DE TRAFÉGO NO DOMINIO MPLS

A partir da metodologia apresentada no capítulo anterior, pode-se descrever, neste capítulo, o processo de implementação do sistema de TE. O sistema de TE proposto utiliza técnicas de Inteligência Computacional. As técnicas de CI utilizadas são lógica nebulosa, redes neurais artificiais e algoritmos genéticos. Os princípios da Computação Autonômica utilizados são auto-configuração, auto-cura, auto-otimização e auto-proteção.

Inicialmente, é feita na seção 4.1 a descrição do problema da TE em um domínio MPLS. Em seguida, na seção 4.2 é apresentada a descrição do SIF utilizado no sistema de TE proposto. Na seção 4.3, é descrito o modelo básico do sistema de TE. Nas seções 4.4 e 4.5 são descritas as modificações feitas no sistema visando o re-encaminhamento da carga de tráfego quando ocorrem falhas em enlaces, e o re-roteamento de LSP´s em caso do aumento da vazão das aplicações prioritárias, respectivamente. A seção 4.6 descreve o CAC utilizando reconhecimento e classificação dos perfis de comportamento de tráfego das aplicações. Nas seções 4.7 e 4.8 são descritos os algoritmos para otimização dos recursos da rede e identificação de enlaces críticos. Na seção 4.9 é apresentado o protótipo para a simulação do sistema. A seção 4.10 finaliza o Capítulo realizando alguns comentários.

4.1 O problema da Engenharia de Tráfego em um domínio MPLS

Dado um sistema autônomo definido pelos seus nós e enlaces, o problema da TE é encontrar uma rota adequada que atravessando o domínio interliga um nó origem de uma

aplicação a um nó destino. Uma forma de implementar a TE de forma efetiva em um sistema autônomo é através do uso do MPLS.

O MPLS consegue forçar o encaminhamento dos pacotes através de certas rotas (ou caminhos) pré-estabelecidas, o que é impossível no esquema IP convencional. Os caminhos que transportarão as LSP´s são estabelecidos pelos LSR´s de entrada. Para que isto seja possível é necessário que os LSR´s de ingresso tenham conhecimento da situação dos enlaces do domínio MPLS. Desta forma, é necessário que constantemente, sejam efetuadas medições de tráfego nos enlaces. Os resultados das medições devem ser constantemente trocados entre si pelos LSR´s do domínio MPLS utilizando protocolos como por exemplo, o OSPF-TE. A vazões nos enlaces são exemplos de medições que podem ser trocadas entre si pelos LSR´s.

Seja como exemplo o domínio MPLS mostrado na Figura 10, onde os pacotes destinados ao receptor B são enviados a partir do LSR1 pelo transmissor A utilizando uma taxa de 40 Mbps. A largura de banda reservada em cada enlace significa que já existem LSP´s estabelecidas neste domínio. Os valores mostrados estão disponíveis para serem utilizados apenas por estas LSP´s. Se no domínio fosse utilizado o roteamento IP padrão, o tráfego seria encaminhado através do enlace 1-3. Isto ocorre pelas seguintes razões:

- O enlace 1-3 é o menor caminho entre o LSR1 e o receptor B, necessitando de apenas dois saltos para alcançar o destino;
- O valor da métricas deste caminho é igual a um (m =1).

Entretanto, com a utilização do roteamento IP padrão ocorreria uma grande perda de pacotes e um alto atraso fim-a-fim, uma vez que o enlace entre o LSR1 e LSR3 não possui folga de largura de banda suficiente.

Com a implementação da TE através da utilização de MPLS, o problema do congestionamento do enlace 1-3 pode ser resolvido. No caso, uma LSP pode ser estabelecida através do caminho formado pelos enlaces 1-2-3. A menor folga de largura de banda nos enlaces deste caminho é 55 Mbps, sendo suficiente para atender a necessidade de vazão da aplicação (40 Mbps).

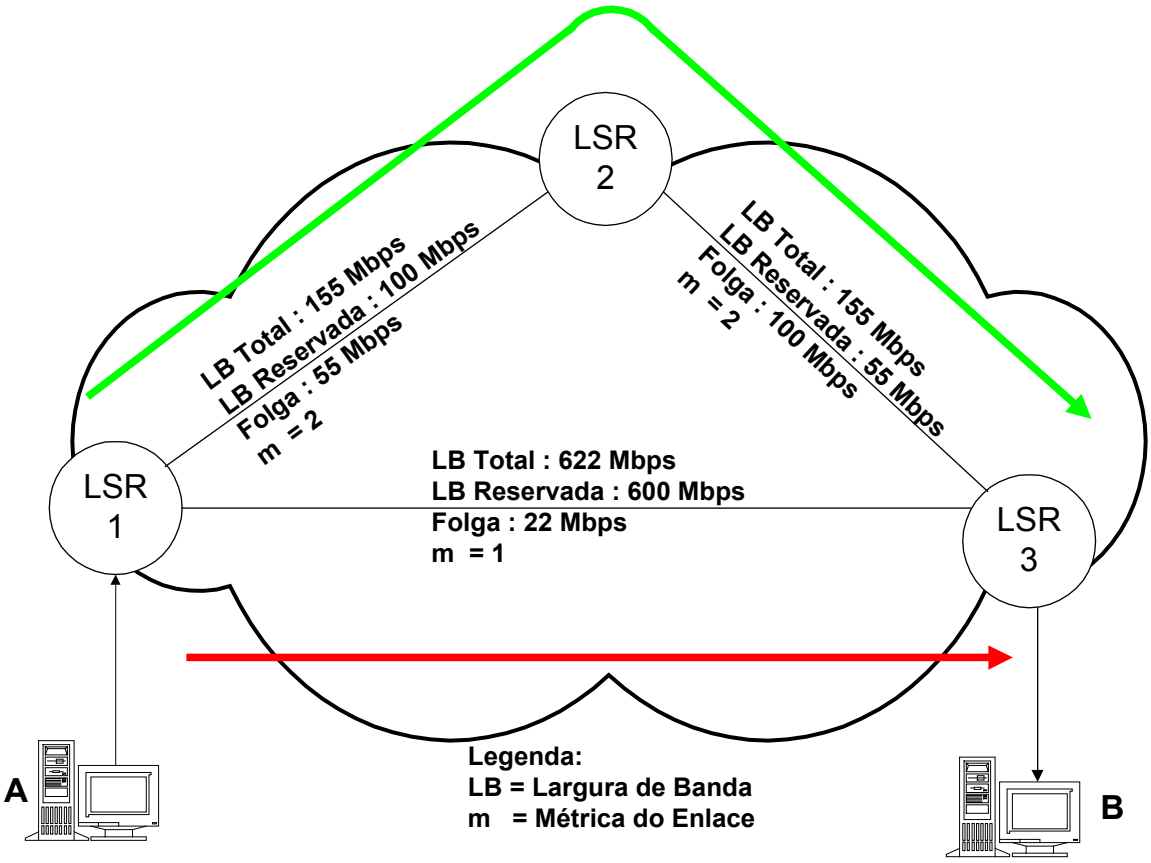

Figura 10 - Exemplo de um domínio MPLS.

Por outro lado, apesar do MPLS viabilizar a implementação efetiva da TE em redes IP, não significa necessariamente que exista garantia de fornecimento de QoS fim-a-fim para as aplicações. Desde que o tráfego gerado pelas aplicações ultrapasse por algum motivo o valor reservado, pode ocorrer congestionamento. Como conseqüência disso, a QoS oferecida às aplicações pode ser afetada. Assim, visando o oferecimento de QoS fim-a-fim para as aplicações em um domínio MPLS, é proposta a implementação de um sistema de TE distribuído e adaptativo utilizando princípios de Computação Autonômica. A medição, a análise do tráfego, a decisão de criar novas LSP´s, e ainda o caminho por onde elas são estabelecidas, são executados automaticamente. O sistema de TE proposto é distribuído no sentido de que cada LSR de ingresso toma suas próprias decisões, e é adaptativo por se ajustar às mudanças no estado da rede. Em cada LSR de entrada a função de TE escolhe o caminho por onde será estabelecida a LSP e depois providencia efetivamente a sua criação e a ativação. O cálculo do caminho para uma LSP leva em

conta a prioridade, a vazão, o atraso fim-a-fim da aplicação e as medições de tráfego nos enlaces do domínio.

Admita-se como exemplo, que existe uma aplicação genérica com necessidades definidas de vazão e atraso fim-a-fim para a qual existe a necessidade de LSP. Supondo-se ainda que existe um conjunto de caminhos candidatos com largura de banda suficiente, qual seria a melhor opção para atender a aplicação? O caminho candidato que resultar em um menor sobra de largura de banda? Ou o caminho candidato que ofereça um valor de atraso fim-a-fim mais adequado às necessidades da aplicação? A melhor opção seria aquela que combinasse as duas opções. Uma forma de obter uma solução deste tipo é utilizando um Sistemas de Inferência Nebulosa. O SIF, que será melhor descrito na Seção 4.2, calcula os custos nebulosos dos caminhos candidatos levando em conta duas variáveis de entrada. A primeira é a necessidade de vazão da aplicação versus a disponibilidade de largura de banda do caminho candidato. A segunda é o atraso fim-a-fim desejado pela aplicação versus o atraso fim-a-fim oferecido. Quanto mais adequados forem os caminhos em termos de vazão e atraso, mais baixos serão os seus custos nebulosos. O problema pode ser formulado matematicamente através da função objetivo apresentada na equação (4). O objetivo é minimizar a soma, para todos os caminhos candidatos e para todas as LSP´s, do produto entre o **custo nebuloso** da cada caminho candidato e a vazão de tráfego de cada LSP. A vazão desejada pela aplicação prioritária i é representada por λi. O custo nebuloso a_{ij} indica o grau de adequação do caminho candidato j para encaminhar o tráfego da LSP i. Se o valor do **custo nebuloso** é baixo então a sua adequação será alta e vice-versa. A variável de decisão x_{ij} determina se a LSP i usa ou não o caminho candidato j. Se $x_{ij} = 1$ então a LSP i usa caminho candidato j, em caso contrário $x_{ij} = 0$.

$$Zx = Minimize \sum_{i=1}^{np} \sum_{j=1}^{nf} a_{ij} \lambda_i x_{ij} \tag{4}$$

A função objetivo está sujeita às restrições apresentadas nas equações (5), (6) e (7).

$$\sum_{i=1}^{nf} \lambda_i x_{ij} \leq \mu_j, \qquad \forall j \in P \tag{5}$$

$$C_j \leq D_i, \qquad \forall i \in F \quad e \quad \forall j \in P \tag{6}$$

$$0 \leq x_{ij} \leq 1 \qquad \forall i \in F \quad e \quad \forall j \in P \tag{7}$$

onde:

P = Conjunto de caminhos candidatos.

np = Número de elementos de P.

F = Conjunto de LSPs.

nf = Número de elementos de F.

μ_J = Largura de banda disponível no caminho candidato j.

D_i = Atraso máximo fim-a-fim permitido para o i-ésimo LSP.

C_j = Atraso representativo do caminho candidato j.

A restrição (5) garante que a capacidade dos caminhos candidatos não seja excedida. Já a restrição (6) garante que o caminho candidato atende as necessidades de atraso da LSP que está sendo criada. Finalmente, a restrição (7) especifica que os valores assumidos pela variável de decisão x_{ij} são 0 ou 1.

Caso os caminhos escolhidos para estabelecer as LSP´s deixem de oferecer uma QoS compatível com as necessidades das aplicações prioritárias, surge a necessidade de re-roteamento. O re-roteamento de LSP´s é realizado quando ocorre falhas em enlaces ou quando as aplicações prioritárias ultrapassam as taxas de vazão acordadas. No caso de falha em enlace, a duração da interrupção pode não ser curta o suficiente para que as aplicações de multimídia e de tempo real consigam manter suas sessões, sem o re-roteamento para uma LSP alternativa. O caminho escolhido para a LSP alternativa deve, portanto, garantir QoS compatível com o oferecido pela LSP principal. Por outro lado, se a vazão de tráfego gerada pela aplicação ultrapassar o valor reservado, pode ocorrer congestionamento e em conseqüência comprometer o QoS oferecido. O sistema de TE, após detectar o aumento da vazão e a queda do QoS, tenta efetuar ajustes na rede visando encontrar um caminho mais adequado para o re-roteamento do tráfego da aplicação. Na seção seguinte é feita a descrição do SIF utilizada no sistema de TE proposto.

4.2 O Sistema de Inferência Nebulosa

A definição de um algoritmo para a escolha de LSP´s que atendam diferentes aplicações com diferentes requisitos de QoS é um problema difícil. Entretanto a complexidade do algoritmo pode ser reduzida através da escolha de um sub-conjunto de parâmetros de QoS. Neste trabalho é usado um SIF para cálculo dos custos dos caminhos candidatos para estabelecimento das LSP´s das aplicações. Os custos nebulosos são calculados levando em conta os valores das métricas associadas aos parâmetros de QoS desejados pelas aplicações e os oferecidos pelos caminhos candidatos. Para cada métrica, é definido um conjunto de funções de pertinência. Os parâmetros destas funções de pertinência refletem dinamicamente as necessidades de QoS das aplicações, como também do estado dos enlaces do caminho candidato. A lógica nebulosa é vantajosa em situações onde os valores associados às métricas não são fixos, mas estão compreendidos numa faixa de valores. Assim, representando as métricas associadas aos parâmetros de QoS desejados pelas aplicações e os oferecidos pelos caminhos candidatos como metas nebulosas, ao invés de valores fixos, facilita a solução.

O SIF utilizado neste trabalho possui duas variáveis lingüísticas de entrada e uma de saída, mostradas na Figura 11. As variáveis lingüísticas de entrada representam as métricas ou parâmetros de QoS requisitados pelas aplicações e também aqueles disponibilizados pelos caminhos candidatos para estabelecimento das LSP´s. A variável lingüística de saída representa o custo nebuloso de cada um dos possíveis caminhos candidatos.

Para cada variável lingüística é definido um conjunto de funções de pertinência. As métricas utilizadas nas variáveis de entrada são a folga de banda (*FB*) e o atraso do caminho candidato, além da vazão e do atraso máximo desejado pela aplicação. A folga de banda residual num enlace i é calculada pela equação (8).

$$FB_i(t) = LB_i(t) - VZ_i(t) \tag{8}$$

Sendo que:

$FB_i(t)$ = Folga de banda residual num enlace i no instante t

$LB_i(t)$ = Largura de banda do enlace i no instante t

$VZ_i(t)$ = Vazão no enlace i no instante t.

Dado um determinado caminho que interliga dois nós origem e destino, a métrica folga de banda representativa do caminho é o resultado da escolha da menor folga de banda residual nos enlaces que o compõem, como mostra a equação (9).

$$FB(t) = \min[FB_1(t), FB_2(t), FB_3(t), FB_4(t), FB_5(t), ...FB_n(t)] \quad (9)$$

Onde:

$FB(t)$ = Folga de banda representativa do caminho que interliga os nós origem e destino no instante *t*.

São apresentadas na Figura 11 as entradas e a saída do SIF. A relBV representa a relação entre a fola de banda representativa do caminho candidato para estabelecimento da LSP e a vazão desejada pela aplicação. A relAT representa a relação entre o atraso máximo desejado pela aplicação e o atraso representativo do caminho candidato para estabelecimento da LSP. O Custo_cam representa o custo nebuloso do caminho candidato que esta sendo avaliado.

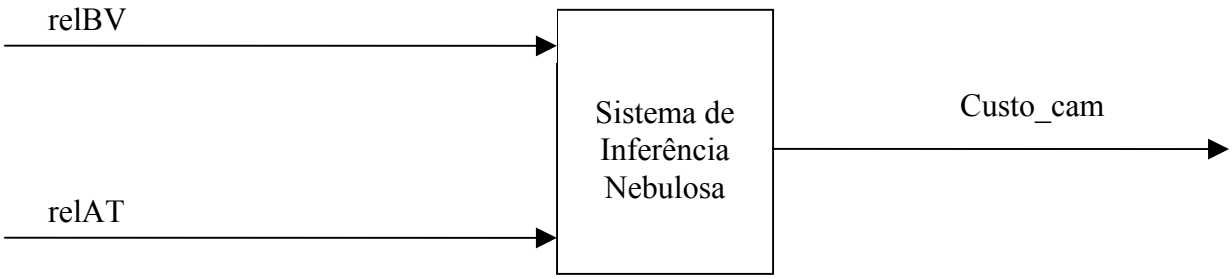

Figura 11 – Entradas e saída do SIF.

Os valores lingüísticos da variável relBV são "Baixa", "Adequada" e "Alta", conforme é mostrado na Figura 12.

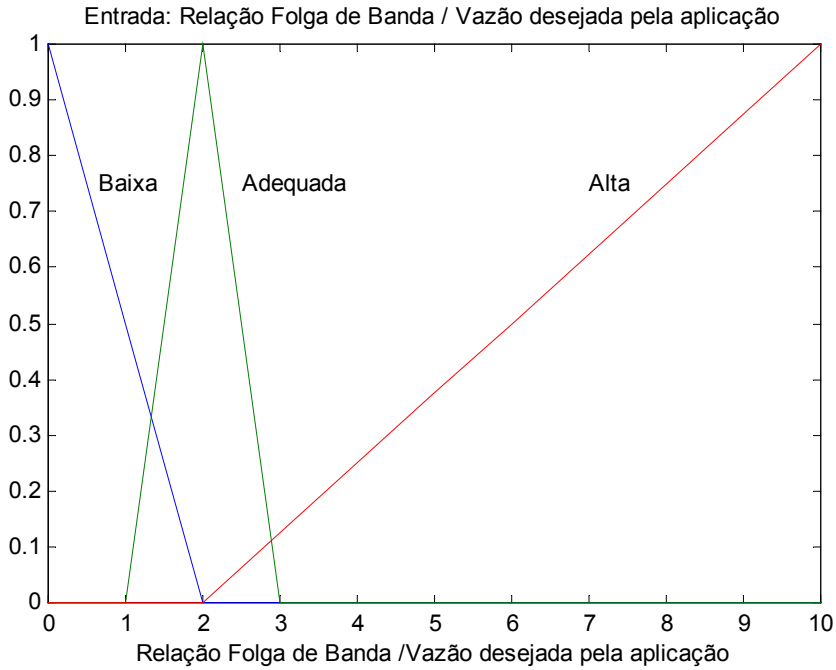

Figura 12 – Entrada: Relação folga/vazão (relBV).

A Figura 13 mostra os valores lingüísticos da variável relAT, ou seja, "Baixa", "Adequada" e "Alta".

Figura 13 – Entrada: Relação atraso máximo / atraso representativo do caminho (relAT).

A Figura 14 mostra os valores lingüísticos da variável Custo_cam, ou seja, “Excelente”, “Bom”, “Regular”, “Ruim” e “Muito ruim”.

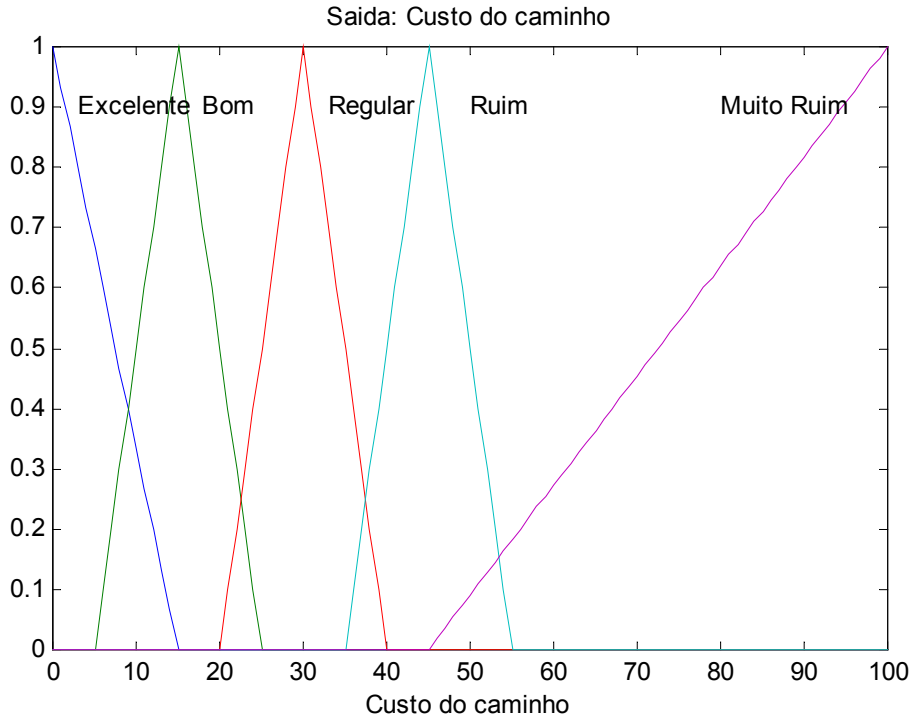

Figura 14 - Saída: Custo do caminho (Custo_cam).

O método de Inferência utilizado neste trabalho é o proposto por MAMDANI (1975). Para produzir a saída Custo_cam, é utilizado o método de defuzificação de centro de área. A base de regras do SIF possui nove regras. O Quadro 2 e o algoritmo da Figura 15 mostram a base de regras para a construção do SIF.

QUADRO 2

Base de regras do SIF.

RelBV	RelAT	Custo_cam
Baixa	Baixa	Muito ruim
Baixa	Adequada	Ruim
Baixa	Alta	Muito ruim
Adequada	Baixa	Ruim
Adequada	Adequada	Excelente
Adequada	Alta	Bom
Alta	Baixa	Ruim
Alta	Adequada	Bom
Alta	Alta	Regular

```
Algoritmo SIF
{Objetivo: Construir a base de regras utilizada para cálculo dos custos dos caminhos      }
{          candidatos para estabelecimento de LPS´s para as aplicações.                   }
Parâmetros de entradas  RelBV,  RelAT
     {Relação entre a folga de banda representativa do caminho e a vazão desejada        }
     {pela  aplicação.                                                                    }
     {Relação entre o atraso máximo desejado pela aplicação e o atraso representativo do  }
     {caminho candidato.                                                                  }
Parâmetro de saída  Custo_cam
     {Custo do caminho.                                                                   }

  Se RelBV é baixa e RelAT é baixa
     então Custo_cam é  muito_ruim
  Se RelBV é baixa e RelAT é adequada
     então Custo_cam é ruim
  Se RelBV é baixa e RelAT é alta
     então Custo_cam é muito_ruim
  Se RelBV é adequada e RelAT é baixa
     então Custo_cam é ruim
  Se RelBV é adequada e RelAT  é adequada
     então Custo_cam é excelente
  Se RelBV é adequada e RelAT é alta
     então Custo_cam é bom
  Se RelBV é alta e RelAT é baixa
     então Custo_cam é ruim
  Se RelBV é alta e RelAT é adequada
     então Custo_cam é bom
  Se RelBV é alta e RelAT é alta
     então  Custo_cam é regular
Fim algoritmo
```

Figura 15 – Algoritmo para a construção da base de regras do SIF.

O SIF foi desenvolvido com a ferramenta JFS (MORTENSEN, 2004). Esta ferramenta oferece um ambiente para o desenvolvimento de protótipo de Sistemas de Inferência Nebulosa (especificação das funções de pertinência, regras de inferência e defuzificador), além de permitir verificação inicial do modelo especificado. Após o desenvolvimento do programa que implementa o SIF na linguagem do JFS, este é convertido para linguagem C++, transformando-se em uma biblioteca. Esta biblioteca é depois incorporada aos módulos de software que juntamente com o **ns2** possibilitam a simulação do sistema de TE no domínio MPLS. O código em JFS do SIF para efetuar o cálculo dos custos nebulosos dos caminhos candidatos é apresentado na Figura 16.

```
title   "Cálculo dos custos nebulosos dos caminhos candidatos para criação das LPS´s ";
  domains
     vazão "MBps" 0 5000;
     atraso "ms" 0 500;
     custo "%" 0 100;

  input
     RelBV "Folga de banda / vazão do caminho" vazão default 1;
     RelAT "Atraso desejado aplicação /Atraso disponibilizado caminho" atraso default 1;
  output
     Custo_cam "Custo nebuloso do caminho" custo;

  adjectives
     RelBV baixa 0:1 2:0;
     RelBV adequada 1:0 2:1 3:0;
     RelBV alta 2:0 10:1;
     RelAT baixa 0:1 2:0;
     RelAT adequada 1:0 2:1 3:0;
     RelAT alta 2:0 10:1;
     Custo_cam excelente 0:1 15:0;
     Custo_cam bom 5:0 15:1 25:0;
     Custo_cam regular 20:0 30:1 40:0;
     Custo_cam ruim 35:0 45:1 55:0;
     Custo_cam muito_ruim 45:0 100:1;

  program
    if RelBV is baixa and RelAT is baixa then Custo_cam is muito_ruim;
    if RelBV is baixa and RelAT is adequada then Custo_cam is ruim;
    if RelBV is baixa and RelAT is alta then Custo_cam is muito_ruim;
    if RelBV is adequada and RelAT is baixa then Custo_cam is ruim;
    if RelBV is adequada and RelAT is adequada then Custo_cam is excelente;
    if RelBV is adequada and RelAT is alta then Custo_cam is bom;
    if RelBV is alta and RelAT is baixa then Custo_cam is ruim;
    if RelBV is alta and RelAT is adequada then Custo_cam is bom;
    if RelBV is alta and RelAT is alta then Custo_cam is regular;
```

Figura 16 - Código em JFS para o cálculo dos custos nebulosos dos caminhos candidatos

4.3 Modelo Básico do Sistema de Engenharia de Tráfego

O sistema proposto é distribuído, ou seja, cada LSR de entrada do domínio MPLS contêm uma unidade de TE. Além disso, é admitida a existência do protocolo OSPF-TE que possibilita a troca de informações sobre o tráfego escoado nos enlaces do domínio

MPLS. Conforme é mostrado na Figura 17, cada unidade de TE é formada pelos seguintes componentes:

- Coletor das aplicações e requisitos de QoS (CARQ);
- Coletor de informações da rede (CIR);
- Coletor e processador de medições de tráfego (CPMT);
- Construtor de lista preliminar de caminhos mais curtos para as LSP´s (CLPC);
- Classificador dos caminhos em famílias (CCF);
- Identificador de necessidade de LSP´s (IANL);
- Sistema de Inferência Nebulosa (SIF);
- Módulo responsável pela escolha dos caminhos para as LSP´s (ECL)**;**
- Módulo responsável pela criação e ativação das LSP´s (CAL).

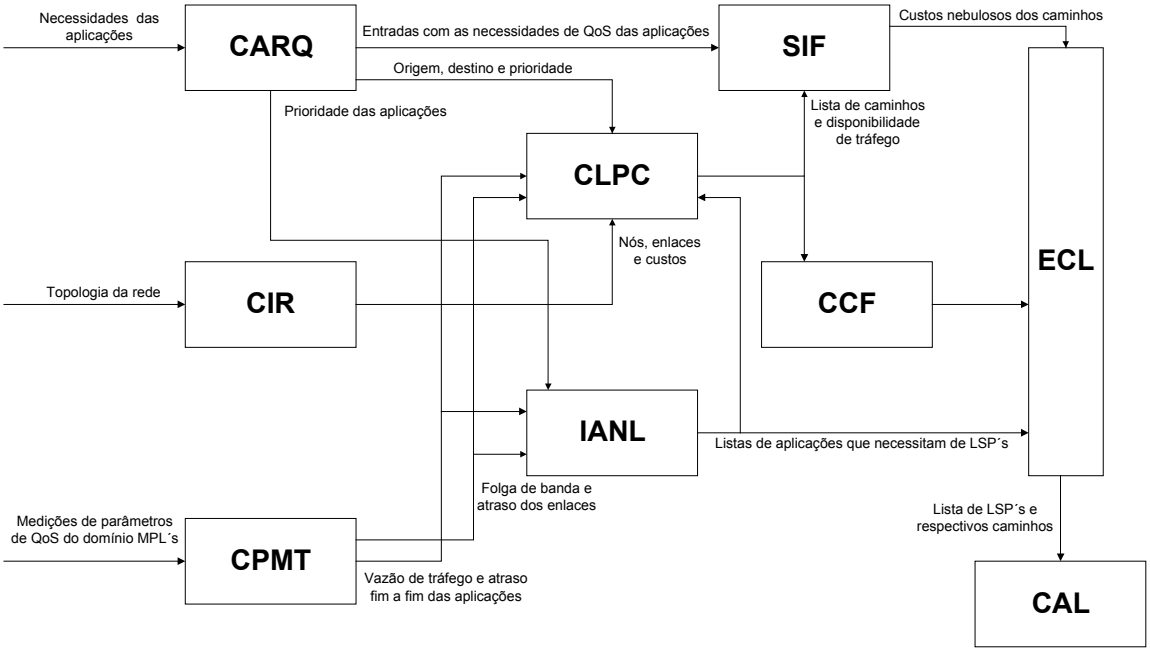

Figura 17 – Componentes do sistema de TE básico.

O **CARQ** cadastra informações sobre as aplicações tais como, origem, destino, prioridades e necessidades de vazão e atraso.

O **CIR** cadastra as informações sobre a arquitetura da rede incluindo a topologia, a largura de banda e o atraso mínimo em cada um dos enlaces.

O **CPMT** recolhe e processa as medições de vazão, atraso, jitter e perda em cada enlace, e também dos fluxos fim-a-fim das aplicações. O CPMT fornece os valores

representativos atuais e anteriores (histórico) dos parâmetros de QoS em cada enlace e também dos fluxos fim-a-fim das aplicações.

O **CLPC** produz uma lista de caminhos mais curtos em condições de estabelecer as LSP´s principais e reservas para as aplicações. O **CLPC** utiliza como base o algoritmo de Dijkstra. O algoritmo do **CLPC** é apresentado na Figura 18.

O **CCF** recebe como entrada a lista preliminar de caminhos mais curtos para as LSP´s. Em seguida ele identifica os caminhos que possuem nós intermediários comuns (excetuando os nós origem e destino). A saída produzida pelo módulo é uma lista de caminhos com a mesma família, e em condições de estabelecer as LSP`s.

O **IANL** recebe como entrada a folga de banda e o atraso dos enlaces, além da prioridade e medições referentes à vazão de tráfego, à perda de pacotes e ao atraso fim-a-fim das aplicações. Para cada aplicação que possui prioridade, o **IANL** verifica se as suas necessidades de vazão de tráfego e atraso fim-a-fim estão sendo atendidas. O módulo produz como saída uma lista de aplicações que necessitam de LSP´s. Para cada necessidade identificada, o **IANL** interage com o **CLPC** para verificar se existe pelo menos um caminho válido para a estabelecer a LSP. Se existir pelo menos um caminho válido então o **IANL** ativa os módulos **CCF**, **SIF** e **ECL**. O algoritmo do **IANL** é apresentado na Figura 19.

```
Algoritmo CLPC
{Objetivo: Construir uma lista preliminar de caminhos mais curtos para estabelecimento   }
{              das LSP´s para as aplicações                                                }
Parâmetros de entrada   maxtent, nf, ne, enl, ffaf
      { número de tentativas, numero de aplicações, numero de enlaces    }
      { vetor de registros contendo informações sobre os enlaces          }
      {    enl[ne].norig          nó origem do enlace                     }
      {    enl[ne].larg_banda     largura de banda                        }
      {    enl[ne].vazao_rep      valor de vazão representativa           }
      {    enl[ne].custo          custo                                   }
      {    enl[ne].atraso_min     atraso mínimo                           }
      { vetor de registros contendo informações sobre as aplicações       }
      {    ffaf[nf].norig  nó origem                                       }
      {    ffaf[nf].nodest  nó destino                                     }
      {    ffaf[nf].vazao_des          vazão desejada                      }
      {    ffaf[nf].atraso_maxdes      atraso máximo aceitável             }
Parâmetros de saída cam
      { vetor de registros com lista preliminar de possíveis caminhos mais curtos }
      { para cada aplicação                                                       }
 i ← 0 { Inicializa o contador de aplicações.                                     }
 enquanto (i < nf)  faça
      noo = ffaf[i].norig
      nod = ffaf[i].nodest
      se (noo <> nod) então
         j  ← 0
         enquanto (j < maxtent) faça
               obtem_conjunto_de_N_enlaces_do_caminho_mais_curto
               soma_atraso ← 0
               caminho_valido ← verdadeiro
               para e ← 0 até N -1 faça
                     folga ← enl[e].larg_banda - enl[e].vazao_rep
                     soma_atraso ← soma_atraso + enl[e].atraso_rep
                     se (folga  <  ffaf[i].vazao_des) então
                        caminho_valido ← falso
                     fim se
               fim para
               se (caminho_valido e (soma_atraso  < =  ffaf[i].atraso_maxdes)) então
                  armazena_caminho
               senão
                  descarta_caminho
               fim se
               j  ← j + 1
         fim enquanto
      fim se
      i ← i +1
 fim enquanto
fim algoritmo
```

Figura 18 – Algoritmo do construtor de lista de caminhos mais curtos para as LSP's.

```
Algoritmo IANL
{Objetivo: Identificar a necessidade de LSP´s para as aplicações                    }
Parâmetros de entrada    nf,  nm, ffaf,  pqos_flux
     {número de fluxos                                                              }
     {número de medições de tráfego                                                 }
     {vetor de registros contendo informações sobre as aplicações                  }
     {    ffaf[nf].norig  nó origem                                                 }
     {    ffaf[nf].nodest  nó destino                                               }
     {    ffaf[nf].vazao_des          vazão desejada                                }
     {    ffaf[nf].atraso_maxdes    atraso fim-a-fim máximo aceitável               }
     {    ffaf[nf].prioridade          prioridade                                   }
     {vetor de registros contendo informações medidas sobre as aplicações          }
     {    pqos_flux [nm].norig  nó origem                                           }
     {    pqos_flux [nm].nodest  nó destino                                         }
     {    pqos_flux [nm].vazao          vazão medida                                }
     {    pqos_flux [nm].atraso          atraso fim-a-fim medido                    }
     {    pqos_flux [nm].perda          perda medida                                }
Parâmetros de saída  nlsp, nativ_lsp
     {número necessário de LSP´s                                                    }
     {vetor contendo a lista de aplicações que necessitam de LSP´s                  }
     {    nativ_lsp[i].nfluxo          numero da aplicação que necessita LSP       }
     {    nativ_lsp[nf]. norig      nó origem                                       }
     {    nativ_lsp [nf].nodest    nó destino                                      }
 k ← 0  {Inicializa o contador de LSP´s a serem criadas.                            }
 i ← 0   {Inicializa o contador de aplicações.                                      }
 enquanto (i < nf)  faça
       j ← 0;
       nec_lsp ←0;
       enquanto (j < nm) faça
          se  ((((ffaf[i].vazao_des > pqos_flux[j].vazao) e (pqos_flux[j].perda > 0.0))
                ou    (ffaf[i].atraso_maxdes < pqos_flux[j].atraso)) e (ffaf[i].prioridade  > 0))
              então
                  nativ_lsp[k].nfluxo = ffaf[i].nfluxo
                  nativ_lsp[k].norig = ffaf[i].norig;
                  nativ_lsp[k].nodest = ffaf[i].nodest
                  k  ← k + 1
          fim se
          j  ← j + 1
       fim enquanto
      i ← i +1
 fim enquanto
 nlsp←k {Numero necessário de LSP´s recebe o valor do contador.                     }
  i ← 0   {Inicializa o contador de LSP´s.                                          }
 enquanto (i < nlsp) faça
    num_caminhos_validos ← 0 {Inicializa o contador de caminhos válidos.            }
    se  (aplicação não possui LSP)
        habilita CLPC
        se  (CLPC encontrou caminhos validos)
```

```
            armazena caminhos validos
            num_caminhos_validos ← num_caminhos_validos + 1
        fim se
    fim se
    se  (num_caminhos_validos > 0) então
        habilita CCF
        j ← 0
        enquanto ( j < num_caminho_validos) faça
              habilita SIF para cálculo e armazenamento dos custos nebulosos dos caminhos
        fim enquanto
        habilita ECL
        senão
              não é criada LSP para a aplicação
    fim se
    i ← i +1
 fim enquanto
fim algoritmo
```

Figura 19 – Algoritmo do identificador de necessidade de LSP´s.

O **SIF,** conforme já foi descrito, calcula os custos nebulosos dos caminhos que estão sendo avaliados. Os custos nebulosos calculados pelo **SIF** são então fornecidos ao **ECL**.

O **ECL** efetua a escolha dos caminhos mais adequados para o estabelecimento das LSP´s para as aplicações. Ele recebe informações sobre as famílias a que pertencem os caminhos avaliados, custos nebulosos e lista de aplicações que necessitam de LSP´s. O **ECL** fornece ao **CAL** a lista de LSP´s que devem ser criadas e os respectivos caminhos. O **CAL** é módulo responsável pela criação e ativação das LSP´s. A Figura 20 mostra o algoritmo do **ECL.**

```
Algoritmo ECL
{ Objetivo:  efetuar a escolha dos caminhos mais adequados para as LSP´s das aplicações.

Parâmetros de entrada   custos_fuzzy,  cam, ffaf,  nf, nlsp
     {vetor de registros contendo os custos nebulosos dos caminhos candidatos.          }
     {vetor de registros contendo famílias de caminhos.                                 }
     {vetor de registros com aplicações que necessitam de LSP´s e suas necessidades.    }
     {número de aplicações.                                                             }
     {número de LSP´s.                                                                  }
Parâmetros de saida   cam_lsp
     {vetor com  caminhos para as LSP´s.                                                }

  i ← 0 {Inicializa o contador de LSP´s a serem criadas.                                }
  enquanto (i < nlsp) faça
     j ← 0 {Inicializa o contador de aplicações.                                        }
     enquanto (j < nf) faça
          k ← 0 {Inicializa o contador de caminhos.                                     }
          enquanto (k < ncam) faça
               se  (ffaf[j].vazao_des <= cam[k].folga_banda)
                   Obtém custo nebuloso do caminho candidato
                   Armazena o custo nebuloso
               fim se
               k ← k + 1
          fim enquanto
          Escolha do caminho candidato com menor custo nebuloso.
          Atualização da folga de banda do caminho escolhido e nos de mesma família.
          ativa CAL para criação e ativação da LSP para a aplicação no caminho escolhido.
          j ← j + 1
     fim enquanto
     i ← i + 1
  fim enquanto
fim algoritmo
```

Figura 20 – Algoritmo para a escolha dos caminhos das LSP´s.

4.4 Engenharia de Tráfego com re-roteamento em caso de falhas em enlaces

O MPLS trouxe benefícios significativos para as redes com arquitetura IP pura, e também para aquelas que utiliza IP e ATM (LEMMA, 2003). Contudo, o MPLS é uma arquitetura orientada a conexão. Em caso de falhas, o MPLS primeiramente tem que estabelecer uma nova LSP e então encaminhar os pacotes através deste novo caminho. Por esta razão o MPLS tem uma baixa resposta de restauração em caso de falha em enlaces ou nós.

Nos anos recentes novos serviços e aplicações têm sido desenvolvidos com fortes características de orientação a conexão em tempo real. Como exemplo, pode-se citar voz sobre IP e aplicações multimídia. A falha em um enlace ou num roteador pode afetar severamente estes serviços. Assim, após o re-roteamento ser completado os serviços podem experimentar uma degradação da sua QoS, se a rota alternativa for mais longa ou estiver mais congestionada (LEMMA, 2003). Note que o tráfego é afetado não somente pela falha em si, mas também pela eventual degradação da rota alternativa para o qual foi desviado. Em função disso, a QoS pode experimentar uma inaceitável redução quando o tráfego das aplicações é re-encaminhado através da rota alternativa.

Por outro lado, a duração da interrupção devido a falha no enlace é na maioria dos casos bastante longa, para que aplicações de multimídia e aplicações de tempo real possam manter suas seções sem a transferência para um caminho alternativo. As aplicações de multimídia tem requisitos específicos de atraso, jitter, vazão e perda de pacotes. Desta forma, para atender as novas necessidades das aplicações, mecanismos de re-roteamento devem ser desenvolvidos para a escolha dos caminhos para as LSP alternativas que garantam QoS compatíveis com os oferecidos pela LSP principal. Assim, é proposta uma adaptação no sistema de TE básico para determinar também os caminhos das LSP´s reservas. No caso, o sistema de TE escolhe os caminhos das LSP´s principais e também das LSP´s reservas.

O sistema de TE modificado é formado pelos mesmos componentes já apresentados anteriormente na Figura 17. Entretanto, foram realizadas alterações nos módulos **IANL**, **CLPC** e **ECL**.

O **CLPC** passa a gerar a lista de caminhos mais curtos para as LSP´s principais e também para as reservas. O **IANL** passa a gerenciar o processo de ativação das LSP´s principais e também das reservas. O algoritmo modificado do **IANL** é apresentado na Figura 21.

O **ECL** foi também modificado de forma a fornecer ao **CAL** a lista de LSP´s principais e reservas que devem ser criadas e respectivos caminhos. O **ECL** escolhe o caminho com menor custo nebuloso para a LSP principal. O caminho com o segundo menor custo nebuloso é escolhido para a LSP reserva.

Algoritmo IANL modificado para gerenciar a ativação de LSP´s principais e reservas

```
{Objetivo: Identificar a necessidade de LSP´s para as aplicações                    }
Parâmetros de entrada    nf,  nm, ffaf,  pqos_flux
     {número de fluxos                                                              }
     {número de medições de tráfego                                                 }
     {vetor de registros contendo informações sobre as aplicações                  }
     {    ffaf[nf].norig  nó origem                                                 }
     {    ffaf[nf].nodest  nó destino                                               }
     {    ffaf[nf].vazao_des          vazão desejada                                }
     {    ffaf[nf].atraso_maxdes    atraso fim-a-fim máximo aceitável               }
     {    ffaf[nf].prioridade          prioridade                                   }
     {vetor de registros contendo informações medidas sobre as aplicações          }
     {    pqos_flux [nm].norig  nó origem                                           }
     {    pqos_flux [nm].nodest  nó destino                                         }
     {    pqos_flux [nm].vazao          vazão medida                                }
     {    pqos_flux [nm].atraso          atraso fim-a-fim medido                    }
     {    pqos_flux [nm].perda          perda medida                                }
Parâmetros de saída  nlsp, nativ_lsp
     {número necessário de LSP´s                                                    }
     {vetor contendo a lista de aplicações que necessitam de LSP´s                  }
     {    nativ_lsp[i].nfluxo           numero da aplicação que necessita LSP      }
     {    nativ_lsp[nf]. norig      nó origem                                       }
     {    nativ_lsp [nf].nodest    nó destino                                      }
 k ← 0 {Inicializa o contador de LSP´s a serem criadas.                             }
 i ← 0  {Inicializa o contador de aplicações.                                       }
 enquanto (i < nf)  faça
       j ← 0;
       enquanto (j < nm) faça
          se  ((((ffaf[i].vazao_des > pqos_flux[j].vazao) e (pqos_flux[j].perda > 0.0))
                ou    (ffaf[i].atraso_maxdes < pqos_flux[j].atraso)) e (ffaf[i].prioridade  > 0))
              então
                  nativ_lsp[k].nfluxo = ffaf[i].nfluxo
                  nativ_lsp[k].norig = ffaf[i].norig;
                  nativ_lsp[k].nodest = ffaf[i].nodest
                  k  ← k + 1
          fim se
          j  ← j + 1
       fim enquanto
      i ← i +1
 fim enquanto
 nlsp ← k {Numero necessário de LSP´s recebe o valor do contador.                   }
 i ← 0 {Inicializa o contador de LSP´s a serem criadas.                             }
 enquanto (i < nlsp)  faça
    num_caminhos_validos ← 0
    se  (aplicação não possui LSP) então
        habilita CLPC
        se  (CLPC encontrou caminhos validos) então
            armazena caminhos_validos
```

```
            num_caminhos_validos ← num_caminhos_validos + 1
        fim se
    fim se
    se  (num_caminhos_validos > 0) então
        habilita CCF
        j ← 0
        enquanto ( j < num_caminhos_validos) faça
              habilita SIF para cálculo e armazenamento dos custos nebulosos
        fim enquanto
        habilita ECL
        senão
              não é criada LSP para a aplicação
    fim se
    se  (houve estabelecimento de LSP principal para a aplicação) então
        num_caminhos_validos ← 0
        habilita CLPC
        analisa resposta do CLPC
        se  (CLPC encontrou caminhos validos) então
            armazena caminhos_validos
            num_caminhos_validos ← num_caminhos_validos + 1
        fim se
        se  (num_caminho_validos > 0) então
            habilita CCF
            j ← 0
            enquanto ( j < num_caminhos_validos) faça
                        habilita SIF para cálculo e armazenamento dos custos nebulosos
            fim enquanto
            habilita ECL
            senão
                  não é criada LSP reserva para a aplicação
        fim se
    fim se
    i ← i +1
 fim enquanto
fim algoritmo
```

Figura 21 - Algoritmo IANL modificado para gerenciar a ativação de LSP´s principais e reservas

4.5 Engenharia de Tráfego com re-roteamento do tráfego em caso de aumento da vazão das aplicações prioritárias

Apesar do MPLS viabilizar a implementação efetiva da TE em redes IP, não significa necessariamente que exista garantia de fornecimento de QoS fim-a-fim para as aplicações. Desde que o tráfego gerado pelas aplicações ultrapasse por algum motivo o

valor reservado, pode ocorrer congestionamento em determinados enlaces do domínio. Como conseqüência disso, a QoS oferecida às aplicações pode ser afetada. Isto pode ocorrer em razão do valor da largura de banda reservada para as LSP´s que é normalmente estimado, o que pode não corresponder as necessidades reais de vazão das aplicações que as utilizam. Além disso, podem surgir novas necessidades de vazão das aplicações não previstas à época da contratação dos serviços pelos clientes. Como ilustração do problema a ser resolvido, seja o domínio MPLS mostrado na Figura 22 onde foram estabelecidas duas LSP´s para o escoamento do tráfego originado nos nós 0 e 1, e destinados aos nós 13 e 14, respectivamente. As duas LSP´s foram estabelecidas originalmente através do mesmo caminho, ou seja, 8-11-12. As larguras de banda dos enlaces 8-11 e 11-12 são suficientes para atender os tráfegos das duas aplicações prioritárias, desde que as taxas estipuladas originalmente para cada uma delas sejam mantidas.

Admita-se agora que o tráfego originado pela aplicação do nó 1, comece a ultrapassar freqüentemente a taxa estipulada originalmente em contrato. Neste caso, provavelmente a QoS oferecida a esta aplicação ficará prejudicada. Desta forma, o que deve ser feito é a tentativa de estabelecimento de uma nova LSP, através de um novo caminho, por exemplo, 8-9-10-12, que consiga escoar o novo tráfego gerado pela aplicação do nó 1. Caso esta tentativa tenha sucesso, pode-se providenciar o re-roteamento do tráfego através da nova LSP. Desta forma, o que está sendo proposto é a implementação de modificações no sistema de TE, para tentar manter a QoS das aplicações mesmo que elas ultrapassem as suas taxas. O sistema de TE modificado é mostrado na Figura 23.

No caso, é feita a inclusão de um novo modulo que será o responsável pela detecção do aumento da vazão de tráfego das aplicações prioritárias. Este módulo ficará responsável pela entrega das novas necessidades das aplicações para o SIF, CLPC e IANL. O algoritmo do **DAV** (Detector do aumento da vazão das aplicações) é apresentado na Figura 24.

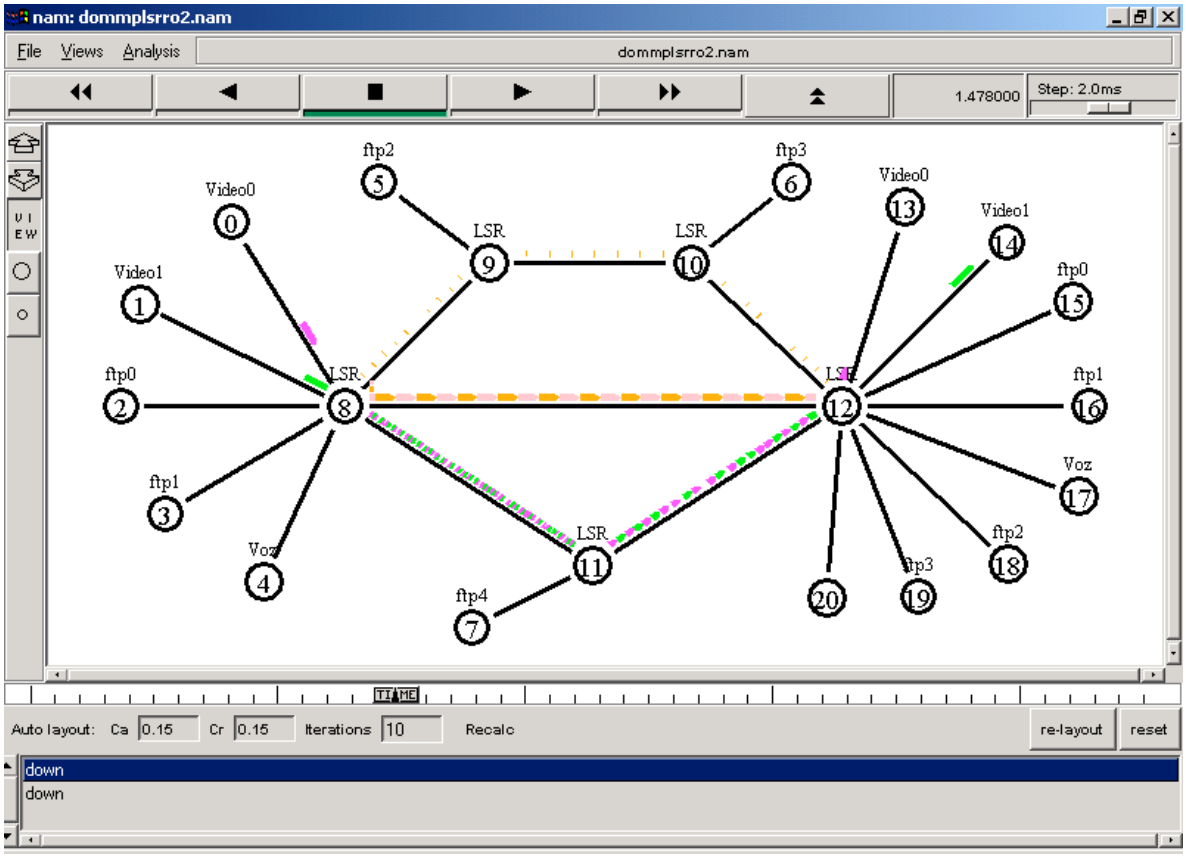

Figura 22 – Um cenário para ilustrar o problema do aumento da vazão de tráfego das aplicações prioritárias.

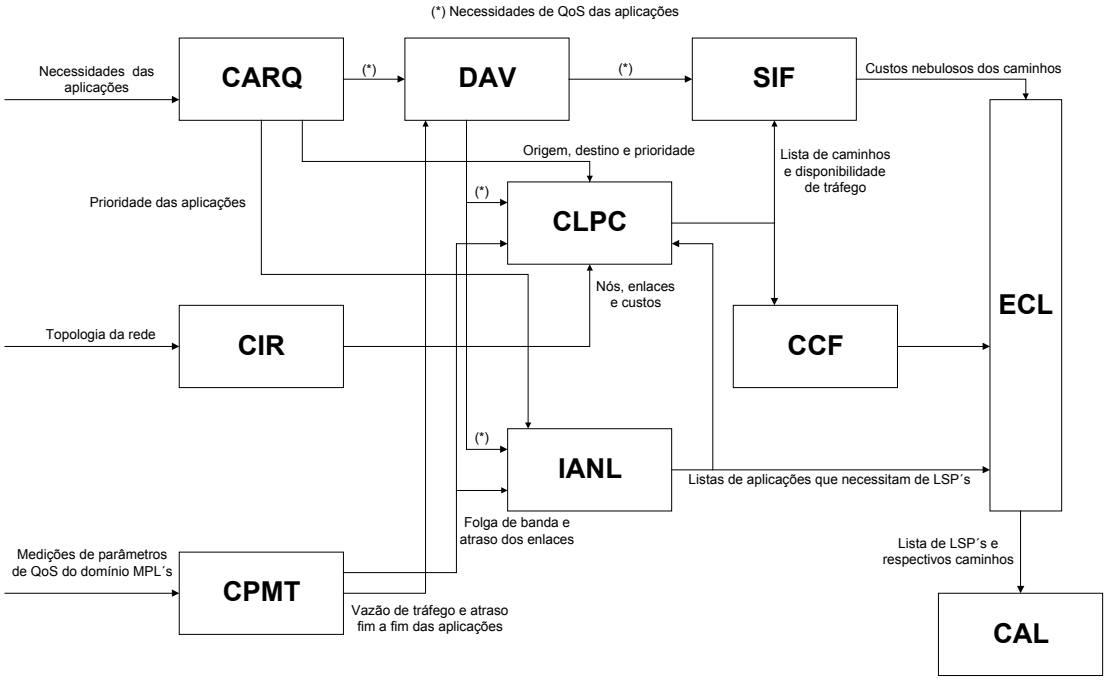

Figura 23 – Sistema de TE com alterações visando a detecção da vazão de tráfego das aplicações prioritárias e re-roteamento de LSP´s.

```
Algoritmo DAV
{Objetivo: Recolher as novas necessidades de tráfego das aplicações                }
Parâmetros de entrada nf, ne, enl, ffaf,  tx_aum_vaz_aval
      {numero de aplicações (fluxos), numero de enlaces                            }
      { vetor de registros contendo informações sobre os enlaces                   }
      {    enl[ne].norig  nó origem do enlace                                      }
      {    enl[ne].vazao_rep valor de vazão representativa                         }
      { vetor de registros contendo informações sobre as aplicações               }
      {    ffaf[nf].norig  nó origem                                               }
      {    ffaf[nf].nodest  nó destino                                             }
      {    ffaf[nf].vazao_des  valor de vazão desejada representativa              }
      {    ffaf[nf].num_ofensas  numero de ofensas                                 }
      {    ffaf[nf].numed          número de medições                              }
      {    ffaf[nf].perc_ofensas  percentual de ofensas                            }
      { taxa de aumento passível de avalização                                     }
Parâmetros de saída ffaf_aut, ffaf
      { vetor de registros com informações sobre as novas necessidades das aplicações }
      {    ffaf_aut[nf].norig  nó origem                                           }
      {    ffaf_aut[nf].nodest  nó destino                                         }
      {    ffaf_aut[nf]. aumento_vazao_des    valor do aumento de vazão desejada   }
      {    ffaf_aut[nf]. autor_aumento_vazao  analise preliminar do aumento de vazão }
      {                    3:   avaliar aumento,          0:   negar  aumento      }
 i ← 0 {Inicializa o contador de aplicações.                                       }
 enquanto (i < nf)  faça
      se (ffaf[i].prioridade <> 0) então
         j  ← 0
         enquanto (j < ne) faça
               se (ffaf[i].norig == enl[j].norig) então
                  se (enl[j].vazao_rep > ffaf[i].vazao_des) então
                     aum_vaz_des ← enl[j].vazao_rep - ffaf[i].vazao_des
                     tx_aum_vaz_des ← aum_vaz_des / ffaf[i].vazao_des
                     se (tx_aum_vaz_des < tx_aum_vaz_aval) então
                        ffaf[i].vazao_des ← enl[j].vazao_rep
                        ffaf_aut[i].aumento_vazao_des ← aum_vaz_des
                        ffaf_aut[i].autor_aumento_vazao ← 3
                     senão
                        ffaf_aut[i].autor_aumento_vazao ← 0
                     fim se
                  fim se
               fim se
               j++
         fim enquanto
        fim se
        i++
 fim enquanto
fim algoritmo
```

Figura 24 – Algoritmo do detector de aumento de vazão das aplicações.

Baseando-se nas informações fornecidas pelo **DAV**, o **IANL** inicia o processo de busca de um novo caminho, que em caso de sucesso possibilitará o sistema de TE estabelecer uma nova LSP para re-encaminhamento do tráfego da aplicação ofensora. Os demais módulos que fazem parte do sistema de TE permanecem inalterados.

4.6 Engenharia de Tráfego com Controle de Admissão de Conexão utilizando reconhecimento e classificação dos perfis de comportamento de tráfego das aplicações

O CAC é um instrumento que decide se uma chamada de entrada pode ser aceita ou rejeitada. Desde que a rede esteja sobrecarregada, ou seja, o tráfego gerado pelas aplicações entrantes ultrapassa a sua capacidade de escoamento, pode ocorrer congestionamento. Desta forma, o sistema de TE deve controlar a admissão do tráfego das aplicações no domínio MPLS. Isto significa que se não for possível o estabelecimento de uma LSP que atenda as necessidades de QoS de uma aplicação, o tráfego desta deve ser bloqueado. Isto muda um pouco a forma de tratamento do trafego das aplicações. Até agora neste trabalho, as LSP´s eram estabelecidas apenas para as aplicações prioritárias. O trafego das aplicações que não conseguiram LSP´S, era encaminhado pelo menor caminho. O problema desta situação é que as aplicações prioritárias que utilizam LSP´s estabelecidas através do menor caminho podem acabar sendo prejudicadas. O menor caminho poderia ser disputado por aplicações que conseguiram e também pelas que não conseguiram LSP´s. Para evitar isso, a partir de agora todas as aplicações passam a ser consideradas prioritárias, mas com diferentes níveis de prioridades. Assim, tenta-se o estabelecimento de LSP´s para todas as aplicações, levando-se em conta a prioridade de atendimento de cada um delas. As aplicações que não conseguiram LSP´s passam a ter o seu tráfego bloqueado.

Entretanto, deve-se observar que a simples introdução do CAC não resolve todos os problemas. Conforme já foi tratado no item 4.5, se o tráfego gerado pelas aplicações ultrapassar o valor estipulado durante a criação da LSP, a QoS poderá ser afetada. Neste caso, deve ser feita uma tentativa de estabelecimento de uma nova LSP que consiga escoar o novo tráfego gerado pela aplicação. Em caso de sucesso, pode-se providenciar o re-roteamento do tráfego através da nova LSP. Em caso de insucesso, o CAC deverá desautorizar total ou parcialmente o aumento da vazão da aplicação. Portanto, para a implementação da modificação

proposta, o sistema de TE deve ser alterado. Assim, são retiradas algumas das atribuições do IANL e transferidas para o CAC que está sendo criado. A nova configuração do sistema de TE é mostrada na Figura 25.

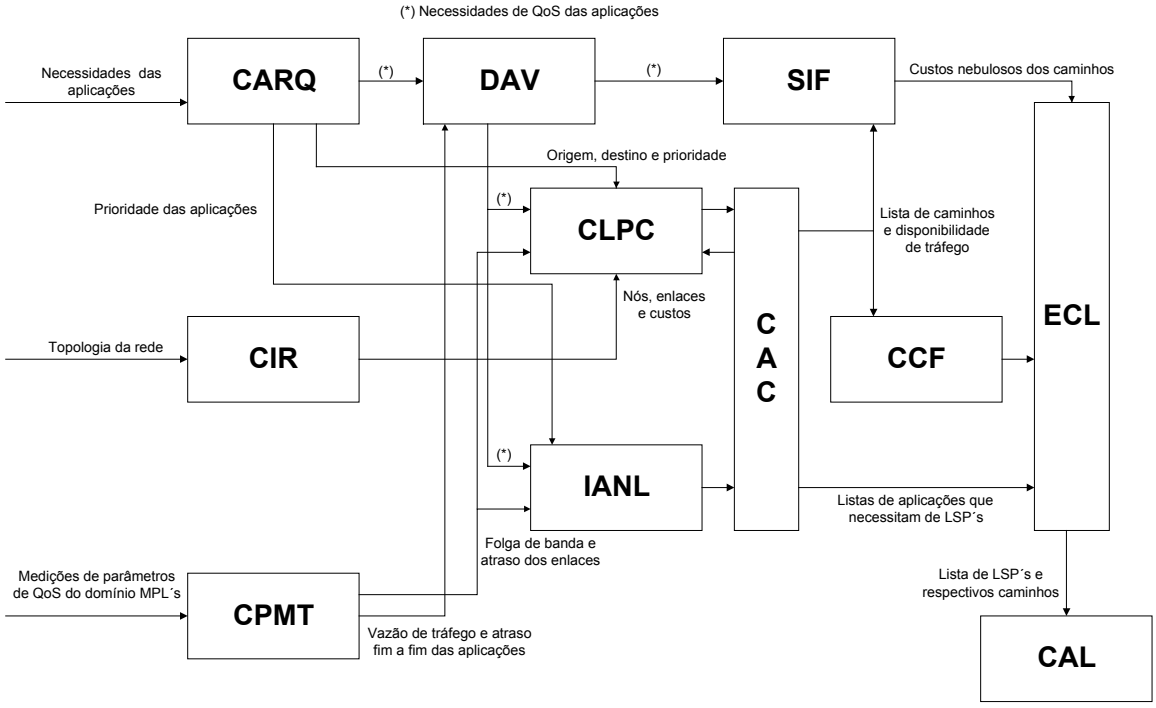

Figura 25 – Sistema de TE com controle de admissão de conexão.

Os algoritmos do **IANL** modificado para inclusão do **CAC** e do próprio **CAC** são apresentados nas Figuras 26 e 27, respectivamente.

Uma outra proposta apresentada neste item é a modificação na forma de atendimento das solicitações das aplicações. Assim, as aplicações "comportadas", ou seja, que mantêm as suas taxas de tráfego conforme os valores acordados, devem ser atendidas com uma maior prioridade. Por outro lado, as aplicações que não cumprem o acordo são atendidas, mas devem ser "penalizadas". Portanto, é proposto um algoritmo para reconhecimento e classificação dos perfis de comportamento de tráfego das aplicações (**RCPT**). A idéia é alterar dinamicamente a prioridade de admissão das aplicações no domínio MPLS levando em conta a análise dos seus comportamentos registrados em um arquivo de histórico. O valor da prioridade de atendimento de cada aplicação é atualizada periodicamente de acordo com a equação (10).

```
Algoritmo IANL modificado para inclusão do CAC
{Objetivo: Identificar a necessidade de LSP´s para as aplicações                    }
Parâmetros de entrada    nf,  nm, ffaf,  pqos_flux
      {número de fluxos                                                              }
      {número de medições de tráfego                                                 }
      {vetor de registros contendo informações sobre as aplicações                  }
      {    ffaf[nf].norig  nó origem                                                 }
      {    ffaf[nf].nodest  nó destino                                               }
      {    ffaf[nf].vazao_des          vazão desejada                                }
      {    ffaf[nf].atraso_maxdes    atraso fim-a-fim máximo aceitável               }
      {    ffaf[nf].prioridade          prioridade                                   }
      {vetor de registros contendo informações medidas sobre as aplicações          }
      {    pqos_flux [nm].norig  nó origem                                           }
      {    pqos_flux [nm].nodest  nó destino                                         }
      {    pqos_flux [nm].vazao          vazão medida                                }
      {    pqos_flux [nm].atraso         atraso fim-a-fim medido                     }
      {    pqos_flux [nm].perda         perda medida                                 }
Parâmetros de saída  nlsp, nativ_lsp
      {número necessário de LSP´s                                                    }
      {vetor contendo a lista de aplicações que necessitam de LSP´s.                 }
      {    nativ_lsp[i].nfluxo          numero da aplicação que necessita LSP        }
      {    nativ_lsp[nf]. norig      nó origem                                       }
      {    nativ_lsp [nf].nodest    nó destino                                       }
  k ← 0 {Inicializa o contador de LSP´s a serem criadas.                             }
  i ← 0  {Inicializa o contador de aplicações.                                       }
  enquanto (i < nf)
        j ← 0;  {Inicializa o contador de medições de tráfego.                       }
        enquanto (j < nm)  faça
            se  ((((ffaf[i].vazao_des > pqos_flux[j].vazao) e (pqos_flux[j].perda > 0.0))
                  ou (ffaf[i].atraso_maxdes < pqos_flux[j].atraso)) e (ffaf[i].prioridade  > 0))
                então
                    nativ_lsp[k].nfluxo = ffaf[i].nfluxo
                    nativ_lsp[k].norig = ffaf[i].norig;
                    nativ_lsp[k].nodest = ffaf[i].nodest
                    k  ← k + 1
            fim se
            j  ← j + 1
        fim enquanto
        i ← i +1
  fim enquanto
  nlsp ← k
fim algoritmo
```

Figura 26 – Algoritmo do IANL modificado para inclusão do CAC.

Algoritmo CAC

```
{Objetivo: autorizar ou não a criação de LSP´s para as aplicações.                    }
{          autorizar ou não a transmissão de pacotes pelas aplicações.                }
Parâmetros de entrada    nf,  ffaf,  nlsp, nativ_lsp
     {número de fluxos                                                                 }
     {número de medições de tráfego                                                    }
     {vetor de registros contendo informações sobre as aplicações                     }
     {    ffaf[nf].norig  nó origem                                                    }
     {    ffaf[nf].nodest  nó destino                                                  }
     {    ffaf[nf].vazao_des          vazão desejada                                   }
     {    ffaf[nf].atraso_maxdes    atraso fim-a-fim máximo aceitável                  }
     {    ffaf[nf].prioridade          prioridade                                     }
     {número necessário de LSP´s a serem criadas                                       }
     {vetor contendo a lista de aplicações que necessitam de LSP´s                     }
     {    nativ_lsp[i].nfluxo          numero da aplicação que necessita LSP          }
     {    nativ_lsp[nf]. norig      nó origem                                          }
     {    nativ_lsp [nf].nodest    nó destino                                         }
Parâmetros de saída  nf, ffaf_aut
     {vetor de registros contendo informações sobre as autorizações de  aplicações    }
     {    ffaf_aut[nf].nfluxo  numero do fluxo                                         }
     {    ffaf_aut[nf].norig  nó origem                                                }
     {    ffaf_aut[nf].nodest  nó destino                                              }
     {    ffaf_aut[nf]. autor_tx       autorização de transmissão de pacotes           }
     {    ffaf_aut[nf]. autor_aumento_vazao        autorização de aumento de vazão.    }
     {    ffaf_aut[nf]. aumento_vazao_des       aumento de vazão desejado.           }
     {    ffaf_aut[nf]. aumento_vazao_pos      aumento de vazão possivel.            }
 i ← 0 { Inicializa contador de LSP´s a serem criadas.                                 }
 enquanto (i < nlsp) faça
     num_caminhos_validos ← 0
     num_caminhos_parcialme_validos ← 0
     se  ((aplicação aumentou vazão e possui LSP) ou (aplicação não possui LSP)) então
         habilita CLPC
         se  (CLPC encontrou caminhos validos) então
             armazena caminhos_validos
             num_caminhos_validos ← num_caminhos_validos + 1
         fim se
         se  (CLPC encontrou caminhos parcialmente validos) então
            se (aplicação aumentou vazão e possui LSP) então
                armazena caminhos_parcialmente_validos
                num_caminhos_parcialme_validos ← num_caminhos_parcialme_validos + 1
           fim se
         fim se
     fim se
     se  (num_caminhos_validos > 0) então
         habilita CCF
         j ← 0
         enquanto ( j < num_caminhos_validos) faça
               habilita SIF para cálculo e armazenamento dos custos nebulosos
```

```
        fim enquanto
         habilita ECL
         senão
              se  (aplicação aumentou vazão e possui LSP) então
                  se  (num_caminhos_parcialme_validos > 0) então
                      habilita CCF
                      j ← 0
                     enquanto (j < num_caminhos_parcialme_validos) faça
                                habilita SIF para cálculo e armazenamento dos custos nebulosos
                     fim enquanto
                     habilita ECL
                     autoriza aumento parcial de vazão
                     senão
                          desautoriza aumento vazão
                  fim se
                  senão
                        não é criada LSP para a aplicação
                        bloqueia transmissão de pacotes da aplicação
              fim se
    fim se
    se  (houve estabelecimento de LSP principal para a aplicação) então
        num_caminhos_validos ← 0
        habilita CLPC
        analisa resposta do CLPC
        se  (CLPC encontrou caminhos validos) então
            armazena caminhos_validos
            num_caminhos_validos ← num_caminhos_validos + 1
        fim se
        se  (num_caminhos_validos > 0) então
            habilita CCF
            j ← 0
           enquanto ( j < num_caminhos_validos) faça
                      habilita SIF para cálculo e armazenamento dos custos nebulosos
           fim enquanto
           habilita ECL
           senão
                não é criada LSP reserva para a aplicação
        fim se
    fim se
    i ← i +1
 fim enquanto
fim algoritmo
```

Figura 27 – Algoritmo do Controle de Admissão de Conexão.

$$Pat = Pant - PercO \tag{10}$$

Onde:

Pat = Prioridade atual da aplicação;

$Pant$ = Prioridade anterior da aplicação;

$PercO$ = Percentual de ofensas da aplicação.

O percentual de ofensas da aplicação ($PercO$) é calculado através da equação (11).

$$PercO = 100\left(\frac{NumOfensas}{Numed}\right) \tag{11}$$

onde:

$NumOfensas$ = Número de vezes que a vazão da aplicação ultrapassou o valor autorizado durante o período de medição de tráfego.

$Numed$ = Número de medições efetuadas durante o período de observação.

Desta forma, o sistema de TE é novamente modificado para a inclusão do **RCPT**. A nova configuração do sistema de TE é mostrada na Figura 28. O algoritmo do **RCPT** é apresentado na Figura 29.

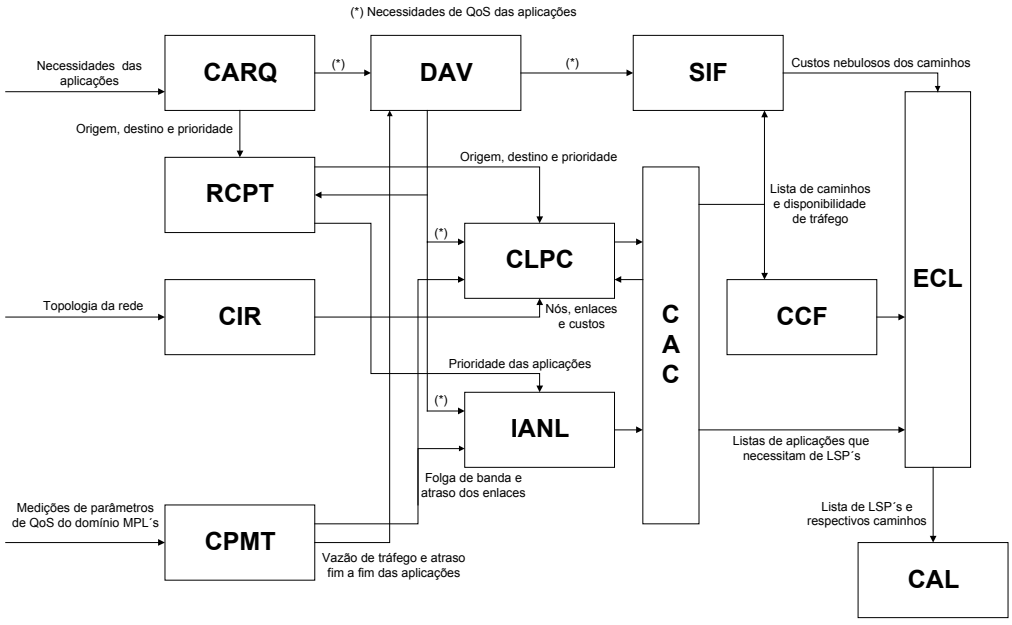

Figura 28 – Sistema de TE com controle de admissão de conexão e reconhecimento e classificação dos perfis de comportamento de tráfego das aplicações.

```
Algoritmo RCPT
{Objetivo:Alterar a prioridade e ordenar as aplicações.                                    }
 Parâmetros de entrada    nf,  ffaf,  ne, enl
      {número de fluxos                                                                    }
      {vetor de registros contendo informações sobre as aplicações                        }
      {    ffaf[nf].norig  nó origem                                                       }
      {    ffaf[nf].vazao_des          vazão desejada                                      }
      {    ffaf[nf].vazao_med          vazão medida                                        }
      {    ffaf[nf].num_ofensas        número de vezes que a aplicação ultrapassou vazão_des}
      {    ffaf[nf].numed              número de medições no período analisado.            }
      {    ffaf[nf].perc_ofensas       percentual  de ofensas.                             }
      {    ffaf[nf].prioridade         prioridade                                          }
      {número de enlaces                                                                   }
      {vetor contendo medições de tráfego nos enlaces do domínio MPLS.                     }
      {    enl[ne].numenlace           numero do  enlace.                                  }
      {    enl[ne].norig               nó origem                                           }
      {    enl[ne].vazao_rep           vazão representativa do enlace.                     }
Parâmetros de saída  nf, ffaf
      {vetor de registros das aplicações ordenado de acordo com a prioridade.              }
  i ← 0
  enquanto (i < nf) faça
          j ← 0
          enquanto (j < ne) faça
                  se  (enl[j].norig = ffaf[i].norig) então
                      ffaf[i].vazao_med ←  enl[j].vazao_rep
                      se  (ffaf[i].vazao_med > ffaf[i].vazao_des) então
                          ffaf[i].num_ofensas ←  ffaf[i].num_ofensas + 1
                      fim se
                      ffaf[i].numed ←  ffaf[i].numed + 1
                      se  (ffaf[i].num_ofensas > 0) então
                          ffaf[i].perc_ofensas ← (ffaf[i].num_ofensas) / ffaf[i].numed) * 100
                      senão
                          ffaf[i].perc_ofensas ← 0.0
                      fim se
                  fim se
                  j ← j + 1
         fim enquanto
        i ← i + 1
  fim enquanto
  i ← 0
  enquanto (i < nf) faça
          ffaf[i].prioridade ← (ffaf[i].prioridade - ffaf[i].perc_ofensas)
          i ← i + 1
  fim enquanto
  ordenação_das_aplicações_segundo_a_prioridade
fim algoritmo
```

Figura 29 – Algoritmo de reconhecimento e classificação dos perfis de comportamento de tráfego das aplicações.

4.7 Engenharia de Tráfego com Algoritmo de Otimização dos recursos da rede

Apesar do sistema de TE proposto viabilizar o fornecimento de QoS fim-a-fim para as aplicações no domínio MPLS, não significa necessariamente que o uso dos recursos da rede esteja sempre otimizado. Isto acontece porque um caminho que num determinado momento é escolhido como a melhor opção para o estabelecimento de uma LSP, pode deixar de sê-lo dependendo das novas condições da rede.

Como ilustração do problema, seja o domínio MPLS mostrado na Figura 30, onde a primeira LSP é criada para encaminhar o tráfego das aplicações de dados (http0/ftp0 e ftp1) originado no nó 2 e destinado ao nó 15. Naturalmente, esta primeira LSP foi estabelecida através do menor caminho livre, ou seja, 8-12. Mais tarde, surgiu a necessidade do estabelecimento de três novas LSP´s para as aplicações de vídeo (Vídeo0 e Vídeo1) e voz (Voz0) originadas nos nós 0, 1 e 4, e destinadas aos nós 13, 14 e 17, respectivamente. Naquele momento, o sistema de TE definiu como melhor opção a criação de duas LSP´s para as aplicações de vídeo através do caminho 8-9-10-12 e uma terceira LSP através do caminho 8-11-12 para a aplicação de voz. A Figura 30, mostra o efeito da criação das LSP´s.

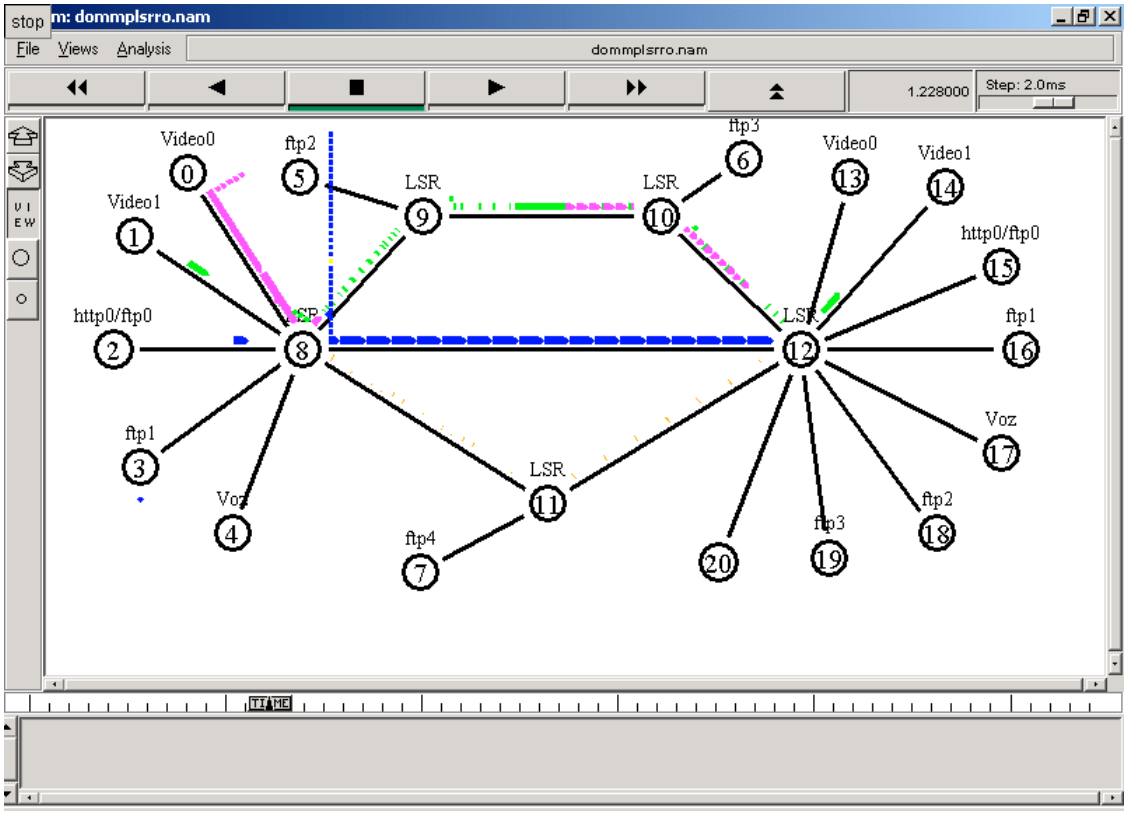

Figura 30 – Um cenário para ilustrar o problema da otimização de recursos da rede.

Admita-se agora que o tráfego originado no nó 2 pela aplicação de dados http0 foi encerrado. Como é mostrado na Figura 31, o menor caminho, ou seja, o enlace 8-12 se tornou ocioso e em condições de encaminhar por exemplo, o tráfego da aplicação de voz.

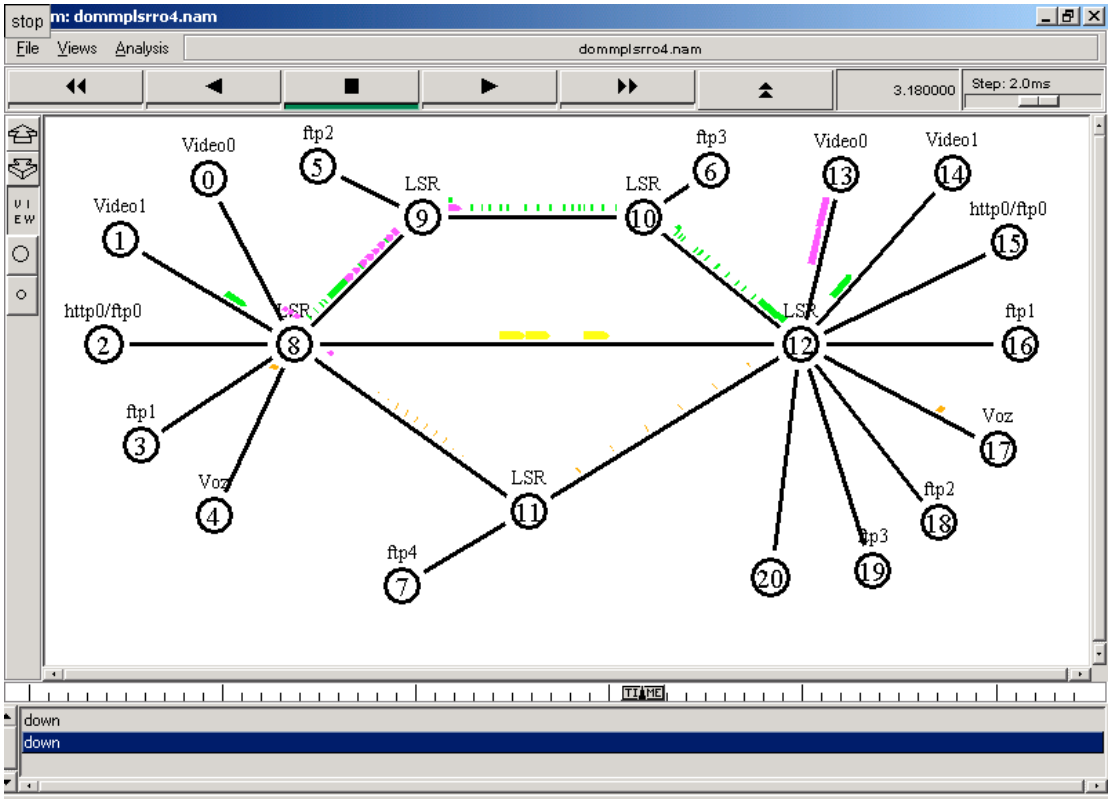

Figura 31 - Efeito do encerramento da transmissão de pacotes originados pelas aplicações de dados http0 no enlace 8-12.

Por outro lado, apesar de não otimizado, o caminho 8-11-12 está atendendo as necessidades atuais de QoS da aplicação de voz. Em razão disso, o sistema de TE não realiza neste caso o re-roteamento. Ocorre que se situações deste tipo tornarem-se comuns, a performance do sistema como um todo poderá ser afetada em função do uso não otimizado dos recursos da rede. Assim, é proposta a inclusão de um módulo de otimização no sistema de TE.

Os algoritmos de otimização podem ser classificados em duas classes, ou seja, os modelos matemáticos (procura a solução ótima) e os heurísticos (procura por aproximação).

Normalmente o tempo de resposta para os modelos matemáticos eleva-se consideravelmente, tornando impraticável a obtenção de soluções ótimas em tempos

satisfatórios. Em razão disso, algoritmos matemáticos são computacionalmente viáveis quando aplicados a problemas reais pequenos.

Para problemas de porte similar ao tratado neste trabalho, costuma-se sacrificar a obtenção de uma solução ótima por métodos heurísticos, que resultem em uma boa solução, com tempo computacionalmente aceitável. Neste trabalho adotou-se como heurística os Algoritmos Genéticos.

Desta forma, o módulo de otimização utilizando algoritmos genéticos (OAG) deverá periodicamente efetuar a otimização dos recursos da rede. Por outro lado, o sistema de TE com módulo de otimização tem um funcionamento diferente do sistema de TE normal. Para que a otimização possa ser realizada é necessário que o OAG tenha um conhecimento prévio de quais são as aplicações que estão utilizando o domínio MPLS naquele momento e suas respectivas LSP´s. O mesmo não ocorre no sistema de TE normal em que as demandas de conexões das aplicações chegam normalmente uma após a outra, sendo que as LSP´s são criadas da mesma forma. Quando duas ou mais demandas chegam ao mesmo tempo, elas são atendidas em função de suas prioridades, conforme já foi apresentado anteriormente. Durante o atendimento de uma determinada solicitação de LSP, é verificada a situação da rede e feita seleção dos caminhos mais curtos que estão em condições de atender a necessidade de QoS da aplicação. O caminho escolhido é aquele que possui o menor valor de custo nebuloso. No caso do sistema de TE com otimização o objetivo não é a obtenção de um conjunto de ótimas soluções individuais, mas a melhor solução possível para o grupo de aplicações. O sistema de TE modificado com a inclusão do módulo de otimização dos recursos da rede é apresentado na Figura 32.

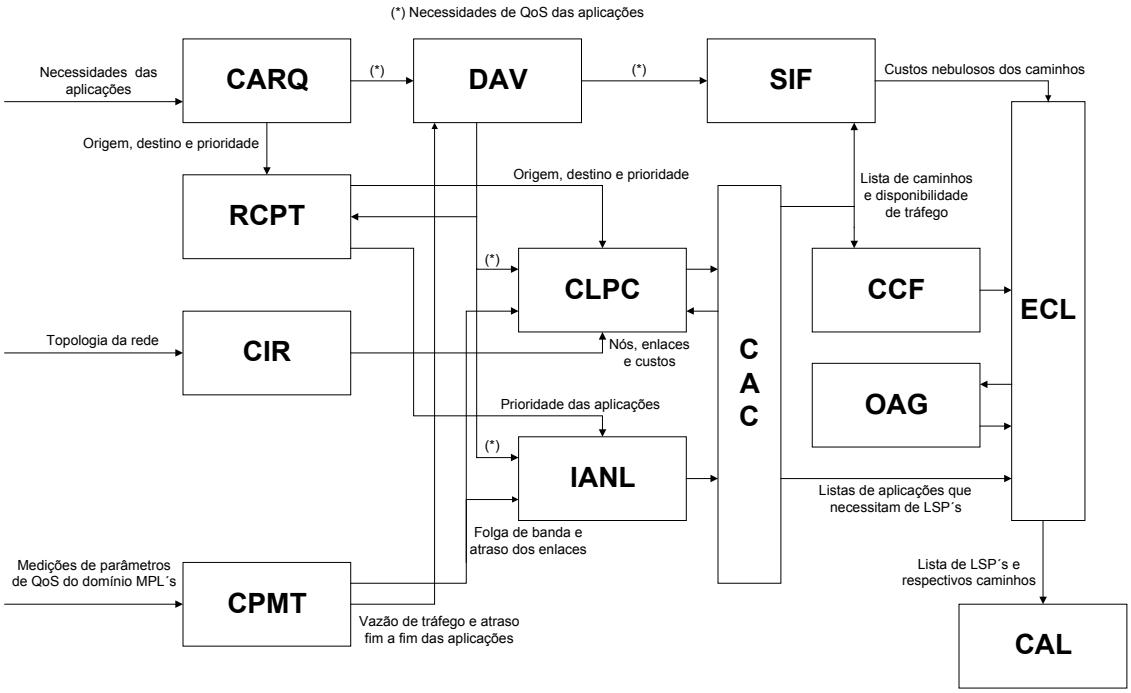

Figura 32 – Sistema de TE modificado com a inclusão do módulo de otimização dos recursos da rede.

Para a implementação da função de otimização no sistema de TE , além da inclusão do **OAG**, houve necessidade também de efetuar modificações no **ECL**. O novo **ECL** é capaz de efetuar a escolha dos caminhos para as LSP´s nas duas situações, ou seja, TE com e sem otimização. O algoritmo modificado do **ECL** é apresentado na Figura 33.

Algoritmo ECL modificado

```
{ Objetivo:  efetuar a escolha dos caminhos mais adequados para as LSP´s das aplicações.}

Parâmetros de entrada   custos_nebulosos, ncam,  cam, ffaf,  nf, nlsp, nsol, otim
     {vetor de registros contendo os custos nebulosos dos caminhos candidatos.          }
     {número de caminhos candidatos.                                                    }
     {vetor de registros contendo famílias de caminhos.                                 }
     {vetor de registros com aplicações que necessitam de LSP´s e suas necessidades.    }
     {número de aplicações.                                                             }
     {número de LSP´s.                                                                  }
     {número de soluções da tabela de soluções.                                         }
     {variável com indicação do momento de ativação da otimização.                      }
     {                  verdadeiro:   ativa otimização.                                 }
     {                  falso:        não ativa otimização.                             }
Parâmetros de saída   nsol, tabsol, crom, Cam_LSP
     {numero de soluções da tabela de soluções informados ao OAG.                       }
     {tabela com soluções válidas para o conjunto de LSP´s informadas ao OAG.           }
     {soluções codificadas informadas ao OAG.                                           }
     {Caminhos escolhidos para a criação das LSP´s informados ao CAL.                   }
  se  (otim) então  {montagem da tabela de soluções e codificação genética.            }
       sx ← 0 {Inicializa o contador de soluções.                                       }
       enquanto (sx < nsol) faça
            i ← 0  {Inicializa o contador de LSP´s  a serem criadas.                    }
           enquanto (i < nlsp) faça
                j ← 0  {Inicializa o contador de aplicações.                            }
               enquanto (j < nf) faça
                    k ← 0 {Inicializa o contador de caminhos candidatos.                }
                    enquanto (k < ncam) faça
                         se  (ffaf[j].vazao_des <= cam[k].folga_banda)
                              ativa SIF para obtenção do custo nebuloso do caminho candidato.
                         fim se
                         k ← k + 1
                    fim enquanto
                    escolha aleatória de um dos caminhos candidatos com menor custo
                       nebuloso.
                    efetua a codificação genética
                    {associação do caminho escolhido para a aplicação j à solução corrente sx}
                    atualização da folga de banda do caminho escolhido e nos caminhos de
                       mesma família.
                    j ← j + 1
               fim enquanto
               i ← i + 1
           fim enquanto
           sx ← sx + 1
       fim enquanto
       ativa OAG
       lê arquivo de sincronização com OAG
       enquanto (sincronização não finalizada) faça
```

```
        lê arquivo de sincronização com OAG
    fim enquanto
    lê melhor solução gerado pelo OAG
    ativa CAL para criação e ativação das LSP´s através dos caminhos da melhor solução.
  senão {Efetua a escolha normal dos caminhos para as aplicações sem Otimização}
  i ← 0  {Inicializa o contador de LSP´s  a serem criadas.                    }
  enquanto (i < nlsp) faça
    j ← 0  {Inicializa o contador de aplicações.                              }
    enquanto (j < nf) faça
        k ← 0 {Inicializa o contador de caminhos candidatos.                  }
        enquanto (k < ncam) faça
            se  (ffaf[j].vazao_des <= cam[k].folga_banda) então
                ativa SIF para obtenção do custo nebuloso do caminho candidato
            fim se
            k ← k + 1
        fim enquanto
        escolha do caminho candidato com menor custo nebuloso para LSP principal.
        atualização da folga de banda do caminho escolhido e nos de mesma família.
        ativa CAL para criação e ativação da LSP para a aplicação no caminho escolhido.
        escolha do caminho candidato com segundo menor custo nebuloso para LSP
            reserva.
        atualização da folga de banda do caminho escolhido e nos de mesma família.
        ativa CAL para criação e ativação da LSP para a aplicação no caminho escolhido.
        j ← j + 1
    fim enquanto
    i ← i + 1
  fim enquanto
 fim se
fim algoritmo
```

Figura 33 - Algoritmo do ECL modificado para a otimização dos recursos da rede

O **ECL** realiza duas importantes atividades, a preparação da tabela de soluções válidas e a codificação genética. Estas informações são depois utilizadas pelo **OAG**.

Durante a montagem da tabela de soluções válidas, os caminhos em condições de estabelecer LSP´s para cada uma das aplicações, são numerados. Como exemplo, o Quadro 3 mostra os caminhos válidos para cada uma das aplicações que alimentam o domínio MPLS da Figura 31. São apresentados também no Quadro 3 as vazão vazões desejadas pelas aplicações e os custos nebulosos dos caminhos candidatos. Os custos nebulosos apresentados são hipotéticos.

A outra atividade executada pelo **ECL** é a codificação genética, ou seja, a escolha aleatória de um caminho para cada LSP para formar uma possível solução. Por exemplo, selecionar o caminho 1 para a primeira LSP, caminho 2 para a segunda LSP, caminho 3

para a terceira LSP e caminho n para a enésima LSP. Assim, cada cromossomo é representado por vários números que indicam os caminhos escolhidos. A Tabela 1 mostra um exemplo de um cromossomo ou possível solução contendo um conjunto de caminhos para as LSP´s, capazes de encaminhar o tráfego das aplicações.

QUADRO 3

Lista de caminhos em condições de estabelecer LSP´s para as aplicações.

Aplicação	Número do caminho para a LSP	Vazão desejada (MBps)	Custo nebuloso	Caminhos possíveis
Video0	1	2	80	8_9_10_12
	2	2	90	8_11_12
Video1	1	2.1	70	8_12
	2	2.1	80	8_9_10_12
	3	2.1	90	8_11_12
http0/ftp0	1	2.2	70	8_12
	2	2.2	80	8_9_10_12
	3	2.2	90	8_11_12
ftp1	1	0.4	70	8_12
	2	0.4	80	8_9_10_12
	3	0.4	90	8_11_12
Voz	1	0.128	70	8_12
	2	0.128	80	8_9_10_12
	3	0.128	90	8_11_12
ftp2	1	0.128	50	9_10_12
	2	0.128	120	9_8_11_12
ftp3	1	0.128	30	10_12
	2	0.128	140	10_9_8_11_12
ftp4	1	0.128	20	11_12
	2	0.128	140	11_8_12

TABELA 1

Exemplo de um cromossomo ou possível solução em condições de estabelecer as LSP´s para o conjunto de aplicações.

video0	Video1	http0/ftp0	ftp1	Voz	ftp2	ftp3	ftp4
1	1	2	2	2	1	1	1

O **OAG** obtêm a melhor solução para o conjunto de aplicações a partir do conjunto de soluções válidas informadas pelo **ECL**. O algoritmo do **OAG** é apresentado na Figura 34.

```
Algoritmo OAG
{ Objetivo:  realiza a escolha da melhor solução de um grupo de soluções possíveis      }
{            utilizando algoritmos genéticos.                                          }

Parâmetros de entrada  nsol, tabsol, crom
     {numero de soluções da tabela de soluções.                                       }
     {tabela com soluções possíveis para o conjunto de LSP´s.                         }
     {soluções codificadas.                                                           }
Parâmetro de saída  msol
     {vetor com melhor solução para estabelecimento das LSP´s para as aplicações.    }

 nindividuos ← 40    {Número de indivíduos                                            }
 NMAXGER ← 100 {Número máximo de gerações                                             }
 IG ← 0.9   {Intervalo de Gerações                                                    }
 gera população inicial de cromossomos.
 avalia população inicial  através do cálculo da aptidão dos indivíduos.
 ger ← 0 {Inicializa contador de gerações.                                           }
 enquanto (ger < NMAXGER) faça
     seleciona os indivíduos para reprodução (geradores).
     recombina os indivíduos selecionados (crossover).
     efetua a mutação sobre a descendência.
     avalia descendência e efetua o cálculo da aptidão dos seus indivíduos.
     reinsere descendência na população corrente.
     avalia população através do cálculo da aptidão dos indivíduos.
     ger ← ger + 1
 fim enquanto
 escolhe o melhor individuo da população.
fim algoritmo
```

Figura 34 – Algoritmo de Otimização dos recursos da rede utilizando AG.

Conforme mostra a Figura 34, inicialmente são apresentadas as variáveis utilizadas no algoritmo genético, ou seja, o número de indivíduos, o número de gerações e o Intervalo de geração (IG). O IG possibilita o controle da porcentagem da população que será substituída para a próxima geração. Um Intervalo de geração com valor igual a 0.9 indica que é implementada uma estratégia elitista, onde os 4 melhores indivíduos (10% da população) sempre propagam através das sucessivas gerações. 0 próximo passo executado no **OAG** é a geração da população inicial. A população inicial é composta por 40 soluções iniciais codificadas na forma de cromossomos, gerada de forma aleatória. O Quadro 4 mostra um exemplo de população inicial contendo um conjunto de ***n*** possíveis soluções em condições de estabelecer as LSP´s para o conjunto de aplicações.

QUADRO 4

Conjunto de soluções em condições de estabelecer as LSP´s para as aplicações.

Solução	Video0	video1	http0/ftp0	ftp1	Voz	ftp2	ftp3	ftp4
1	1	1	2	2	2	1	1	1
2	1	1	3	3	3	2	2	2
3	1	3	3	3	3	1	1	2
4	1	2	2	2	2	1	1	2
5	2	3	3	3	3	1	1	2
.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.
N	C0n	C1n	C2n	C3n	C4n	C5n	C6n	C7n

Em seguida é realizada a avaliação da população inicial. A avaliação da população inicial é feita utilizando-se a função objetivo apresentada anteriormente na equação (4). São calculados os valores de Zx para cada indivíduo (ou possível solução) da população avaliada. Os indivíduos mais aptos são aqueles para os quais são obtidos os menores valores de Zx.

Depois disso, os indivíduos geradores (pais) são selecionados de acordo com o seu valor de aptidão. O número de geradores é obtido através do produto entre o Intervalo de geração e o número de indivíduos da população original. É adotado neste trabalho Intervalo de geração com valor igual a 0.9. Desta forma, como a população original possui 40 indivíduos, são obtidos 36 indivíduos geradores após a seleção. É empregado o método de seleção por giro de roleta (Roullete Wheel Selection) sendo que a fatia de cada indivíduo é definida com a utilização da técnica de *ranking*. Nesta técnica, a fatia é definida não pela nota relativa de cada individuo, mas pela posição que eles ocupam no *ranking* de todas as notas (BRAGA et al., 2000).

O próximo passo é a recombinação dos indivíduos selecionados. É utilizado um cruzamento de único ponto, com um valor de probabilidade igual a 0.7. A nova população, ou descendência, obtida a partir dos recombinação dos indivíduos geradores possui também 36 cromossomos.

Após a recombinação dos indivíduos selecionados, pode-se efetuar a mutação sobre a descendência. Em seguida, é feita a avaliação da descendência e efetuada a re-inserção dos seus indivíduos na população corrente. A re-inserção é baseada na aptidão, sendo que os indivíduos menos aptos da população corrente são substituídos pelos mais aptos da descendência.

Depois disso, a nova população gerada é novamente avaliada. O processo continua até atingir o numero de gerações estipulado. Após isto, é escolhido o melhor individuo da população, ou seja, a melhor solução.

4.8 Engenharia de Tráfego com Algoritmo de Identificação de Enlaces Críticos

Numa rede real o número de aplicações conectadas pode variar com o tempo. Se a infra-estrutura da rede permanece a mesma, é lógico esperar que em um determinado momento futuro, alguns enlaces da rede poderão eventualmente apresentar congestionamento. Portanto, a ação da TE deve ser proativa, ou seja, deve periodicamente identificar os enlaces mais críticos e providenciar a ampliação de suas capacidades, antes que o congestionamento efetivamente ocorra.

É proposta nesta seção uma modificação no sistema de TE, com a inclusão de um módulo para identificação dos enlaces mais críticos do domínio MPLS e a sinalização da necessidade de ampliação de suas capacidades. A descrição da modificação proposta no sistema de TE é apresentada a seguir.

4.8.1 O Algoritmo de Identificação de Enlaces Críticos

Seja um domínio MPLS onde o sistema de TE dispõe de informações sobre a largura de banda disponível em todos os enlaces. Além disso, são realizadas medições periódicas das vazões nos enlaces do domínio e armazenadas em um arquivo de histórico. São considerados críticos os enlaces nos quais os valores de vazão já ultrapassaram 80% das respectivas larguras de banda. O módulo proposto, denominado **AIEC**, identifica periodicamente os enlaces mais críticos e sinaliza a necessidade de ampliação de suas capacidades. A nova configuração do sistema de TE com o **AIEC** é mostrada na Figura 35.

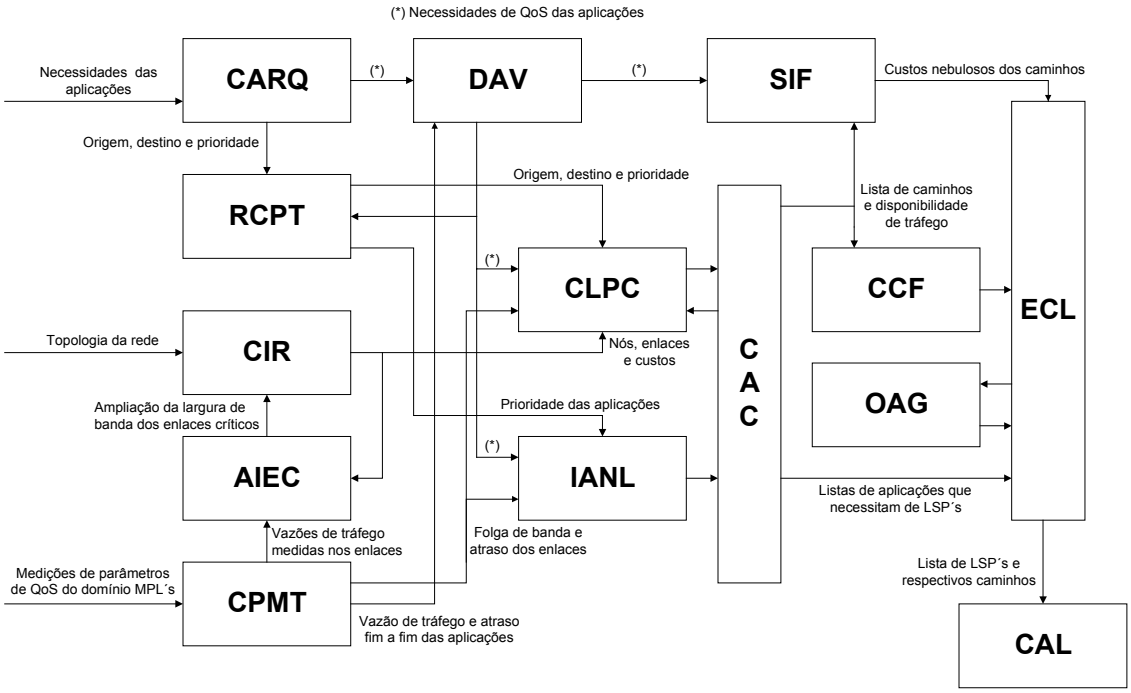

Figura 35 – Sistema de TE com Algoritmo de Identificação de Enlaces Críticos.

O **AIEC** trabalha em conjunto com redes neurais artificiais do tipo MLP. Para cada enlace crítico identificado é implementada uma RNA. As MLP´s são treinadas para prever a vazão futura nos enlaces críticos identificados, baseando-se nos valores de vazão armazenados em um arquivo de histórico. Os maiores valores de vazão previstos são depois utilizados para a avaliação da necessidade de ampliação da largura de banda dos enlaces. O algoritmo de identificação de enlaces críticos é apresentado na Figura 36.

```
Algoritmo AIEC
{ Objetivo:  efetuar a identificação dos enlaces críticos de um domínio MPLS,  previsão  }
{              da vazão e ampliação da largura de banda.                                    }
Parâmetros de entrada    num_enlaces,  enl
      { numero de enlaces  }
      { vetor de registros contendo informações sobre os  enlaces                          }

   util_enl ← 0.8
   j ← 0   { Inicializa numero de enlaces.                                                  }
  nec ← 0   { Inicializa numero de enlaces críticos.                                       }
  enquanto (j < num_enlaces) faça
       enl[j].utiliz  ← enl[j].vazao_rep / enl[j].larg_banda
       se  ((enl[j].vazao_rep / enl[j].larg_banda) > util_enl)  então
            enlac[nec] ← enl[j]
            nec ←nec + 1
       fim se
        j ← j + 1
  fim enquanto

  ordena enlaces críticos de forma crescente em função do percentual de utilização

   j ← 0 { Inicializa numero de enlaces.                                                   }
  enquanto (j < nec) faça
     lê registro de enlace critico
     prepara o arquivo de histórico de medições da vazão no enlace para uso pela RNA
     ativa RNA para previsão da vazão de trafego futura no enlace critico lido.
     le arquivo de sincronização para verificar se a RNA terminou previsão
     se  (RNA terminou a previsão)  então
         salva resultados da previsão.
     fim se
     j ← j + 1
  fim enquanto
  j ← 0 { Inicializa numero de enlaces.                                                   }
 enquanto (j < nec) faça
    processa o arquivo de vazão prevista no enlace gerado pela RNA
    obtém a maior vazão representativa prevista no enlace critico.
    se  (vazão representativa > (0.8 * enlac[j].larg_banda)) então
        sinaliza a necessidade de ampliação da largura de banda do enlace
    fim se
     j ← j + 1
  fim enquanto
fim algoritmo
```

Figura 36 – Algoritmo de Identificação de Enlaces Críticos.

O item seguinte faz a descrição da rede neural artificial MLP utilizada para previsão da vazão nos enlaces críticos.

4.8.2 A Rede Neural Artificial MLP

A predição de séries temporais envolve a construção de modelos a partir de observações em um determinado período. As técnicas clássicas para predição, tais como ARMA (*Autoregressive Moving Average*) e ARIMA (*Autoregressive Integrated Moving Average*), são bem-estabelecidas. Porém, a construção destes modelos pode depender de um entendimento mais profundo do problema, além de poder ter alta complexidade, dependendo do número de variáveis consideradas. A alternativa de modelagem por RNA´s se torna atraente, já que estas se caracterizam por um modelos não-paramétricos, em que não há necessidade de se entender o processo propriamente dito. A modelagem pode ser feita utilizando apenas amostragens de valores de entrada e saída do sistema em intervalos de tempos regulares (BRAGA et al., 2000).

Neste trabalho, optou-se por utilizar RNA´s do tipo MLP para realizar previsão da vazão nos enlaces críticos. O primeiro passo para o desenvolvimento das redes MLP´s foi a definição de suas entradas. Após a realização de testes foi definida a utilização de MLP com realimentação dos valores de tráfego de saída na entrada. Foram adotadas como entradas os valores atuais e anteriores da perda de pacotes, jitter, atraso e vazão medidos nos enlaces do domínio MPLS. A saída de cada uma das redes MLP é a vazão de tráfego futura prevista.

O passo seguinte foi a definição da configuração das redes MLP´s. Foram utilizadas rede MLP´s com duas camadas, uma escondida e outra de saída. Durante os testes, verificou-se que os melhores resultados foram obtidos com a utilização de 12 (doze) neurônios na camada escondida. O algoritmo de treinamento utilizado foi Back-propagation com taxa de aprendizado adaptativa. A taxa de aprendizado inicial utilizada para todos os enlaces foi 0.01. A taxa de erro desejada durante o treinamento das redes neurais também foi 0.01. A Figura 37 mostra a estrutura da rede MLP.

Conforme pode ser observado na figura 37, a MLP com realimentação possui 20 entradas e uma única saída. A saída refere-se a vazão futura prevista no enlace crítico. As vinte entradas das MLP´s referem-se as medições atuais e anteriores da perda de pacotes, jitter, atraso e vazão de tráfego no enlace crítico.

Os dados de entrada foram padronizados para dar a cada um deles a mesma importância. Neste trabalho foi utilizado o método de padronização Z-score (HINES, 1997). Neste método, os valores padronizados (Y) são obtidos a partir da equação (12).

$$Y = \frac{(X - Média)}{Variância} \tag{12}$$

Na equação (12), os dados de entrada não padronizados são representados por X.

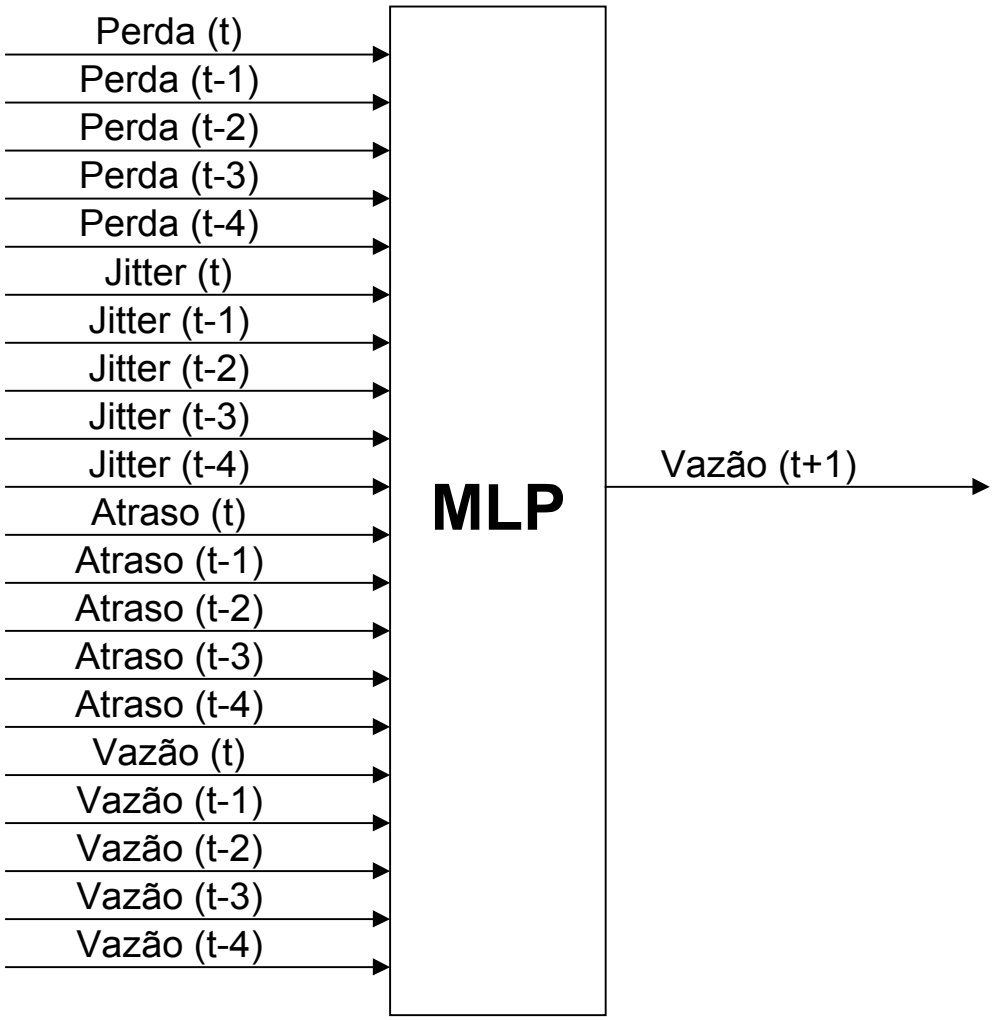

Figura 37 – Estrutura da rede MLP utilizando realimentação.

Os desempenhos das MLP´s foram avaliados utilizando o Erro Absoluto (*EABS*) e o Erro Médio Absoluto (*EMA*). O *EABS* e o *EMA* são calculados através das equações (13) e (14).

$$EABS = 100\left(\frac{Módulo(\text{Valor Real - Valor Previsto})}{\text{Valor Real}}\right) \tag{13}$$

$$EMA = \left(\frac{1}{npp} \right) \sum_{j=1}^{npp} EABS_j \qquad (14)$$

onde:

npp = Número de padrões previstos.

Com o objetivo de verificar o desempenho das redes MLP´s construídas, os valores de vazão previstos pela MLP e os EMA´s foram comparados com os obtidos por um preditor Naive. O preditor Naive realiza a operação Y(t+1) = Y(t), fornecendo uma resposta considerada um patamar mínimo de predição. O preditor Naive é utilizado apenas para comparação. A MLP deve logicamente apresentar resultados melhores que o preditor Naive.

4.9 Protótipos para simulação do sistema de Engenharia de Tráfego

O sistema de TE foi simulado utilizando o software **ns2**. Desta forma, para possibilitar a simulação do sistema foi necessário o desenvolvimento de um conjunto de programas, que implementam os módulos de software descritos nas seções anteriores. Os programas desenvolvidos foram GEDOM.CPP, CADTOP.CPP, D*.TCL, PROMED.CPP, PQUEDA.CPP, AIEC.CPP, BPTRAIN.M, RNAPREV.M, CFL*.CPP, CFLAG.CPP, SGA_TE.M e SALVAMED.TCL. Os programas acessam os seguintes arquivos:

- ARQLSPP.TR: cadastra os dados das LSP´s principais;
- ARQLSPR.TR: cadastra os dados das LSP´s reservas;
- ARQCAMIN.TR: cadastra os caminhos a serem utilizados pelas LSP´s;
- HABIL*.TR: arquivo de habilitações utilizado para comunicação entre os programas que implementam o sistema de TE;
- D*.TR: cadastra os traces de medição de tráfego gerados pelo **ns2** para cada topologia;
- MEDENREP.TR: cadastra informações sobre as características dos enlaces e também as medições de tráfego representativas;
- FENLACE.TR: cadastra as medições de tráfego dos enlaces do domínio MPLS;

- FHENLACE.TR: cadastra o histórico de medições de tráfego dos enlaces do domínio MPLS;
- FLUXOS.TR: cadastra as necessidade de QoS das aplicações como vazão, atraso máximo, jitter, e perda máxima;
- FMEDFLUX.TR: cadastra as medições de tráfego fim a fim das aplicações;
- FHMEDFLUX.TR: cadastra o histórico de medições de tráfego fim a fim das aplicações;
- QUEDA.TR: cadastra informações sobre as quedas nos enlaces;
- HQUEDA*.TR: cadastra o histórico de quedas nos enlaces;
- AUTORFLUX*.TR: contêm informações de controle sobre as autorizações das aplicações realizadas pelo CAC (bloqueios, desbloqueios, vazão autorizada, etc).

O diagrama apresentado na Figura 38 mostra a interação entre os programas durante as simulações do sistema de TE.

O programa CADTOP.CPP implementa o **CARQ** descrito nas seções anteriores. Ele possibilita o cadastro em arquivos das informações referentes ao domínio MPLS, e necessidades em termos de parâmetros de QoS desejados pelas aplicações. As informações referentes aos enlaces do domínio MPLS armazenadas no arquivo MEDENREP.TR são nó origem, nó destino, largura de banda, e atraso mínimo. Os demais campos do arquivo MEDENREP.TR como, por exemplo, vazão representativa, atraso representativo, jitter representativo e perda representativa, serão depois preenchidos automaticamente pelo PROMED. As informações referentes as aplicações submetidas ao domínio armazenadas no arquivo FLUXOS.TR são nó origem, nó destino, prioridade, vazão desejada, atraso máximo desejado, Jitter máximo desejado e perda máxima desejada. O programa CADTOP é o primeiro a ser executado, uma vez que os outros programas da simulação necessitam das informações cadastradas nestes arquivos.

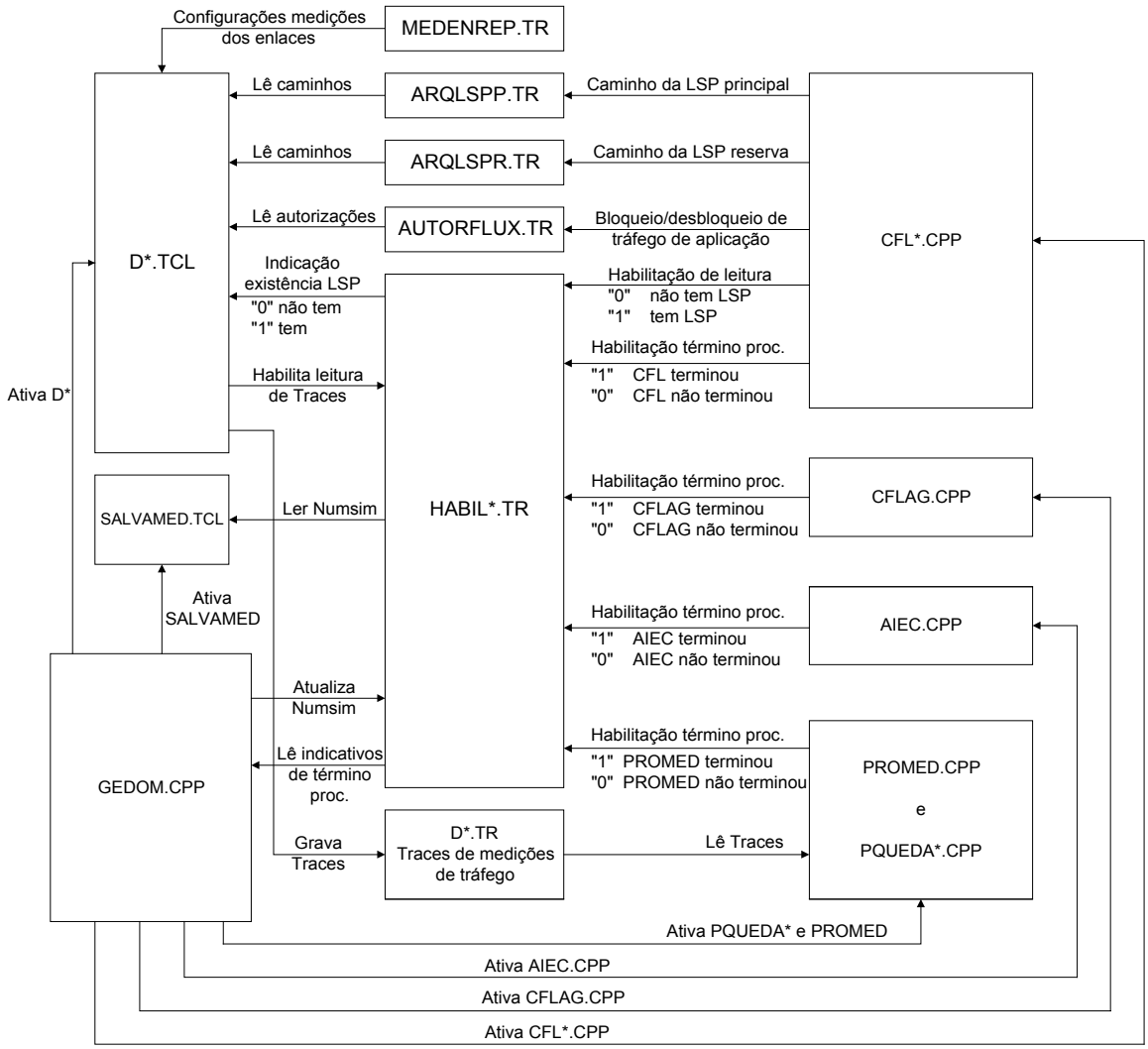

Figura 38 - Interação entre os programas para a simulação do sistema de TE.

O GEDOM.CPP faz a integração dos programas construídos para a simulação do sistema de TE no domínio MPLS. O Algoritmo mostrado na Figura 39 apresenta as atividades executadas pelo programa.

```
Algoritmo GEDOM
{ Objetivo:  efetuar a simulação do sistema de TE.                                  }
   numsim ← 10 {Define o número de simulações.                                      }
   sim ← 0 {Inicializa numero de simulações.                                        }
   Inicializa campos do arquivo de habilitações  HABIL.TR
   hab.numsim ← 0
   hab.hablecfl ← 0
   hab.habletrace ← 0
   hab.habesctrace ← 0
   hab.hablspp ←  0
   hab.hablspr ← 0
   hab.habaiec ←  0
   hab.habcflag ← 0
   grava campos em HABIL*.TR
   apaga arquivos de simulações anteriores
   apaga conteúdos dos arquivos de LSP´s (ARQLSPP.TR e  ARQLSPR.TR)
   lê HABIL*.TR
   enquanto (sim < numsim) faça
      se (sim > 0)
         Lê HABIL*.TR
         enquanto (hab.hablecfl = 0) {cfl* não terminou o processamento.           }
             lê HABIL*.TR
         fim enquanto
         hab.hablecfl ← 0
         atualiza HABIL*.TR
         tempoaiec ← resto (sim/8)
         tempoOtim ← resto (sim/6)
      fim se
      ativa D*.TCL
      lê HABIL*.TR
      enquanto (hab.habletrace = 0) {D*.TCL não terminou a simulação.              }
          lê HABIL*.TR
      fim enquanto
      hab.habletrace ←  0
      sim ← sim + 1
      atualiza HABIL*.TR
      ativa PQUEDA.CPP
      lê HABIL*.TR
      enquanto (hab.habletrace = 0)    {PQUEDA.CPP não terminou.                   }
          lê HABIL*.TR
      fim enquanto
      hab.habletrace ←  0
      atualiza HABIL*.TR
      ativa PROMED.CPP
      lê HABIL.TR
      enquanto (hab.habesctrace = 0)  {PROMED não terminou o processamento.       }
          lê HABIL*.TR
      fim enquanto
```

```
        hab.habletrace ← 0
        atualiza HABIL*.TR
        se ((sim < numsim) e (tempoaiec = 0))
            ativa AIEC.CPP
        FimSe
        lê HABIL.TR
        enquanto (hab.habaiec = 0)  {AIEC não terminou.                    }
            lê HABIL*.TR
        fim enquanto
        hab. habaiec ← 0
        atualiza HABIL*.TR
        ativa SALVAMED*.TCL
        se ((sim < numsim) e (tempoOtim > 0))
            ativa CFL*.CPP
        fim se
        se ((sim < numsim) e (tempoOtim = 0))
            ativa CFLAG.CPP
        fim se
        lê HABIL.TR
        enquanto (hab.habcflag = 0)  {CFLAG não terminou.                  }
            lê HABIL*.TR
        fim enquanto
        hab. habcflag ← 0
        atualiza HABIL*.TR
    fim enquanto
fim algoritmo
```

Figura 39 - Algoritmo para a simulação do sistema de TE

O D*.TCL é um conjunto de versões de programas que implementam no simulador ns2, as diferentes topologias do domínio MPLS utilizadas neste trabalho. Por exemplo, para a topologia I foram construídas versões do programa para simulação do modelo básico de TE, TE com re-encaminhamento em caso de falhas em enlaces do domínio, TE com aumento do tráfego das aplicações prioritárias, TE com CAC, TE com CAC e algoritmo de otimização e TE com CAC e algoritmo de identificação de enlaces críticos. O mesmo ocorreu com a topologia II. O D*.TCL além das atividades normais de implementação do domínio MPLS, também lê os arquivos de LSP´s (ARQLSPP.TR e ARQLSPR.TR) criados pelo CFL*.CPP, e em seguida cria e ativa as LSP´s necessárias.

O programa PQUEDA.CPP é o responsável pelo armazenamento das informações referentes às quedas ocorridas em enlaces do domínio MPLS. Estas informações serão utilizadas posteriormente para cálculo das confiabilidades dos enlaces. Será proposto

como trabalho futuro uma alteração no sistema de TE, com a inclusão de uma terceira função de pertinência de entrada no SIF, representando a confiabilidade do caminho candidato para estabelecimento da LSP. A confiabilidade será mais alta quanto menor for o número de ocorrências de falhas em enlaces do caminho.

O programa PROMED.CPP implementa o **CPMT** e o **DAV,** descritos nas seções anteriores. Ele é responsável pelas seguintes tarefas:

- Recolhe as medições de tráfego referentes aos enlaces do domínio e entre os nós origem e destino das aplicações (fim-a-fim);
- Armazena, processa, e classifica as medições de tráfego por enlaces e fluxos;
- Grava as medições em arquivos de histórico de enlaces e fluxos;
- Escolhe as medições representativas por enlaces e depois providencia a gravação em arquivos.

O AIEC implementa o algoritmo de identificação de enlaces críticos. O AIEC.CPP utiliza os programas BPTRAIN.M e RNAPREV.M, que são responsáveis pela implementação da MLP. A MLP, como já descrito, efetua a previsão da vazão no enlace crítico.

O CFL*.CPP é um conjunto de versões de programas que implementam os diferentes sistemas de TE descritos nas seções anteriores. Foram construídas versões do programa para simulação do modelo básico de TE, TE com re-encaminhamento em caso de falhas em enlaces do domínio, TE com aumento do tráfego das aplicações prioritárias, TE com CAC e aplicações comportadas, TE com CAC e aplicações não comportadas com prioridade estática e TE com CAC e aplicações não comportadas mas com prioridade dinâmica (RCPT).

O CFL implementa os módulos **CLPC**, **CCF**, **IANL**, **SIF, ECL e CAL.** O módulo **CAL** é executado em conjunto com o programa D*.TCL.

O CFLAG.CPP implementa o módulo de otimização dos recursos da rede utilizando algoritmos genéticos (OAG). O CFLAG.CPP utiliza o programa SGA_TE.M. O SGA_TE.M implementa a otimização usando algoritmos genéticos.

O programa SALVAMED.TCL é o responsável pela atualização dos arquivos durante as simulações.

Os programas foram desenvolvidos utilizando MatLab e as linguagens C/C++ e Otcl. Uma das funções utilizadas pelo programa CFL*.CPP implementa o SIF. Esta função foi originalmente desenvolvida na linguagem JFS (MORTENSEN, 2004) e depois foi convertida para C/C++.

4.10 Considerações finais

Foi apresentada neste capítulo, a descrição das implementações do sistema de TE no domínio MPLS. Inicialmente, foi feita a descrição do problema da TE em um domínio MPLS. Em seguida, foram apresentados o SIF e o modelo utilizado para a implementação do sistema básico de TE no domínio MPLS. Além disso, foram também apresentadas as modificações efetuadas no sistema de TE visando o tratamento do re-encaminhamento da carga de tráfego quando ocorrem falhas em enlaces, e re-roteamento de LSP´s em caso do aumento da vazão das aplicações prioritárias. Foram apresentados também os algoritmos da TE com CAC utilizando reconhecimento e classificação dos perfis de comportamento de tráfego das aplicações e TE com otimização. Foi apresentado também o algoritmo de identificação de enlaces críticos. Finalizando, foi apresentado o protótipo para a simulação do sistema.

Após o desenvolvimento do sistema de TE proposto, foram realizados dez experimentos para cada uma das implementações efetuadas. Os resultados obtidos com as simulações efetuadas são apresentados no próximo capítulo.

CAPITULO 5

RESULTADOS

Neste capítulo, são apresentados os resultados da avaliação do sistema de TE. O sistema foi implementado em um domínio MPLS com a utilização do ns2 (VINT, 2003). A avaliação levou em conta a QoS oferecida às aplicações do tipo dados, voz e vídeo, que cruzam o domínio MPLS e também os indicadores de desempenho da rede. Os parâmetros utilizados para a avaliação da QoS oferecida às aplicações foram atraso, jitter, vazão e perda de pacotes. As medições dos parâmetros de QoS são realizadas fim-a-fim. Os indicadores utilizados para avaliar o desempenho da rede foram RTeo e utilização média da rede. Além disso, foram avaliados também a perda média de pacotes obtida para o conjunto de aplicações com e sem a implementação da TE. Foram realizados dez experimentos para cada uma das implementações do sistema de TE efetuadas.

O capítulo está organizado da seguinte forma. Inicialmente, são descritas as topologias de simulação, na seção 5.1. A seguir na seção 5.2 são apresentados os resultados da implementação da TE nas topologias I e II. Na seção 5.3 são apresentados os resultados obtidos durante a avaliação da influência das falhas em enlaces sobre a QoS oferecida às aplicações. Na seção 5.4 são apresentados os resultados obtidos durante a avaliação da TE quando as aplicações prioritárias aumentam a vazão acima dos valores acordados. Na seção 5.5 são apresentados os resultados da avaliação da TE com CAC. Na seção 5.6 são apresentados os resultados obtidos durante a avaliação da TE com otimização dos recursos da rede utilizando algoritmos genéticos. Na seção 5.7 são apresentados os resultados da avaliação da TE com algoritmo de identificação de enlaces críticos. A seção 5.8 finaliza o capítulo realizando alguns comentários sobre os resultados obtidos.

5.1 Topologias de simulação

Foram utilizadas neste trabalho duas topologias de simulação. O primeiro exemplo verificado foi da topologia I, mostrada na seção 5.1.1. O objetivo desta simulação foi verificar o comportamento da TE em uma topologia simples bem conhecida, de forma a poder avaliar o seu comportamento sob condições controladas.

O segundo exemplo verificado foi da topologia II, descrita na seção 5.1.2. O objetivo desta simulação foi verificar o comportamento da TE em uma situação semelhante às encontradas no mundo real.

5.1.1 Topologia I

A topologia I é apresentada na Figura 40. Esta topologia forma um domínio MPLS com cinco nós (LSR´s), o qual está interligado a outros dezesseis nós. Os dezesseis nós correspondem a oito nós de origem e oito nós de destino de aplicações. A disciplina de escalonamento utilizada nas filas dos LSR´s foi a FIFO (First In, First Out). No caso, os pacotes recebidos são armazenados e enviados na mesma ordem em que chegaram.

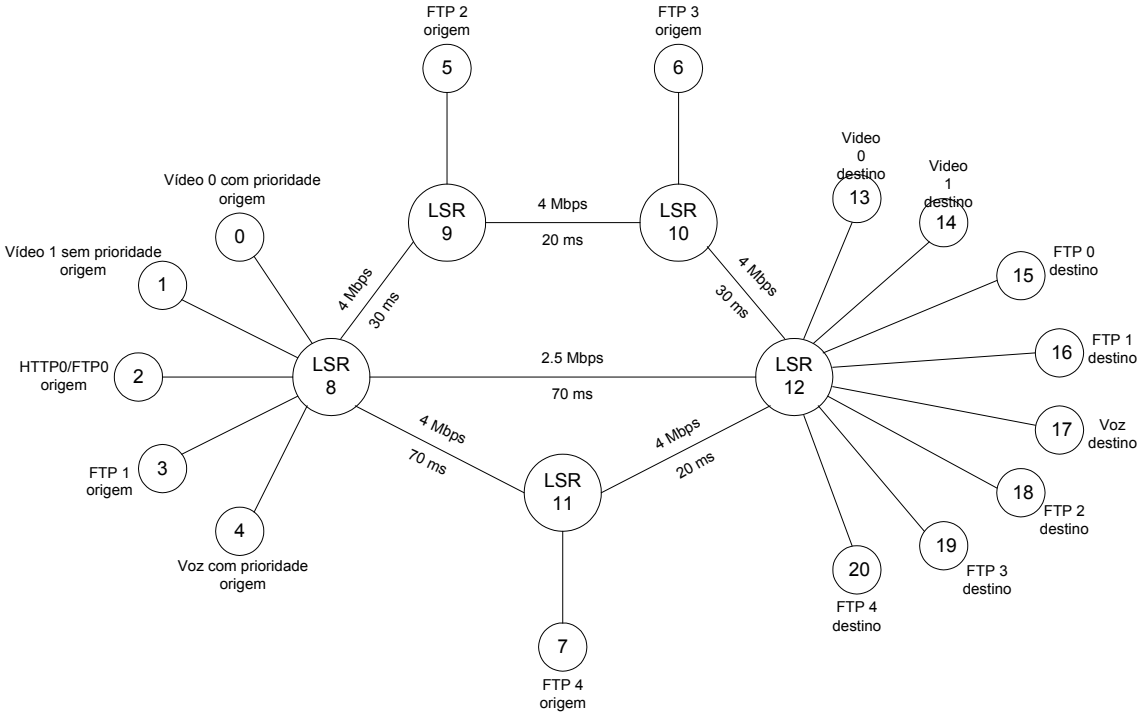

Figura 40 – Topologia I de simulação.

Durante os testes efetuados no item 5.4.1 a largura de banda do enlace 8-9 foi alterada para 2.5 MBps. As oito aplicações (Video0, Video1, ftp0, ftp1, ftp2, ftp3, ftp4 e Voip) são conectadas aos respectivos destinos, através do domínio MPLS. As aplicações estão conectadas aos LSR´s através de enlaces a 10 Mbps e atraso de 1 ms.

O domínio MPLS na topologia I possui três caminhos alternativos, os quais são necessários para realizar a TE. Os cincos LSR´s (8, 9, 10, 11 e 12) estão conectados através de enlaces de longa distância (e conseqüentemente alto atraso). Os caminhos foram configurados com as seguintes características:

- Caminho 1: é a seqüência de LSR´s 8-9-10-12, que possui um atraso mínimo de 90 ms.
- Caminho 2: é a caminho direto 8-12, que possui um atraso mínimo de 70 ms. Esse caminho é o mais curto, portanto seria o escolhido naturalmente pelos protocolos de roteamento padrão. Este caminho é candidato a se transformar num ponto de congestionamento.
- Caminho 3: representado pela seqüência de LSR´s 8-11-12, que possui um atraso mínimo de 90 ms.

A quantidade de fontes de tráfego foi definida de maneira a se verificar o comportamento da rede e os efeitos da implementação da TE em uma situação controlada. O sistema foi testado utilizando aplicações de voz, dados e vídeo. Para gerar os tráfegos das aplicações de voz foram utilizadas fontes de dados constantes (CBR). Os tráfegos das aplicações de vídeo foram obtidos com a utilização de traces reais. No caso das aplicações de dados foram utilizadas fontes FTP e traces reais HTTP. As fontes FTP foram configuradas com pacotes de tamanho de 1000 Bytes. As aplicações possuem as seguintes necessidades e características:

- Vídeo0: Vazão média 2 Mbps e atraso máximo 150ms. Esta aplicação foi cadastrada com prioridade 5, e portanto tem alta prioridade;
- Video1: Durante os testes efetuados nos itens 5.2.1 e 5.3.1 foi adotada a vazão de 1.1 MBps. No caso do item 5.4.1 a vazão adotada foi 0.35 MBps. No caso dos itens 5.5.1 e 5.5.2 a vazão foi 2.1 MBps. A aplicação foi cadastrada com prioridade 5 e o atraso máximo considerado foi 150ms;
- Dados (http0/ftp0 e ftp1): Não foram consideradas exigências, sendo que os pacotes são encaminhados pelo menor caminho. Estas aplicações foram cadastradas com

prioridade 0. Foi injetado tráfego http obtido de trace real no nó 2. Além disso, foram configuradas cinqüenta fontes de dados (FTP), sendo vinte originando-se no nó 2, e trinta originando-se no nó 3. O momento da ativação destas fontes durante as simulações foi aleatório.

- Dados (ftp2, ftp3 e ftp4): Destinam-se a gerar tráfego de fundo nos caminhos 1 e 3. Foram configuradas três fontes de dados (FTP) originando-se nos LSR´s 9, 10 e 11. Estes dados destinam-se respectivamente aos nós 18, 19 e 20. Estas aplicações não possuem prioridade. O momento da ativação destas fontes foi aleatório.
- Voz: vazão média 128 Kbps e atraso máximo 150 ms. Esta aplicação foi cadastrada com prioridade 5 (alta prioridade).

5.1.2 Topologia II

A topologia II é apresentada na Figura 41. Esta topologia forma um domínio MPLS composto de dezesseis LSR´s. Uma parte deste conjunto, ou seja dez LSR´s, são roteadores de entrada e/ou saída de trafego das aplicações. Os outros seis LSR´s são roteadores internos e apenas fazem o re-encaminhamento do tráfego através do domínio MPLS. A disciplina de escalonamento utilizada nas filas dos LSR´s foi a FIFO.

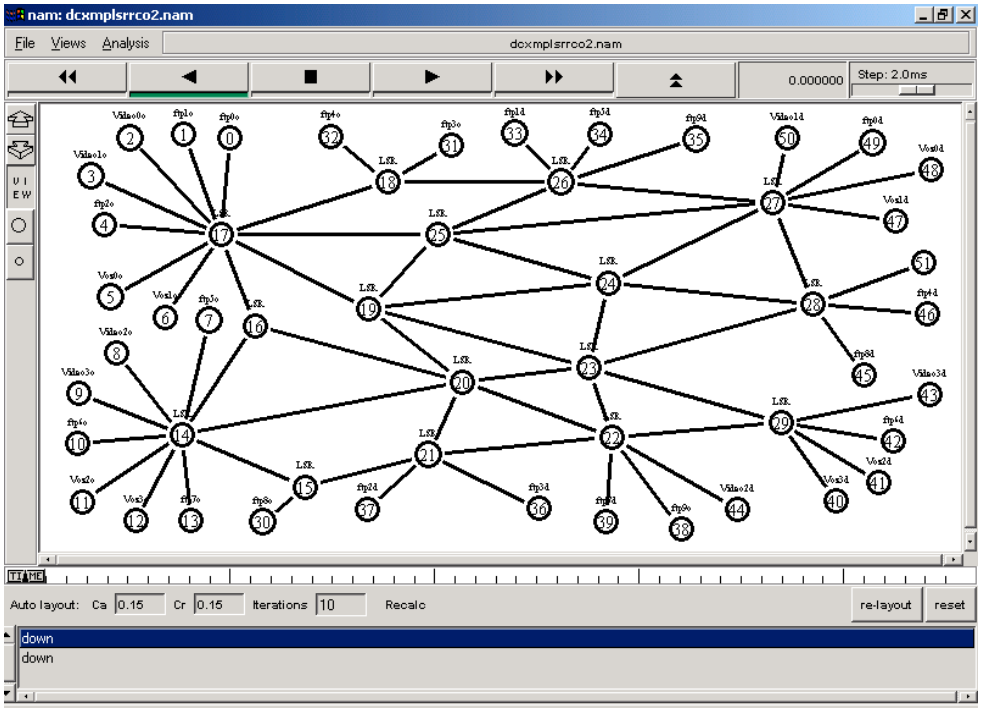

Figura 41 – Topologia II de simulação.

As aplicações tanto na origem, como no destino estão conectadas aos LSR´s do domínio através de enlaces a 10 Mbps e atraso de 1 ms. Os enlaces que compõem o domínio MPLS na topologia II possuem as características mostradas no Quadro 5.

QUADRO 5

Características dos enlaces do domínio MPLS na topologia II.

LSR de origem	LSR de destino	Largura de banda (MBPS)	Atraso mínimo (ms)
14	15	4	30
14	16	4	20
14	20	1	30
15	21	4	20
16	20	4	20
17	16	4	20
17	18	2	10
17	19	2	30
17	25	1	10
18	26	2	10
19	20	4	20
19	23	4	20
19	24	8	20
19	25	4	40
20	21	4	20
20	22	4	10
20	23	4	10
21	22	8	20
22	29	8	10
23	22	4	20
23	24	4	10
23	28	2	20
23	29	4	20
24	25	4	30
24	27	8	30
24	28	2	10
25	26	2	30
25	27	8	10
26	27	2	10
27	28	2	80

A exemplo do que foi realizada na topologia I, o sistema de TE foi simulado na topologia II utilizando aplicações de voz, dados e vídeo. Para gerar os tráfegos das

aplicações de voz foram utilizadas fontes de dados constantes (CBR). Os tráfegos das aplicações de vídeo foram obtidos com a utilização de traces reais. No caso das aplicações de dados foram utilizadas fontes FTP e traces reais http, sendo que o momento da ativação destas fontes durante as simulações foi aleatório. As fontes FTP foram configuradas com pacotes de tamanho de 1000 Bytes. As necessidades e características das aplicações utilizadas nas avaliações da função de TE são apresentadas no Quadro 6.

QUADRO 6

Necessidades e características das aplicações de dados, voz e vídeo.

Aplicação	Origem	Destino	Vazão (MBps)	Atraso (ms)	Prioridade
dados (ftp0)	0	49	-	-	0
dados (ftp1)	1	33	-	-	0
vídeo0	2	51	2.20	< 150	0
vídeo1	3	50	1.20	< 150	5
dados (ftp2)	4	37	-	-	0
dados (ftp3)	31	36	-	-	0
dados (ftp4)	32	46	-	-	0
voz0	5	48	0.128	< 150	5
voz1	6	47	0.128	< 150	5
dados (ftp5)	7	34	-	-	0
vídeo2	8	44	1.20	< 150	5
vídeo3	9	43	1.20	< 150	0
voz2	11	41	0.128	< 150	5
dados (ftp6)	10	42	-	-	0
voz3	12	40	0.128	< 150	5
dados (ftp7)	13	39	-	-	0
dados (ftp8)	30	45	-	-	0

A aplicação Video0 foi considerada prioritária (mudando de prioridade 0 para 5) durante a avaliação da função de TE efetuada no item 5.4.2. A avaliação efetuada no item 5.4.2 diz respeito a verificação da influencia do aumento da vazão das aplicações prioritárias na QoS oferecida. Nas demais simulações a aplicação Video0 permaneceu com a prioridade apontada no Quadro 6, ou seja, zero. Além disso, não foram consideradas exigências para as aplicações de dados, sendo que os seus pacotes serão encaminhados pelo menor caminho. As aplicações de dados destinam-se a gerar tráfego de fundo nos enlaces do domínio MPLS.

5.2 Resultados da implementação da Engenharia de Tráfego

São apresentados, nesta Seção os resultados obtidos durante a avaliação da TE nas topologias I e II.

5.2.1 Implementação da Engenharia de Tráfego na topologia I

Nas simulações realizadas na topologia I, as aplicações de dados, voz e vídeo iniciaram a transmissão de seus pacotes sem que tenham sido criadas LSP`s, ou seja, sem a atuação da TE. Neste caso, o roteador 8 (LSR 8) é um roteador normal, que realiza o encaminhamento baseado no caminho mais curto. Assim, todos os pacotes originados pelas aplicações foram encaminhados através do enlace 8-12, conforme mostra a Figura 42.

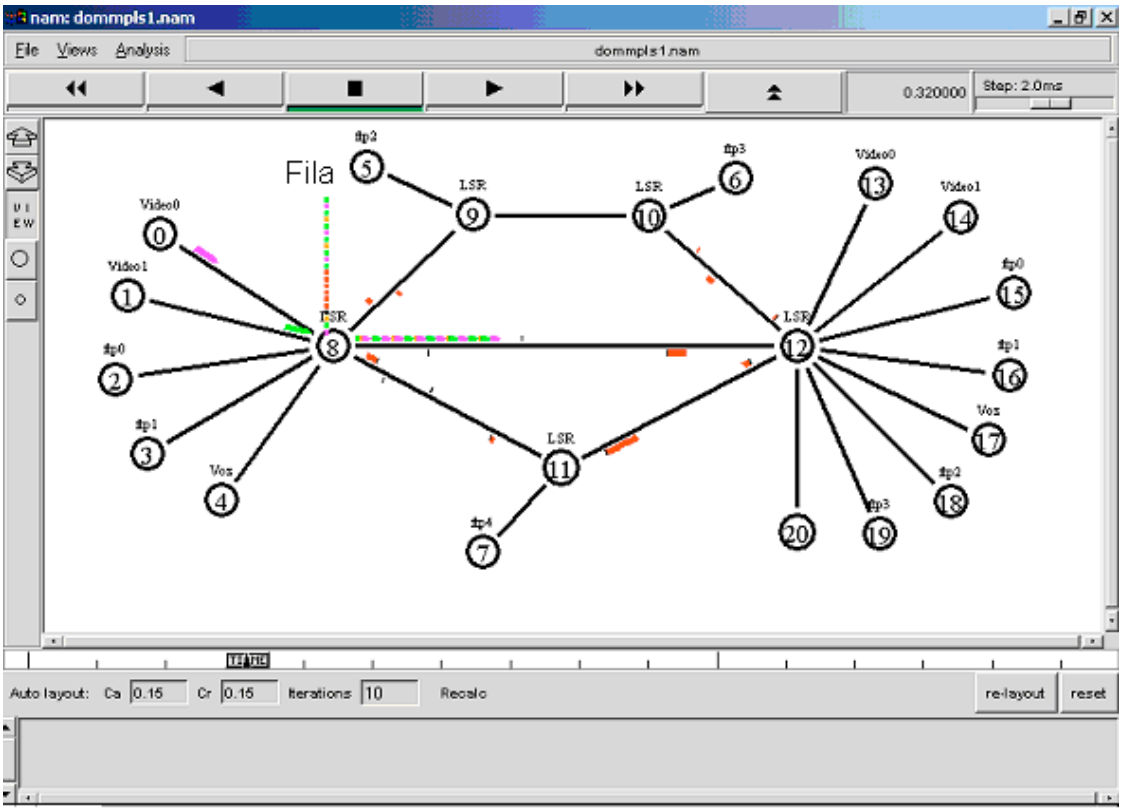

Figura 42 – Pacotes originados no roteador 8 com destino ao roteador 12 sendo encaminhados pelo menor caminho.

Pode-se observar na Figura 42 que em razão do enlace 8-12 não possuir capacidade para escoar todo o tráfego entrante, os pacotes que chegam são encaminhados para a fila. A partir do momento em que o limite máximo da fila é atingido, começa a haver descarte

de pacotes, conforme mostra a Figura 43. Pode ser observado também que os outros dois eventuais caminhos alternativos (8-9-10-12 e 8-11-12) estão completamente ociosos.

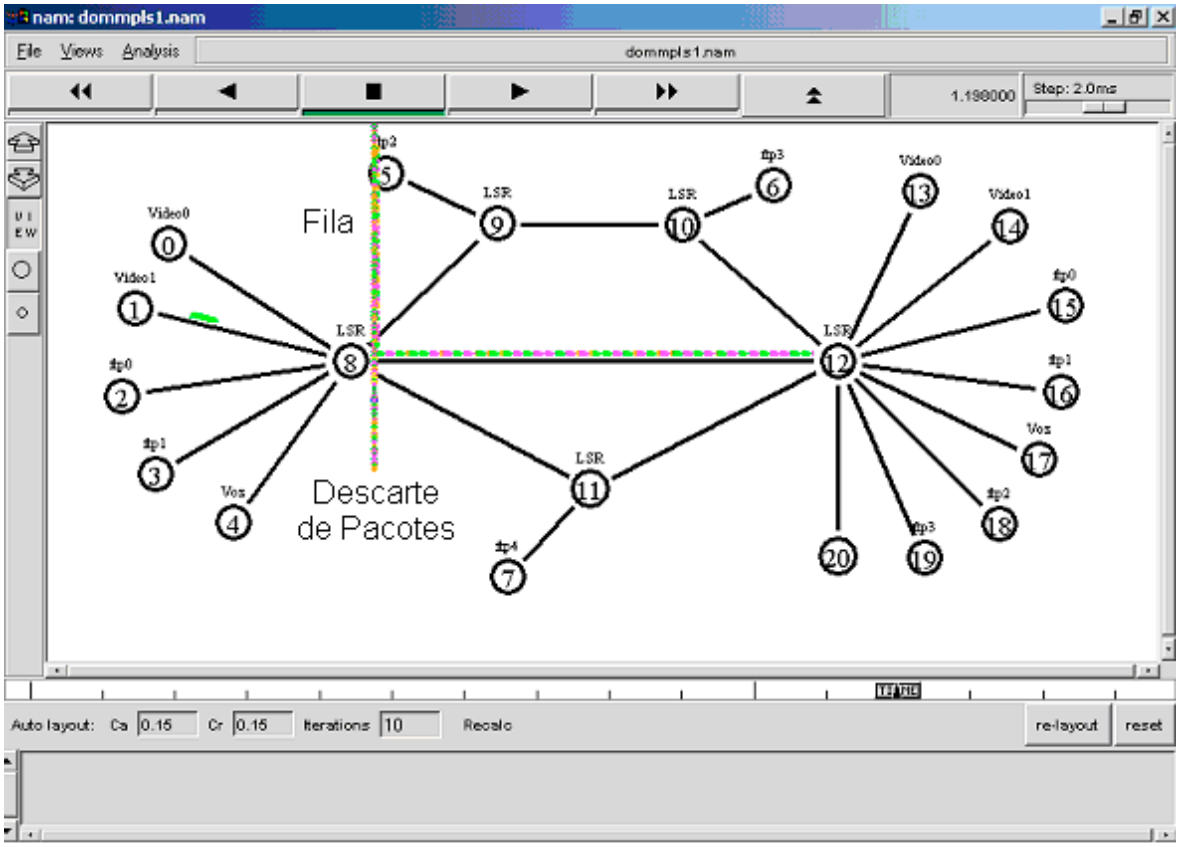

Figura 43 – A fila atinge o limite máximo de pacotes e começa a haver descartes.

Esta situação permanece até o momento ($0 \leq t < 4s$) em que o sistema de TE, através do CPMT, recebe e processa as medições dos parâmetros de QoS (vazão, atraso, jitter e perda) em cada enlace do domínio, e também as medições dos fluxos fim a fim das aplicações.

Para cada aplicação que possui prioridade, a função de TE do LSR de entrada (LSR 8) verifica se as suas necessidades de vazão de tráfego e atraso fim-a-fim estão sendo atendidas. Após a escolha do melhor caminho para cada LSP, o LSR 8 efetua a troca de sinais com os outros LSR´s e as LSP´s para cada uma das aplicações são estabelecidas. A Figura 44 mostra o efeito da implementação da TE no domínio MPLS, ou seja, os pacotes das aplicações sendo encaminhados pelas LSP´s recém criadas.

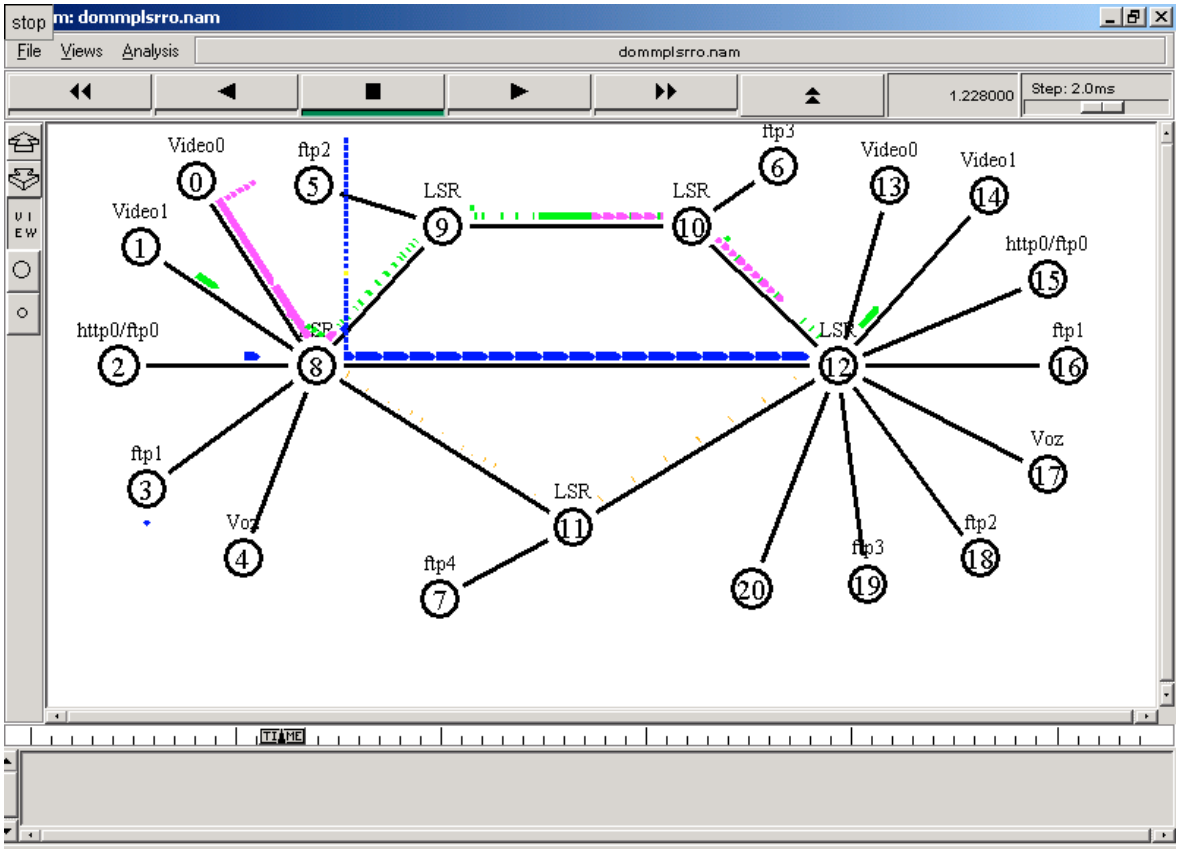

Figura 44 - Efeito da implementação da TE no domínio MPLS.

Observando-se a Figura 44, nota-se que os pacotes das aplicações de vídeo (Vídeo0 e Vídeo1) são encaminhados através de LSP´s estabelecidas através do caminho 8-9-10-12. Os pacotes da aplicação de voz são encaminhados através de LSPs estabelecida através do caminho 8-11-12. As aplicações não prioritárias (http0/ftp0 e ftp1) são encaminhadas através do enlace 8-12. Pode-se observar também que os pacotes provenientes das aplicações foram distribuídos pelos enlaces do domínio. Este fato produz como conseqüência o aumento da utilização dos enlaces e também do percentual de tráfego escoado através da rede. A Tabela 2 mostra o desempenho da rede (média e desvio padrão) antes e após a implementação da TE.

TABELA 2

Desempenho da rede antes e após a implementação da TE.

Indicadores de desempenho da rede	Sem TE	Com TE
RTeo	0.655 (0.003)	1.000 (0.001)
Utilização média da rede (%)	16.7 (0.058)	36.3 (0.100)

Pode-se observar na Tabela 2 que a RTeo, ou seja, a Relação entre o tráfego escoado e o tráfego oferecido ao domínio MPLS, teve uma melhora de 52% com a implementação da implementação da TE na topologia I. Além disso, verificou-se também que a utilização média da rede é melhorada em 117%. Estes resultados mostraram que com a implementação da TE obtêm-se uma melhor distribuição do tráfego entrante através dos enlaces do domínio MPLS, resultando no aumento do tráfego escoado.

Foi analisada em seguida a influência da implementação da TE na QoS oferecida às aplicações. Inicialmente foi verificada a influência da implementação da TE para as aplicações que possuem prioridade, ou seja, Vídeo0, Vídeo1 e voz. As Figuras 45 e 46 mostram a comparação do atraso, da variação do atraso, da vazão e perda de pacotes, antes e após a realização da TE para as aplicações de Vídeo0 (fluxo 0) e Vídeo1 (fluxo 1), respectivamente.

Os resultados apresentados nas Figuras 45 e 46 mostram que as necessidades dos fluxos de vídeo em termos de atraso máximo e vazão foram atendidas. Após a implementação da TE ($t \geq 4s$), a vazão se manteve de acordo com os valores desejados (valores de pico próximos a 2 Mbps (Video0) e 1.2 Mbps (Video1)). Da mesma forma o atraso caiu para valores próximos a 100 ms, o que atende perfeitamente as necessidades. Deve ser observada também a sensível melhora do parâmetro perda de pacotes, que antes da TE ($0 \leq t < 4s$) era bastante alto, chegando a 76.3 % (Video0) e 100% (Vídeo1) no pico, mas caiu para zero após a sua implementação. Com relação ao jitter os valores foram mantidos em 2 ms (Video0) e próximos de 1 ms (Video1) durante todo o período de observação. Deve-se observar que os valores obtidos, após a implementação da TE, para o atraso (< 150 ms) e perda de pacotes (< 1%) atendem as metas de QoS apresentados na recomendação G.1010 (ITU-T, 2002) para aplicações de vídeo. A recomendação G.1010 não propõe um valor limite para o jitter, no caso das aplicações de vídeo.

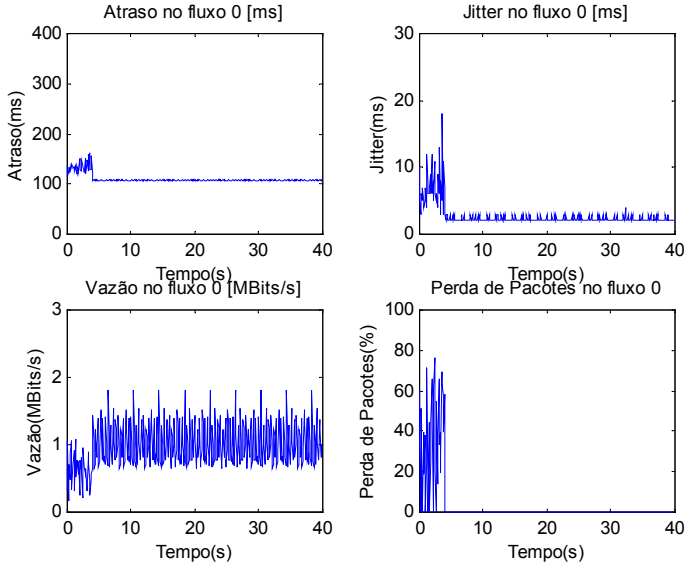

Figura 45 - Evolução da vazão, perda de pacotes, atraso e jitter para a aplicação de Vídeo0 na topologia I.

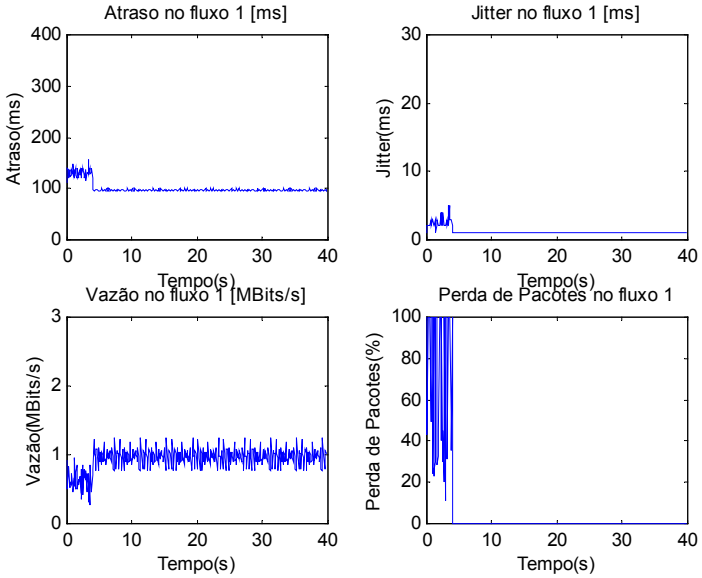

Figura 46 - Evolução da vazão, perda de pacotes, atraso e jitter para a aplicação de Vídeo1 na topologia I.

A Figura 47 mostra a evolução do atraso, do jitter, da vazão e perda de pacotes, respectivamente, antes e após a realização da TE para o fluxo de voz (fluxo 4). Os resultados apresentados na Figura 47 mostram que as necessidades do fluxo de voz em termos de atraso máximo e vazão também foram atendidas. Após a implementação da TE ($t \geq 4s$), a vazão se mantêm dentro dos valores desejados (128 kbps). Pode ser observado

também que o atraso ficou abaixo de 100 ms, o que atende perfeitamente as necessidades da aplicação. Da mesma forma que ocorreu com as aplicações de vídeo, observa-se uma sensível melhora do parâmetro perda de pacotes. Antes da TE este parâmetro era bastante alto, chegando a 100 % no pico, mas baixou para zero após a sua implementação. As exigências com relação ao jitter também são atendidas, sendo que ele permanece sempre abaixo de 1 ms**.** Deve-se observar que os valores obtidos, após a implementação da TE, para o atraso (< 150 ms), perda de pacotes (< 3%) e jitter (=< 1 ms) atendem as metas de QoS apresentados na recomendação G.1010 (ITU-T, 2002) para aplicações de voz.

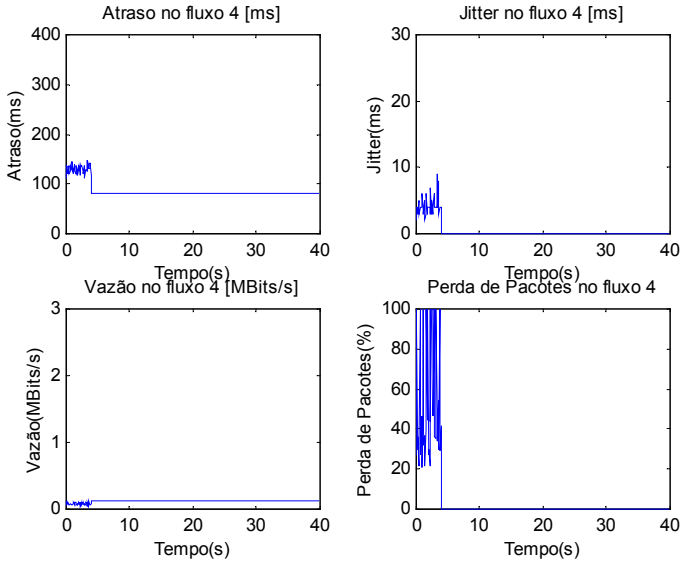

Figura 47 - Evolução da vazão, perda de pacotes, atraso e jitter para a aplicação de voz.

Comparando-se os parâmetros de QoS oferecidos às aplicações não prioritárias, antes e após a implementação da TE, nota-se que elas também foram beneficiadas. Como exemplo, é mostrada na Figura 48, a evolução da vazão, perda de pacotes, atraso e jitter para a aplicação de dados sem prioridade (fluxo 2). A evolução dos parâmetros de QoS para a aplicação de dados sem prioridade (fluxo3) é apresentada na Figura 49. Conforme pode ser verificado nas Figuras 48 e 49, não foram observadas perdas de pacotes para as aplicações de dados. Como conseqüência disto, foram atendidas as metas de QoS apresentados na recomendação G.1010 (ITU-T, 2002) para as aplicações de dados. A Tabela 3 mostra a perda média de pacotes para o conjunto de aplicações, antes e após a introdução da TE.

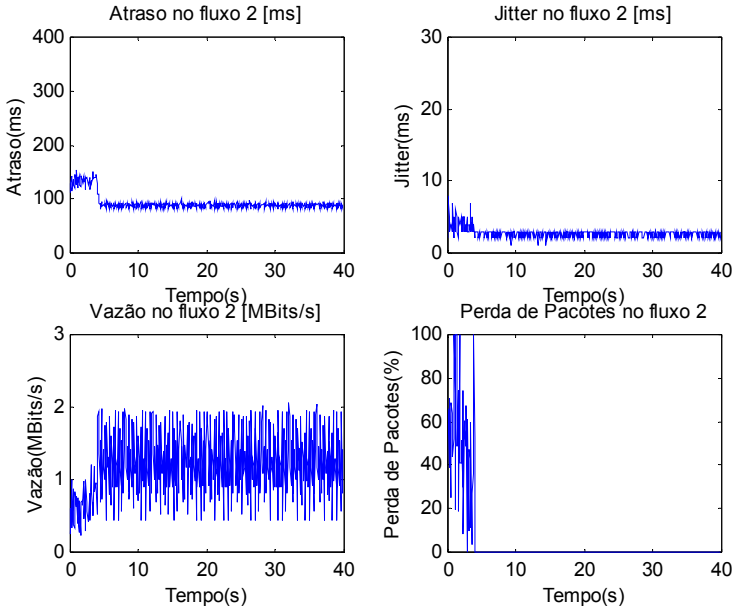

Figura 48 - Evolução da vazão, perda de pacotes, atraso e jitter para a aplicação de dados sem prioridade (fluxo 2).

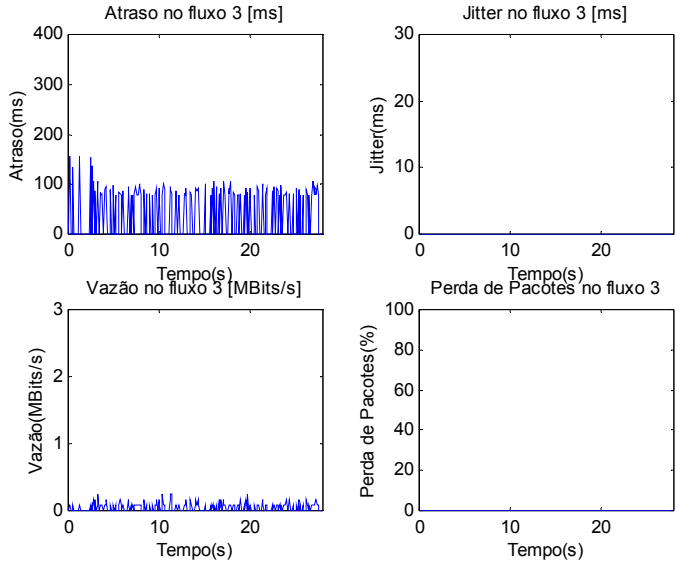

Figura 49 - Evolução da vazão, perda de pacotes, atraso e jitter para a aplicação de dados sem prioridade (fluxo 3).

TABELA 3
Perda média de pacotes para o conjunto de aplicações.

Conjunto de aplicações	Perda média de pacotes (%)	
	Sem TE	Com TE
Prioritárias	53.6 (0.493)	0.0 (0.006)
Não-prioritárias	12.4 (0.212)	0.0 (0.001)

Pode-se observar na Tabela 3 que, ocorreu uma redução significativa da perda média de pacotes para as aplicações com a introdução da TE.

5.2.2 Implementação da Engenharia de Tráfego na topologia II

Da mesma forma que foi feito para a topologia I, na topologia II as aplicações de dados, voz e vídeo iniciaram a transmissão de seus pacotes sem que tenham sido criadas LSP`s, ou seja, sem TE. Nos primeiros quatro segundos, os LSR´s são roteadores normais, que realizam o encaminhamento dos pacotes através dos caminhos mais curtos, o que resulta em congestionamento. O Quadro 7 mostra os menores caminhos utilizados pelas aplicações antes da atuação da TE.

QUADRO 7

Caminhos utilizados pelas aplicações antes da atuação da TE.

Tipo de aplicação	Prioridade	Nó de origem	Nó de destino	Menor caminho
Dados (ftp0)	0	0	49	17_25_27
Dados (ftp1)	0	1	33	17_18_26
Vídeo0	5	2	51	17_25_24_28
Vídeo1	5	3	50	17_25_27
Dados (ftp2)	0	4	37	17_19_20_21
Voz0	5	5	48	17_25_27
Voz1	5	6	47	17_25_27
Dados (ftp5)	0	7	34	14_20_19_25_26
Vídeo2	5	8	44	14_20_22_29
Vídeo3	5	9	43	14_20_22_29
Voz2	5	11	41	14_20_22_29
Dados (ftp6)	0	10	42	14_20_22_29
Voz3	5	12	40	14_20_22_29
Dados (ftp7)	0	13	39	14_20_22

A Figura 50 mostra o encaminhamento do trafego através do domínio antes da atuação da TE.

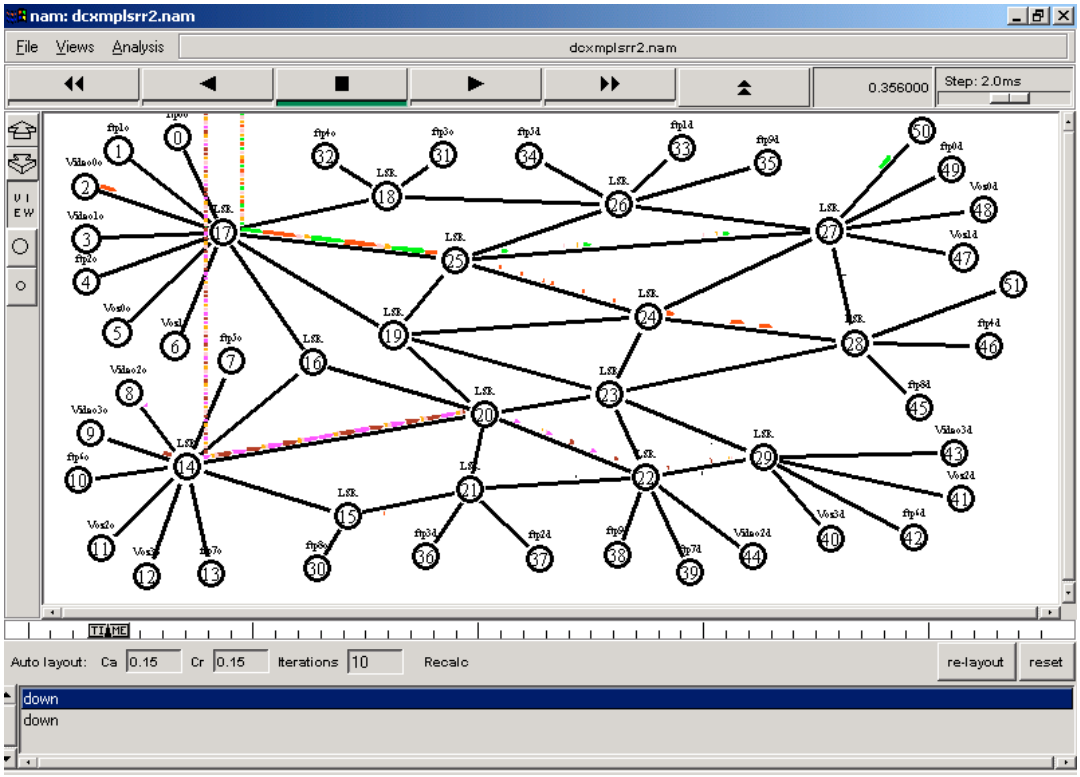

Figura 50 - Encaminhamento do trafego através do domínio antes da atuação da TE.

A situação apresentada na Figura 50 permaneceu até o momento em que a função de TE começou a atuar ($0 \leq t < 4s$). O **CPMT** recolhe e processa as medições dos parâmetros de QoS (vazão, atraso, Jitter e perda) em cada enlace do domínio, e também as medições dos fluxos fim-a-fim das aplicações. Para cada aplicação que possui prioridade, a função de TE, através do **IANL**, verifica se as suas necessidades de vazão de tráfego e atraso fim-a-fim estão sendo atendidas. O **IANL** produz uma lista de aplicações que necessitam de LSP´s. O Quadro 8 mostra os caminhos escolhidos pelas funções de TE, para o estabelecimento das LSP´s para as aplicações que possuem prioridade.

QUADRO 8

Caminhos escolhidos pelas funções de TE para o estabelecimento das LSP´s.

Tipo de aplicação	Nó de origem	Nó de destino	Caminho da LSP
Vídeo0	2	51	17_16_20_23_28
Vídeo1	3	50	17_18_26_27
Voz0	5	48	17_18_26_27
Voz1	6	47	17_19_25_27
Vídeo2	8	44	14_16_20_22
Vídeo3	9	43	14_15_21_22_29
Voz2	11	41	14_15_21_22_29
Voz3	12	40	14_15_21_22_29

A Figura 51, mostra o efeito da implementação da TE no domínio MPLS, ou seja, os pacotes das aplicações prioritárias sendo encaminhados pelas LSP´s criadas. O tráfego originado pelas aplicações sem prioridade continuou a ser encaminhado pelo menor caminho.

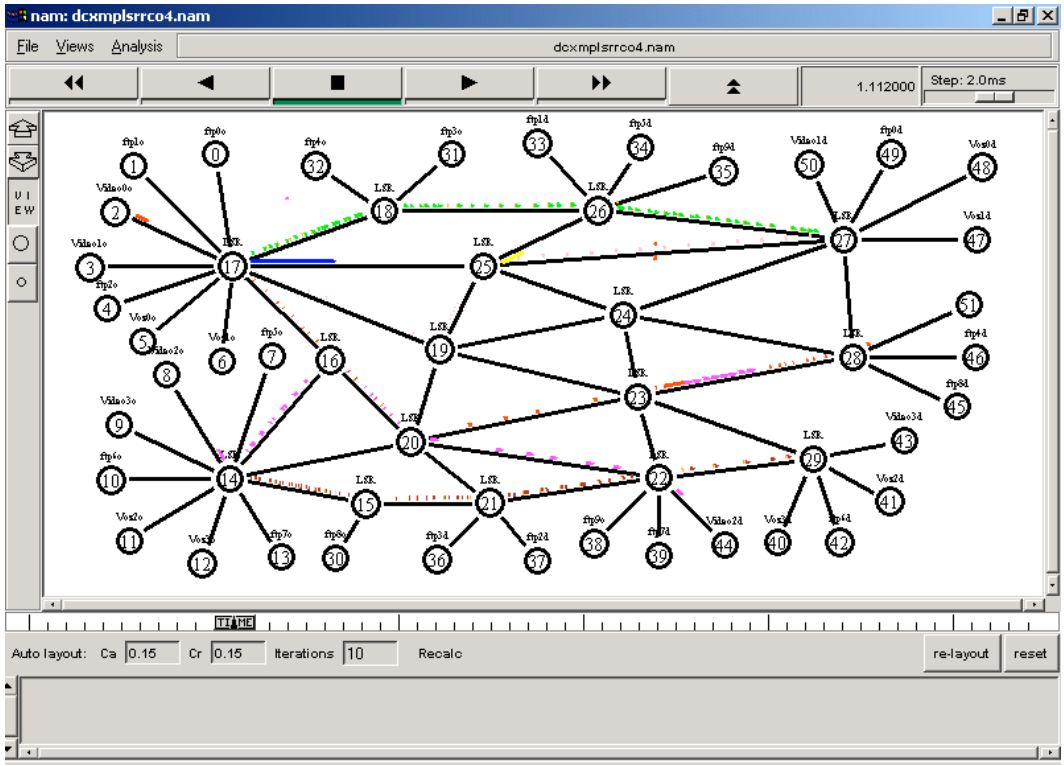

Figura 51 - Efeito da implementação da TE no domínio MPLS.

Em seguida, foi verificada a influência da implementação da TE na QoS oferecida às aplicações originadas nos LSR´s 14 e 17. Os resultados obtidos mostram que as necessidades das aplicações em termos de atraso máximo e vazão foram atendidas. Assim, apenas como exemplo, são apresentadas nas Figuras 52, 53, 54 e 55, as evoluções do atraso, do jitter, da vazão e perda de pacotes, antes e após a realização da TE para as aplicações de Vídeo0 (fluxo 2), Vídeo1 (fluxo 3), Voz0 (fluxo 5) e Voz1 (fluxo 6) originadas no LSR 17.

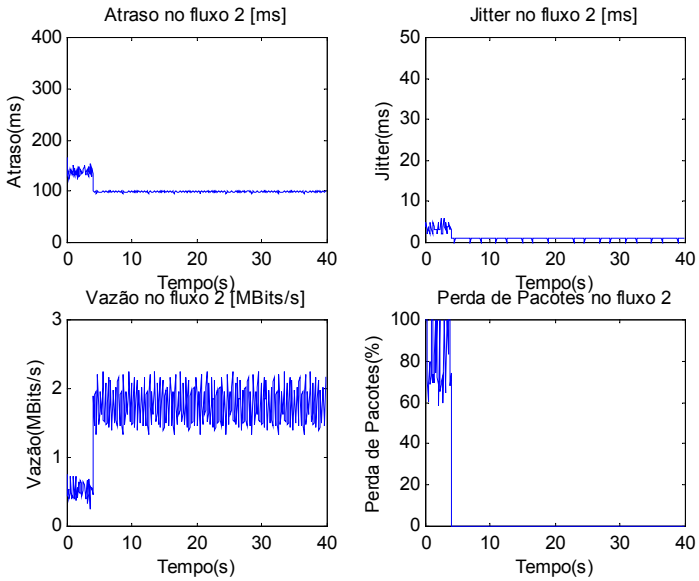

Figura 52 - Evolução da vazão, perda de pacotes, atraso e jitter para a aplicação de Vídeo0 na topologia II.

Os resultados apresentados na Figuras 52 mostram que após a implementação da TE ($t \geq 4s$), a vazão se manteve próxima dos valores desejados, ou seja, 2.2 MBps. Da mesma forma o atraso caiu para 90 ms, o que atende perfeitamente as suas necessidades, segundo a recomendação G.1010 (ITU-T, 2002). Deve ser observada também a sensível melhora da perda de pacotes, que antes da TE chegava a 100 % no pico e caiu para zero após a sua implementação. Com relação ao jitter, os resultados foram satisfatórios segundo a recomendação G.1010, mantendo-se abaixo de 1 ms durante todo o período de observação.

A Figura 53 apresenta a evoluções do atraso, do jitter, da vazão e perda de pacotes, antes e após a realização da TE para a aplicação de Vídeo1 (fluxo 3) também originada no LSR 17. Os resultados são semelhantes aos obtidos pela aplicação Vídeo0 (fluxo 2), apresentados na Figura 52.

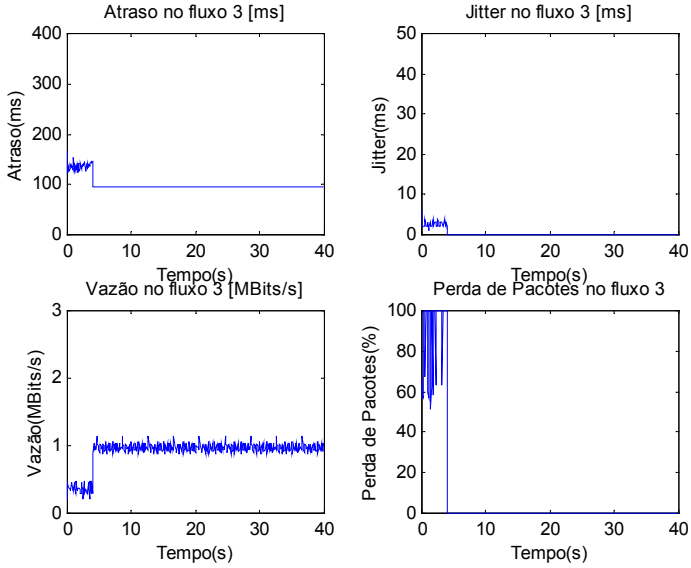

Figura 53 - Evolução da vazão, perda de pacotes, atraso e jitter para a aplicação de Vídeo1 (fluxo 3) na topologia II

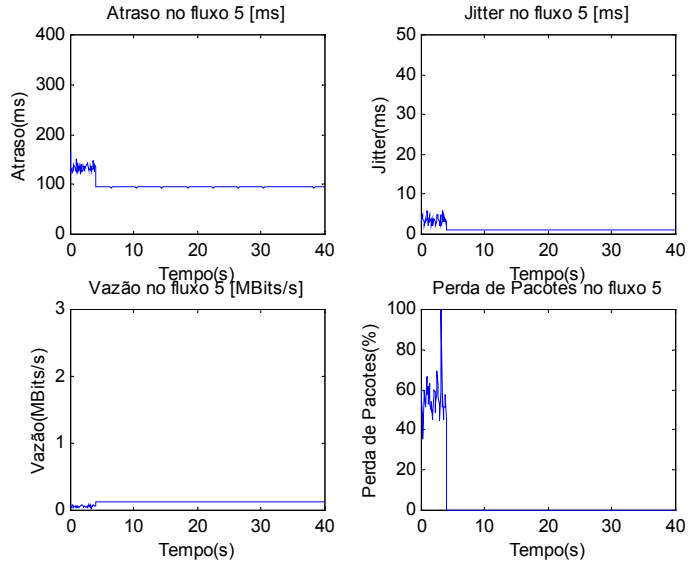

Figura 54 - Evolução da vazão, perda de pacotes, atraso e jitter para a aplicação de Voz0 (fluxo 5) na topologia II.

Os resultados apresentados na Figura 54 mostram que as necessidades da aplicação de Voz0 em termos de atraso máximo e vazão também foram atendidas. Após a implementação da TE, a vazão se manteve próxima do valor desejado (128 kbps). Pode ser observado também que o atraso foi 90 ms, o que atende perfeitamente as necessidades da aplicação (ITU-T, 2002). Da mesma forma que ocorreu com as aplicações de vídeo,

observou-se uma sensível melhora do parâmetro perda de pacotes. Antes da TE este parâmetro era bastante alto, oscilando em torno de 60% e chegando a 100% no pico, sendo que baixou para zero após a sua implementação. Os resultados obtidos para o jitter são satisfatórios (ITU-T, 2002), sendo que permaneceram sempre abaixo de 1 ms durante o período de atuação da função de TE.

A Figura 55 apresenta a evoluções do atraso, do jitter, da vazão e perda de pacotes, antes e após a realização da TE para a aplicação de Voz1 (fluxo 6) também originada no LSR 17. Os resultados são semelhantes aos obtidos pela aplicação Voz0 (fluxo 5), apresentados na Figura 54.

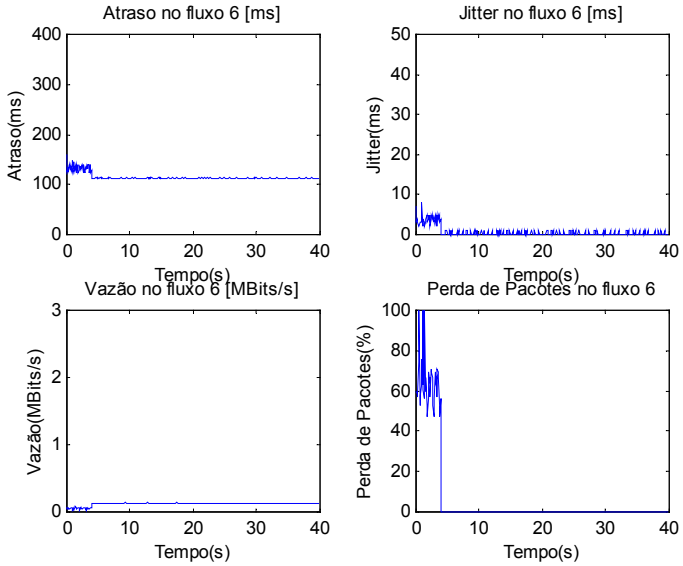

Figura 55 - Evolução da vazão, perda de pacotes, atraso e jitter para a aplicação de Voz1 (fluxo 5) na topologia II.

Após a análise dos resultados obtidos para as aplicações prioritárias entrantes no LSR 17, foi feito o mesmo para o LSR 14. Os resultados observados para o LSR 14 foram semelhantes ao obtidos para as aplicações entrantes no LSR 17. As necessidades das aplicações de Vídeo2 (fluxo 8), Vídeo3 (fluxo 9), Voz2 (fluxo 11) e Voz3 (fluxo 12) foram atendidas. Após a atuação da TE, as vazões medidas foram adequadas e os valores dos atrasos fim-a-fim ficaram sempre abaixo de 100 ms. As perdas de pacotes caíram praticamente a zero após a atuação da TE. Além disso, os valores obtidos para o jitter permaneceram sempre abaixo de 1 ms durante o período de atuação da função de TE**.** A

Tabela 4 resume os resultados obtidos, apresentando os indicadores de desempenho globais antes e após a implementação da TE para a topologia II.

TABELA 4

Desempenho da rede na topologia II antes e após a implementação da TE.

Indicadores de desempenho da rede		Sem TE	Com TE
RTeo		0.464 (0.001)	1.000 (0.001)
Perda média de pacotes (%)	Conjunto de aplicações prioritárias	71.1 (0.500)	0.0 (0.577)
	Conjunto de aplicações não-prioritárias	0.0 (0.574)	0.0 (0.001)
Utilização média da rede (%)		11.8 (0.044)	17.0 (0.265)

Os resultados apresentados na Tabela 4 indicam que a implementação da TE na topologia II possibilitou uma melhoria de 116% para a RTeo e 44% para a utilização média da rede. Pode-se observar também que ocorreu uma redução significativa da perda média de pacotes para as aplicações prioritárias. Desta forma, a exemplo do ocorreu para a topologia I, a implementação da TE na topologia II também proporcionou uma melhoria dos indicadores de desempenho globais da rede.

5.3 Influência das falhas em enlaces sobre a QoS oferecida às aplicações

São apresentados, nesta Seção os resultados obtidos durante a avaliação da influência das falhas em enlaces sobre a QoS oferecida às aplicações nas topologias I e II.

5.3.1 Influência das falhas em enlaces sobre a QoS oferecida às aplicações na topologia I

Uma das formas como o MPLS provê restauração do fluxo dos dados em caso de falha em enlace do domínio MPLS é através de uma LSP pré-alocada, que serve de reserva para a LSP principal. Detectada a falha, o nó-origem do fluxo de dados usa seu mecanismo de re-roteamento. Assim, em primeiro lugar, é descartada temporariamente a LSP principal, sendo o fluxo de dados desviado para a Reserva. No caso da simulação

realizada na topologia I, o tráfego originado pelas aplicações prioritárias de vídeo, que chegam ao LSR 8, flui normalmente pelo caminho 8_9_10_12 (LSP principal) conforme já foi mostrado na Figura 44. Por outro lado, a função de TE estabelece uma LSP reserva através do caminho 8_11_12. Enquanto não houver a ocorrência de falhas em enlaces do caminho da LSP principal, a LSP reserva permanece inativa sem ocupar largura de banda. Quando ocorre uma falha em algum enlace do caminho principal, como no caso do enlace 8_9 na Figura 56, o LSR 8 detecta o fato e redireciona os pacotes para a LSP reserva.

Conforme é mostrado nas Figuras 56 e 57, imediatamente após a ocorrência de falha no enlace 8_9 que faz parte do caminho principal 8_9_10_12, é realizado o re-encaminhamento para o caminho alternativo 8_11_12 (LSP reserva).

Após a recuperação do enlace 8_9, o tráfego das aplicações de vídeo e voz com prioridade volta a ser encaminhado através do caminho principal 8_9_10_12 conforme é mostrado nas Figuras 58 e 59.

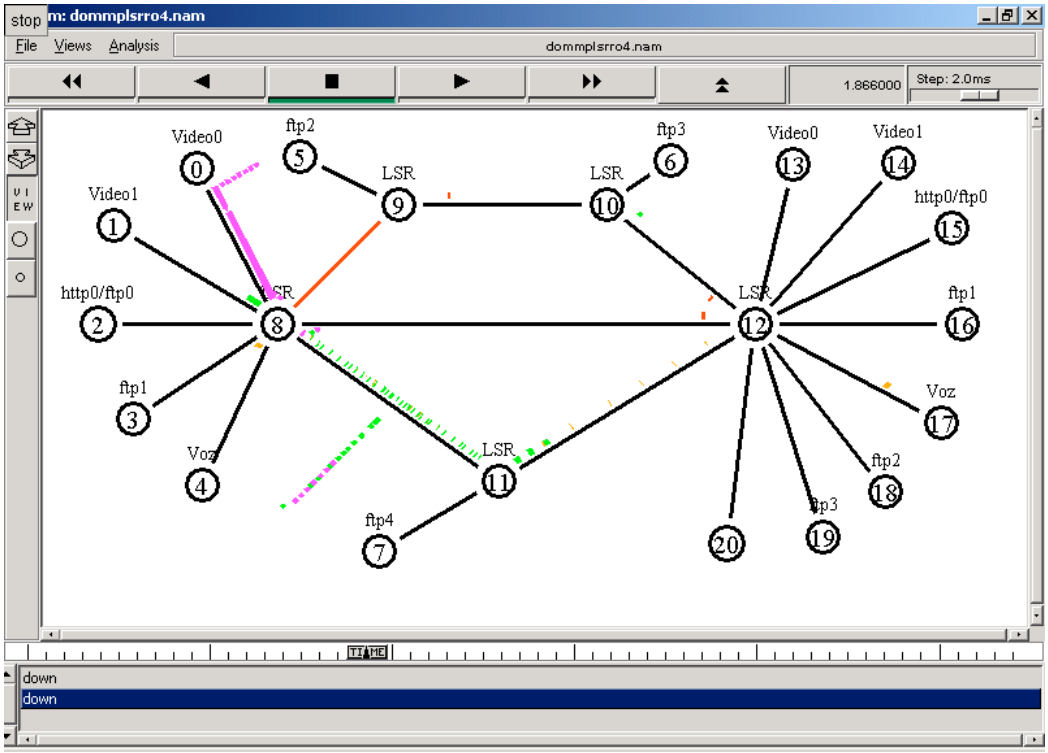

Figura 56 - Queda no enlace responsável pelo encaminhamento das aplicações de vídeo com prioridade.

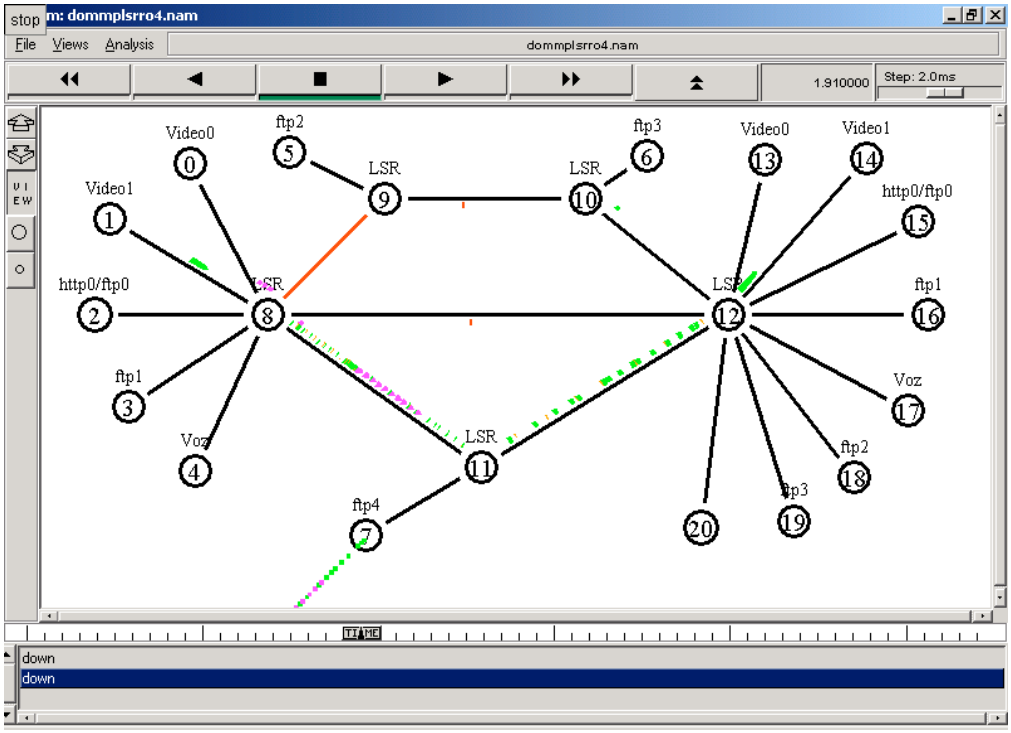

Figura 57 - Re-encaminhamento do tráfego das aplicações de vídeo com prioridade para o caminho alternativo (LSP reserva).

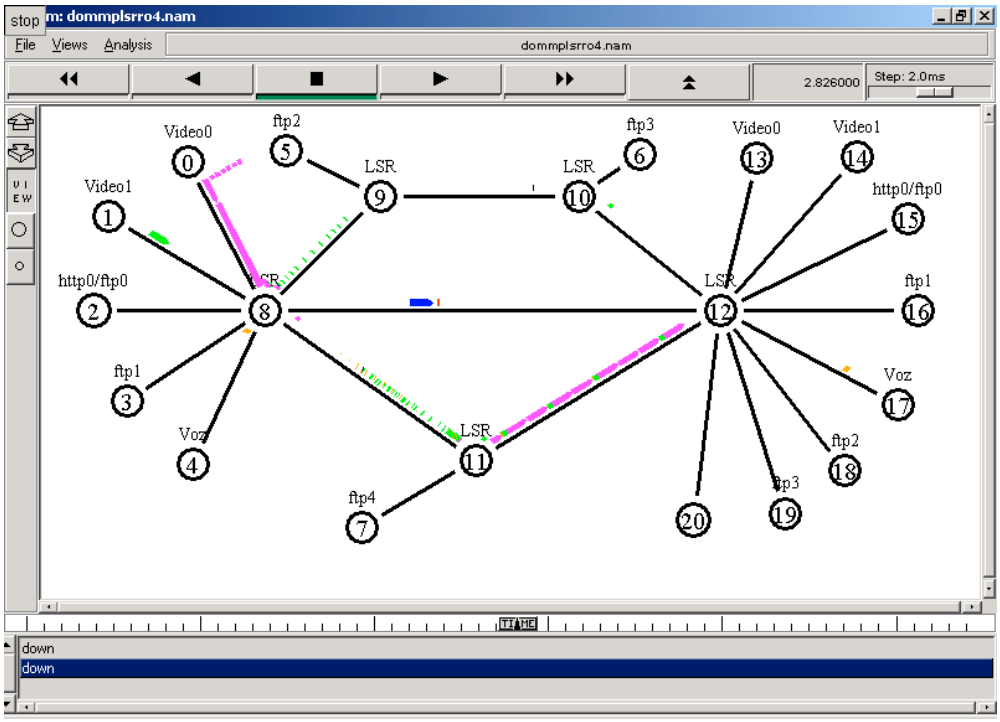

Figura 58 - O tráfego das aplicações de vídeo com prioridade volta a ser encaminhado através do caminho principal 8_9_10_12.

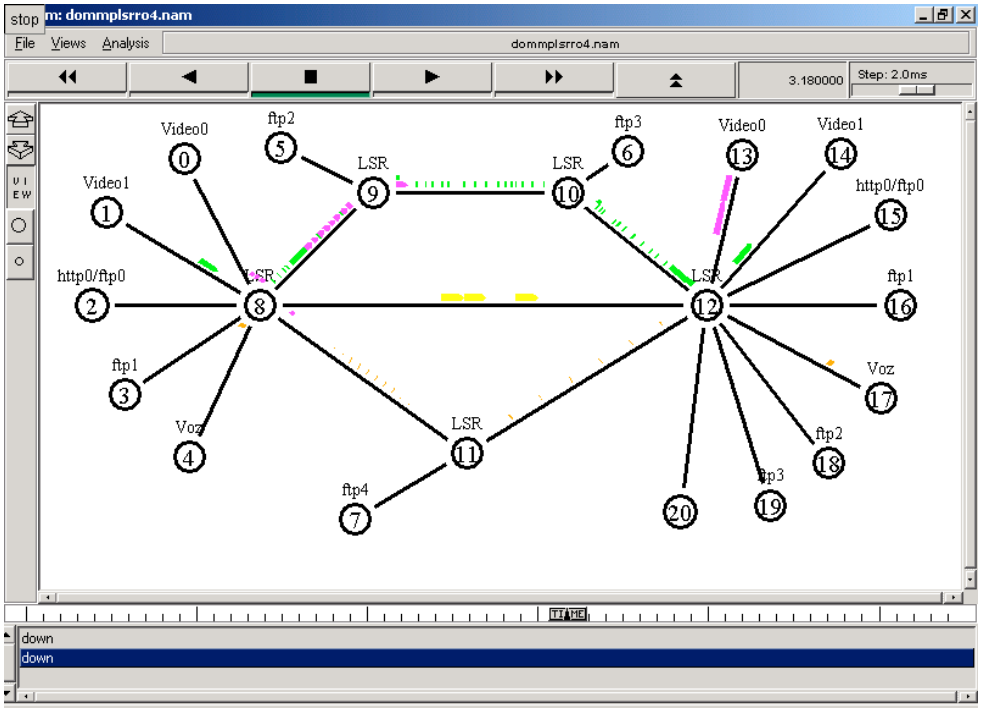

Figura 59 - O tráfego das aplicações de vídeo com prioridade sendo encaminhado novamente através do caminho principal.

As Figuras 60 e 61 mostram o efeito da falha no enlace 8_9 e do re-encaminhamento do tráfego através do caminho alternativo no nível de QoS oferecido às aplicações de vídeo com prioridade.

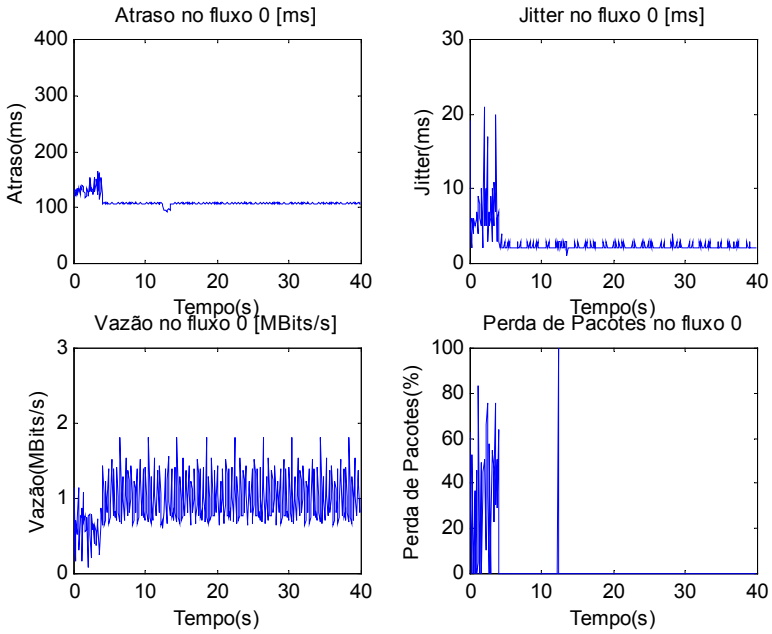

Figura 60 - Efeito da falha no enlace 8-9 no nível de QoS oferecido à aplicação de vídeo com prioridade (Vídeo0).

Os resultados apresentados na Figura 60 mostram que a QoS oferecida as aplicação de Vídeo0 continuou a ser mantida apesar da falha no enlace 8_9. Após a implementação da TE, a vazão se manteve na faixa acordada. A perda de pacotes, que antes da implementação da TE era alta, caiu para zero após a sua implementação, mas voltou a ser significativa (100%) durante um curtíssimo espaço de tempo devido a queda no enlace. O atraso e o jitter praticamente não foram afetados pela queda do enlace 8_9.

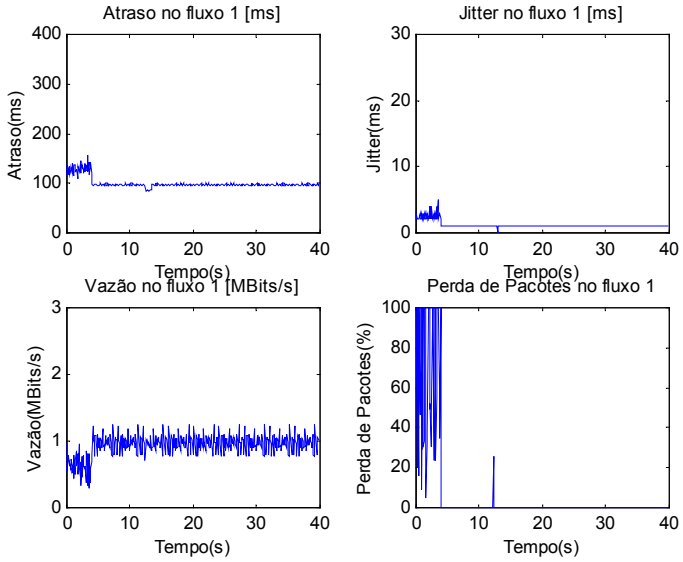

Figura 61 - Efeito da falha no enlace 8-9 no nível de QoS oferecido à aplicação de vídeo com prioridade (Vídeo1).

Conforme mostram os resultados apresentados na Figura 61, a QoS oferecida a aplicação de Vídeo1 também continuou a ser mantida apesar da falha no enlace 8_9. Após a implementação da TE, a vazão se manteve dentro dos valores desejados. A perda de pacotes, que antes da implementação da TE era bastante alta (100%) no período entre 0 e 4s, caiu para zero, mas voltou a ser significativa (25.5 %) devido a queda no enlace. No caso, pode-se observar que a queda do enlace afetou mais a aplicação de Vídeo0 do que a aplicação de Video1 com relação ao parâmetro perda de pacotes. Isto ocorreu em função da maior valor de vazão da aplicação de Video0. Por outro lado, o atraso e o jitter oferecidos a aplicação de Vídeo1 praticamente não foram afetados pela queda do enlace 8_9, como também ocorreu com a aplicação de Vídeo0.

Em termos globais o desempenho da rede foi pouco afetado pela queda no enlace 8-9. A Tabela 5 mostra os indicadores de desempenho da rede submetida a atuação da TE na

ausência e presença de falha em enlace. Os resultados mostram que devido a ocorrência de falha no enlace 8-9, houve uma pequena diminuição da RTeo, e um aumento da perda media de pacotes para o conjunto de aplicações prioritárias. Entretanto, a perda média de pacotes obtida para o conjunto de aplicações prioritárias atende as metas de QoS apresentadas na recomendação G.1010 do ITU-T para aplicações de voz (< 3%) e vídeo (< 1%). A utilização média da rede permanece inalterada.

TABELA 5

Efeito da falha no desempenho da rede submetida à atuação da TE.

Indicadores de desempenho		Domínio MPLS Com TE e sem falha em enlace	Domínio MPLS Com TE e com falha em enlace
RTeo		1.000 (0.001)	0.999 (0.052)
Perda média de pacotes (%)	Conjunto de aplicações prioritárias	0.0 (0.006)	0.1 (0.015)
	Conjunto de aplicações não-prioritárias	0.0 (0.001)	0.0 (0.050)
Utilização média da rede (%)		36.3 (0.100)	36.3 (0.153)

5.3.2 Influência das falhas em enlaces sobre a QoS oferecida às aplicações na topologia II

Conforme foi mostrado anteriormente no Quadro 8, a LSP principal da aplicação prioritária Vídeo2 foi estabelecida através do caminho 14_16_20_22. Visando proteger o tráfego da aplicação prioritária em caso de falhas em enlaces da LSP principal, a função de TE atuante sobre a LSR 14 criou uma LSP reserva através do caminho 14_15_21_22.

A Figura 62 mostra a ocorrência de falha no enlace 14_16, que faz parte do caminho utilizado pela LSP principal da aplicação de Vídeo2. Imediatamente após a ocorrência da queda do enlace 14-16, foi realizado o re-roteamento do tráfego da aplicação prioritária Vídeo2 para a LSP reserva através do caminho 14_15_21_22. Após a recuperação do enlace 14_16, o tráfego da aplicação voltou a ser encaminhado através do caminho 14_16_20_22 original. Este encaminhamento é exatamente igual ao mostrado anteriormente na Figura 51.

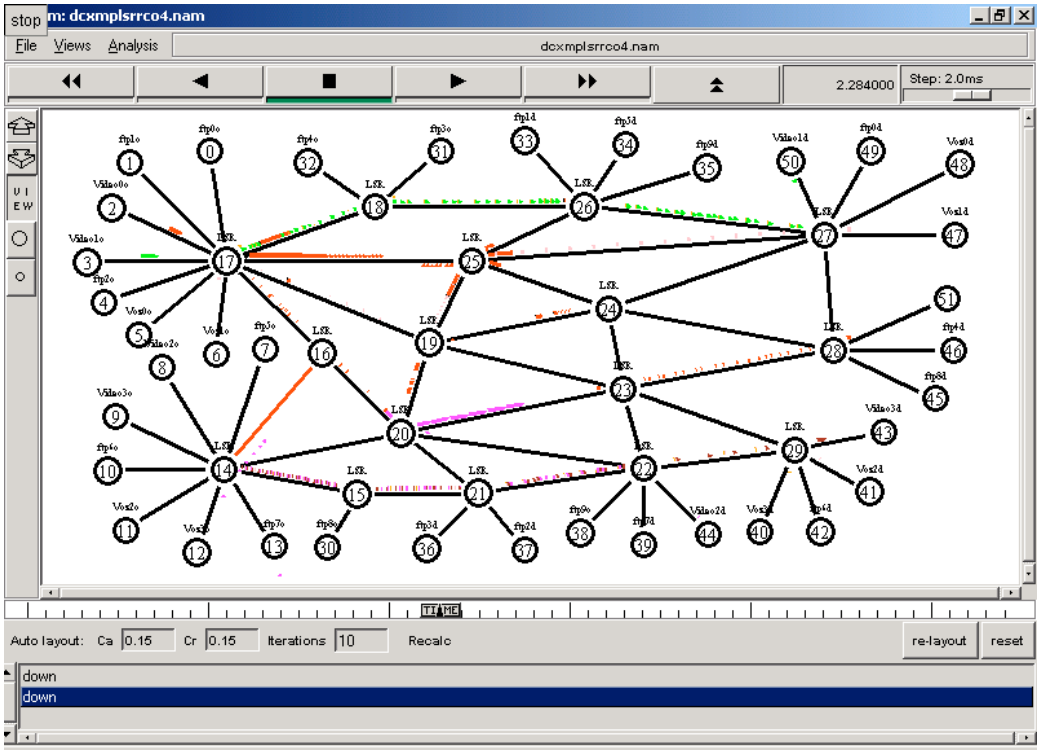

Figura 62 - Queda no enlace 14-16 da LSP principal e re-encaminhamento do tráfego para a LSP reserva.

A Figura 63 mostra o efeito da falha no enlace 14-16 e do re-encaminhamento do tráfego através da LSP alternativa sobre a QoS oferecida à aplicação de Vídeo2 (fluxo 8).

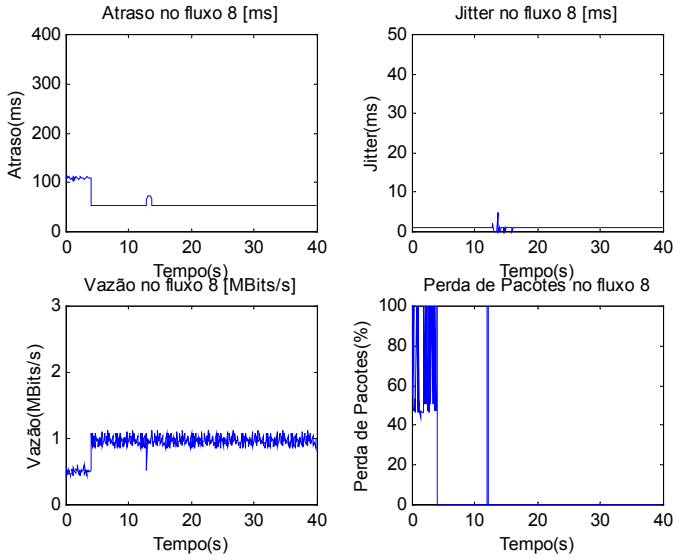

Figura 63 - Efeito da falha no enlace 14-16 e do re-encaminhamento do tráfego sobre a QoS oferecida às aplicações de Vídeo2.

Os resultados apresentados na Figura 63 mostram que a queda no enlace 14_16, afetou pouco a QoS oferecida à aplicação. A LSP reserva estabelecida pela função de TE supriu as necessidades da aplicação. Após a atuação da função de TE, a vazão se mantêm dentro dos valores desejados até o momento da queda do enlace 14_16 (t=13.8s). Durante um curto período de tempo, correspondente a detecção da falha e ativação da LSP reserva, ocorreu perda de pacotes, diminuição da vazão e aumento do atraso e também do Jitter. Após este momento a situação voltou ao normal, sendo que a QoS oferecida à aplicação foi semelhante aquela disponibilizada quando na ausência de falha.

Em termos globais o desempenho da rede foi pouco afetado pela queda no enlace 14-16. A Tabela 6 mostra os indicadores de desempenho da rede submetida a atuação da TE na ausência e presença de falha em enlace.

TABELA 6

Efeito da falha no desempenho da rede II submetida à atuação da TE.

Indicadores de desempenho		Domínio MPLS Com TE e sem falha em enlace	Domínio MPLS Com TE e com falha em enlace
RTeo		1.000 (0.001)	0.999 (0.048)
Perda média de pacotes (%)	Conjunto de aplicações prioritárias	0.0 (0.577)	0.1 (0.551)
	Conjunto de aplicações não-prioritárias	0.0 (0.001)	0.0 (0.115)
Utilização média da rede (%)		17.0 (0.265)	16.6 (0.321)

Pode-se observar na Tabela 6, que devido a ocorrência de falha no enlace 14-16, houve uma leve queda da RTeo, um aumento da perda media de pacotes para o conjunto de aplicações prioritárias e uma diminuição da utilização média da rede. A perda média de pacotes para o conjunto de aplicações não-prioritárias permanece inalterada.

5.4 Influência do aumento da vazão das aplicações prioritárias na QoS oferecida

São apresentados, nesta seção os resultados obtidos sobre a avaliação da influência do aumento da vazão das aplicações prioritárias na QoS oferecida nas topologias I e II.

5.4.1 Influência do aumento da vazão das aplicações prioritárias na QoS oferecida na topologia I

Desde que o tráfego gerado pelas aplicações prioritárias ultrapasse por algum motivo o valor reservado para a LSP, pode ocorrer congestionamento. Como conseqüência disso, a QoS oferecida às aplicações pode ser afetada. Desta forma, a função de TE deve realizar ajustes no encaminhamento das aplicações com o objetivo de tentar manter a QoS mesmo que as aplicações aumentem o seu tráfego. No caso, foram consideradas prioritárias nesta simulação uma aplicação de voz e duas de vídeo (Video0 e Vídeo1). Foram consideradas as taxas de 2 MBps e 0.35 MBps como necessidades iniciais de vazão das aplicações de Vídeo0 e Vídeo1, respectivamente. Em função das necessidades de vazão destas aplicações foram estabelecidas inicialmente as três LSP´s mostradas no Quadro 9.

QUADRO 9

LSP´s iniciais estabelecidas para as aplicações prioritárias.

Tipo de aplicação	Nó de origem	Nó de destino	Caminho da LSP
Vídeo0	0	13	8_9_10_12
Vídeo1	1	14	8_9_10_12
Voz	4	17	8_11_12

A Figura 64 mostra o encaminhamento do tráfego das aplicações prioritárias em função de suas necessidades iniciais de vazão e atraso. O tráfego das aplicações não prioritárias é encaminhado através do menor caminho (enlace 8-12).

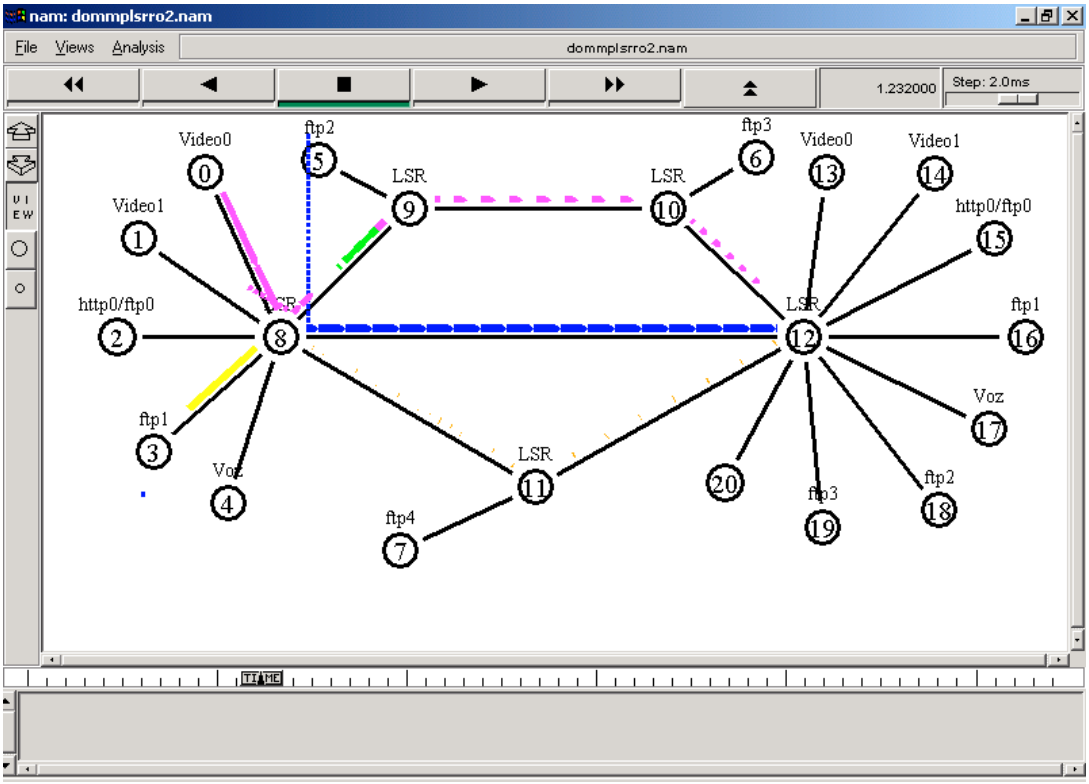

Figura 64 - Encaminhamento do tráfego das aplicações em função de suas necessidades iniciais de vazão e atraso.

Devido ao crescimento da vazão das aplicações de vídeo, a função de TE, na tentativa de ajustar o tráfego da rede, criou uma nova LSP para encaminhar o tráfego da aplicação Vídeo0, conforme é mostrado no Quadro 10.

QUADRO 10

LSP´s estabelecidas visando ajustar o tráfego da rede.

Tipo de aplicação	Nó de origem	Nó de destino	Caminho da LSP
Vídeo0	0	13	8_11_12
Vídeo1	1	14	8_9_10_12
Voz	4	17	8_11_12

A Figura 65 mostra o encaminhamento do tráfego das aplicações prioritárias através das LSP´s estabelecidas, visando ajustar o tráfego da rede em função de suas novas necessidades de vazão. O tráfego das aplicações não prioritárias continuou sendo encaminhado através do menor caminho.

A partir deste ponto o processo é estabilizado de acordo com os caminhos mostrados no Quadro 10. As Figuras 66, 67 e 68 mostram a evolução dos parâmetros de QoS das

aplicações de vídeo e voz durante um período de 40s. Durante os primeiros 4s não houve atuação da TE e o tráfego de todas as aplicações foi encaminhado pelo menor caminho.

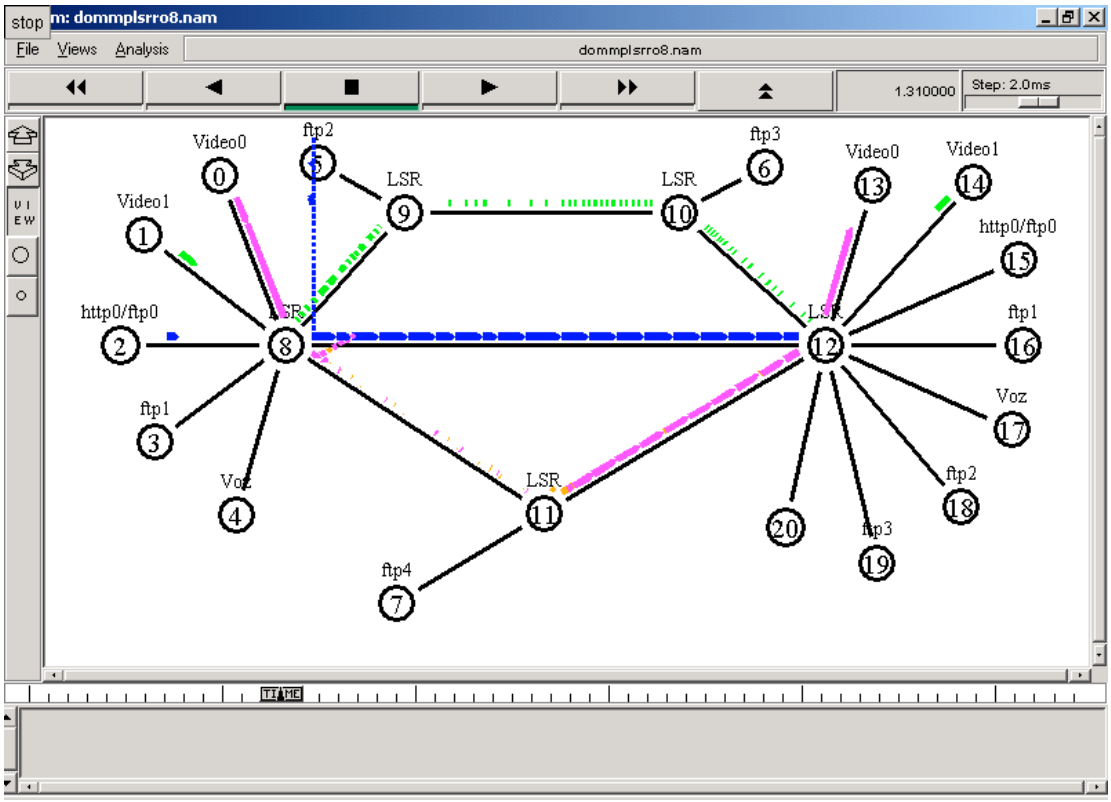

Figura 65 - Encaminhamento do tráfego das aplicações através das novas LSP´s estabelecidas visando ajustar o tráfego da rede.

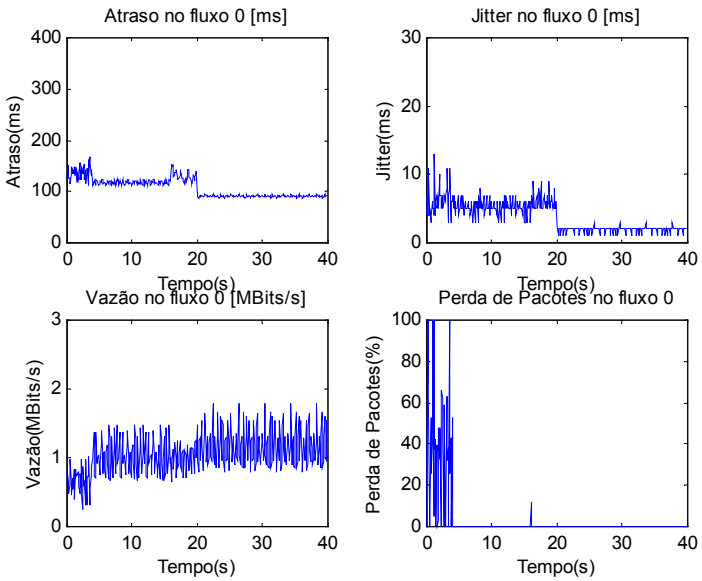

Figura 66 - Evolução da vazão, perda de pacotes, atraso e jitter para a aplicação de Vídeo0.

Conforme pode ser observado na Figura 66, durante a atuação da TE (t ≥ 4s) a QoS oferecida a aplicação de Vídeo0 sofreu uma queda em função do crescimento imprevisto da sua própria vazão e principalmente pelo aumento da vazão da aplicação de Vídeo1. Antes do aumento da vazão das aplicações, o atraso fim-a-fim ficava em torno de 123ms e o jitter em 6 ms. Durante a fase de ajustes o atraso chegou a 153ms, a perda média de pacotes chegou a 11.7 %, o jitter ficou oscilando na faixa de 6 a 8ms e a vazão caiu para 0.6 MBps. Entretanto, após a fase de ajuste efetuados pela função de TE, as necessidades da aplicação Video0 voltaram a ser atendidas. Após a estabilização e apesar do crescimento da vazão da aplicação, o atraso caiu para 90ms, a perda de pacotes caiu para zero e o jitter foi reduzido ficando na faixa de 1 a 2 ms.

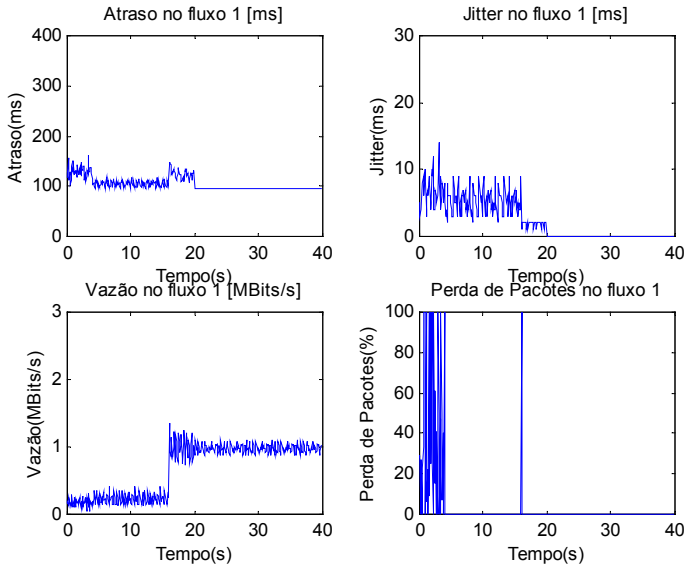

Figura 67 - Evolução da vazão, perda de pacotes, atraso e jitter para a aplicação de Vídeo1.

Pode ser observado também na Figura 67 que a partir do instante t=16s a vazão da aplicação de Vídeo1 tem um importante crescimento. Este fato acaba provocando a queda no nível de QoS oferecido as aplicações de vídeo. Isto ocorre porque as suas LSP´s foram estabelecidas inicialmente através do mesmo caminho (8-9-10-12), conforme mostra o Quadro 9, apresentado anteriormente. Antes do aumento da vazão das aplicações, o atraso fim-a-fim para a aplicação de Vídeo1 ficava em torno de 112 ms e o jitter em 6 ms. Devido ao aumento da vazão o atraso chegou a 149 ms, a perda média de pacotes chegou a 100 %, e o jitter ficou oscilando na faixa de 5 a 7 ms. Entretanto, após a fase de ajuste, as

necessidades da aplicação voltaram a ser atendidas. Assim, após a estabilização e mesmo com o crescimento da vazão das aplicações, o atraso caiu para 94 ms, e a perda de pacotes e o jitter foram reduzidos para zero. Esta melhora ocorreu pelo fato da função de ajuste ter encontrado um caminho menos utilizado.

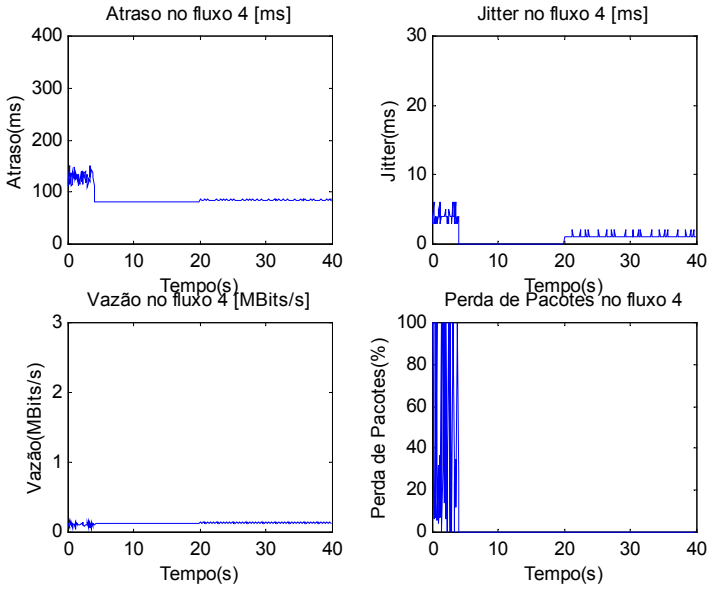

Figura 68 - Evolução da vazão, perda de pacotes, atraso e jitter para a aplicação de Voz.

Conforme pode ser observado na Figura 68, durante o período de ajustes efetuados pela função de TE, a QoS oferecida à aplicação de voz é pouco afetada. A perda de pacotes permaneceu em zero e o atraso aumentou de 82 para 84 ms. O jitter teve um leve aumento, sendo que a média ficou em 1 ms.

A partir da observação da QoS oferecida às aplicações de vídeo e voz, pode-se concluir que os ajustes efetuados pela função de TE proposta foram bem sucedidos na topologia I. Apesar do aumento da vazão das aplicações de vídeo, a QoS oferecida às aplicações voltou a ser mantida dentro de valores aceitáveis, segundo a recomendação G.1010 (ITU-T, 2002).

Com o objetivo de verificar quantitativamente o efeito da função de ajuste na perda de pacotes, foram realizadas também simulações onde o domínio MPLS foi submetido a aplicações mal comportadas, mas sem a habilitação desta função. A Tabela 7 mostra os resultados obtidos na topologia I submetida à aplicações mal comportadas para as duas versões da TE, ou seja, TE sem e com habilitação da função de ajuste.

TABELA 7

Perda média de pacotes nas duas abordagem de TE na topologia I quando as aplicações prioritárias ultrapassam a vazão estipulada.

Perda média de pacotes (%)	Domínio MPLS com TE sem função de ajuste	Domínio MPLS com TE com função de ajuste
Conjunto de aplicações prioritárias	9.5 (0.076)	0.1 (0.021)
Conjunto de aplicações não-prioritárias	0.0 (0.001)	0.0 (0.055)

Comparando-se os resultados mostrados na Tabela 7, nota-se que o desempenho da TE com função de ajuste é muito superior. No caso, obtém-se um percentual muito menor de perdas de pacotes para o conjunto de aplicações prioritárias, o que confirma a eficiência da TE com função de ajuste.

5.4.2 Influência do aumento da vazão das aplicações prioritárias na QoS oferecida na topologia II

Da mesma forma que foi feito para a topologia I, foi analisado também para a topologia II o efeito do aumento da vazão das aplicações prioritárias na QoS oferecida. No caso, foram separadas para análise duas aplicações de voz (Voz0 e Voz1) e duas de vídeo (Video0 e Vídeo1) interligadas ao LSR 17. Nas simulações efetuadas as aplicações originadas no nós 2 (Video0) e 3 (Video1) não respeitam as vazões estipuladas. Inicialmente a função de TE atuante sobre a LSR 17 criou 4 LSP´s utilizando os caminhos mostrados no Quadro 11.

QUADRO 11

LSP´s iniciais estabelecidas para as aplicações prioritárias.

Tipo de aplicação	Nó de origem	Nó de destino	Caminhos iniciais das LSP´s
Vídeo0	2	51	17_19_23_28
Vídeo1	3	50	17_18_26_27
Voz0	5	48	17_19_25_27
Voz1	6	47	17_19_25_27

Para criar as LSP´s relacionadas no Quadro 11, a função de TE levou em conta as necessidades iniciais de vazão e atraso das aplicações prioritárias apresentadas no Quadro 4. Além disso, periodicamente (nesta simulação a cada 4s) foram realizadas medições de tráfego nos enlaces do domínio MPLS. Em função do crescimento das vazões das aplicações Video0 e Vídeo1 separadas para análise, a função de TE tenta ajustar o tráfego através da rede criando novas LSP´s que sejam adequada às novas necessidades. O Quadro 12 mostra os caminhos finais utilizados pelas LSP´s quando a rede atinge a estabilização.

QUADRO 12

Caminhos utilizados pelas LSP´s após a fase de ajuste.

Tipo de aplicação	Nó de origem	Nó de destino	Caminhos finais das LSP´s
Vídeo0	2	51	17_19_23_28
Vídeo1	3	50	17_18_26_27
Voz0	5	48	17_19_25_27
Voz1	6	47	17_19_25_27

Observando o Quadro 12, nota-se que os caminhos finais atribuídos as LPS´s responsáveis pelo escoamento do tráfego das aplicações Vídeo0, Voz0 e Voz1 são os mesmos apresentados anteriormente no Quadro 11. Isto significa que o caminho inicial atribuído para escoar o tráfego da aplicação Video0 conseguiu suportar o aumento da vazão.

A Figura 69 mostra o encaminhamento do tráfego das aplicações prioritárias através das LSP´s estabelecidas com os caminhos estabilizados. O tráfego das aplicações não prioritárias continuou sendo encaminhado através do menor caminho. Deve-se observar que as aplicações prioritárias entrantes na LSR 14 mantiveram as suas taxas originais, e portanto os caminhos de suas LSP´s não sofreram ajustes.

As Figuras 70, 71, 72, e 73 mostram a evolução dos parâmetros de QoS das aplicações de Vídeo0 (fluxo 2), Vídeo1 (fluxo 3), Voz0 (fluxo 5) e Voz1 (fluxo 6) entrantes no LSR 17 durante um período de 40s. Durante os primeiros 4s ($t < 4s$) não houve atuação da TE e o tráfego de todas as aplicações foi encaminhado pelo menor caminho.

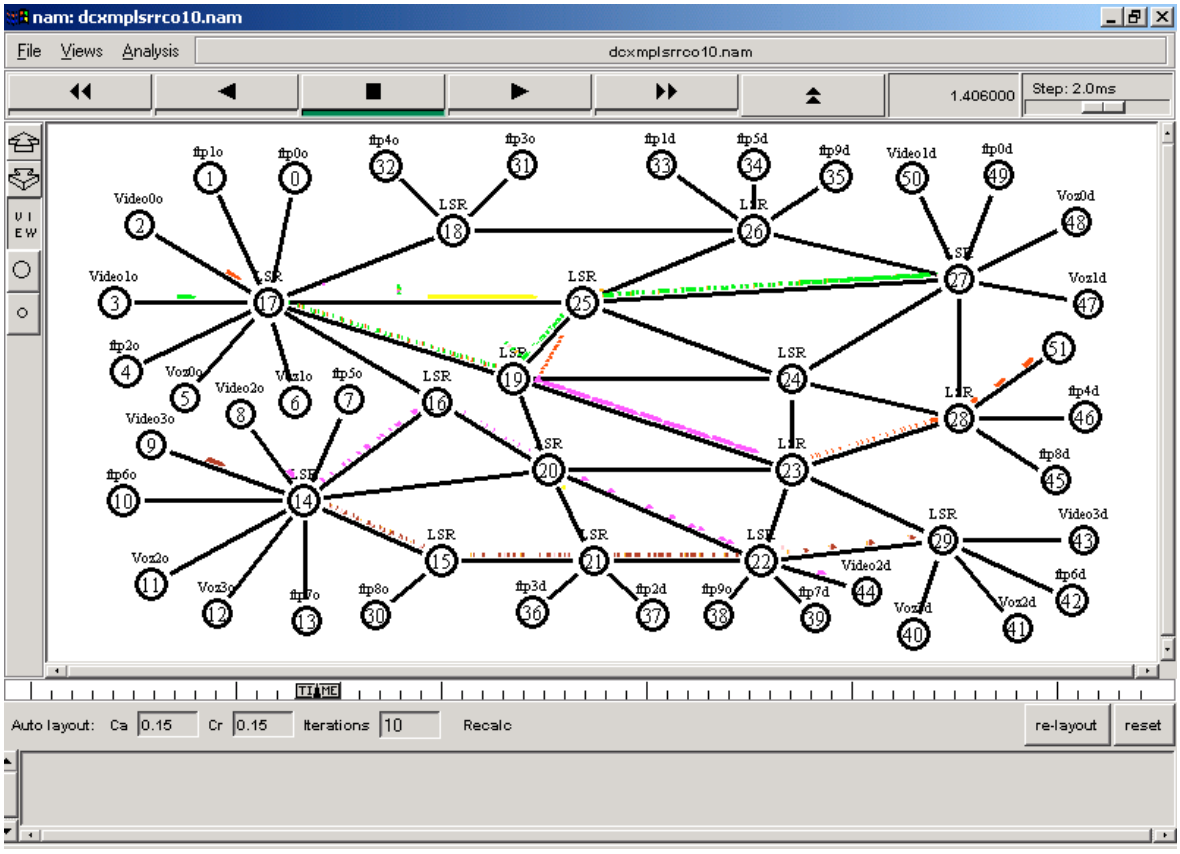

Figura 69 - Encaminhamento do tráfego das aplicações prioritárias através de LSP´s com caminhos estabilizados após a atuação da função de ajuste.

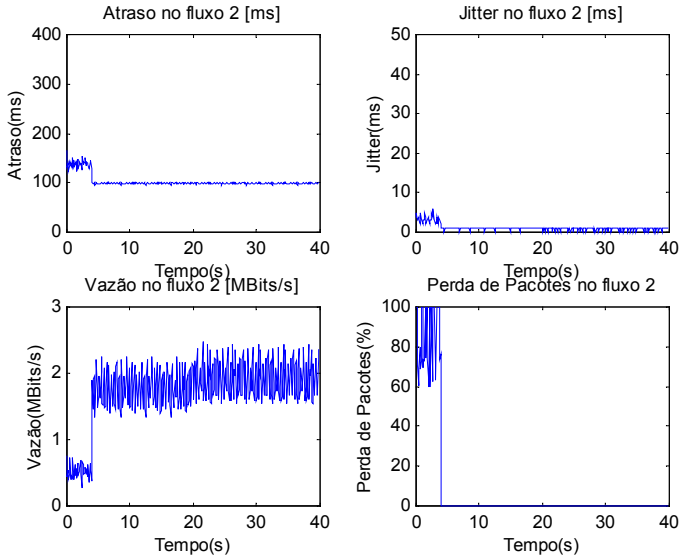

Figura 70 - Evolução da vazão, perda de pacotes, atraso e jitter para a aplicação de Vídeo0.

Conforme pode ser observado na Figura 70, durante a atuação da TE ($t \geq 4s$), a QoS oferecida a aplicação de Vídeo0 não foi afetada pelo crescimento da vazão. A perda de pacotes se manteve em zero, o atraso permaneceu em 100ms e o jitter não ultrapassou 1 ms.

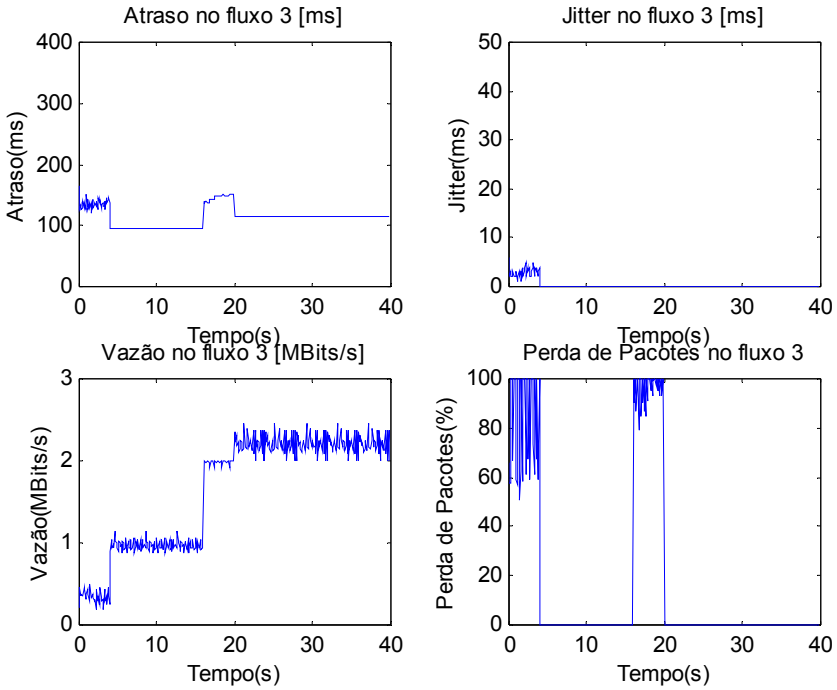

Figura 71 - Evolução da vazão, perda de pacotes, atraso e jitter para a aplicação de Vídeo1.

Observando-se a Figura 71 nota-se que, durante um período de tempo ($16s \leq t \leq 20s$), a QoS oferecida às aplicações de Vídeo1 foi afetada devido ao forte crescimento da sua vazão. Este período corresponde ao tempo necessário para detecção do aumento da vazão e a realização de ajustes (re-roteamento de LSP´S) pela função de TE. A perda de pacotes sofreu um aumento, mas após o ajuste, foi estabilizada em zero. O atraso também aumentou, mas depois voltou ao normal estabilizando-se em 114ms. O jitter permaneceu em zero ms durante o período de atuação da TE.

A QoS oferecida às aplicações de Voz0 não foi afetada pelo crescimento das vazões das aplicações de Vídeo0 e Vídeo1, conforme pode ser observado na Figura 72. A perda de pacotes se manteve em zero, o atraso permaneceu em 113ms, e o jitter permaneceu entre 0 e 1 ms.

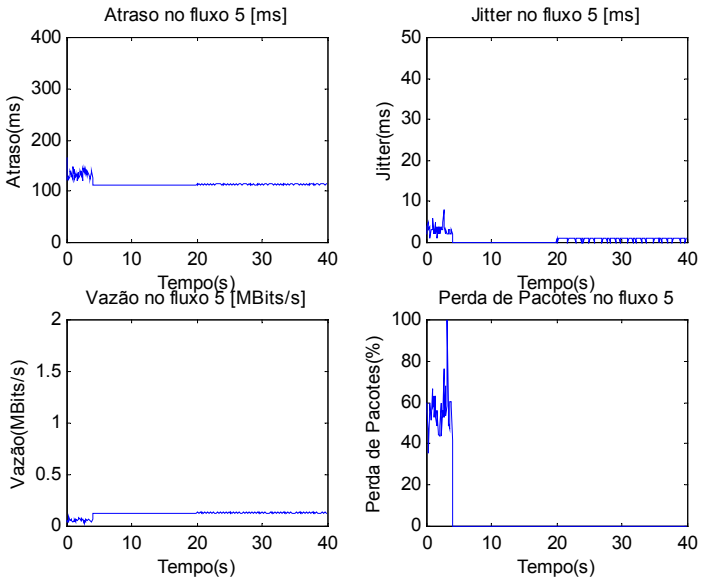

Figura 72 - Evolução da vazão, perda de pacotes, atraso e jitter para a aplicação de Voz0.

A Figura 73 mostra que a QoS oferecida à aplicação de Voz1 também não foi afetada pelos ajustes efetuados pela função de TE. A perda de pacotes se manteve próxima de zero, o atraso permaneceu em 114ms, e o jitter ficou entre 0 e 1ms durante o período de atuação da função de TE.

A partir da observação da QoS oferecida às aplicações de Vídeo0, Vídeo1, Voz0, e Voz1, pode-se concluir que os ajustes efetuados pela função de TE também foram bem sucedidos na topologia II. Apesar do aumento da vazão das aplicações de Vídeo0 e Vídeo1, após os ajustes a QoS oferecida foi mantida dentro de valores aceitáveis, segundo a recomendação G.1010 (ITU-T, 2002).

Da mesma forma que foi feito para a topologia I, foram também realizadas simulações onde o domínio MPLS foi submetido a aplicações mal comportadas, mas sem a habilitação da função de ajuste O Quadro 8 mostra os resultados obtidos na topologia II, com relação a perda média de pacotes para as duas formas de atuação da TE, ou seja, sem e com a habilitação da função de ajuste.

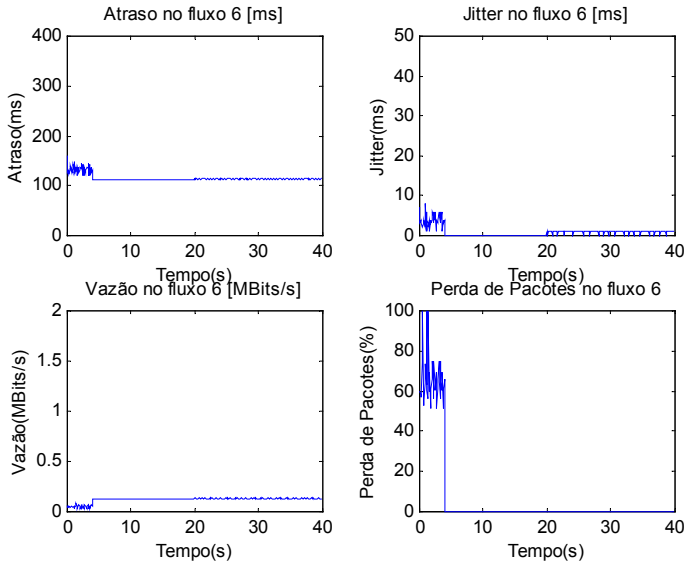

Figura 73 - Evolução da vazão, perda de pacotes, atraso e jitter para a aplicação de Voz1.

TABELA 8

Perda média de pacotes das duas abordagem de TE na topologia II quando as aplicações prioritárias ultrapassam a vazão estipulada.

Perda média de pacotes (%)	Domínio MPLS com TE sem função de ajuste	Domínio MPLS com TE com função de ajuste
Conjunto de aplicações prioritárias	6.3 (0.012)	1.4 (0.021)
Conjunto de aplicações não-prioritárias	0.0 (0.007)	0.0 (0.057)

Da mesma forma que ocorreu com a topologia I, a Tabela 8 mostra que o desempenho da TE é superior. No caso, obtém-se um percentual muito menor de perdas de pacotes para o conjunto de aplicações prioritárias, o que confirma a eficiência da TE com a função de ajuste.

5.5 Engenharia de Tráfego com Controle de Admissão de Conexão

Para verificar a eficiência da TE com CAC, neste item as simulações foram realizadas em domínios MPLS sobrecarregados. Isto significa que durante as simulações foi oferecido à rede um volume de tráfego que ultrapassa a capacidade de escoamento do domínio. Para gerar a sobrecarga foi adicionado trafego CBR às aplicações de dados ftp/http de fundo. Inicialmente a TE com CAC foi testada com aplicações comportadas. Os resultados foram comparados com os obtidos pela TE normal nas mesmas condições. Em seguida, os testes foram realizados considerando a TE com CAC, mas com aplicações não comportadas e prioridade estática. Os resultados também foram comparados com os obtidos pela TE normal. Finalmente, a TE com CAC foi testada novamente com aplicações não comportadas, mas com prioridades dinâmicas. Neste caso, as prioridades são alteradas dependendo dos comportamentos das aplicações.

5.5.1 Engenharia de Tráfego com CAC na rede sobrecarregada com aplicações comportadas

As aplicações de voz, dados e vídeo não ultrapassam os valores de vazão acordados. As simulações foram realizadas inicialmente na rede I sobrecarregada e depois também na rede II. Elas foram realizadas nas duas situações, ou seja, com a atuação da TE normal (sem o CAC) e com a atuação da TE com CAC. A Figura 74 mostra o resultado da atuação da TE normal em um domínio MPLS sobrecarregado na topologia II.

Pode-se observar na Figura 74, que vários enlaces ficaram congestionados, o que fez com que a rede não fosse capaz de escoar todo o tráfego entrante. Em função do congestionamento, não se conseguiu manter a QoS acordada para todas as aplicações prioritárias apesar da TE. Isto aconteceu porque, apesar da função de TE não ter conseguido a criação de LSP´s para todas as aplicações, o tráfego destas aplicações não é bloqueado. Assim, as aplicações que não conseguiram LSP´s continuaram a transmitir através do menor caminho, o que acaba prejudicando as outras com maior prioridade que conseguiram LSP´s. A Figura 75 mostra um exemplo desta situação, onde a QoS oferecida à aplicação prioritária Voz3 não atendeu as suas necessidades no período entre 0 e 8s

(0s ≤ t < 8s). Pode-se observar que mesmo após a implementação da TE (t ≥ 4s), continuou a existir uma alta perda de pacotes até o instante 8s. A partir do instante 8s, a função de TE obteve um caminho que possibilitou a diminuição da perda. Mas isto não foi suficiente, uma vez que o jitter oscilou entre 1 e 8 ms durante todo o período de observação, o que não atende as necessidades da aplicação de voz.

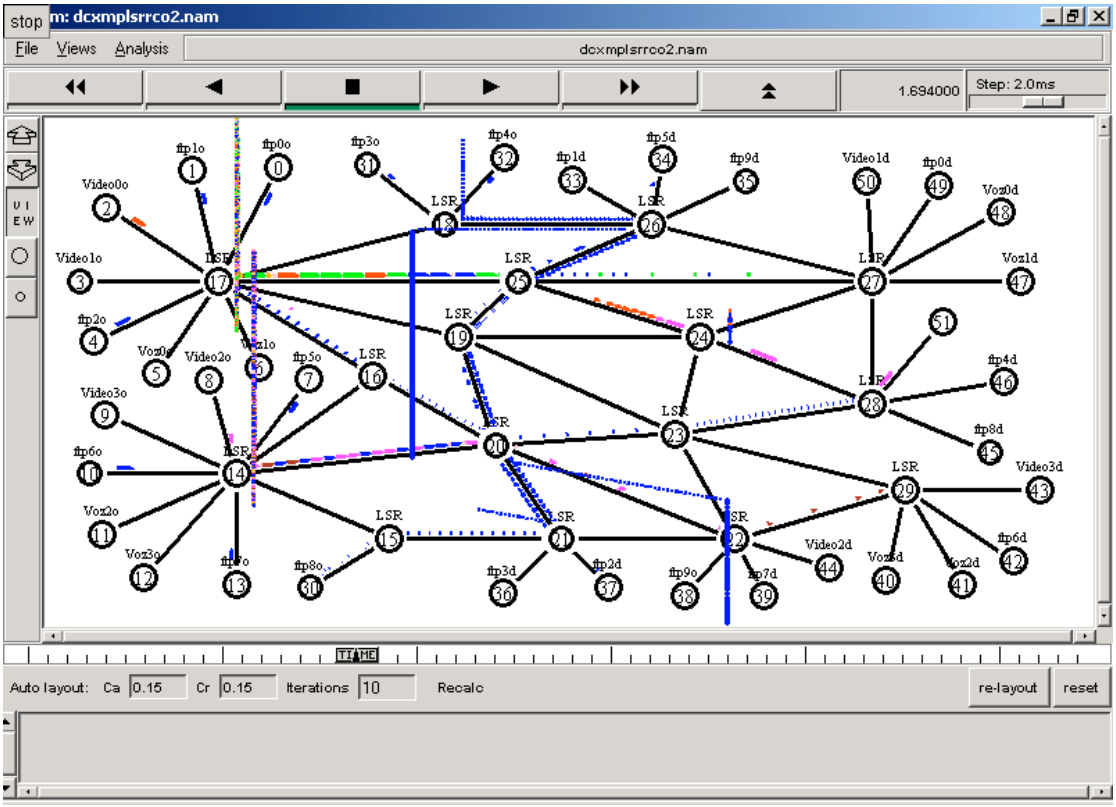

Figura 74 - Encaminhamento do tráfego em um domínio MPLS sobrecarregado utilizando TE sem CAC.

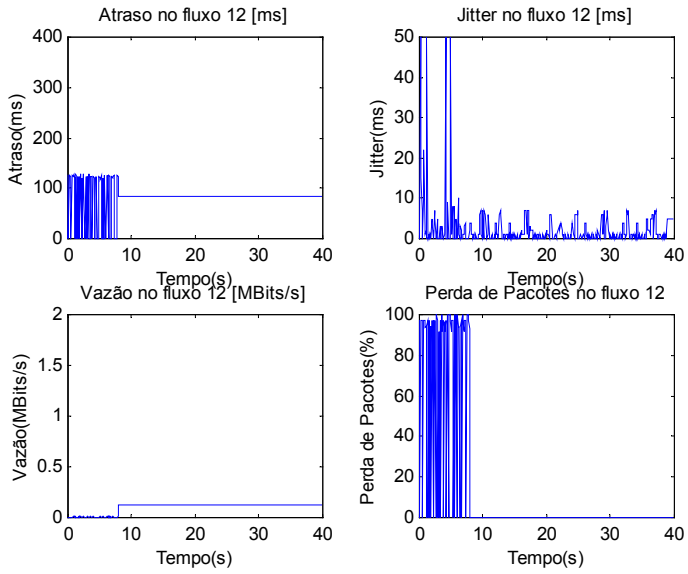

Figura 75 - Evolução da vazão, perda de pacotes, atraso e jitter para a aplicação de Voz3.

A próxima etapa foi realizar as simulações utilizando a TE com CAC. A Figura 76 mostra o resultado da atuação da TE com o CAC em um domínio MPLS sobrecarregado com topologia II.

Comparando-se os encaminhamentos de pacotes mostrados nas Figuras 74 e 76, pode-se notar que o congestionamento antes existente foi totalmente eliminado com a utilização da TE com CAC. Isto aconteceu porque o tráfego das aplicações que não conseguiram LSP´s foi bloqueado pelo CAC. Desta forma, as aplicações com menor prioridade que não conseguiram LSP´s, simplesmente são impedidas de encaminhar os seus pacotes através do menor caminho. Em função disso, apesar da sobrecarga de tráfego, conseguiu-se manter a QoS acordada para todas as aplicações que obtiveram LSP´s.

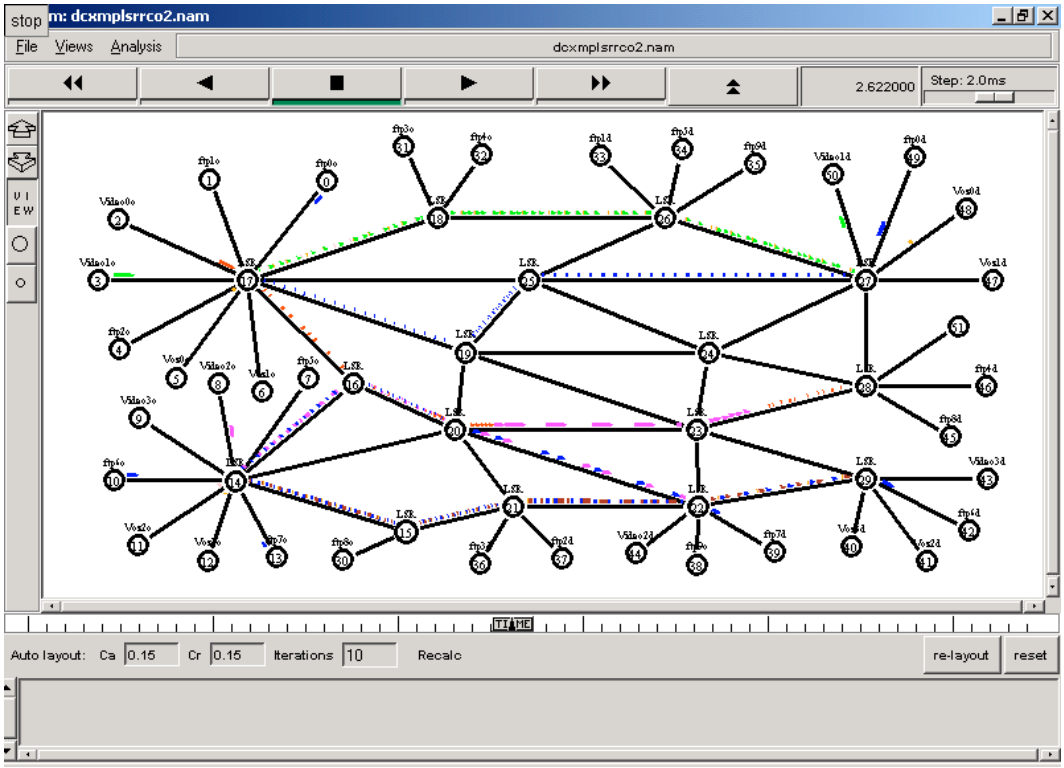

Figura 76 - Encaminhamento do tráfego em um domínio MPLS sobrecarregado utilizando TE com CAC.

A Figura 77 mostra um exemplo desta situação, onde a QoS oferecida à aplicação de Voz3 agora atende as suas necessidades. Os resultados apresentados na Figura 77 mostram que as necessidades da aplicação de Voz3 foram atendidas. Com a atuação da TE com CAC, a vazão se manteve dentro dos valores desejados (128 kbps) e a perda de pacotes se manteve em 0%. Além disso, o atraso fim-a-fim foi de 83 ms e o jitter variou entre 0 e 1ms, o que atende perfeitamente as necessidades da aplicação (ITU-T, 2002).

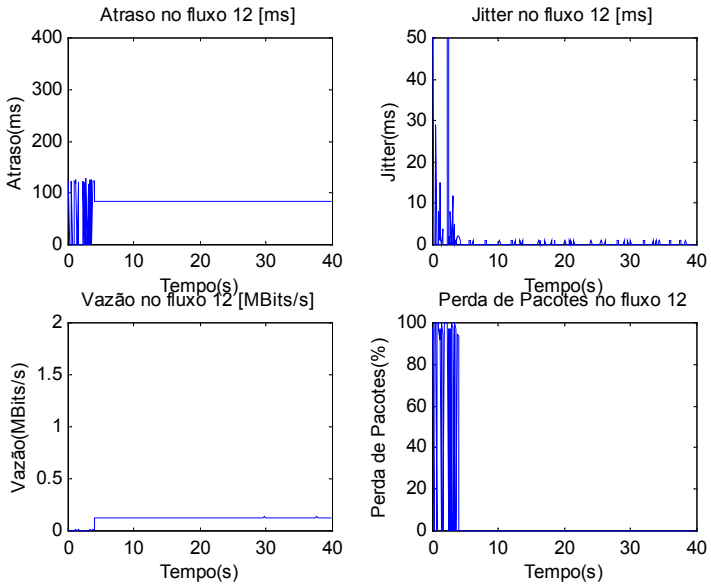

Figura 77 - Evolução da vazão, perda de pacotes, atraso e jitter para a aplicação de Voz3.

A Tabela 9 apresenta um resumo dos resultados obtidos para as topologias I e II nas duas situações, ou seja, com TE sem e com CAC.

TABELA 9

Efeito da TE com CAC nas redes I e II sobrecarregadas e com aplicações comportadas.

Indicadores de desempenho	Domínio MPLS Com TE normal		Domínio MPLS com TE e CAC	
	I	II	I	II
RTeo	0.941 (0.002)	0.690 (0.003)	1.000 (0.002)	1.000 (0.003)
Perda média de pacotes (%)	8.71 (0.040)	37.9 (0.058)	0.0 (0.067)	0.0 (0.069)
Utilização média da rede (%)	46 (0.153)	41.3 (0.572)	13.7 (0.153)	24.5 (0.497)

Os resultados da Tabela 9 mostram que a TE com CAC é mais eficiente para efetuar escoamento do tráfego das aplicações em uma rede sobrecarregada. No caso da TE com CAC, o tráfego autorizado foi totalmente escoado. Já no caso da utilização média da rede, a TE normal obteve um valor melhor. Isto acontece porque, apesar da função de TE não ter conseguido a criação de LSP´s para algumas aplicações, o tráfego destas aplicações não foi

bloqueado. Este fato acaba contribuindo para melhorar a utilização da rede, uma vez que as aplicações continuam a transmitir os seus pacotes através do menor caminho, mesmo que ocorra perda. Por outro lado, o uso da TE com CAC possibilitou a redução do valor da perda media de pacotes obtida pelo conjunto de aplicações para zero. Portanto, a utilização média da rede quando se utiliza a TE com CAC é menor, mas em compensação a QoS oferecida às aplicações é superior.

5.5.2 Engenharia de Tráfego com CAC na rede sobrecarregada com aplicações não comportadas e com prioridade estática

A função de TE com CAC deve realizar ajustes no encaminhamento das aplicações com o objetivo de tentar manter a QoS mesmo que as aplicações aumentem o seu tráfego. Se este ajuste não for possível a aplicação deve ser forçada a diminuir a sua vazão. Em caso contrário, o aumento da vazão da aplicação ofensora é autorizado. Neste item, as simulações foram realizadas inicialmente na topologia I e depois na II. Foram realizados testes com a atuação da TE normal (sem o CAC) e depois com a atuação da TE com CAC. As prioridades das aplicações permaneceram inalteradas durante todo o período de simulação.

As Figuras 78 e 79 mostram os resultados da atuação da TE com CAC na topologia II, onde aplicações ofensoras de vídeo (Vídeo0 e Video1) aumentam as suas vazões.

Conforme pode ser observado na Figura 78, o aumento da vazão da aplicação ofensora Video0 foi autorizado pela função de TE com CAC. Desta forma durante a atuação da TE com CAC a QoS oferecida a aplicação de Vídeo0 foi mantida apesar do aumento da sua vazão.

No caso da aplicação Vídeo1, o aumento da sua vazão de tráfego não foi autorizado conforme mostra a Figura 79. Pode-se notar que o aumento da vazão da aplicação provocou a elevação da perda de pacotes de 0% para 100% e o atraso fim-a-fim de 95 ms para 125 ms. A função de TE com CAC, após detectar o aumento da vazão da aplicação, tenta encontrar um novo caminho que consiga atender as novas exigências. Como um caminho com capacidade suficiente não foi encontrado, o aumento da vazão foi

desautorizado e a aplicação foi forçada a diminuir a vazão. Como resultado, a perda de pacotes voltou a 0% e o atraso fim-a-fim caiu de 125 ms para 95 ms.

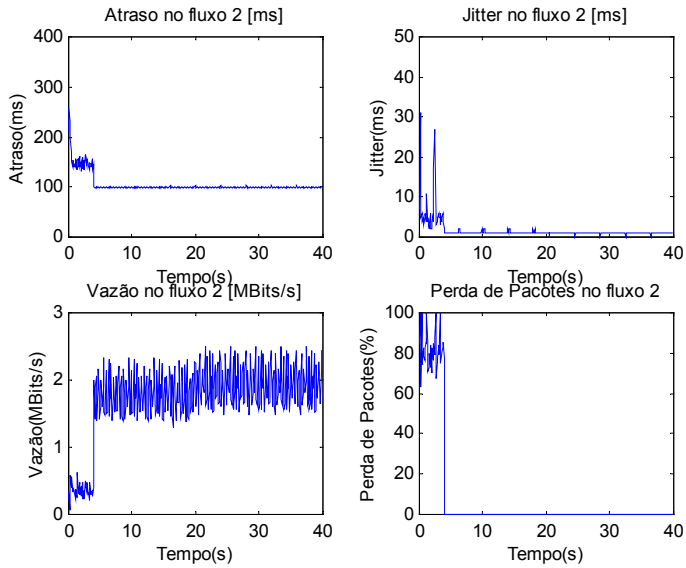

Figura 78 - Evolução da vazão, perda de pacotes, atraso e jitter para a aplicação de Vídeo0.

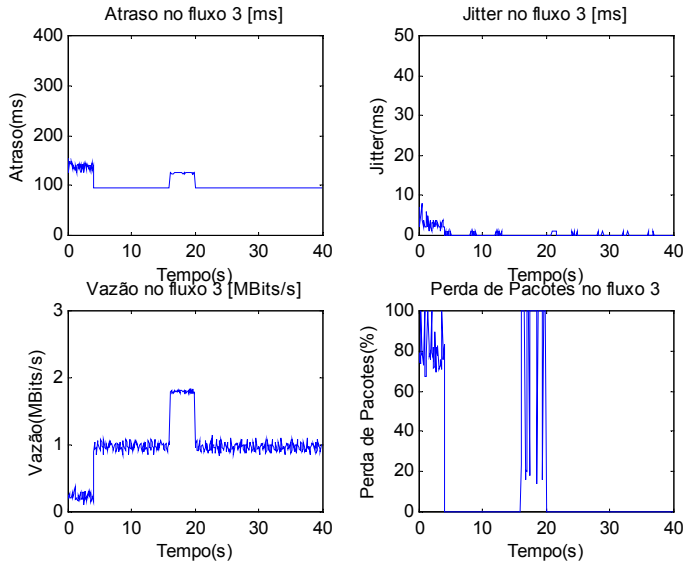

Figura 79 - Evolução da vazão, perda de pacotes, atraso e jitter para a aplicação de Vídeo1.

A Tabela 10 apresenta um resumo dos resultados obtidos para as topologias sobrecarregadas submetidas a aplicações de vídeo não comportadas com prioridade estática nas duas situações, ou seja, com TE sem e com CAC.

TABELA 10

Efeito da TE com CAC nas redes I e II sobrecarregadas e com aplicações não comportadas com prioridade estática.

Indicadores de desempenho	Domínio MPLS com TE normal		Domínio MPLS com TE e CAC	
	I	II	I	II
RTeo	0.941 (0.002)	0.693 (0.003)	1.000 (0.003)	0.997 (0.002)
Perda média de pacotes (%)	8.71 (0.040)	37.9 (0.058)	0.0 (0.072)	0.70 (0.075)
Utilização média da rede (%)	46 (0.153)	41.3 (0.572)	13.7 (0.153)	24.5 (0.497)

Os resultados da Tabela 10 mostram que a RTeo obtida para a TE com CAC é superior aquela obtida para a TE sem CAC. A utilização média da rede nas topologias I e II submetidas às aplicações não comportadas é maior quando se utiliza a TE sem CAC. Por outro lado, a TE com CAC obtém um valor muito menor de perda média de pacotes para o conjunto de aplicações. No caso da TE sem CAC nas topologias I e II, a perda média de pacotes obtida para o conjunto de aplicações não atende as metas de QoS apresentadas na recomendação G.1010 do ITU-T para aplicações de voz (< 3%) e vídeo (< 1%). No caso da topologia I, o intervalo de confiança obtido para a perda média de pacotes da TE sem CAC foi 1.332. Neste caso, o limite superior obtido para a perda média de pacotes foi 10 % e o limite superior foi 7.37%, o que nas duas situações ultrapassam o valor estipulado pela recomendação G.1010. Para a topologia II, o intervalo de confiança obtido para a perda média de pacotes da TE sem CAC foi 1.121. Neste caso, o limite superior obtido para a perda média de pacotes foi 39.02 % e o limite inferior foi 36.77%, o que nas duas situações, como ocorreu com a topologia I, ultrapassam o valor estipulado pela recomendação G.1010. Por outro lado, os resultados de perda média de pacotes obtidos com a utilização da TE com CAC atendem as metas de QoS apresentadas na recomendação G.1010 do ITU-T para aplicações de voz e vídeo. Na topologia I o intervalo de confiança obtido para a perda média de pacotes foi zero. Para a topologia II, o intervalo de confiança foi 0.22. Neste caso, o limite superior obtido para a perda média de pacotes foi 0.92 % e o limite inferior foi 0.28%.

5.5.3 Engenharia de Tráfego com CAC na rede sobrecarregada com aplicações não comportadas e com prioridade dinâmica

Nesta seção, as prioridades originais das aplicações podem ser alteradas pela TE com CAC dependendo da análise dos seus históricos de comportamento. As aplicações comportadas, ou seja, que obedecem aos valores de vazão acordados são favorecidas com o aumento da prioridade. As aplicações não comportadas, ao contrário, têm as suas prioridades diminuídas. Pode acontecer que uma aplicação comportada que possuía menor prioridade acabe, em função do aumento da sua prioridade, ultrapassando uma aplicação não comportada de prioridade original maior. Caso os recursos disponíveis da rede não sejam suficientes para atender a todos, a aplicações que possuem menores prioridades não são atendidas. Os testes foram realizados no domínio MPLS considerando as duas situações, ou seja, a atuação da TE com CAC com prioridade estática e TE com CAC com prioridade dinâmica.

Da mesma forma que ocorreu no domínio MPLS com topologia II e TE com CAC com prioridade estática, as aplicações ofensoras Vídeo0 e Vídeo1 aumentaram as suas vazões acima dos valores acordados. Os resultados obtidos com TE com CAC com prioridade dinâmica foram exatamente os mesmos obtidos com o uso da TE com CAC e prioridade estática mostrados anteriormente nas Figuras 78 e 79. O aumento da vazão da aplicação ofensora Video0 foi autorizado pela função de TE com CAC, sendo que a QoS oferecida foi mantida apesar do aumento da sua vazão. Já no caso da aplicação Vídeo1, o aumento da sua vazão de tráfego não foi autorizado.

A Tabela 11 apresenta um resumo dos resultados obtidos para as topologias I e II sobrecarregadas submetidas a aplicações de vídeo não comportadas com prioridade dinâmica nas duas situações, ou seja, com TE sem e com CAC. Os resultados mostram que os valores de RTeo obtidos para as duas abordagens são semelhantes, com uma pequena superioridade, no caso da topologia II, para a TE com CAC com prioridade dinâmica. A TE com CAC com prioridade dinâmica possibilitou o escoamento de praticamente todo o tráfego entrante no domínio MPLS.

TABELA 11

Efeito da TE com CAC nas redes I e II sobrecarregadas e com aplicações não comportadas com prioridades estática e dinâmica.

Indicadores de desempenho	Domínio MPLS com TE e CAC prioridade estática		Domínio MPLS com TE e CAC prioridade dinâmica	
	I	II	I	II
RTeo	1.000 (0.003)	0.997 (0.002)	1.000 (0.003)	0.998 (0.002)
Perda média de pacotes (%)	0.0 (0.072)	0.70 (0.075)	0.0 (0.069)	0.60 (0.100)
Utilização média da rede (%)	13.7 (0.153)	24.5 (0.497)	13.7 (0.153)	24.5 (0.115)

Além disso, para as duas abordagens, os valores obtidos para a perda média de pacotes obtida para o conjunto de aplicações atendem as metas de QoS apresentadas na recomendação G.1010 do ITU-T para aplicações de voz (< 3%) e vídeo (< 1%). Por outro lado, a perda média de pacotes é menor quando é utilizada a TE com CAC com prioridade dinâmica. Com relação ao percentual de utilização média da rede os valores obtidos são os mesmos.

5.6 Engenharia de Tráfego com Algoritmo de Otimização dos recursos da rede

A otimização dos recursos da rede deve ser feita periodicamente. A meta da otimização é fazer com que as LSP´s que atendem as aplicações, utilizem sempre os melhores caminhos possíveis. Assim, um caminho que num determinado momento é escolhido como a melhor opção para o estabelecimento de uma LSP, pode deixar de sê-lo dependendo das novas condições da rede.

Com o objetivo de verificar o efeito da função de otimização na rede foram realizadas simulações nas topologias I e II. O Quadro 13 mostra os caminhos utilizados pelas LSP´s das aplicações antes e após a otimização na topologia II. A seqüência apresentada está de acordo com o ordenamento estabelecido pela TE a partir da prioridade de atendimento das aplicações. A prioridade de atendimento foi estabelecida em função do histórico de comportamento.

QUADRO 13

Caminhos utilizados pelas LSP´s antes e após a otimização na topologia II.

Aplicação	Origem	Destino	Caminhos das LSP´s antes da otimização	Caminhos das LSP´s após a otimização
Voz2	11	41	14_15_21_22_29	14_15_21_22_29
Voz3	12	40	14_15_21_22_29	14_20_23_29
Voz0	5	48	17_19_25_27	17_18_26_27
Voz1	6	47	17_19_25_27	17_19_25_27
Vídeo3	9	43	14_15_21_22_29	14_15_21_22_29
Vídeo2	8	44	14_16_20_22	14_16_20_22
Vídeo0	2	51	17_16_20_23_28	17_19_23_28
Vídeo1	3	50	17_18_26_27	17_18_26_27
Dados8	30	45	15_21_20_23_28	15_21_22_20_23_28
Dados7	13	39	14_15_21_22	14_15_21_22
Dados0	0	49	17_19_25_27	17_19_25_27
Dados6	10	42	14_15_21_22_29	14_15_21_22_29

Observando-se o Quadro 13 pode-se notar que os caminhos das LSP´s de algumas aplicações foram modificados com a otimização. Dentre estas, pode-se ressaltar como exemplo as aplicações Voz0 e Voz3. As Figuras 80 e 81 apresentam a vazão, perda, atraso e jitter das aplicações Voz0 e Voz3 durante as simulações.

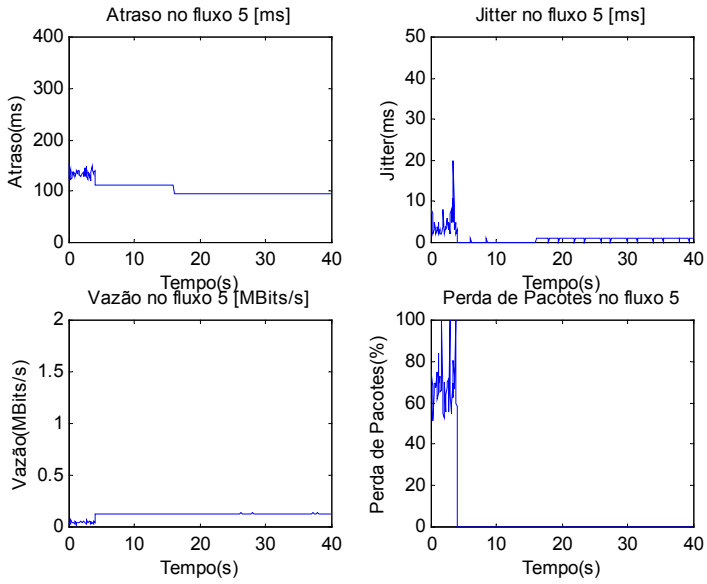

Figura 80 – Evolução da vazão, perda de pacotes, atraso e jitter para a aplicação de Voz0 antes e após a otimização.

Os resultados apresentados na Figura 80 mostram que a QoS oferecida a aplicação Voz0 é mantida durante a atuação da TE, ou seja, no período compreendido entre 4 e 40 s.

Além disso, deve-se ressaltar que após a otimização, instante 16 s, o atraso fim-a-fim é diminuído de 112 ms para 94 ms. Isto ocorre devido à escolha pela função de otimização de um caminho mais curto em termos de custo para atender a aplicação Voz0. O mesmo resultado pode ser observado na Figura 81 para a aplicação Voz3. Após a otimização, instante 16 s, o atraso fim-a-fim da aplicação Voz3 é diminuído de 83 ms para 63 ms.

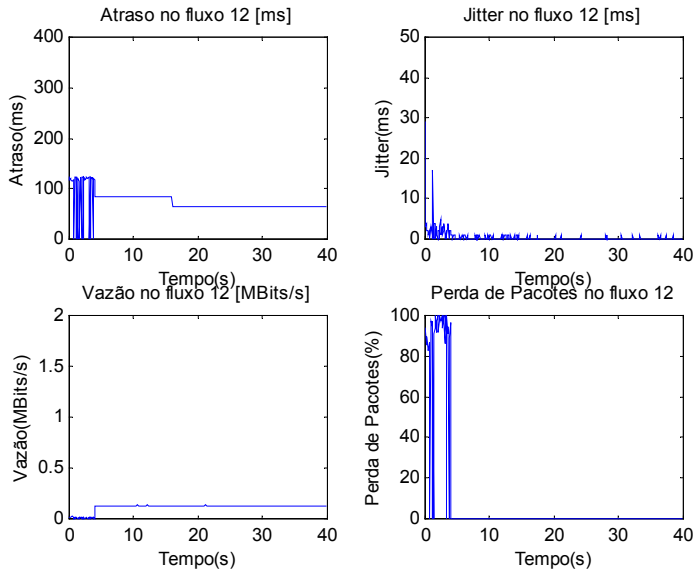

Figura 81 – Evolução da vazão, perda de pacotes, atraso e jitter para a aplicação de Voz3 antes e após a otimização.

A Tabela 12 apresenta um resumo dos resultados obtidos para as topologias sobrecarregadas submetidas a aplicações de vídeo comportadas com prioridade dinâmica nas duas situações, ou seja, com TE com **CAC** sem e com otimização. Os resultados mostram que a RTeo e a perda média de pacotes não sofreram alteração quando foi realizada a otimização. Quando foi implementada a TE com otimização observou-se que a utilização média da rede foi ligeiramente superior a TE sem otimização. Na topologia I a média dos atrasos fim-a-fim das aplicações foi de 76.4 ms quando foi feita a otimização. Sem a otimização a média dos atrasos fim-a-fim das aplicações subiu para 77 ms.

Na topologia II a média dos atrasos fim-a-fim das aplicações quando se utiliza TE com otimização foi de 91 ms. Sem a otimização a media observada foi 93 ms. Esta menor média de atrasos fim-a-fim aconteceu por que o módulo de AG responsável pela otimização escolheu os caminhos mais curtos possíveis para o conjunto de aplicações.

Assim, pode-se concluir que os resultados obtidos quando se utiliza TE com otimização são ligeiramente superiores, conforme esperado.

TABELA 12

Efeito da TE sem e com otimização nas topologias I e II.

Indicadores de desempenho	Domínio MPLS com TE sem otimização		Domínio MPLS com TE com otimização	
	I	II	I	II
RTeo	1.000 (0.002)	1.000 (0.003)	1.000 (0.002)	1.000 (0.003)
Perda média de pacotes (%)	0.0 (0.067)	0.0 (0.069)	0.0 (0.069)	0.0 (0.070)
Utilização média da rede (%)	13.7 (0.153)	24.5 (0.497)	14.7 (0.153)	24.8 (0.200)
Atraso médio fim-a-fim das aplicações (ms)	77 (0.193)	93 (0.306)	76.4 (0.108)	91 (0.351)

5.7 Engenharia de Tráfego com Algoritmo de Identificação de Enlaces Críticos

São apresentados nesta seção os resultados obtidos durante a avaliação da TE com algoritmo de identificação de enlaces críticos nas topologias I e II.

5.7.1 Engenharia de Tráfego com Algoritmo de Identificação de Enlaces Críticos na topologia I

Os testes realizados na topologia I duraram 40s sendo que abrangeram vinte simulações de 2s de duração cada uma. Durante as cinco primeiras simulações ($0 \leq t < 10s$), a TE com CAC não estabeleceu LSP´s para todas as aplicações devido a diminuição proposital da capacidade de escoamento do tráfego do domínio MPLS. As larguras de banda de alguns enlaces foram configuradas para forçar a geração de gargalos ou enlaces críticos, impedindo a atendimento das necessidades de tráfego de todas as aplicações. As aplicações que não obtiveram LSP´s foram bloqueadas. A cinco primeiras simulações foram utilizadas apenas para gerar o histórico de medições de tráfego.

Após a obtenção do histórico de medições de tráfego nos enlaces do domínio MPLS, o **AIEC** é ativado na sexta simulação. Com a ativação do **AIEC**, são feitas a identificação dos enlaces críticos, as previsões de vazões futuras e depois as ampliações das respectivas

larguras de banda, quando necessário. Na seqüência, a TE com CAC verifica a possibilidade de criação de LSP´s para as aplicações que estavam bloqueadas. A partir deste ponto as simulações continuam normalmente até o final.

Uma das aplicações que foram penalizadas com o bloqueio do seu tráfego nas 5 primeiras simulações foi a aplicação Video0. Isto ocorreu em função da impossibilidade de estabelecimento de uma LSP que atendesse as suas necessidades de QoS. O **AIEC** foi ativado na sexta simulação, sendo que em seguida, identificou o enlace 8-9 como crítico e depois ativou a rede MLP. A rede MLP, com a utilização de um conjunto de amostras de medições passadas de tráfego armazenadas no arquivo de histórico, foi treinada para tentar prever o tráfego futuro no enlace crítico identificado. As amostras com duração de 0.6 segundos cada uma, foram retiradas das cinco primeiras simulações. O tempo total do conjunto de cinco amostras foi 3 segundos. A utilização de amostras com duração menor que o período total de cada simulação foi realizada para facilitar o treinamento da MLP. Durante os testes foi observado que o uso de amostras com longo período de duração dificultava a convergência da MLP. A Figura 82 mostra a evolução do tráfego real e previsto pela MLP no enlace crítico 8-9.

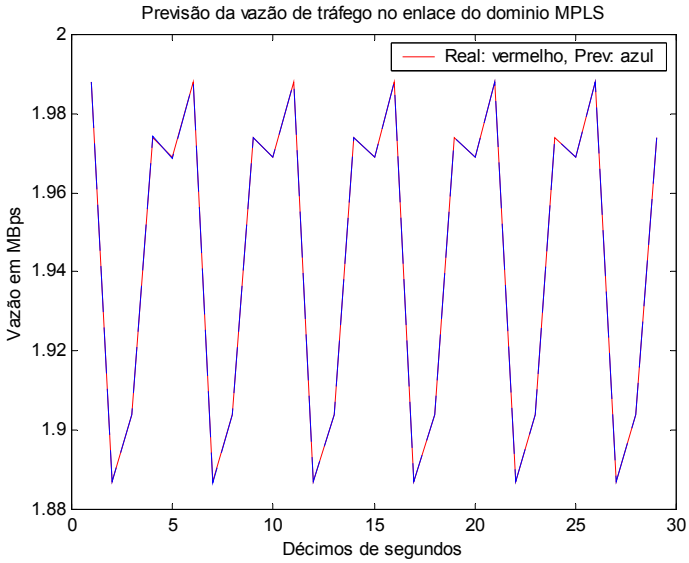

Figura 82 – Evolução da vazão de tráfego real e prevista pela MLP no enlace 8-9.

Observando-se a Figura 82, pode-se notar que a vazão real e a vazão prevista no enlace crítico 8-9 são praticamente coincidentes. Isto mostra que o uso de amostras de medições facilitou o treinamento ($0 \leq t < 16$ décimos de segundos) da MLP e a predição

(16 ≤ t< 30 décimos de segundos) realizada. Com o objetivo de checar os resultados, a previsão realizada pela MLP foi comparada com a do preditor Naive. A Tabela 13 apresenta os resultados do Erro Médio Absoluto (EMA) obtidos durante a fases de treinamento, testes e global para a MLP e para o preditor Naive.

TABELA 13

Erros Médios Absolutos obtidos pela MLP e preditor naive no enlace 8-9.

Preditor	EMA (%)		
	Treinamento	Testes	Global
MLP	0.0031	0.0033	0.0032
Naive	2.0498	4.8289	2.2415

Observando-se a Tabela 13, nota-se que os resultados obtidos com o uso da MLP são superiores ao do preditor Naive conforme era esperado.

Baseando-se no resultados previstos pela MLP para a vazão de tráfego no enlace 8-9, o AIEC identificou 1.99 MBps como o maior valor. Como o maior valor se aproxima do valor da largura de banda (2 MBps), é providenciada a ampliação. A largura de banda do enlace crítico 8-9 foi ampliada de 2 para 4 MBps. O critério utilizado para a ampliação foi a duplicação da capacidade atual. A ampliação tornou possível a autorização pelo CAC da criação de uma LSP para a aplicação de Vídeo0. A Figura 83 mostra a evolução dos parâmetros de QoS da aplicação de Vídeo0 durante 40s de simulação.

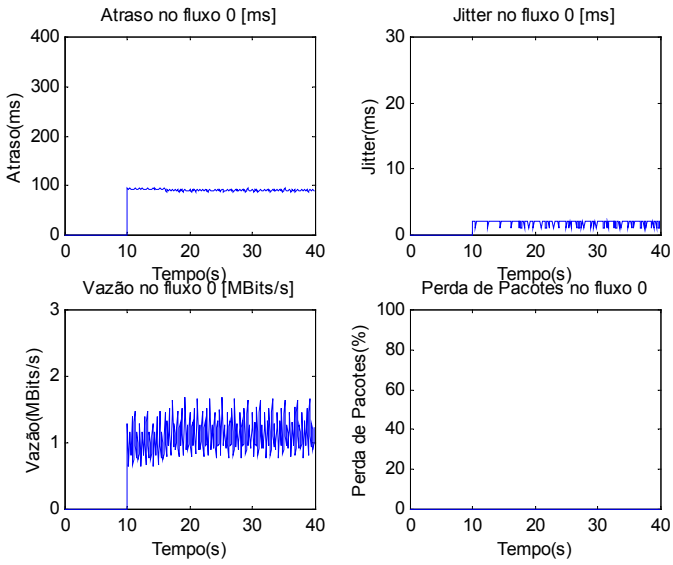

Figura 83 - Evolução da vazão, perda de pacotes, atraso e jitter para a aplicação de Vídeo0.

Conforme pode ser observado na Figura 83, durante o intervalo $0 \leq t < 10s$ correspondente às cinco simulações iniciais, a aplicação Video0 foi impedida de transmitir pacotes devido a ação do CAC. Após a criação da LSP, a aplicação é autorizada a transmitir ($10 \leq t \leq 40s$). Conforme era esperado, a QoS oferecida às aplicações de Vídeo0 se mantêm em valores aceitáveis, segundo a recomendação G.1010 (ITU-T, 2002).

5.7.2 Engenharia de Tráfego com Algoritmo de Identificação de Enlaces Críticos na topologia II

Os testes realizados na topologia II também abrangeram, como na topologia I, vinte simulações de 2 segundos de duração cada uma. As cinco primeiras simulações ($0 \leq t < 10s$) também foram utilizadas para gerar o histórico de medições de tráfego. Após a obtenção do histórico de medições de tráfego nos enlaces do domínio MPLS, o AIEC também é ativado na sexta simulação. Com a ativação do AIEC, são realizadas a identificação dos enlaces críticos, as previsões de vazões futuras e depois as ampliações das respectivas largura de banda, quando necessário. Na seqüência, a TE com CAC verifica a possibilidade de criação de LSP´s para as aplicações que estavam bloqueadas. A partir deste ponto as simulações continuam normalmente até o final.

Durante as primeiras cinco primeiras simulações na topologia II, a TE com CAC não estabeleceu LSP´s para todas as aplicações. Isto ocorreu em função da diminuição artificial da capacidade escoamento de tráfego do domínio MPLS. Assim, visando a verificação do funcionamento do AIEC, as larguras de banda de alguns enlaces foram configuradas para forçar a geração de gargalos ou enlaces críticos no domínio, impedindo a atendimento das necessidades de tráfego de todas as aplicações. As aplicações que não obtiveram LSP´s foram bloqueadas.

Nas cinco primeiras simulações ($0 \leq t < 10s$), a aplicação interligada ao nó 1 (CBR/FTP1) foi penalizada com o bloqueio do seu tráfego, em função da impossibilidade de estabelecimento de uma LSP. As simulações continuaram com a ativação do AIEC na sexta simulação. Em seguida, o AIEC identificou o enlace 17-18 como crítico e depois ativou a MLP. A rede MLP, com a utilização de um conjunto de amostras de medições passadas de tráfego armazenadas no arquivo de histórico, foi treinada para tentar prever o tráfego futuro no enlace crítico identificado. As amostras com duração de 0.6 segundos

cada uma, foram retiradas das cinco primeiras simulações. O tempo total do conjunto de cinco amostras foi 3 segundos. A Figura 84 mostra a evolução do tráfego real e previsto pela MLP no enlace crítico 17-18.

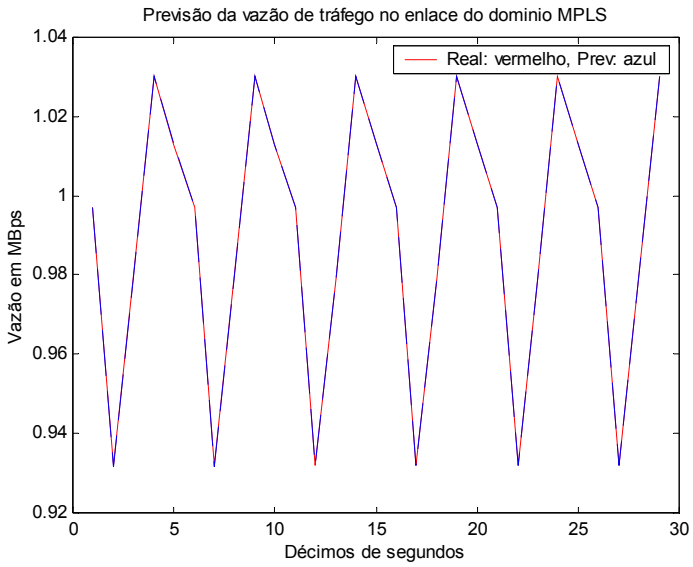

Figura 84 – Evolução da vazão de tráfego real e prevista pela MLP no enlace 17-18.

Da mesma forma que ocorreu com a previsão realizada no enlace crítico 8-9 da topologia I mostrada na Figura 82, pode-se notar na Figura 84, que a vazão real e a vazão prevista no enlace crítico 17-18 da topologia II são praticamente coincidentes. Isto mostra que o uso de amostras de medições também facilitou o treinamento ($0 \leq t < 16$ décimos de segundos) da MLP e a predição ($16 \leq t < 30$ décimos de segundos) realizada na topologia II.

Com o objetivo de checar os resultados, as previsões realizadas pela MLP e pelo preditor Naive foram também comparadas. A Tabela 14 apresenta os resultados do Erro Médio Absoluto (EMA) obtidos durante a fases de treinamento, testes e global para a MLP e para o preditor Naive.

TABELA 14

Erros Médios Absolutos obtidos pela MLP e preditor naive no enlace 17-18.

Preditor	EMA (%)		
	Treinamento	Testes	Global
MLP	0.0093	0.0095	0.0094
Naive	3.7625	8.3859	4.0814

O AIEC identificou 1.03 MBps como o maior valor de vazão de tráfego previsto pela MLP no enlace crítico 17-18. Como o maior valor previsto para a vazão se aproxima do valor da largura de banda (1.256 MBps) do enlace, o AIEC providencia a sua ampliação. A largura de banda do enlace crítico 17-18 foi ampliada de 1.256 para 2.512 MBps. O critério utilizado na ampliação das larguras de banda do enlace crítico foi a duplicação da sua capacidades atual.

A ampliação da largura de banda dos enlace 17-18 torna possível a autorização pelo CAC da criação de uma LSP para as aplicações originadas no nó 1. O tráfego originado no nó 1 combina CBR e FTP e foi gerado inicialmente apenas para prover tráfego de fundo, sendo que não possui maiores exigências de QoS. A Figura 85 mostra a evolução dos parâmetros de QoS das aplicações CBR/FTP1 durante um período de 40s de simulação.

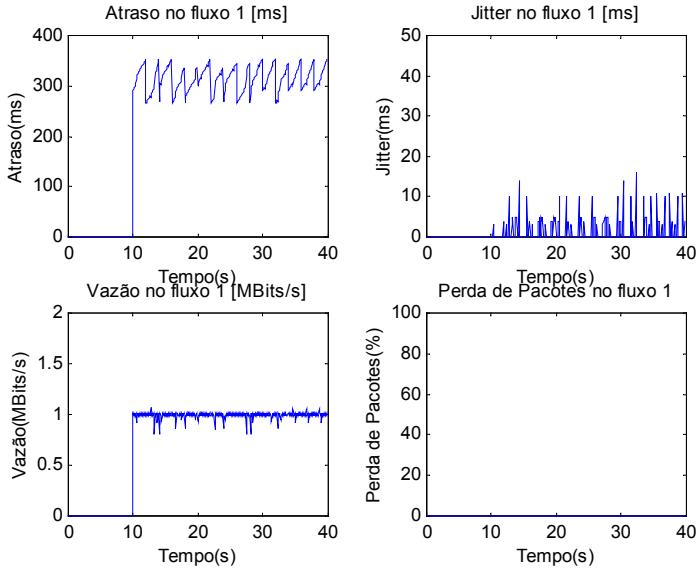

Figura 85 - Evolução da vazão, perda de pacotes, atraso e jitter para a aplicação CBR/FTP1.

Conforme pode ser observado na Figura 85, durante o intervalo $0 \leq t < 10s$ as aplicações **CBR/FTP1** foram impedidas de transmitir pacotes devido à ação da TE com CAC. Após a ampliação dos enlaces críticos e a criação da LSP, as aplicações **CBR/FTP1** são autorizadas a transmitir ($10 \leq t \leq 40s$). Conforme era esperado, a QoS oferecida às aplicações CBR/FTP1 se mantêm em valores aceitáveis.

5.8 Considerações finais

Foram apresentados, neste Capítulo, os resultados da avaliação do desempenho do sistema de TE. Ele utiliza técnicas de CI e apresenta características de um sistema autonômico, ou seja, aquele que tem a capacidade de se autoproteger, autocurar, autoconfigurar e auto-otimizar. O sistema foi implementado em um domínio MPLS simulado com a utilização do ns2 (VINT, 2003). A avaliação do sistema de TE foi feita analisando-se a QoS oferecida as aplicações que cruzam o domínio MPLS e também os indicadores de desempenho da rede. O domínio MPLS foi alimentado com aplicações do tipo dados, voz e vídeo. Os parâmetros utilizados para a avaliação da QoS oferecida as aplicações foram atraso, jitter, vazão e perda de pacotes. As medições dos parâmetros de QoS foram realizadas fim-a-fim, ou seja, entre os nós de origem e de destino das aplicações. Os indicadores utilizados para avaliar o desempenho da rede foram a utilização média dos enlaces e a relação entre o tráfego oferecido e escoado. Além disso, foram avaliadas também nas simulações a perda média de pacotes obtida para o conjunto de aplicações com e sem a implementação da TE.

Os resultados dos experimentos mostraram que é possível obter benefícios em termos de desempenho e oferecimento de QoS em domínio MPLS auto-gerenciado através de um sistema de TE com características autonômicas. O sistema de TE combina a contínua medição de tráfego no domínio, escolha dos melhores caminhos utilizando um SIF, e o estabelecimento automático das LSP´s para as aplicações que delas necessitam. O encaminhamento do tráfego das aplicações prioritárias através de caminhos alternativos possibilitou uma melhor utilização dos recursos da rede, além de melhorar a qualidade dos serviços oferecidos a estas aplicações, nas duas topologias avaliadas. Com relação ao desempenho da rede, na topologia I ocorreu uma melhora de 117% no percentual de utilização da rede, além de um aumento de 52% na RTeo. No caso da topologia II, as melhorias obtidas no percentual de utilização da rede e RTeo, foram 44% e 116%, respectivamente.

Foi avaliada também a influência de falhas em enlaces do domínio MPLS sobre a QoS oferecida às aplicações. A avaliação foi feita inicialmente utilizando a topologia I, e depois foi estendida para a topologia II. Nas duas situações as LSP´s reservas estabelecidas utilizando a abordagem nebulosa supriram as necessidades das aplicações prioritárias

quando ocorreu queda em enlace. Durante um curtíssimo período de tempo, correspondente a detecção da falha e ativação das LSP´s reservas, ocorreu perda de pacotes, diminuição da vazão e alterações no jitter, principalmente para as aplicações de vídeo. Após este momento a situação voltou ao normal, sendo que a QoS oferecida às aplicações foi semelhante aquela disponibilizada quando na ausência de falhas. Ocorreu também uma pequena diminuição da RTeo (0.1%) e um aumento da perda média de pacotes para o conjunto de aplicações prioritárias (0.1%) nas duas topologias avaliadas.

Foi avaliado também o sistema de TE com ajustes nos caminhos das LSP´s quando as aplicações prioritárias ultrapassam as suas taxas de vazão originais. O sistema se mostrou eficiente para atender as novas necessidades de vazão das aplicações prioritárias. Apesar do aumento da vazão das aplicações, a QoS oferecida voltou a ser mantida dentro de valores aceitáveis após os ajustes.

Foram avaliadas também as modificações efetuadas no sistema de TE para inclusão do CAC. As simulações foram realizadas em domínios MPLS sobrecarregados. Inicialmente, a TE com CAC foi testada na rede sobrecarregada submetida à aplicações comportadas. Os resultados obtidos mostraram que a TE com CAC é mais eficiente que a TE normal sem CAC para efetuar o escoamento do tráfego das aplicações. No caso da TE com CAC, o tráfego autorizado é totalmente escoado, enquanto na TE normal sem CAC a RTeo ficou em 94.1% na topologia I e 69% na topologia II. Já no caso da utilização média da rede, a TE normal obtém um valor melhor. Por outro lado, o uso da TE com CAC possibilitou a redução da perda media de pacotes do conjunto de aplicações para zero. No caso da TE normal sem CAC, a perda media de pacotes do conjunto de aplicações foi 8.71% para a topologia I e 37.9% na topologia II. Portanto, a utilização média da rede quando se utiliza a TE com CAC é menor, mas em compensação a QoS oferecida às aplicações é superior.

Em seguida, a TE com CAC foi testada também com aplicações não comportadas e prioridades inalteradas (estáticas). Os resultados também mostraram que a RTeo obtida para a TE com CAC é superior a obtida para a TE normal sem CAC. A utilização média da rede nas topologias I e II submetidas às aplicações não comportadas também é maior quando se utiliza a TE sem CAC. Por outro lado, a TE com CAC obtém um valor muito menor de perda média de pacotes para o conjunto de aplicações. Finalmente, a TE com CAC foi testada novamente com aplicações não comportadas, mas com prioridades

dinâmicas. Os resultados obtidos com a TE com CAC e prioridades dinâmicas foram comparados com os obtidos pela TE com CAC e prioridades estáticas nas mesmas condições. Os resultados obtidos mostraram que os valores de RTeo para as duas abordagens são semelhantes, com uma pequena superioridade, no caso da topologia II, para a TE com CAC com prioridade dinâmica. Além disso, para as duas abordagens, os valores obtidos para a perda média de pacotes obtida para o conjunto de aplicações atendem as metas de QoS apresentadas na recomendação G.1010 do ITU-T para aplicações de Voz (< 3%) e Vídeo (< 1%). Entretanto, a perda média de pacotes é menor quando é utilizada a TE com CAC com prioridade dinâmica. Com relação ao percentual de utilização média da rede os valores obtidos são os mesmos.

Foi realizada também a avaliação da TE com CAC e AIEC. As primeiras simulações foram utilizadas para gerar o histórico de medições de tráfego nos enlaces do domínio MPLS. A TE com CAC não estabeleceu LSP´s para todas as aplicações devido a impossibilidade do escoamento do tráfego com QoS. Após a obtenção do histórico de medições de tráfego o AIEC foi ativado. Com a ativação do AIEC, foram realizadas a identificação dos enlaces críticos, as previsões de vazões futuras e depois as ampliações das suas respectivas larguras de banda. Na seqüência, a TE com CAC verificou a possibilidade de criação de LSP´s para as aplicações que estavam bloqueadas. Conforme esperado, as ampliações das larguras de banda dos enlaces críticos possibilitaram a autorização pelo CAC da criação de LSP´s para as aplicações que antes estavam bloqueadas.

Foram realizadas também simulações para verificar o funcionamento da TE com otimização utilizando Algoritmos Genéticos. Os resultados obtidos mostraram que a RTeo e a perda média de pacotes não sofreram alteração quando foi realizada a otimização. Por outro lado, a TE com otimização possibilitou um pequena melhoria na utilização média da rede. Observou-se também que a média dos atrasos fim-a-fim das aplicações foi menor quando se utiliza TE com otimização. Isto ocorreu porque a TE com otimização utilizando AG escolhe os caminhos mais curtos possíveis para o conjunto de aplicações. Assim, pode-se concluir que os resultados obtidos quando se utiliza TE com otimização são superiores ao obtidos pela TE sem otimização, conforme esperado.

CAPITULO 6

CONCLUSÕES

Apresentou-se neste trabalho uma proposta para o desenvolvimento de um sistema de TE em um domínio MPLS visando oferecer QoS para aplicações de dados, voz e vídeo. O sistema de TE utiliza técnicas de CI e apresenta características de um sistema autonômico, ou seja, aquele que tem a capacidade de se autoproteger, autocurar, autoconfigurar e auto-otimizar. O próprio sistema, representado por unidades de TE, posicionadas nos LSR´s de entradas do domínio MPLS, após avaliar a necessidade providencia automaticamente a criação e a ativação das LSP´s para atender as aplicações sob sua responsabilidade. O sistema foi implementado em um domínio MPLS simulado com a utilização do ns2 (VINT, 2003). A avaliação do sistema de TE foi feita analisando-se a QoS oferecida às aplicações que cruzam o domínio MPLS e também os indicadores de desempenho da rede. Os parâmetros utilizados para a avaliação da QoS oferecida às aplicações foram atraso, jitter, vazão e perda de pacotes. As medições dos parâmetros de QoS foram realizadas fim-a-fim. Os indicadores utilizados para avaliar o desempenho da rede foram a utilização média dos enlaces e a relação entre o tráfego oferecido e o escoado. Além disso, foi avaliada a perda média de pacotes obtida para o conjunto de aplicações, com e sem a implementação da TE.

6.1 Resultados obtidos com o desenvolvimento do sistema de TE

Os resultados das simulações mostraram que é possível obter benefícios em termos de desempenho e oferecimento de QoS em domínio MPLS auto-gerenciado através de um sistema de TE. O sistema desenvolvido combina a contínua medição de tráfego no domínio, escolha dos melhores caminhos utilizando um SIF e o estabelecimento

automático das LSP´s para as aplicações que delas necessitam. O encaminhamento do tráfego das aplicações prioritárias através de caminhos alternativos possibilitou uma melhor utilização dos recursos da rede, além de melhorar a qualidade dos serviços oferecidos a estas aplicações. Na topologia I ocorreu uma melhora de 117% no percentual de utilização da rede, além de um aumento de 52% na RTeo. Na topologia II, as melhorias obtidas no percentual de utilização da rede e RTeo, foram de 44% e 116%, respectivamente.

O sistema foi eficiente para manter a QoS das aplicações quando ocorrem falhas em enlaces do domínio MPLS. As avaliações feitas nas topologias I e II mostraram que as LSP´s reservas estabelecidas utilizando a abordagem nebulosa supriram as necessidades das aplicações prioritárias quando ocorreu queda em enlace. Durante um curtíssimo período de tempo, correspondente à detecção da falha e ativação das LSP´s reservas, ocorreu perda de pacotes, diminuição da vazão e alterações no jitter, principalmente para as aplicações de vídeo. Após este momento a situação voltou ao normal, sendo que a QoS oferecida às aplicações foi semelhante àquela disponibilizada quando na ausência de falhas. Ocorreu também uma pequena diminuição da RTeo (0.1%) e um aumento da perda média de pacotes para o conjunto de aplicações prioritárias (0.1%) nas duas topologias avaliadas.

Foram avaliadas também as modificações efetuadas no sistema visando ajustes nos caminhos das LSP´s quando as aplicações prioritárias ultrapassam as suas taxas de vazão originais. O sistema se mostrou eficiente, tanto na topologia I como na II, para efetuar ajustes nos caminhos das LSP´s visando atender as novas necessidades de vazão das aplicações prioritárias. Como resultado disso, apesar do aumento da vazão das aplicações, a QoS oferecida voltou a ser mantida dentro de valores aceitáveis após os ajustes efetuados pela TE.

O sistema foi novamente modificado para a inclusão do CAC. A TE com CAC manteve a performance do sistema em caso de sobrecarga, apresentando melhores resultados que a TE normal. As simulações foram realizadas em domínios MPLS sobrecarregados submetidos a aplicações comportadas e não comportadas com prioridades estática e dinâmica.

No caso da rede sobrecarregada submetida a aplicações comportadas, os resultados obtidos mostraram que a TE com CAC é mais eficiente que a TE normal sem CAC para

efetuar o escoamento do tráfego das aplicações nas duas topologias avaliadas. No caso da TE com CAC, o tráfego autorizado é totalmente escoado, enquanto na TE normal sem CAC a RTeo ficou em 94.1% na topologia I e 69% na topologia II. Entretanto, a TE normal obtém um valor melhor para a utilização média da rede. Isto acontece porque, apesar da TE normal não ter conseguido a criação de LSP´s para algumas aplicações, o tráfego destas aplicações não foi bloqueado. Este fato acaba contribuindo para melhorar a utilização da rede, uma vez que as aplicações continuam a transmitir os seus pacotes através do menor caminho, mesmo que ocorra perda. Por outro lado, o uso da TE com CAC possibilitou a redução da perda media de pacotes do conjunto de aplicações para zero. No caso da TE normal sem CAC, a perda media de pacotes do conjunto de aplicações foi de 8.71% para a topologia I e de 37.9% na topologia II. Portanto, a utilização média da rede quando se utiliza a TE com CAC é menor, mas em compensação a QoS oferecida às aplicações é superior.

Os testes efetuados na rede sobrecarregada submetida a aplicações não comportadas com prioridades estáticas também mostraram que a TE com CAC é mais eficiente que a TE normal sem CAC. A RTeo obtida para a TE com CAC é superior aquela obtida para a TE normal sem CAC. A utilização média da rede também é maior quando se utiliza a TE sem CAC, da mesma forma que ocorreu quando a rede foi submetida a aplicações comportadas. Entretanto, a TE com CAC obtém um valor muito menor de perda média de pacotes para o conjunto de aplicações.

Finalmente, a TE com CAC foi testada novamente com aplicações não comportadas, mas com prioridades dinâmicas. Os resultados foram comparados com os obtidos pela TE com CAC e prioridades estáticas nas mesmas condições. Os valores de RTeo para as duas abordagens são semelhantes, com uma pequena superioridade, no caso da topologia II, para a TE com CAC com prioridade dinâmica. Entretanto, a perda média de pacotes é menor quando é utilizada a TE com CAC com prioridade dinâmica. Com relação ao percentual de utilização média da rede os valores obtidos são os mesmos.

Alem disso, foram realizadas também simulações para verificar o funcionamento da TE com otimização utilizando Algoritmos Genéticos. A RTeo e a perda média de pacotes não sofreram alteração quando foi realizada a otimização. Por outro lado, a TE com otimização possibilitou um pequena melhoria na utilização média da rede. Observou-se também que a média dos atrasos fim-a-fim das aplicações foi menor quando se utiliza TE

com otimização. Isto ocorreu porque a TE com otimização escolhe os caminhos mais curtos possíveis para o conjunto de aplicações. Portanto, os resultados obtidos quando se utiliza TE com otimização são superiores aos obtidos pela TE sem otimização, conforme esperado.

O sistema de TE com CAC foi depois modificado para a inclusão do AIEC. O AIEC representa uma melhoria introduzida no sistema de TE ao possibilitar a identificação de enlaces críticos. A partir da atuação do AIEC, gargalos existentes na rede puderam ser identificados, possibilitando as ampliações necessárias das larguras de banda. Conforme esperado, as ampliações das larguras de banda dos enlaces críticos tornaram possível a criação de novas LSP´s pela TE com CAC. A criação das LSP´s possibilitou que aplicações antes bloqueadas fossem autorizadas a transmitir dados.

6.2 Limitações do estudo

A principal limitação deste estudo refere-se ao fato do sistema de TE não ter sido implementado em um ambiente formado por equipamentos e aplicações reais. Este trabalho foi realizando em ambiente simulado. A troca de informações de controle entre os LSR´s para estabelecimento das LSP´s estão sendo feitas em um tempo desprezível, o que não corresponde a realidade. O desenvolvimento de um ambiente real, onde estes tempos serão considerados, será realizado em um trabalho futuro.

As previsões das vazões futuras nos enlaces críticos foram realizadas por redes neurais do tipo MLP. O treinamento das MLP´s foi realizado com a utilização de um conjunto de amostras de medições passadas de tráfego armazenadas no arquivo de histórico. Para facilitar o treinamento das MLP´s, foram utilizadas amostras com duração menor que o período total de cada simulação. Esta decisão foi tomada porque durante os testes observou-se que o uso de amostras com longo período de duração dificultava a convergência das MLP´s. O estudo de outros tipos de redes neurais artificiais que possam diminuir o tempo de treinamento e melhorar a convergência, será objeto de estudo futuro.

Um outro ponto que merece comentário é o dimensionamento dos enlaces da rede. Em redes com integração de serviços como o domínio MPLS tratado neste trabalho, além da complexidade no tratamento dos diversos tipos de aplicações, exige-se um controle para garantir os requisitos de QoS de cada uma delas. Esta situação faz com que os

dimensionamentos destas redes enfrentem desafios não observados nas redes tradicionais. Por exemplo, o dimensionamento de enlaces em redes de telefonia tradicionais é baseado somente no critério de perda de chamadas, admitindo que todas as chamadas necessitam de largura de banda fixa. O dimensionamento dos enlaces das redes com integração de serviços é muito mais complexo. A complexidade do dimensionamento está relacionada com a diversidade de parâmetros (largura de banda, atraso, topologia da rede, custos, etc). Portanto, o dimensionamento dos enlaces críticos identificados no domínio MPLS não foi tratado neste trabalho, sendo uma sugestão para trabalho futuro.

6.3 Trabalhos futuros

Os resultados obtidos são promissores, entretanto mais estudos são necessários para melhorar o sistema de TE. Assim, como continuação deste trabalho pode-se propor as seguintes atividades:

a) Alteração do SIF com a inclusão de uma terceira função de pertinência de entrada representando a confiabilidade do caminho candidato para estabelecimento da LSP. A confiabilidade será mais alta quanto menor for o número de ocorrências de falhas em enlaces do caminho;
b) Avaliação de outros algoritmos de treinamento de RNA´s para diminuir o tempo de convergência durante a previsão da vazão de tráfego nos enlaces críticos;
c) Realização de um estudo sobre o dimensionamento dos enlaces do domínio MPLS;
d) Implementar e avaliar o sistema de TE desenvolvido em um ambiente formado por equipamentos e aplicações reais.

Além disso, pode-se propor também a realização do mapeamento dos componentes da arquitetura do sistema de TE proposta para Elementos Autonômicos da Computação Autonômica. Deverão ser definidas as políticas utilizadas, descritas as interfaces dos Elementos Autonômicos e a forma como negociam entre si durante a troca de informações/serviços, além das restrições aplicáveis. Os resultados do mapeamento deverão ser incorporadas depois ao sistema de TE.

Um outro ponto que poderá ser estudado é a interoperabilidade entre domínios. O foco principal deste trabalho foi o provimento de QoS fim-a-fim para aplicações pertencentes a um mesmo domínio. Entretanto, uma grande rede, como por exemplo a Internet, é formada pela interligação de vários domínio diferentes. Portanto, pode-se propor também como trabalho futuro, o estudo de alternativas para garantir a interoperabilidade entre domínios diferentes e ao mesmo tempo manter a QoS fim-a-fim fornecida às aplicações.

6.4 Considerações finais

O objetivo geral deste trabalho, que consistiu em desenvolver um sistema de Engenharia de Tráfego, capaz de sustentar tráfego misto (dados, voz, vídeo) com QoS na rede, utilizando MPLS, princípios de Computação Autonômica e técnicas de CI, foi alcançado.

Os objetivos específicos, que consistiram em implementar algoritmos para descoberta dos caminhos, criação e ativação das LSP´s, monitoração e re-roteamento de LSP´s, controle de admissão de conexões utilizando reconhecimento e classificação dos perfis de comportamento de tráfego das aplicações, otimização dos recursos da rede, identificação dos enlaces críticos e sinalização da necessidade de ampliação de suas capacidades de forma proativa e avaliação dos resultados, foram realizados de acordo com o método proposto, o que proporcionou a obtenção do objetivo geral.

Os resultados da avaliação mostraram que o sistema de TE desenvolvido pode ser considerado como uma alternativa viável para o oferecimento diferenciado de QoS para aplicações de dados, voz e vídeo nas redes IP atuais. Além disso, a combinação de MPLS, técnicas de CI e princípios de Computação Autonômica possibilitou a construção de um sistema de TE auto gerenciável que se adapta às condições do ambiente, automatizando funções hoje executadas por seres humanos. Este comportamento inteligente apresentado pelo sistema de TE proposto vem ao encontro das necessidades das atuais redes de Telecomunicações, desafiadas pela crescente complexidade com relação ao gerenciamento. A crescente complexidade de gerenciamento das redes atuais começa a alcançar níveis além da habilidade humana. Como conseqüência, a intervenção humana se torna um ponto fraco no processo de gerenciamento e controle.

REFERÊNCIAS BIBLIOGRAFICAS

ABDENNOUR, A. *TShort-term MPEG-4 video traffic prediction using ANFIS*. In: International Journal of Network Management, vol. 15, n. 3, p.377-392, novembro 2005.

ADILÉA, W. Extração de Conhecimento a partir de Redes Neurais aplicada ao Problema da Cinemática Inversa na Robótica. Dissertação de Mestrado. Universidade do Vale do Rio dos Sinos, 2003.

ANJALI, T.; SCOGLIO, C.; OLIVEIRA, J. C. *New MPLS Network Management Techniques Based on Adaptative Learning*. In: IEEE transactions on Neural Networks, vol. 16. n. 5, p.1242-1255, setembro 2005.

ASH, G. *Traffic Engineering & QoS Methods for IP-, ATM-, & TDM-Based Multiservice Network*. Internet Draft, março 2001. Acesso em 10 dezembro 2001. Disponível em: ftp://search.ietf.org/internet-drafts/draft-ietf-tewg-qos-routing-04.txt.

AWDUCHE, D. O.; CHIU, A.; ELWALID, A.; WIDJAJA, I.; XIAO, X. *A Framework for Internet Traffic Engineering*. Internet-Draft, julho 2000. Acesso em 3 dezembro 2003. Disponível em: http://www3.ietf.org/proceedings/01aug/I-D/draft-ietf-tewg-framework-05.txt.

AWDUCHE, D. O.; BERGER, L.; GAN, D.; LI, T.; SRINIVANSAN, V.; SVALOW, G. *RSVP-TE: Extensions to RSVP for LSP Tunnels*. RFC 3209, dezembro 2001. Acesso em 17 outubro 2003. Disponível em: http://www.rfc-archive.org/getrfc.php?rfc=3209.

AWDUCHE, D.; CHIU A.; ELWALID, WIDJAJA A.; XIAO, X. *Overview and Principles of Internet Traffic Engineering*. RFC 3272, maio 2002. Acesso em 18 novembro 2003. Disponível em: http://www.rfc-archive.org/getrfc.php?rfc=3272.

AWDUCHE, D.; MALCOLM, J.; AGOGBUA, J.; O'DELL, M.; MCMANUS, J. *Requirements for traffic engineering over MPLS*. RFC 2702, 1999. Acesso em janeiro 2004. Disponível em: http://www.faqs.org/rfcs/rfc2702.html.

AWDUCHE, D.; REKHTER, Y. *Multiprotocol lambda switching: Combining MPLS Traffic Engineering Control with Optical Crossconnets*. In: IEEE Communications Magazine, vol. 39, n.3, p.111-116, março 2001.

BEZDEK, J. C. *What is Computational Intelligence,*. In: ZURADA, J. M.; MARKS II, R. J.; ROBINSON, C. J. *Computational Intelligence – Imitating Life*, IEEE Press, 1994.

BLAKE, S.; BLACK, D.; CARLSON, M.; DAVIES, E.; WANG, Z.; WEISS, W. *RFC 2475 - An Architecture for Differentiated Services.* Experimental RFC, dezembro 1998. Acesso em janeiro 2003. Disponível em: http://www.ietf.org/rfc/rfc2475.txt.

BRADEN, R.; CLARK, D; SHENKER, S. *Integrated Services in the Internet Architecture: an Overview*. RFC 1633, junho 1994. Acesso em janeiro 2003. Disponível em: http://rfc.sunsite.dk/rfc/rfc1633.html.

BRADEN, R.; ZHANG, L; BERSON, S; HERZOG, S; JAMIN S. *Resource ReSerVation Protocol (RSVP) – Version 1 Functional Specification*. RFC 2205, setembro 1997. Acesso em janeiro 2002. Disponível em: http://www.ietf.org/rfc/rfc2205.txt.

BRAGA, A. P.; LUDENIR, T. B.; CARVALHO, A. C. P. L. F. *Redes Neurais Artificiais – Teoria e aplicações.* LTC, 2000.

BUTENWEG, S. *Two Distributed Reactive MPLS Traffic Engineering Mechanisms for Throughput Optimization in Best Effort MPLS Networks.* In: Eighth IEEE International Symposium on Computers and Communications (ISCC), vol. 1, p. 379-384, julho 2003.

CRAWLEY E.; NAIR R., RAJAGOPALAN, B.; SANDICK, H. *A framework for qos-based routing in the internet.* RFC 2386, agosto 1998. Acesso em março 2003. Disponível em: http://www.rfc-archive.org/getrfc.php?rfc=2386.

CELESTINO JR., J., PONTE, P.R.X., TOMAZ, A.C.F. E DINIZ, A.L.B.P.B. *FuDyLBA: um esquema de Engenharia de Tráfego para Balanceamento de Carga em Redes MPLS Baseado em Lógica Difusa*. In: anais do XXII Simpósio Brasileiro de Redes de Computadores (SBRC), maio, 2004.

CUI, B.; YANG, Z.; DING, W. *A Load Balancing Algorithm Supporting QoS for Traffic Engineering in MPLS Networks.* In: Proceedings of the The Fourth International Conference on Computer and Information Technology (CIT'04), vol. 00, pp.436-441, setembro 2004.

DIN, N. M.; ABIDIN, H. Z; RAHMAN, S. F. A.; FISAL, N. *A Fuzzy LSP regulator for Preemption Control in DiffServ-Aware MPLS Internet*. In: the 2005 IEEE 7th Malaysia International Conference on Communication, vol. 1, p. 6, novembro 2005.

DONG, X.; HARIRI, S.; XUE, L.; CHEN, Z.; HUOPING, M.; PAVULURI, S.; RAO, S. *Autonomia: an autonomic computing environment*. In: Proceedings of the IEEE International Performance, Computing, and Communications Conference, p. 61-68, abril 2003.

DIAS, R. A.; CAMPONOGARA, E.; FARINES, J.; WILLRICH, R.; CAMPESTRINI, A. *Engenharia de Tráfego Dinâmica em Redes IP sobre Tecnologia MPLS:*

Otimização Baseada em Heurísticas. In: anais do XXII Simpósio Brasileiro de Redes de Computadores (SBRC), maio 2004.

DIN, M. N.; FISAL, N. *Dynamic Resource Allocation of IP Traffic for a DiffServ-MPLS Interface using Fuzzy Logic.* In: The 9th Asia-Pacific IEEE Conference Proceedings, vol. 1, p. 339-343, setembro 2003.

ELWALID, A.; JIN , C.; LOW, S**.;** WIDJAJA, I. *MATE: MPLS adaptive traffic engineering*. In**:** The Twentieth Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM'01), vol. 3, p.1300-1309, abril 2001.

FERGUSON, P.; HUSTON, G. *Quality of Service: Delivering QoS on the Internet and in Corporate Networks*, John Wiley & Sons, 1998.

FERNANDEZ, M. P. *Provisionamento de recursos em arquitetura DiffServ para melhoria da Qualidade de Serviços (QoS)*. Tese de doutorado. Programa de Pós-Graduação em Engenharia Elétrica/COPPE/UFRJ, outubro 2002.

GANEK, A. G.; CORBI, T. A. *The dawning of the autonomic computing era*. In: IBM Systems Journal, vol. 42, p. 5-18, janeiro 2003.

GIRISH, M. K.; ZHOU, B.; HU, J. Q. *Formulation of the traffic engineering problems in MPLS based IP networks*. In: Proceedings of the Fifth IEEE Symposium on Computers and Communications (ISCC'00), p. 214-219, março 2000.

GOLDBERG, D. E. *Genetic Algorithms in Search, Optimization and Machine Learning.* Addison Wesley, 1989.

GOZDECKI, J.; JAJSZCZYK, A.; STANKIEWICZ, R. *Quality of Service Terminology in IP Networks.* In: IEEE Communications Magazine, vol. 41, n.3, p. 153-159, março 2003.

HARDY, W. C. *QoS Measurement and Evaluation of Telecommunications Quality of Service*. John Wiley and Sons LTD, 2001.

HAYKIN, S. *Neural Networks-A comprehensive foundation.* Prentice Hall, 1994.

HENRIQUES, L. O. A. P. *Compensação das oscilações de torque de um acionamento de relutância chaveado utilizando técnicas de controle neuro-fuzzy,* Dissertação de mestrado*,* COPPE/UFRJ, abril 1999.

HINES, J. W. *Fuzzy and Neural Approaches in Engineering MatLab supplement*, John Wiley & Sons, 1997.

INTERNATIONAL TELECOMMUNICATION UNION – ITU-T. G.1010. *End-user multimidia QoS categories*. Series G: Transmission Systems and Media Digital Systems and Networks, 2002. Acesso em junho 2004. Disponível em:

ftp.tiaonline.org/tr-30/tr303/Public/0312%20Lake%20Buena%20Vista/G1010%20-%2011-01.doc.

INTERNATIONAL TELECOMMUNICATION UNION – ITU-T. *Recommendation E.800: Terms and Definitions Related to Quality of Service and Network Performance Including Dependability.* Agosto 1993. Acesso em junho 2005. Disponível em: ttp://www.itu.int/rec/T-REC-E.800-199408-I/en.

INTERNATIONAL TELECOMMUNICATION UNION – ITU-T. *Recommendation G.1000: Communications Quality of Service – A framework and definitions.* Novembro 2001. Acesso em junho 2005. Disponível em: http://www.itu.int/rec/T-REC-G.1000/en.

IYODA, E. M.; *Inteligência computacional no projeto automático de redes neurais híbridas e redes neurofuzzy heterogêneas,* Dissertação de Mestrado, FEEC/ UNICAMP, janeiro 2000

JAMOUSSI, B.; ANDERSSON, L.; CALLON, R.; DANTU, R.; WU, L.; DOOLAN, P.; WORSTER, T.; FELDMAN, N.; FREDETTE, A.; GIRISH, M.; GRAY, E.; HEINANEN, J.; KILTY, T.; MALIS, A. *Constraint-Based LSP Setup using LDP*. RFC 3212, fevereiro 2002. Acesso em junho 2005. Disponível em: http://www.rfc-archive.org/getrfc.php?rfc=3212.

KHAN, J. A. ALNUWEIRI, H. *A Traffic Engineered Routing Algorithm Based on Fuzzy Logic*. In: IEEE Pacific Rim Conference on Communications Computers and Signal Processing (PACRIM), vol. 1, p. 454-457, agosto 2003.

KEPHART, J. O.; CHESS, D. M. *The vision of autonomic computing.* In: IEEE Computer, vol. 36, n.1, p. 41-50, janeiro 2003.

LEMMA, H. G. *Enhanced Fast Rerouting Mechanisms for Protected Traffic in MPLS Networks*. PHD Thesys, UPC, abril 2003

LIU, H.; BAI, D.; DING, W. *An Explicit Routing Optimization Algorithm for Internet Traffic Engineering.* In: IEEE International Conference on Communication Technology (ICCT), vol. 1, p. 445-449, abril 2003.

MANDANI, E. H.; ASSILIAN, S**.** *An experiment in linguistic synthesis with a fuzzy logic controller.* In: International Journal of Man Machine Studies, vol. 7, n. 1, p. 1-13, 1975.

MORTENSEN, J, E. *JFS Fuzzy System.* Última atualização junho 2004. Acesso em julho 2004. Disponível em: <http://www.inet.uni2.dk/~jemor/jfs.htm>.

MÜLLER, H. A.; O'BRIEN, L.; KLEIN, M.; WOOD, B. *Autonomic Computing*. Technical Note. Software Engineering Institute. Carnegie Mellon University. Abril,

2006. Acesso em agosto 2006. Disponível em: http://www.sei.cmu.edu/pub/documents/06.reports/pdf/06tn006.pdf.

NOBRE, E.G.; PONTE, P. R. X.; FERNANDEZ, M. P. CELESTINO JR., J. *IntelliDyLBA: um esquema de Balanceamento de Carga em Redes MPLS com Aprendizado Desassistido baseado em Lógica Difusa e Algoritmos Genéticos*. In: anais do XXIII Simpósio Brasileiro de Redes de Computadores (SBRC), maio 2005.

OLIVEIRA, S. S. de. *Análise de tráfego na integração de redes IP e ATM usando simulação*. *Dissertação de Mestrado*. Programa de Pós-Graduação em Ciência da Computação/UFSC, outubro 2001.

REFENES, A. N; AZEMA-BARAC, M; CHEN, L.;KAROUSSOS, S. A**.** *Currency exchange rate prediction and neural network design strategies*. In: Neural computing & Aplications Journal, vol. 1, p. 46-58, 1993.

RESENDE, R. A.; ROSSI, S. M.; YAMAKAMI A.;BONANI, L. H.; MOSCHIM, E. *Traffic Engineering with MPLS Using Fuzzy Logic for Application in IP Networks*. In:. The 12th IEEE International Conference on Fuzzy Systems **(**FUZZ '03), vol. 2, p.1146 –1151, maio 2003.

SALVADORI, E.; SABEL, M; BATTITI, R. *A reative Scheme for Traffic Engineering in MPLS Networks.* Technical report*,* Universitá di Trento, Departimento di Informática e Telecommunicazioni, dezembro 2002.

SALEHIE, M.; TAHVILDARI, L. *Autonomic Computing: Emerging trends and open problems.* In: Proceedings of the 2005 workshop on Design and evolution of autonomic application software (DEAS '05), p. 1-7, maio 2005.

SHENKER, S.; PARTRIDGE, C.; GUERIN, R. *Specification of Guaranteed Service*. *RFC 2212*, setembro 1997. Acesso em agosto 2004. Disponível em: http://www.ifla.org/documents/rfcs/rfc2212.txt.

SHIGANG, C.; NAHRSTEDT, K.. *An overview of quality of service routing for next-generation high-speed networks: problems and solutions*. In: IEEE Network, vol. 12, n. 6, p. 64 –79, novembro/dezembro 1998.

SPRAGGS, S. *Traffic engineering.* In: BT Technology Journal, vol. 18, n. 3, p. 137-150, julho 2000.

STERRITT, R.; PARASHAR, M.; TIANFIELD, T.; UNLAND, R. *A concise introduction to autonomic computing*. In: Journal of Advanced Engineering Informatics, Engineering Applications of Artificial Intelligence, Special Issue on Autonomic Computing and Automation, Elsevier Publishers, vol. 19, n. 3, p.181-187, julho 2005.

SVALOW, G. *MPLS Advantages for Traffic Engineering*. In: IEEE Comunications Magazine, vol. 37, n. 12, p.54-57, dezembro 1999.

TANEMBAUM, A. *Computer Networks*. Englewood Cliffs, NJ: Prentice Hall, 1996.

VINT Network Simulator – version 2. Acesso em Março 2003. Disponível em: <http://www- mash.cs.berkeley.edu/ns>.

WHITE, S. R.; HANSON, J. E.; WHALLEY, I.; CHESS, D. M.; KEPHART, J. O. *An Architectural Approach to Autonomic Computing*. In: Proceedings of the 1st International Conference on Autonomic Computing (ICAC'04), IEEE Computer Society, p.17-19, maio 2004.

WROCLAWSKI, J.; *Specification of the Controlled-Load Network Element Service*. RFC 2211, 1997. Acesso em agosto 2003. Disponível em: http://www.rfc-archive.org/getrfc.php?rfc=2211.

WOLF, L. C.; GRIWODZ, C.; STEINMETZ, R. *Multimedia Communication.* In: Proceedings of the IEEE, vol. 85, n. 12, p. 1915-1933, dezembro 1997.

XIAO, X.; HANNAN , A.;BAILEY, B.; NI, L. *Traffic Engineering with MPLS in the Internet.* In: IEEE Network magazine, vol. 14, n. 2, p. 28-33, março/abril 2000.

XIAO, X.; NI, L. *Internet QoS: a big picture*. In: IEEE Networks Magazine, vol. 13, n. 2, p. 8-18, março/abril 1999.

ZEBULUM, R. S. *Previsão de carga em sistemas elétricos de potência por redes neurais*. Dissertação de mestrado. Pontifícia Universidade Católica do Rio de Janeiro, 1995.