UNIVERSIDADE ESTADUAL DE CAMPINAS

FACULDADE DE ENGENHARIA ELÉTRICA E DE COMPUTAÇÃO

Tese de Mestrado

Roteamento de Tráfego Adaptativo Baseado em Caminho Mínimo em Redes MPLS

Autor: Raulison Alves Resende
Orientador: Prof. Dr. Akebo Yamakami.

Campinas - SP
Outubro de 2001

UNIVERSIDADE ESTADUAL DE CAMPINAS

FACULDADE DE ENGENHARIA ELÉTRICA E DE COMPUTAÇÃO

Área de Concentração

Telecomunicações e Telemática

Roteamento de Tráfego Adaptativo Baseado em Caminho Mínimo em Redes MPLS

Dissertação apresentada à Faculdade de Engenharia Elétrica e de Computação da Universidade Estadual de Campinas, como parte dos requisitos exigidos para obtenção do titulo de Mestre em Engenharia Elétrica.

Autor: Raulison Alves Resende
Orientador: Prof. Dr. Akebo Yamakami.

Banca Examinadora:

Dr. Alberto Paradisi - Fundação CPqD

Prof. Dr. Edson Moschim - FEEC-UNICAMP

Prof. Dr. Yuzo Iano - FEEC-UNICAMP

Campinas - SP
Outubro de 2001

Dedicatória

À Luciana, minha esposa e grande companheira,
A meu pai, Manoelino e a minha mãe Maria de Fátima,
À minha sogra, Dinorá Maria e a meu sogro Adão,
dedico este trabalho.

Agradecimentos

A Deus e a Ave-Maria.

Além do apoio familiar, de parentes e amigos, este trabalho contou com a colaboração de várias pessoas e instituições, às quais deixo aqui meus sinceros agradecimentos:

Ao Prof. Dr. Akebo Yamakami, pela confiança depositada, incentivo e orientação.

Ao amigo Adrian, pelas orientações iniciais sobre a pós-graduação da FEEC.

Aos meus amigos Adriano Neto e João de Deus pela ajuda fundamental na correção de vários textos, bem como do trabalho de digitação desta tese.

Aos meus amigos Fabiano Pádua e Tulius pelo companheirismo demonstrado durante este trabalho.

Aos professores e colegas do Departamento de Telemática, pelo apoio, amizade e ensinamentos proporcionados.

Aos membros da Banca Examinadora, Dr. Alberto Paradisi (Fundação CPqD), Prof. Dr. Edson Moschim (UNICAMP) e Prof. Dr. Yuzo Iano (UNICAMP) pelas críticas e sugestões apresentadas.

Ao CNPq, pela bolsa de mestrado concedida.

A UNICAMP, pelo acolhimento e infra-estrutura.

As demais instituições e pessoas que, de alguma forma, contribuíram para a realização deste trabalho.

"A razão para os problemas é vencê-los. Porque esta é a verdadeira natureza do homem, ir além dos limites. Não é o desafio com que nos deparamos que determina quem somos e o que estamos nos tornando, mas a maneira que respondemos ao desafio, se tocamos fogo nos destroços ou trabalhamos até o fim". R. Bach

Resumo

Este trabalho apresenta uma proposta de roteamento de tráfego adaptativo em redes MPLS (MultiProtocol Label Switching). Esta proposta baseia-se na representação da rede MPLS através de grafo, sobre a qual é implementado um algoritmo que procura enlaces com o menor custo para formar um LSP (Label Switched Path). Os custos são gerados através de uma função penalizadora dos enlaces sobrecarregados buscando com isto uma mínima perda de pacotes e uma utilização homogenea do domínio MPLS. Será utilizado o critério de seleção de rotas com menor custo representando menos congestionamento através do algoritmo Dijkstra no nó de ingresso LER (Label Edge Router). O algoritmo proposto foi simulado numa rede exemplo utilizando a ferramenta NS-2 (Network Simulator – version 2) e os resultados são analisados e comparados com o modelo tradicional de SPF (Short Path First) estático.

Abstract

In this work is presented a proposal of adaptative traffic routing in MPLS (MultiProtocol Label Switching) networks. This proposal is based on the modelling of MPLS networks through graphs, on which was implemented an algorithm that seeks links with optimal cost to form LSP (Label Switched Path). The costs are generated by a penalty function for overloaded links in order to get a minimal package loss and a homogeneous utilization of the MPLS domain. We will use the link selection criterion with the lower cost, which represents less overloading, through the Dijkstra algorithm in the enter node LER (Label Edge Router). The proposed algorithm was simulated on an example network by the use of NS-2 (Network Simulator – version 2) tool. We analyse and compare the obtained results to traditional SPF (Short Path First) model.

Trabalhos Publicados

R. A. Resende, A. Yamakami, "Roteamento de Tráfego Baseado em Restrições em Redes MPLS", Anais do 19º Simpósio Brasileiro de Telecomunicações, Fortaleza-Ceará, Setembro 2001.

F. J. L. Pádua, R. A. Resende, E. Moschim e A. Yamakami, "Roteamento e Atribuição de Lambdas em Redes GMPLS/DWDM", Anais do 19º Simpósio Brasileiro de Telecomunicações, Fortaleza-Ceará, Setembro de 2001.

Índice

Lista de Figuras

Lista de Tabelas

Lista de Acrônimos

ATM	*Asynchronous Transfer Mode*
BGP	*Border Gateway Protocol*
BTT	*Bidirectional Traffic Trunk*
CBR	*Constant Bit Rate*
DLCI	*Data Link Control Identifier*
EIA	*Electronic Industries Association*
FEC	*Forwarding Equivalence Class*
FTN	*FEC-to-NHLFE*
FTP	*File Transfer Protocol*
HTTP	*hyper text transfer protocol*
IEEE	*Institution of Electrical Engineers*
IETF	*Internet Engineering Task Force*
IGP	*Interior Gateway Protocol*
ILM	*Incoming Label Map*
IP	*Internet Protocol*
IS-IS	*Intermediate System to Intermediate System*
ISO	*International Organization for Standardization*
ISP	*Internet Service Provider*
ITU	*International Telecommunications Union*
ITU-T	*ITU - Telecommunications Standardization Sector*
LDP	*Label Distribution Protocol*
LER	*Label Edge Router*
LIB	*Label Information Base*
LSA	*Link State Advertisement*
LSP	*Label Switched Path*
LSR	*Label Switched Router*
MAM	*Maximum Allocation Multiplier*
MATE	*MPLS Adaptive Traffic Engineering*
MPLS	*MultiProtocol Label Switching*
NBMA	*Non-Broadcast Multiple Access*
NHLFE	*Next Hop Label Forwarding Entry*
OSI	*Open System Interconnect*
OSPF	*Open Shortest Path First*
PASTE	*Provider Architecture for Differentiated Services and Traffic Engineering*
QoS	*Quality of Service*
RFC	*Request for Comments*
RIP	*Routing Information Protocol*
RSVP	*Resource ReSerVation Protocol*
SPF	*Shortest Path First*
TCP	*Transmission Control Protocol*
TE	*Traffic Engineering*

TIA	*Telecommunications Industtry Association*
TLV	*Type-Length-Value*
UDP	*User Datagram Protocol*
VBR	*Variable Bit Rate*
VCI	*Virtual Channel Identifier*
VPI	*Virtual Path Identifier*
VPN	*Virtual Private Network*

1

Introdução

"O Futuro já não é o que era! O crescimento explosivo da capacidade dos computadores e de outras tecnologias vai ocasionar, na próxima década, mudanças profundas na forma de viver do homem. Convivendo com um novo universo que abrange negócios, governo, comunicações pessoais, educação e entretenimento, o ser humano, com uma interface de multimídia, vai se tornar mais produtivo, melhor informado e capaz de desfrutar e contribuir para uma melhor qualidade de vida".(Jacob, 1997).

Introdução – 1.1. Descrição do Trabalho.

Na década de 50, computadores eram máquinas enormes e complexas onde o processamento em lote era a opção de interação para os usuários. Já na década de 60 surgiram os primeiros terminais interativos, permitindo aos usuários o acesso ao computador central, possibilitando a interação direta com o computador, ao mesmo tempo em que avanços nas técnicas de processamento davam origem a sistemas de tempo compartilhado (*time-sharing*), o que permitia que diferentes usuários ocupassem simultaneamente o computador central, por uma espécie de revezamento no tempo de ocupação do processador.

Em 1969 surgia a primeira rede de computadores do mundo, a ARPAnet[1], que interligava quatro universidades americanas. Inúmeras outras redes de caráter militar e científico surgiriam nos anos seguintes. Durante a década de 70 começa a descentralização do sistema de grande porte, ocorrendo à distribuição do poder computacional.

O desenvolvimento de microcomputadores de alto desempenho permitiu a instalação de considerável poder computacional em várias localizações de uma organização. Uma vez a rede

[1] Uma das mais antigas redes de comutação de pacotes projetada pela ARPA (Advanced Research Projects Agency)

instalada, passou a existir a necessidade de troca de informações ocasionando uma razão importante para a interconexão. Ambientes de trabalho cooperativos tornaram-se realidades tanto em empresas como em universidades, exigindo a interconexão dos equipamentos nessas organizações. Em 1981 surgiu o IBM-PC, o primeiro micro com arquitetura de 16 bits a ser um sucesso comercial. Rapidamente suas vendas dispararam e as empresas passaram a adquirir muitos PCs. Este foi um pequeno passo na interligação de alguns deles na forma de uma rede local de micros. A padronização em 1978 da tecnologia Ethernet, desenvolvida pela Xerox PARC no início da década de 70, proporcionou o surgimento de adaptadores de 10 Mbps (10 milhões de bits por segundo) para essas primeiras redes de PCs.

Ao longo das décadas de 80 e 90, conforme crescia o nível de informatização da sociedade e a tecnologia de comunicação de dados e de redes, foram surgindo padronizações para atender aos requisitos de compatibilidade exigidos pelo mercado que não mais se contentava com as tecnologias proprietárias das décadas anteriores.

Siglas como ISO/OSI, EIA/TIA, IEEE e termos como sistemas abertos tornaram-se comuns. Criada na década de 80 a partir da ARPAnet, inicialmente a Internet era apenas uma rede de caráter acadêmico que interligava centros de pesquisa e universidades americanas. A evolução da informatização e a forma democrática como essa rede foi criada, a qual é mantida até hoje, fizeram com que, lentamente, se estendessem para além das fronteiras americanas. Com o lançamento da interface gráfica Web em 1993, estava dado o passo na direção da popularização dessa rede, o que se concretizou em 1995, considerado o ano da explosão da Internet. Nesse ano foi aberto o acesso comercial, surgiram as primeiras Intranets e a rede passou a crescer a um ritmo alucinante.

A Internet popularizou as interfaces e aplicativos gráficos. Esses, por sua vez, exigem bandas crescentes para atenderem às expectativas de tráfego e aos novos serviços, que incluem voz e vídeo digital. As primeiras redes podiam transportar apenas dados, mas hoje devem prover recursos multimídia, o que implica num constante aumento da velocidade dos dispositivos e tecnologias de comunicação de dados, num crescimento sem fim.

As redes de telecomunicações sofreram uma grande evolução tecnológica desde o tempo de sua criação até aos nossos dias. Evoluíram de redes analógicas comutadas manualmente às modernas centrais digitais com transmissão através de cabos de fibra ótica. No que diz respeito ao serviço oferecido, as redes de telecomunicações, tradicionalmente, eram projetadas para um

tipo específico de serviço: uma rede para transmissão de voz, outra para telex, outra para dados e assim por diante. Neste cenário, um usuário corporativo necessita contratar diversos serviços para atender às diversas demandas de sua empresa.

No mundo moderno, onde a informação é a grande mercadoria, levar informações de um lugar a outro, sem limitar a sua natureza (dados, voz, vídeo, documentos, etc), garantindo segurança, qualidade e rapidez, tornou-se uma exigência do mercado e objetivo das empresas que atuam no setor das telecomunicações.

Para atender as exigências do mercado de comunicações, o ITU-T (*International Telecommunications Union - Telecommunications*) escolheu o modo de transferência assíncrono (ATM - *Asynchronous Transfer Mode*) como a tecnologia para a implementação da rede digital de serviços integrados de faixa larga (RDSI-FL) devido ao grau de flexibilidade do seu princípio onde quaisquer tipos de informação (dados, som, vídeo ou fotos), podem ser divididas em pequenos pacotes digitais de tamanho fixo (células) e transmitidas independentes de seus conteúdos. O ATM utiliza as vantagens dos princípios de operação orientada à conexão, que significa: 1) que antes da transmissão da informação útil (*payload*) deve existir uma conexão; 2) a informação útil é transmitida via esta conexão; 3) esta conexão é derrubada novamente via sinalização do usuário. A informação é empacotada em células em ordem para serem transportadas. As células consistem num campo de informação mais o cabeçalho. As células ATM são geradas apenas quando requeridas. Se não há informação útil a ser transmitida em uma determinada seção de transmissão num certo período de tempo, são geradas células chamadas "*idle cells*" ou células vazias. Desta forma, a banda disponível pode ser explorada de maneira flexível para se preencher as necessidades variáveis de transmissão, garantir seus requisitos de tráfego de forma integrada e, com isso, ser capaz de prover os recursos necessários a multimídia.

A plataforma ATM, mesmo com todas as funcionalidades apresentadas para solucionar os problemas de banda larga e convergência de serviços, nunca encontrou uma aceitação maciça. Uma das justificativas é a necessidade de reformulação das redes. Isso demanda enormes investimentos sem garantia de retorno. Esse fato restringiu momentaneamente o uso de ATM em redes de alto volume de dados e especializadas.

No final dos anos 90, o Protocolo Internet (IP) começa a despontar como uma tecnologia barata e acessível para todos os problemas, porém, sem apresentar solução convincente para as redes de acesso na substituição das centrais locais. Muitos testes e experiências foram feitos, mas

a questão do custo e da aplicação que permitam uma utilização em larga escala ainda não foi identificada. Os arraigados conceitos de qualidade, confiabilidade e disponibilidade, típicos das operadoras telecomunicações, limitam a área de atuação dessa nova tecnologia às "*start-ups*" ou empresas especializadas, retardando sua disseminação no mercado de voz e multisserviço das grandes operadoras.

Na década de 90, surgiu um número grande de tecnologias de comutação de pacotes baseada na troca de rótulos. O IETF (*Internet Engineering Task Force*) observando a potencialidade, funcionalidade e ao mesmo tempo incompatibilidade entre elas, resolveu criar um grupo de trabalho visando uma padronização destas tecnologias.

O MPLS (*MultiProtocol Label Switching*) é o principal resultado do trabalho deste grupo. Através desta tecnologia pode-se resolver alguns dos problemas apresentados pelo roteamento IP convencional, do tipo *hop-by-hop*, que são atrasos de propagação dos datagramas através da rede e, com isso, não é mais prioridade aquisição de roteadores com grande capacidade de roteamento a um custo elevado.

Desta maneira, o MPLS minimiza as despesas além de oferecer um melhor desempenho de roteamento e novas tecnologias de serviços, por exemplo VPN (*Virtual Private Network*), QoS (*Quality of Service*) e Engenharia de Tráfego. Desta forma, passa-se a ter uma eficiência na transmissão de dados ponto-a-ponto.

O MPLS agrega, no nível de enlace, a capacidade de roteamento, acrescentando algumas funcionalidades inexistentes no roteamento IP convencional, do tipo hop-by-hop, que são atrasos de propagação dos datagramas através da rede, e o congestionamento da rede. Ele ocasiona a comutação IP como solução para o primeiro problema e a engenharia de tráfego para o segundo.

1.1. Descrição do Trabalho

O objetivo principal deste trabalho, é o de propor um método que realize roteamento adaptativo baseado em caminho mínimo em tempo real, utilizando para isto uma rede de computadores com roteadores com MPLS (*MultiProtocol Label Switching*).

Assim, para expor este novo método, dividimos o trabalho em 6 capítulos descritos abaixo.

No Capítulo 2, procuramos dar uma noção geral do *MultiProtocol Label Switching*, mostrando a arquitetura através de uma definição de suas características e seus componentes.

Em seguida, o Capítulo 3, trata dos conceitos e elementos fundamentais da Engenharia de Tráfego. São estudados aspectos como propiciar a melhor utilização dos recursos da rede, exercendo influência direta sobre o roteamento.

No Capítulo 4 é apresentada a nossa proposta, roteamento adaptativo baseado em caminho mínimo em rede *MultiProtocol Label Switching* para comparar com o método de roteamento caminho mínimo estático para que, através de parâmetros de medida estatística, possamos avaliar o desempenho do método proposto.

No Capítulo 5 são apresentados as simulações efetuadas e os resultados obtidos. São analisados os resultados computacionais resultantes da aplicação do método descrito no Capítulo 4.

No Capítulo 6 apresentamos as conclusões da dissertação e algumas sugestões de trabalhos futuros.

2

Arquitetura MPLS (Multiprotocol Label Switching)

Introdução – 2.1. Estrutura do MPLS – 2.2. Componentes do MPLS – 2.3. Encaminhamento e Troca de Rótulos – 2.4. Distribuição de Rótulo – 2.5. Transmissão de Pacotes em Rede MPLS – 2.6. Determinação de Rota – 2.7. Aplicações MPLS

O MPLS é uma tentativa do IETF de padronizar a comutação de pacotes baseada na troca de rótulos. A incapacidade de interação entre as tecnologias IP Switching (Ipsilon), CSR - Cell Switched Router (Toshiba), Tag Switching de **[REK97]** (Cisco Systems), ARIS - Aggregate Route-based IP Switching (IBM) e SITA - Switching IP Through ATM (Telecom Finland) fez com que a Força Tarefa de Engenharia da Internet (*IETF - Internet Engineering Task Force*) criasse, em dezembro de 1996, um grupo de trabalho visando uma padronização destas tecnologias.

Segundo Rosen **[ROS01]**, a comutação de rótulos multiprotocolos (MPLS - *MultiProtocol Label Switching*), combina a funcionalidade dos protocolos de roteamento da camada de rede e a comutação por rótulos, além de fornecer significativos benefícios a redes com IP (*Internet Protocol*) e ATM (*Asynchronous Transfer Mode*) ou uma combinação de outras tecnologias no nível da camada de rede.

Esta tecnologia visa minimizar as despesas, melhorar o desempenho de roteamento e fornecer novas alternativas de serviços, tais como, VPNs (*Virtual Private Network*), QoS (*Quality of Service*) do inicio ao fim e engenharia de tráfego permitindo, com isso, a utilização eficiente de redes existentes para atender o crescimento futuro e uma rápida correção de falhas de enlace e nó. O MPLS também fornece serviços IP diferenciados, gerenciamento e aprovisionamento mais simples para provedores e assinantes da rede.

2.1. Estrutura do MPLS

Em uma rede IP (*Internet Protocol*) não orientada à conexão tem-se uma estrutura baseada em roteadores. Esta maneira como é estruturada traz algumas vantagens e desvantagens. Como vantagem, ela torna-se flexível e tem seus endereços distribuídos organizadamente. Contudo, quando a rede opera em altas taxas de tráfego, os roteadores facilmente atingem o gargalo eletrônico tornando-se pontos críticos. A Tabela 2.1 faz uma comparação entre os métodos.

Para solucionar o problema mencionado da rede IP, tem-se adotado o uso de comutadores da tecnologia ATM. Outras formas de agregar as vantagens dos equipamentos de comutação às redes IP têm sido estudadas, sendo uma delas o MPLS **[ROS01]**.

O MPLS torna a função de encaminhamento mais simples. A análise dos pacotes é feita fundamentalmente nos roteadores de borda, passando assim a trabalhar com o mecanismo orientado à conexão dentro do IP. O MPLS pode ser lógica e funcionalmente dividido em duas estruturas, para oferecer a funcionalidade *Label Switching*, como mostra a Figura 1.

A primeira estrutura do MPLS é a de controle, **[MAG01]**. No plano de controle estão localizadas as funções de controle tais como sinalização, roteamento, conversão de endereços, policiamento de tráfego, dentre outras. Neste plano a representação da rede é a mesma dos protocolos de roteamento (por exemplo, um grafo).

Um comutador não convencional que suporta MPLS assume a característica de roteador IP. Caracterizada esta funcionalidade, eles adotam como protocolos de roteamento: OSPF (*Open Shortest Path First*) **[MOY98]**, RIP (*Routing Information Protocol*) **[MAL98]** e BGP (*Border Gateway Protocol*) **[REK01]**.

A integração facilitada de IP com ATM – a comutação de rótulos usada pelo MPLS é muito semelhante à forma com que os comutadores ATM trabalham, baseados nos campos VPI/VCI das células ATM. Conforme Davie **[DAV01]** especifica, os comutadores ATM permitem a utilização de campos de VPI/VCI como rótulos MPLS podendo, dessa forma, ser utilizados em redes que faz uso de tal tecnologia. A reutilização de equipamentos de duas redes distintas em uma nova rede MPLS, sugere uma integração entre redes com equipamentos IP e ATM, num mesmo "backbone".

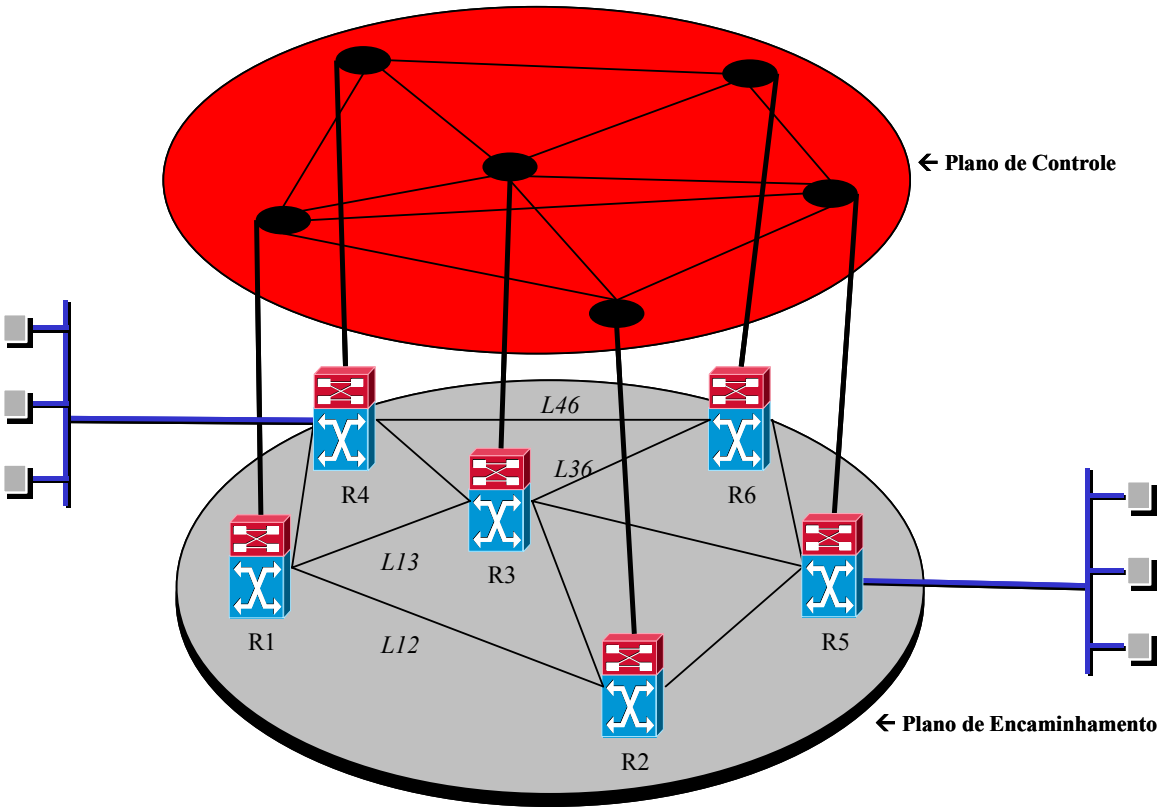


Figura 1- Estrutura de funcionamento do MPLS.

A segunda estrutura do MPLS é a de encaminhamento, cuja operação é ditada pelo plano de controle e agrega funções relativas à propagação de datagramas IP, como por exemplo, encapsulamento, segmentação, remontagem e rotulação de datagrama.

A superposição de um rótulo ao datagrama tem a propriedade de imprimir à comunicação uma característica de orientação à conexão. Além de usufruir os benefícios gerados da sua união com o ATM, os ISPs (*Internet Service Provider*) passam a ter uma excelente infra-estrutura de transporte de grandes volumes de tráfego IP.

Tabela 1 - Comparação entre Roteamento IP e Label Switching

Diferenças entre Roteamento IP Convencional e Label Switching		
	Roteamento IP convencional	**Label Switching**
Análise inteira do cabeçalho IP	Verifica os pacotes em cada *salto* do caminho na rede	Verifica os pacotes somente uma vez no ingresso do caminho virtual
Suporte para dados *Unicast* e *Multcast*	Necessita de roteamento especial para *Multicast* e algoritmos de encaminhamento	Necessita de somente um algoritmo de encaminhamento
Decisão de roteamento	Baseado no endereço de destino no cabeçalho IP	Baseado em vários parâmetros: endereço de destino no cabeçalho IP, QoS, tipo de dados, etc.

2.2. Componentes do MPLS

A Figura 2 mostra os principais dispositivos que fazem parte da arquitetura MPLS definidas em **[ROS01]**. Cada dispositivo executa um único protocolo de roteamento IP. Eles são indispensáveis no processo de mapeamento de rótulos para classes equivalentes de encaminhamento ao próximo nó através de um circuito virtual.

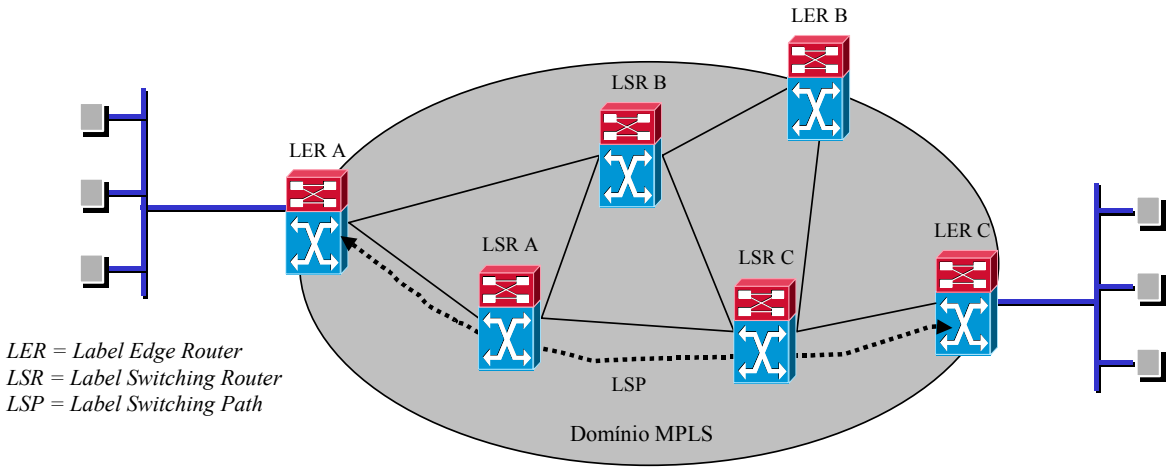


Figura 2 - Infra-estrutura básica MPLS.

2.2.1. LER e LSR

O LER (*Label Edge Router*) é um dispositivo que atua nas bordas das redes de acesso e MPLS. As múltiplas portas dão sustentação aos LERs para se conectarem aos mais variados tipos de redes existentes (Frame Relay, ATM, e Ethernet) e enviarem seus tráfegos sobre rede MPLS. Eles usam os protocolos de sinalização (LDP **[AND01]**, **[THO01]**, RSVP-TE **[AWD01]** e BGP **[REK01]**) de rótulo que são responsáveis pelo estabelecimento e remoção dos LSPs para distribuir e remover o tráfego das redes de acesso nos nós ingresso e egresso.

O LSR (*Label Switching Router*) é um dispositivo para roteamento de alta velocidade localizado no núcleo da rede MPLS. Ele participa ativamente no estabelecimento de LSPs, usando o protocolo de sinalização de rótulo e no encaminhamento de tráfego baseado nos trajetos estabelecidos. Em qualquer rede MPLS, um roteador que suporte tal técnica será um LSR.

2.2.2. FEC

A FEC (*Forwarding Equivalence Class*) é uma representação de um grupo de pacotes que compartilham das mesmas exigências para o transporte. Todos os pacotes pertencentes a este grupo recebem o mesmo tratamento por todo encaminhamento (origem – destino).

2.2.3. LSP

O LSP (*Label Switched Path*) é um conjunto de dispositivos do MPLS, o qual habilitado, representa o seu domínio, formando caminho de origem LSR ingresso e destino LSR egresso pelo qual passam pacotes pertencentes a uma determinada FEC, conforme ilustrado na Figura 2.

2.3. Encaminhamento e Troca de Rótulo

O roteamento convencional IP analisa uma grande quantidade de informações contidas no cabeçalho do datagrama desnecessariamente, para apenas tomar a decisão de qual será o próximo salto (hop), causando um atraso da informação.

Neste sentido, o MPLS apresenta dois procedimentos que são suficientes para solucionar este problema em qualquer rede. O primeiro procedimento analisa todos os pacotes ingressantes e os classificam em FEC. O segundo busca a interface associada à FEC para enviar aos LSRs intermediários. Os pacotes pertencentes a mesma FEC terão o mesmo tratamento em todo trajeto.

A atribuição de pacotes com endereços de destino variados a um conjunto mínimo de FECs deve ser realizada apenas uma vez, a partir do instante em que o pacote entra na rede MPLS. Para esta atribuição será utilizado um único rótulo para identificar determinada FEC. O rótulo é um identificador com legibilidade local, tamanho fixo de 20 bits, afixado no cabeçalho de cada pacote e torna-se o indexador para todas as decisões tomadas com relação ao encaminhamento do pacote. O processo de análise do rótulo é denominado por Rosen **[ROS01]** como Label Swapping, para todas as redes. A essência deste processo é que em cada salto, o LSR

descarta o rótulo existente e aplica um novo rótulo que indica ao próximo nó como enviar o pacote. Os valores dos rótulos não são constantes e podem mudar de roteador para roteador.

O rótulo pode possuir várias funções, uma delas é a de precedência do pacote de acordo com **[ROS01]**. A análise do cabeçalho do nível de enlace, além de determinar o próximo salto, serve também para aplicar políticas de prioridades. No MPLS, os protocolos não determinam a prioridade ou classe de serviços com base no cabeçalho. A determinação é com base na verificação a que classe de serviços ou precedência as FECs dos pacotes pertencem, indicadas pelos rótulos. Analisado, pode-se observar que o rótulo assume as funções de indicar, ao mesmo tempo, rota e classes de serviço de um pacote.

Após um pacote receber um rótulo, ele passa a pertencer a uma FEC e pode ser comutado. O encapsulamento de um pacote pode defrontar-se com tecnologias de enlace que não empregam identificadores de conexão, por exemplo, PPP (Poitn-to-Point Protocol) **[SIM99]** e Ethernet (IEEE 802.3). Ocorrida esta situação, deve-se empregar a técnica de Encapsulamento Genérico. Uma estrutura chamada Shim Header é posicionada entre o cabeçalho de enlace e sua carga (datagrama IP) e armazena o rótulo associado ao datagrama (Figura 4). Esta estrutura possui quatro tipos de campos como mostra a Figura 3.

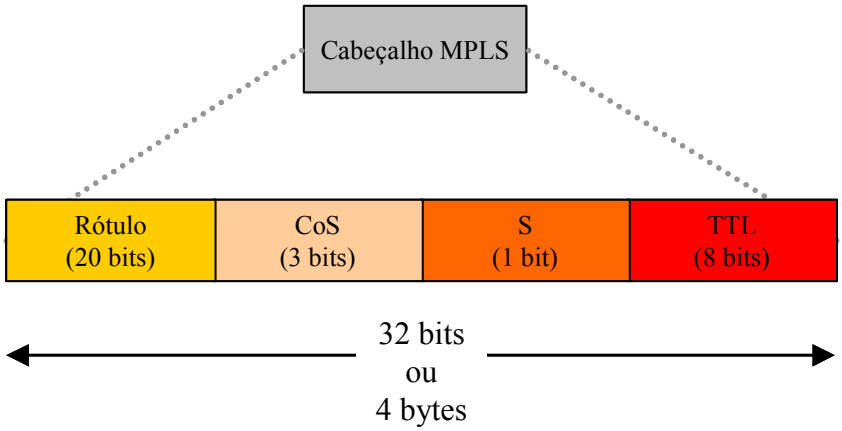


Figura 3 - Cabeçalho do MPLS.

Os 32 bits do cabeçalho MPLS contém os seguintes campos:

O campo (Rótulo – 20 bits) transporta o valor atual do rótulo MPLS;

O campo (CoS – 3 bits) pode afetar a fila e aplicar algoritmos para descartar pacotes, além de especificar como ele é transmitido através da rede;

O campo Stack (S – 1 bit) suporta a hierarquia de empilhamento de rótulo;

O campo TTL (Time-To-Live – 8 bits) provê a convencional funcionalidade TTL do IP (é um contador usado para limitar a vida útil do pacote).

Para redes ATM, segundo **[DAV01]** e **[HÄG98]**, o rotulamento ocorre no campo Identificador de Vias Virtuais/Identificador de Canal Virtual (VPI/VCI) (Figura 5). Já o cabeçalho do IPv6 ocorre no nível de enlace (Figura 6), no campo de Flow Label **[DEE98]**, e por último, em dispositivo Frame Relay definido por **[CON01]**, o rótulo deverá ser codificado no campo DLCI (Data Link Connection Identifier) no cabeçalho de nível 2 (Figura 6).

Figura 4 - Localização do rótulo MPLS entre os cabeçalhos camada 2 e 3.

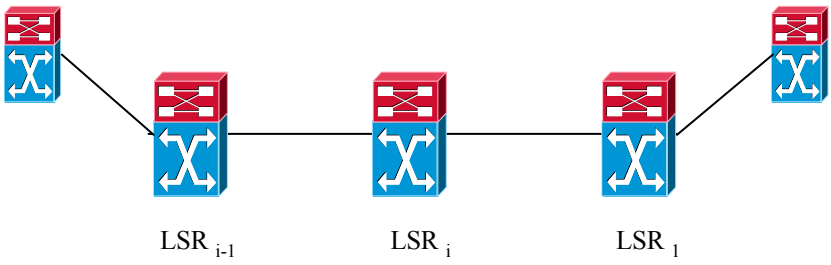
Figura 5 - Localização do rótulo MPLS no cabeçalho camada 3 - Rede.

Figura 6 - Localização do rótulo MPLS no cabeçalho camada 2 - Enlace.

2.4. Distribuição de Rótulo

Na estrutura MPLS, a atividade de distribuição de rótulo é de suma importância para o encaminhamento de pacotes e, para esta atividade, é necessária a utilização de um protocolo. Pode-se trabalhar com duas formas de protocolos: os BGP **[REK01]** ou RSVP-TE **[AWD01]** existentes adaptados e a criação de um protocolo com todas as funcionalidades necessárias para que tal tarefa seja bem desempenhada. Tal atividade deve ser em comum acordo entre os roteadores com relação ao significado dos rótulos utilizados para encaminhar o tráfego entre e através deles.

Quando um par de LSRs está de comum acordo em usar um rótulo qualquer para indicar a transmissão de um para o outro com relação a uma FEC, diz-se que o LSR emissor está em upstream na transmissão e o LSR receptor está em downstream conforme mostra a Figura 7.



LSRi-1 = vizinho upstream de LSRi
LSRi = vizinho upstream de LSRi+1
LSRi+1= vizinho downstream de LSRi
LSRi = vizinho downstream de LSRi-1

Figura 7 - Relação de vizinhança no MPLS.

Na especificação do MPLS, a determinação do rótulo é sempre feita pelo LSR downstream no caminho do pacote. A escolha do rótulo pode ser feita segundo uma requisição do LSR upstream ou diretamente através de iniciativa do LSR downstream no envio do pacote. Os protocolos de distribuição de rótulos são responsáveis pela troca de mensagens entre os LSRs. No sentido de gerar as tabelas de rótulos (Figura 8), cada vez que se associa um rótulo a uma FEC, este passa a ter certos atributos que são definidos também pelo protocolo de distribuição. A arquitetura MPLS suporta diferentes tipos de protocolo de distribuição de rótulos. Outros protocolos de roteamento ou serviços como o BGP **[REK95]** e o RSVP **[BRA97]** foram modificados para padronizarem mecanismos de indicação de rótulos.

Ao invés de um único rótulo, o MPLS permite que os pacotes de dados carreguem vários rótulos (Figura 9), de modo que o último rótulo a ser colocado no pacote deve ser o primeiro a sair. Um pacote de n rótulos é dito de profundidade n, para todo $n \geq 0$. O rótulo mais baixo é considerado rótulo 1 e os demais seguem uma ordem crescente. O processamento de cada nível de rótulo segue o mesmo princípio e é realizado de maneira completamente independente.

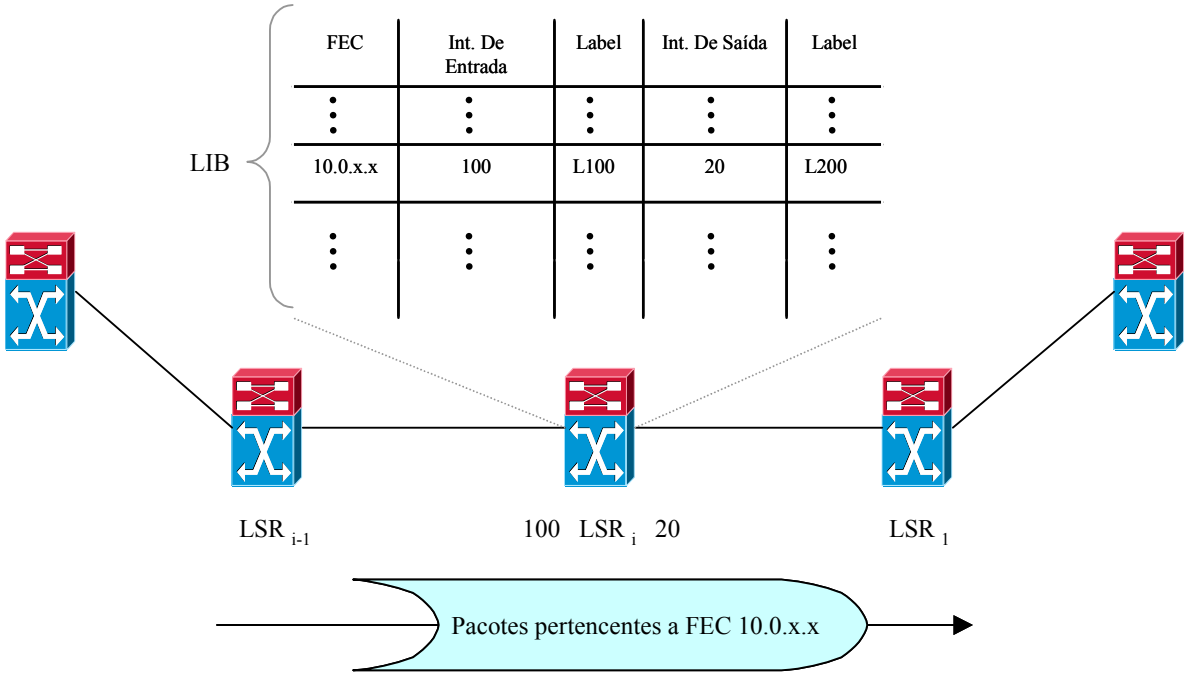


Figura 8 - Estabelecimento de LSPs: Distribuicao de Rótulo Doustream.

A arquitetura MPLS oferece a possibilidade ao pacote de carregar mais de um rótulo, formando assim uma pilha de rótulos. A ordem de retirada dos rótulos é: o último que entra deverá ser o primeiro a sair. A questão fundamental envolvendo a pilha de rótulos é: a decisão de encaminhamento realizada nos LSRs será sempre baseada no rótulo que se encontra no topo da pilha. Se um pacote tem uma pilha com profundidade α, então será assumido que:

Se $\alpha = 0$, a pilha está vazia;

Se $\alpha > 0$, o rótulo armazenado no seu topo tem nível α e o nível final será 1.

Os LSRs utilizam a tabela NHLFE (Next Hop Label Forwarding Entry) no procedimento de envio de pacotes rotulados. Pode-se encontrar as seguintes informações na tabela:

i) Qual é o próximo salto;

ii) As possíveis operações realizadas na tabela de rótulos do pacote:

a) Retirada de rótulo da pilha;

b) Substituição do rótulo que se encontra no topo da pilha por um novo;

c) O processo de substituição seguido de um impulso (push) em um (mais de um) novo(s) rótulo(s) na pilha;

Além dessas informações, consta também na NHLFE, o encaminhamento de nível de enlace a ser usado para o envio, a forma de codificar a pilha e qualquer outra informação necessária para enviar o pacote de forma precisa.

Às vezes, a NHLFE indica que o próximo nó para enviar o pacote é o próprio LSR emissor. Nesse caso, é feita uma operação de pop: retira-se o rótulo no topo da pilha e, se o pacote ainda tenha rótulo, este deve ser interpretado como se tivesse chegado por outro LSR. Caso contrário, o pacote ficará apenas com o endereço IP nativo, o qual será interpretado normalmente para realizar o seu encaminhamento. Nota-se portanto que o MPLS, em certas situações, faz uso do roteamento IP convencional.

Os procedimentos ILM (Incoming Label Map) e FTN (FEC-to-NHLFE) são responsáveis pelo mapeamento dos pacotes recebidos por LSRs em conjunto com NHLFEs. O ILM mapeia os rótulos recebidos nas respectivas NHLFEs. É utilizado no recebimento de pacotes rotulados. Já o FTN, organiza as FECs em grupos de NHLFEs. Este processo é utilizado no encaminhamento de pacotes que são recebidos sem rótulos, mas que devem ser enviados com rótulos.

Indiferente do método de mapeamento, se o grupo de NHLFEs retem mais de um elemento, uma única entrada deve ser escolhida para a sua transmissão. Para isto, o método comutação de rótulo (Label Swapping), inicialmente analisa um pacote com a finalidade de saber se o mesmo está ou não rotulado. Para um pacote rotulado, usa-se o ILM para mapear este rótulo em uma NHLFE e, usando suas informações, determina para onde direcioná-lo (interface de saída). Já para pacote sem rótulo, o LSR analisa o cabeçalho da camada de rede para determinar a FEC do pacote. Ele então usa a FTN para mapear a FEC na NHLFE correspondente. De posse das informações da NHLFE, o LSR determina para onde encaminhá-lo e executa uma operação no topo da pilha de rótulo. Neste caso, não seria prudente executar um pop em um pacote não-rotulado.

2.5. Transmissão de Pacotes em Rede MPLS

Uma transmissão de pacotes em rede MPLS ocorre em circuitos virtuais, denominados de Label Switched Path (LSP). A Figura 9, ilustra um LSP de n LSRs que faz o roteamento dos pacotes pertencentes a uma determinada FEC. Cada LSP, $<R_1, ..., R_n>$ de nível m, possui as seguintes propriedades:

i) R_1 é o LSR de ingresso do LSP (LSP ingress), tendo como função, a inserção do rótulo inicial nos pacotes do LSP a que pertence, ocasionando pilhas de profundidade m;

ii) Para todo i, 1 < i < n, o pacote P possui uma pilha de rótulos de profundidade m, quando recebido pelo R_i.

iii) Para todo i, 1 < i < n, R_i transmite o pacote P para R_{i+1}, por meio da arquitetura MPLS, podendo assim usar o rótulo do topo da pilha como índice em ILM.

Para todo i, 1 < i < n, se um pacote P for recebido e reenviado por um sistema S depois de transmitido por R_i, antes de ter sido recebido por R_{i+1} então a decisão de passagem em S não poderá usar o rótulo de nível da pilha nem o cabeçalho do nível de rede. Isto só será possível se S for parte de uma sub-rede de comutação intermediária entre R_i e R_{i+1}. O sistema S poderá fazer a passagem usando outras informações não descritas acima ou um rótulo em nível superior na pilha MPLS.

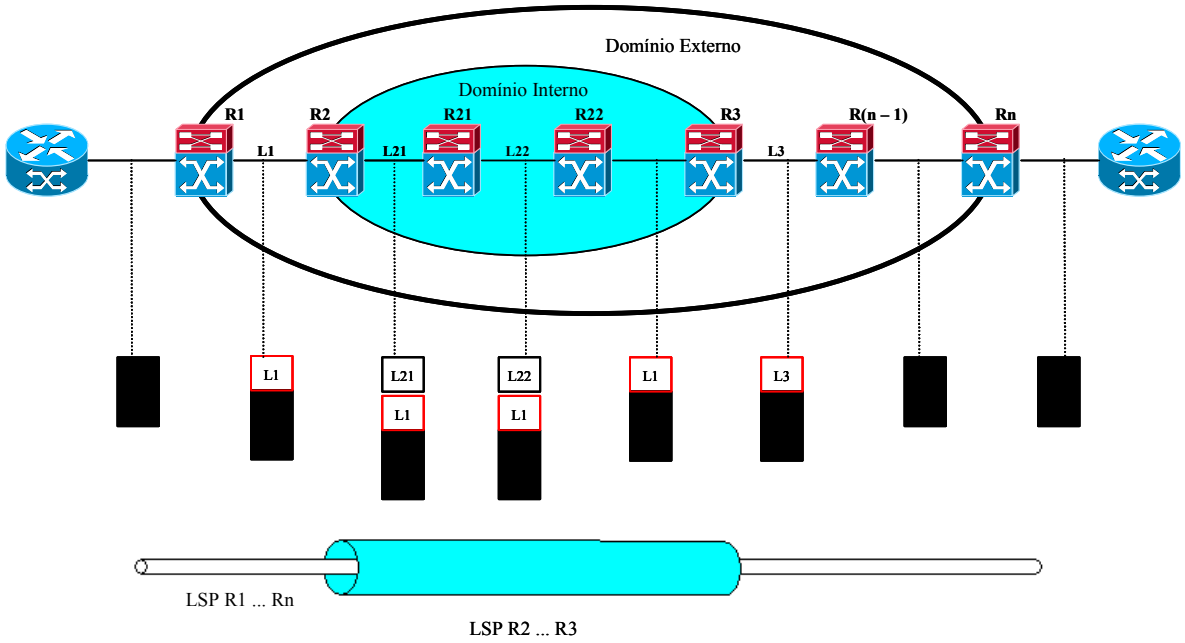


Figura 9 - Empilhamento de Rótulos: Túneis e Hierarquias.

A operação de retirada do rótulo pelo penúltimo salto é de extrema importância nos LSPs. Esta ação, quando executada, permite que o LSR egresso realize uma única busca na pilha de rótulos para encaminhar o pacote, esteja este rotulado ou não, impossibilitando a redundância de operação. Para que tal operação ocorra, o pacote deverá solicitar explicitamente ao *upstream*.

O roteamento de nível 3 para ser realizado, necessita das informações contidas na NHLFE, como visto anteriormente. A determinação do caminho dentro de um circuito depende

das informações gravadas nas NHLFEs dos LSRs ao longo do LSP, que podem ser geradas a partir das rotas criadas pelo algoritmo de roteamento do protocolo de rede.

Para evitar comportamentos indesejáveis na rede tais como ocorrência de loops, o LSR deve descartar pacotes com rótulos que não pertencem ao grupo de rótulos de entradas válidos.

Entre os protocolos de roteamento IP, alguns têm funcionamento dinâmico. As FECs que se baseiam nos prefixos de endereços distribuídos por estes protocolos controlam um LSP nas formas ordenado e independente. Na forma ordenada, o LSR toma a decisão de associar um rótulo a FEC somente se ele for o LSR egresso da FEC ou se ele recebeu uma associação de rótulo do próximo salto correspondente à FEC. A sua principal utilização acontece quando uma determinada FEC necessita do mesmo conjunto de propriedades durante o caminho a percorrer.

Já na forma independente, a alocação de rótulos para as FECs é uma decisão autônoma do LSR, bem como a distribuição da alocação aos seus pares *upstream*. A sua utilização é para casos onde alguns LSRs necessitem comutar o tráfego de uma FEC, antes mesmo de ter ocorrido o completo estabelecimento do LSP e o tráfego poderá não usufruir do mesmo conjunto de propriedades.

A noção de classes de endereçamento é utilizada pelo IP através dos endereços hierárquicos, que servem para dividir os grupos de endereços nas tabelas de roteamento. A mesma noção, no MPLS, corresponde às FECs. Contudo, a atribuição de uma FEC para cada grupo de endereçamento IP, nem sempre corresponde à melhor estratégia para a criação dos circuitos MPLS, porque não é raro ter grupos diferentes de endereços IP que possuam a mesma interface de saída em um roteador. Se forem criadas FECs diferentes em um LSR que levem ao mesmo LSR adjacente, haverá uma carga de sinalização para estabelecimento e controle de LSP desnecessária. Logo, foi proposto em **[ROS01]** um mecanismo de agregação para permitir que grupos de FECs que possuam um destino em comum no escopo de um LSR sejam integrados em uma única FEC.

Uma estrutura de agregação pode ter diferentes níveis de granularidade. A granularidade mais fina corresponde a uma FEC para cada conjunto distinto de endereços IP de um LSR, e a granularidade mais espessa corresponde a existência de uma única FEC para cada interface de saída, possivelmente agrupando vários conjuntos diferentes de endereços IP. Quando se utiliza controle ordenado de LSPs, a granularidade escolhida para a determinação de rótulos deve ser a mesma do nó downstream. Quando o controle é Independente, é possível ter LSRs adjacentes se

comunicando com FECs de granularidades diferentes. Se um LSR envia pacotes para um outro que tenha uma granularidade mais espessa que ele, não há problemas, pois o LSR receptor terá apenas que mapear os pacotes de entradas adequadamente para suas FECs. Caso contrário, se as FECs do emissor forem mais espessos do que do LSR receptor, haverá incompatibilidade e o LSR receptor terá que modificar suas FECs. Prefere-se normalmente redistribuir as FECs do LSR emissor para que fiquem com a mesma granularidade (mais fina) das FECs do nó receptor.

2.6. Determinação de Rota

O MPLS essencialmente trabalha com duas políticas para determinar qual será a rota durante o processo de criação do LSP para uma FEC. As opções de políticas do MPLS são as seguintes:

- ✓ Análise do cabeçalho do pacote em todos os roteadores do percurso. Esta política é semelhante ao procedimento de roteamento executado no modelo IP tradicional, conhecido por roteamento *hop-by-hop*. Já o MPLS analisa apenas o rótulo.
- ✓ Criação explícita de um LSP (caminho) normalmente a partir do LSR de ingresso ou egresso, determina explicitamente quais são os LSRs do caminho por onde passarão os pacotes.

Neste tipo de procedimento, diferentes LSPs podem ser determinados para cada tipo de tráfego. Além disso, as informações de roteamento podem ser distribuídas em cada LSR, eliminando, a menos do rótulo, a necessidade de outras informações no pacote.

O MPLS, ao oferecer a possibilidade de se criar explicitamente uma rota, ele propicia também QoS e TE. Com isto, pode-se criar rotas e fazer re-roteamento de tráfego com base em diversos fatores, por exemplos, largura de banda disponível, taxa de perda de pacotes, carga na rede ou qualquer outro fator necessário para manter um bom nível de QoS.

A política baseada em restrições não é encontrada na maioria dos roteadores IP (que se baseiam na métrica do menor caminho), mas pode-se aplicá-la para fazer classificação de fluxos, realização de divisão das classes de serviços e utilização de rótulos diferentes para representar fluxos com prioridades distintas. Esta abordagem pode ser considerada uma forma de prover QoS.

Segundo Awduche **[AWD99]**, o MPLS pode utilizar outros métodos para produzir QoS baseados em critérios de classificação de fluxo com as informações de origem/destino de pacote IP, número de porta do protocolo TCP/UDP, bits TOS do pacote IP (Serviços Diferenciados) ou oferecer qualidade de serviço com reserva de recurso (Serviço Integrado) com o uso do protocolo RSVP.

2.7. Aplicações MPLS

Os serviços oferecidos atualmente pela Internet têm expandido com grande rapidez, provocando uma necessidade cada vez mais crítica de banda passante no backbone da rede. Aplicações de multimídia em tempo real tornam-se cada vez mais factíveis e espalham-se por toda a rede. Atualmente existem duas aplicações para MPLS em redes de provedores de serviços da Internet (ISPs) para garantir um bom funcionamento das aplicações.

2.7.1. Engenharia de Tráfego (TE)

A engenharia de tráfego é, atualmente, a principal aplicação do MPLS devido a possibilidade de controlar o fluxo de tráfego na rede, com o objetivo de reduzir problemas de congestionamento e conseguir uma utilização homogênea dos recursos disponíveis. A engenharia de tráfego vai contra o processo de roteamento pelo caminho mais curto através dos algoritmos IGPs[2]. Eles possuem algumas limitações, no que diz respeito a tráfego e controle.

O IETF tem duas propostas para engenharia de tráfego. A primeira é denominada Engenharia de Tráfego Adaptativa com MPLS (MPLS Adaptive Traffic Engineering - MATE). Esta proposta **[ELW01]**, visa estabelecer múltiplos LSPs entre uma dada origem e um destino, empregando-se roteamento explicito para o estabelecimento destes caminhos, seja ele através do LDP ou do RSVP. O LSR de ingresso pode então distribuir o tráfego entre os diversos LSPs, de maneira a balancear o tráfego através da rede e possibilitar uma equalização no uso dos recursos.

A segunda proposta é uma Arquitetura para Provedores de Serviços diferenciados e Engenharia de Tráfego (PASTE) **[LI98]** para os ISPs. A PASTE é fundamentada em MPLS e o RSVP para criar arquiteturas escaláveis de gerência de tráfego para suportar serviços diferenciados.

[2] Algoritmo que os roteadores internos usam quando fazem intercâmbio de acessibilidade de rede e de informações sobre roteamento.

Ao aplicar técnicas de TE, pode-se observar os seguintes grandes benefícios: minimização de pontos de congestionamento na rede, fácil re-roteamento dos fluxos em caso de falha e, consequentemente, a diminuição da perda de pacotes, atraso e "jitter". Com este conjunto de benefícios torna-se fácil para os provedores de serviços e acessos a Internet (ISP) oferecerem QoS aos seus usuários. O desdobramento da TE é apresentado no próximo capítulo.

2.7.2. VPN – Rede Virtual Privada

Visando minimizar custos e maximizar os recursos tecnológicos, as empresas procuram alternativas para solucionar as limitações apresentadas na infra-estrutura de redes privadas corporativas através da utilização da Internet para trafegar os seus dados.

Desta idéia surgiu o conceito de Rede privada virtual ou VPN (Virtual Private Network) que consiste de uma solução simples, flexível e poderoso mecanismo de tunelamento oferecido pelos ISPs. Ela é desenvolvida nos moldes de uma rede geograficamente distribuída ou WAN. Conforme Tanenbaum **[TAN97],** ela pode abranger uma ampla área geográfica, freqüentemente um país ou continente com todos atributos de segurança, gerenciamento e processamento (Figura 10). Para criar uma VPN, deve-se analisar os seguintes fatores: compatibilidade, disponibilidade, interoperabilidade e segurança.

A compatibilidade ocupa-se do método de tornar a rede privada compatível com a Internet, usando as seguintes opções:

- ✓Conversão dos endereços privados em Internet (publico);
- ✓Gateways IP;
- ✓Segundo Muthukrishnan **[MUT00],** tunelamento no MPLS atende bem as necessidades, que consiste em encapsular os pacotes no nó de origem e transmiti-lo, via Internet, através do LSP denominado túnel até o nó de destino onde vai acontecer o desencapsulamento do mesmo. Este encapsulamento protege dos usuários não-autorizados, usando técnicas de criptografia.

Os túneis podem ser estáticos ou dinâmicos. Túneis estáticos, que permanecem ativos por longos períodos de tempo, são apropriados para VPNs site-para-site (ou Lan-para-Lan) enquanto que túneis dinâmicos são ativados apenas quando requer tráfego, sendo apropriados para VPNs cliente-para-Lan.

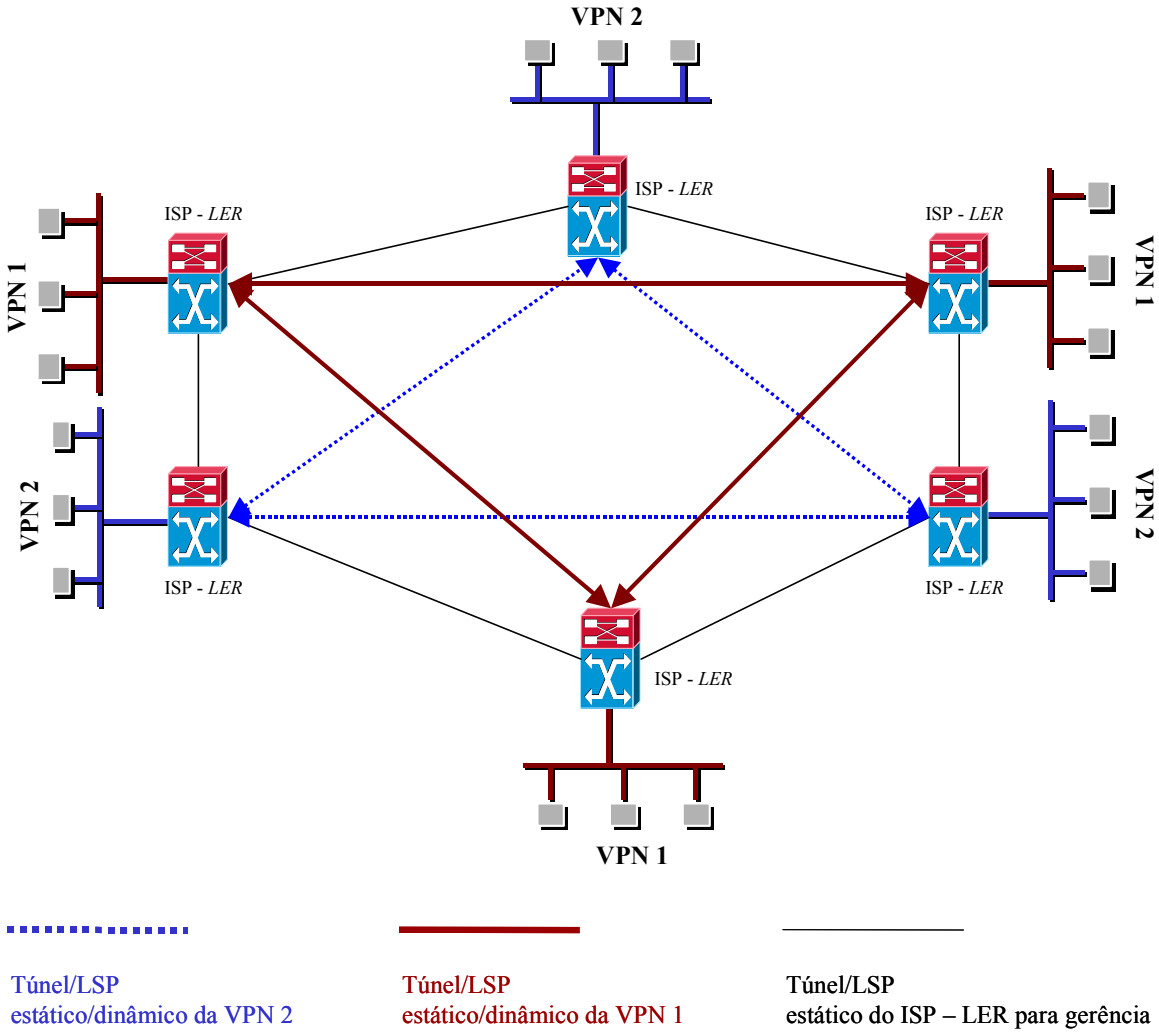


Figura 10 - MPLS propiciando a criação de VPNs.

3

Engenharia de Tráfego e MPLS

Introdução – 3.1. Engenharia de Tráfego – 3.2. Desempenho Orientado a Tráfego e Recursos – 3.3. Engenharia de Tráfego e Problemas – 3.4. Atributos e Características do Tronco de Tráfego – 3.5. Roteamento Baseado em Restrições – 3.6. Enquadramento para Serviços com Garantias.

A Internet hoje é e será, por um longo período, o atrativo na área de comunicação de dados. O serviço de Engenharia de Tráfego propicia a melhor utilização dos recursos da rede, exercendo influência direta sobre o roteamento e garantindo a banda necessária para uma determinada aplicação.

3.1. Engenharia de Tráfego

A engenharia de tráfego é a tarefa de realizar o mapeamento dos fluxos de tráfego em uma infra-estrutura física de transporte, de modo a atender critérios definidos pela operação da rede. Ao ter como enfoque a otimização do desempenho da rede, em relação a aplicação de tecnologias e princípios científicos para a medição, modelação, caracterização e controle de tráfego da Internet, ela se tornou um instrumento indispensável nos Sistemas Autônomos, devido o alto custo de recursos de rede e por causa da natureza comercial e competitiva da Internet. Estes fatores enfatizam a necessidade de uma máxima eficiência operacional. A engenharia de tráfego procura maximizar o desempenho orientado a tráfego ou orientado a recursos.

3.2. Desempenho Orientado a Tráfego e Recursos

Desempenho orientado a tráfego, inclui aspectos que aumentam o QoS de fluxos de tráfego. Trabalhando na Internet com modelo de serviço melhor esforço (classe única), os objetivos de desempenho orientados a tráfego incluem: minimização de perda de pacotes, minimização de atrasos, maximização de processamento e execução de acordos de nível de serviço (Service Level Agreements - SLAs), sendo a minimização da perda de pacotes um dos mais importantes objetivos de desempenhos orientados a tráfego. Já o de desempenhos orientados

a recursos, incluem aspectos tocantes a otimização da utilização de recursos de rede. Gerenciamento eficiente de recursos de rede é o veículo para obtenção desses objetivos. Em particular, devemos evitar que subconjuntos de recursos de rede não se tornem super-utilizados e congestionados enquanto outros subconjuntos, ao longo de possíveis caminhos alternativos, permanecem sub-utilizados. Assim, uma função central de engenharia de tráfego é administrar eficazmente recursos de banda.

Minimizar congestionamento é um objetivo tanto de desempenhos orientados a tráfego quanto a recursos. O interesse recai sobre problemas de congestionamento que são prolongados, ao invés de congestionamentos passageiros que resultam de rajadas instantâneas. Congestionamento, geralmente se manifesta em dois cenários: quando os recursos de rede são insuficientes ou inadequados para acomodar a carga oferecida e quando os fluxos de tráfego são ineficientemente mapeados sobre recursos disponíveis, fazendo com que os subconjuntos de recursos de rede se tornem super-utilizados, enquanto outros permanecem pouco utilizados.

Segundo Awduche **[AWD01]**, o primeiro tipo de problema de congestionamento pode ser resolvido ou por expansão da capacidade, ou por aplicação de técnicas clássicas de controle de congestionamento, ou ambos. Técnicas clássicas de controle de congestionamento tentam regular a demanda de forma que o tráfego se ajuste sobre os recursos disponíveis, e incluem: limitação de taxa, controle de fluxo de janela, gerenciamento de filas de roteamento, controle baseado em escalonamento e outros;

O segundo tipo de problema de congestionamento é resultante da alocação ineficiente de recursos e pode, normalmente, ser resolvido através de engenharia de tráfego. Em geral, o congestionamento resultante da má alocação de recursos pode ser reduzido adotando-se políticas de balanceamento de carga. O objetivo de tais estratégias é minimizar congestionamento através da alocação eficiente de recursos. Quando o congestionamento é minimizado através de uma alocação eficiente de recursos, a perda de pacotes diminui, assim como o atraso de trânsito, e aumenta a vazão agregada. Com isso, o usuário passa a perceber um significativo acréscimo de QoS na rede. Dessa forma, fica claro que balanceamento de carga é uma importante política de otimização do desempenho.

3.3. Engenharia de Tráfego e Problemas

A engenharia de tráfego em MPLS apresenta basicamente três problemas: como mapear pacotes em FECs, como mapear FECs em troncos de tráfego e como mapear troncos de tráfego sobre topologia de uma rede física através de LPSs **[AWD01]**.

Esta seção descreve as capacidades funcionais necessárias para apoiar a engenharia de tráfego sobre MPLS em grandes redes. As capacidades propostas consistem em um conjunto de atributos associados ao tronco de tráfego que, coletivamente, especificam suas características comportamentais, em conjunto de atributos associados com recursos que restringem a colocação de troncos de tráfego entre eles. Estes também podem ser vistos como atributos de restrição de topologia.

Um sistema de roteamento baseada em restrições é usado para selecionar caminhos para troncos de tráfego sujeito a restrições impostas pelos itens que dizem respeito ao desempenho.

O sistema de roteamento baseado em restrições não tem que ser parte do MPLS, porém, ambas tem necessidade de serem integradas. Os atributos associados com troncos de tráfego e com recursos, como também os parâmetros associados ao roteamento, coletivamente representam as variáveis de controle que podem ser modificadas por ação administrativa ou por agentes automatizados que controlam a rede para um estado desejado. Em uma rede operacional, é desejável que estes atributos possam ser dinamicamente modificados por um operador sem interromper operações de rede.

3.4. Atributos e Características do Tronco de Tráfego

O tronco de tráfego é um agregado de fluxos de tráfego que pertencem à mesma classe. Esta definição pode ser ampliada e permitir que troncos de tráfego incluam agregados de tráfego multi-classes **[AWD01]**. Em um modelo de serviço de classe única, como o da Internet atual, um tronco de tráfego poderia encapsular todo tráfego entre um LSR de ingresso e um LSR de egresso. Troncos de tráfego são objetos roteáveis (semelhante aos VCs do ATM). Um tronco de tráfego é distinto do LSP. Em contextos operacionais, um tronco de tráfego pode ser movido de um caminho para outro.

3.4.1. Tronco de Tráfego Unidirecional

Na prática, um tronco de tráfego pode ser caracterizado por seus LSRs de ingresso e de egresso. A FEC é mapeada sobre ele, assim como um conjunto de atributos que determinam suas

características comportamentais. Dois assuntos básicos possuem um significado particular: parametrização de troncos de tráfego e colocação de caminhos e regras de manutenção para troncos de tráfego.

3.4.2. Tronco de Tráfego Bidirecional

Embora troncos de tráfego sejam conceitualmente unidirecionais, em muitos contextos práticos, é útil iniciar simultaneamente dois troncos de tráfego com os mesmos pontos, mas que transmitem pacotes em direções opostas. Os dois troncos de tráfego são acoplados logicamente. Um tronco "forward", transmite tráfego de um nó de origem para um nó de destino. Já o tronco backward, leva tráfego do nó de destino para o nó de origem. O acoplamento definido chama-se **tronco de tráfego bidirecional** (Bidirecional Traffic Trunk, ou BTT) quando as duas condições seguintes forem verdades:

- Ambos os troncos de tráfego estão inicializados por uma ação atômica em um LSR, chamado de nó originador, ou por uma ação atômica em uma estação de administração de rede.
- Nenhum dos troncos de tráfego compostos pode existir sem o outro, ou seja, ambos são inicializados e destruídos em conjunto.

Também devem ser consideradas as propriedades topológicas dos BTTs. Um BTT pode ser topologicamente simétrico ou topologicamente assimétrico. Um BTT é topologicamente simétrico se seus troncos de tráfego constituintes são roteados pelo mesmo caminho físico, embora eles operem em direções opostas, porém, se os troncos de tráfego de componente são roteados por caminhos físicos diferentes, o BTT é topologicamente assimétrico.

Deve-se notar que troncos de tráfego bidirecionais são apenas uma conveniência administrativa. Na prática, as maiorias das funções da engenharia de tráfego podem ser implementadas usando-se apenas troncos de tráfego unidirecionais. A Figura 3.1 ilustra a relação entre os vários conceitos descritos anteriormente.

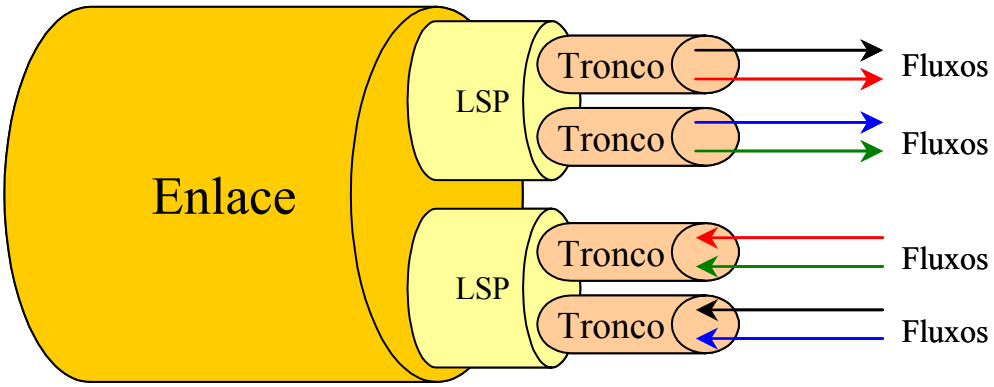


Figura 11 - Relação entre Fluxos, Troncos, LSPs e Enlaces.

3.4.3. Operações Básicas no Troco de Tráfego

As operações básicas em troncos de tráfego significativos aos propósitos da engenharia de tráfego são resumidas abaixo.

- Estabelecer: criar uma instância de um tronco de tráfego.
- Ativar: fazer com que um tronco de tráfego comece a passar tráfego. O estabelecimento e ativação de um tronco de tráfego são eventos logicamente separados, porém, eles podem ser implementados ou podem ser invocados como uma ação atômica.
- Desativar: fazer com que um tronco de tráfego pare de passar tráfego.
- Modificar Atributos: fazer com que os atributos de um tronco de tráfego sejam modificados.
- Re-encaminhar: fazer com que um tronco de tráfego possa mudar sua rota. Isto pode ser feito por ação administrativa ou automaticamente pelos protocolos subjacentes.
- Destruir: remover uma instância de um tronco de tráfego da rede e recuperar todos os recursos alocados a ele. Tais recursos incluem espaço de rótulo e possivelmente banda disponível.

As considerações acima são as operações básicas em troncos de tráfego. Operações adicionais também são possíveis como policiamento e acondicionamento do tráfego.

3.4.4. Monitoramento de Desempenho e Tarifação

As capacidades de monitoramento de desempenho e contabilidade são importantes para as funções de faturamento (billing) e caracterização de tráfego. As estatísticas de desempenho obtidas de sistemas de tarifação e de monitoração do desempenho podem ser usadas na caracterização de tráfego, otimização de desempenho e capacidade de planejamento da engenharia de tráfego. A capacidade para obter estatísticas ao nível de tronco de tráfego, deve ser considerada uma exigência essencial para engenharia de tráfego sobre MPLS.

3.4.5. Atributos do Tronco de Tráfego

Um atributo de um tronco de tráfego é um parâmetro que influencia suas características comportamentais. Atributos podem ser explicitamente atribuídos a troncos de tráfego por ação administrativa ou implicitamente nomeados pelos protocolos subjacentes, quando pacotes são classificados e mapeados em FEC no ingresso do domínio MPLS. Não importando como os atributos foram originalmente nomeados, para os propósitos da engenharia de tráfego, deverá ser possível modificá-los administrativamente. A combinação de parâmetros de tráfego e atributos de policiamento é análoga ao controle de parâmetros usados em redes ATM. A maioria dos atributos listados acima possuem análogos em tecnologias bem estabelecidas.

Prioridade e preempção podem ser consideradas como atributos relacionados, porque elas expressam certas relações binárias entre troncos de tráfego. Conceitualmente, estas relações binárias determinam a maneira pela qual tronco de tráfego interagem entre si, uma vez que eles competem por recursos de rede durante estabelecimento de caminhos e manutenção de caminhos.

3.4.6. Atributos de Parâmetro de Tráfego

Parâmetros de tráfego podem ser usados para capturar as características dos fluxos de tráfego a serem transportados pelo tronco de tráfego. Tais características podem incluir taxas de pico, taxa média, tamanho permissível da rajada, etc. Da perspectiva de engenharia de tráfego, os parâmetros de tráfego são importantes, porque eles indicam as exigências de recursos do tronco de tráfego. Isto é útil para alocação de recursos e prevenção de congestionamento através de políticas de prevenção. Com a finalidade de alocação de banda, pode ser calculado um único valor canônico de exigência de banda dos parâmetros de tráfego de um tronco.

3.4.7. Seleção de Caminho e Manutenção de Atributos

Seleção de caminho genérico e manutenção de atributos definem as regras de seleção de rota tomada por um tronco de tráfego, como também, as regras para manutenção de caminhos já estabelecidos. Caminhos podem ser calculados automaticamente pelos protocolos de roteamento ou eles podem ser definidos administrativamente por um operador da rede. Se não há exigência de recurso ou restrições associadas com um tronco de tráfego, então um protocolo baseado em topologia pode ser usado para selecionar seu caminho. Porém, se existirem exigências de recurso ou restrições de política, então um esquema de roteamento baseado em restrições poderia ser usado para seleção de caminho.

A administração de caminho consiste em todos os aspectos que pertencem à manutenção de caminhos percorridos por troncos de tráfego. Em alguns contextos operacionais, é desejável que uma implementação de MPLS possa se reconfigurar dinamicamente para adaptar a alguma noção de mudança em estado de sistema. Adaptabilidade e capacidade de recuperação são aspectos de administração dinâmica de caminhos.

3.4.7.1. Caminhos Explícitos Especificados Administrativamente

Um caminho explícito administrativamente especificado para um tronco de tráfego é aquele configurado por ação de um operador. Um caminho administrativamente especificado pode ser completo ou parcialmente especificado. Um caminho é completamente especificado se todos os saltos intermediários entre os pontos são indicados. Um caminho é especificado parcialmente se apenas um subconjunto de saltos intermediários é indicado. Neste caso, é necessário que os protocolos subjacentes completem o caminho. Devido a erros de operador, um caminho administrativamente especificado pode ser incompatível ou ilógico. Os protocolos subjacentes devem possuir capacidade de descobrir tais inconsistências e prover um "feedback" apropriado.

Um atributo de critério de caminho preferencial deve ser associado com caminhos explícitos especificados administrativamente. Esse atributo seria uma variável binária que indicaria se o caminho explícito configurado administrativamente é obrigatório ou não. Se um caminho explícito especificado administrativamente é selecionado com um atributo obrigatório, então somente esse caminho deve ser usado. Se um caminho obrigatório não é factível, do ponto de vista da topologia, ou se o caminho não pode ser instanciado, porque os recursos disponíveis

são inadequados, então ocorrerá uma falha no processo. Se um caminho é especificado como obrigatório então, sob qualquer hipótese, um caminho alternativo não poderá ser usado. Um caminho obrigatório que é instanciado com sucesso também é implicitamente fixado. Uma vez instanciado o caminho, ele não pode ser mudado, exceto através da sua exclusão e instanciação de um caminho novo.

Entretanto, se um caminho explícito especificado administrativamente é selecionado com o atributo regra de caminho preferencial não-obrigatório então, se possível, o caminho deve ser usado. Caso contrário, um caminho alternativo pode ser escolhido pelos protocolos subjacentes.

3.4.7.2. Hierarquia de Regras de Preferência Para Multi-Caminhos

Pode ser útil a habilidade de especificar administrativamente um conjunto de caminhos explícitos, candidatos para um determinado tronco de tráfego, e definir uma hierarquia de relações de preferência nos caminhos. Durante o estabelecimento do caminho, critérios preferenciais são aplicados para selecionar um caminho satisfatório da lista de candidato. Também, sob condições de falha, as regras de preferência são aplicadas para selecionar caminhos alternativos da lista de caminhos candidatos.

3.4.7.3. Atributos de Afinidade de Classes de Recursos

Atributos de afinidade entre classes de recursos podem ser usados, associados com um tronco de tráfego, para especificar a classe de recursos que estão para ser explicitamente incluídas ou excluídas do caminho do tronco de tráfego. Estes são atributos de policiamento que podem ser usados para impor restrições adicionais no caminho percorrido por um determinado tronco de tráfego. Podem ser especificados atributos de afinidade entre classes de recursos para um tráfego como uma sucessão de duplas (classe-recurso, afinidade).

O parâmetro de classe-recurso identifica uma classe de recurso na qual a relação de afinidade é definida com respeito ao tronco de tráfego. O parâmetro de afinidade indica a relação de afinidade, isto é, se os membros da classe de recurso estão ou não para serem incluídos ou excluídos do caminho do tronco de tráfego. Especificamente, o parâmetro de afinidade pode ser uma variável binária que leva um dos valores seguintes: (1) inclusão explícita, e (2) exclusão explícita.

Se o atributo de afinidade é uma variável binária, pode ser possível usar expressões booleanas para especificar as afinidades de classe de recurso associadas a um determinado tronco de tráfego. Se nenhum atributo de afinidade entre classes de recursos é especificado, então se espera que uma relação de afinidade "don't care" seja estabelecido entre o tronco de tráfego e todos os recursos, ou seja, não há exigência para incluir ou excluir explicitamente qualquer recurso do caminho do tronco de tráfego. Este deve ser o procedimento padrão.

Atributos de afinidade entre classes de recursos são construtores muito úteis e poderosos, porque eles podem ser usados para implementar uma variedade de políticas. Por exemplo, eles podem ser usados para conter certos troncos de tráfego dentro de específicas regiões topológicas da rede. Um sistema de roteamento baseado em restrições pode ser usado para calcular um caminho explícito para um tronco de tráfego sujeito a restrições de afinidade entre classes de recursos da seguinte maneira: para inclusão explícita, podar todos os recursos que não pertencerem às classes especificadas anteriores a computação de caminho. Para exclusão explícita, podar todos os recursos que pertencerem às classes especificadas, antes de executar computações de alocação de caminho.

3.4.7.4. Atributo de Adaptabilidade

Características de rede e estados mudam com o passar do tempo. Por exemplo, recursos novos se tornam disponíveis, recursos com falhas se tornam indisponíveis e recursos alocados são desalocados e deixando muitas vezes caminhos eficientes disponíveis. Em geral, os caminhos mais eficientes se tornam disponíveis. Então, da perspectiva da engenharia de tráfego, é necessário ter parâmetros de controle administrativos que possam ser usados para especificar como troncos de tráfego deve responder a este dinamismo. Em alguns casos, poderia ser desejável mudar dinamicamente os caminhos de certos troncos de tráfego em resposta a mudanças no estado da rede. Este processo é chamado re-otimização. Em outros casos, re-otimização poderia ser indesejável.

Um atributo de adaptabilidade é uma parte dos parâmetros de manutenção de caminho associada aos troncos de tráfego. O atributo de adaptabilidade associado com um tronco de tráfego indica se o tronco é sujeito a re-otimização, ou seja, um atributo de adaptabilidade é uma variável binária que leva um dos seguintes valores: (1) habilita re-otimização e (2) desabilita re-otimização.

Se re-otimização estiver habilitado, então um tronco de tráfego pode ser re-encaminhado por caminhos diferentes pelos protocolos subjacentes em resposta às mudanças no estado de rede (principalmente mudanças em disponibilidade de recurso). Reciprocamente, se re-otimização é inválido, então o tronco de tráfego é fixado ao caminho estabelecido e não pode ser re-encaminhado em resposta a mudanças no estado de rede.

A estabilidade é uma preocupação principal quando a re-otimização é permitida. Para promover estabilidade, uma implementação de MPLS não deveria ser muito refratária à dinâmica evolutiva da rede. Ao mesmo tempo, ela deve-se adaptar bem rapidamente de forma que o uso de recursos de rede possa ser feito de forma adequada. Isto implica que a freqüência de re-otimização deveria ser configurável administrativamente.

A re-otimização é distinta da capacidade de recuperação. Um atributo diferente é usado para especificar as características de capacidade de recuperação de um tronco de tráfego. Na prática, pareceria razoável esperar que troncos de tráfego sujeito a re-otimização fossem implicitamente elásticos para fracassos ao longo dos caminhos deles/delas. Entretanto, um tronco de tráfego que não esteja sujeito a re-otimização e cujo caminho não seja especificado administrativamente com um atributo obrigatório, também pode ser exigido que tenha uma capacidade de recuperação a falhas de enlaces e nos nós, ao longo de seu caminho estabelecido formalmente. Podemos dizer que adaptabilidade a evolução de estado através de re-otimização implica na capacidade de recuperação contra falhas, considerando que a capacidade de recuperação para fracassos não implica adaptabilidade através de re-otimização mudanças de estado.

3.4.7.5. Distribuição de Carga Entre Troncos de Tráfego Paralelos

Distribuição de carga por múltiplos troncos de tráfego paralelos entre dois nós, é uma consideração importante. Em vários contextos práticos, o tráfego agregado entre dois nós pode ser tal que nenhum enlace sozinho (caminho) possa levar a carga. Entretanto, o fluxo agregado poderia ser menos que o máximo fluxo permissível de um corte mínimo, que divide os dois nós. Neste caso, a única solução possível é dividir o tráfego agregado adequadamente em subfluxos e dirigi-los através de múltiplos caminhos entre os dois nós.

Em um domínio de MPLS, este problema pode ser resolvido através do instanciamento de múltiplos troncos de tráfego entre os dois nós, tal que cada tronco de tráfego leva uma porção

do tráfego agregado. Então, são necessários meios flexíveis de atribuição de carga para múltiplos troncos de tráfego paralelos que levam tráfego entre um par de nós.

Especificamente, de uma perspectiva operacional, em situações onde tronco de tráfego paralelos estão garantidos, seria útil ter algum atributo que pudesse ser usado para indicar a proporção relativa de tráfego a ser levado em cada tronco de tráfego. Os protocolos subjacentes mapearão a carga sobre os troncos de tráfego de acordo com as proporções especificadas. Também é desejável manter a ordenação dos pacotes pertencentes ao mesmo fluxo (mesmo endereço de origem, endereço de destino e número de porta).

3.4.8. Atributo de Prioridade

O atributo de prioridade define a importância relativa de troncos de tráfego. Se um sistema de roteamento baseado em restrições for com MPLS, então prioridades se tornam muito importantes, porque eles podem ser usados para determinar a ordem na qual a seleção de caminho será feita para troncos de tráfego no estabelecimento de conexões e em casos de falha. Prioridades também são importantes em implementações que permitem preempção, porque os atributos de prioridade podem ser usados para impor uma ordem parcial no conjunto de troncos de tráfego de acordo com quais políticas de preempção possam ser atualizadas.

3.4.9. Atributo de Preempção

O atributo de preempção determina se um tronco de tráfego pode fazer preempção em outro tronco de tráfego de um determinado caminho e se outro tronco de tráfego pode fazer preempção em um tronco de tráfego específico. Preempção é útil para os objetivos de desempenho orientados a tráfego e os orientados a recurso. Pode ser também usada para assegurar que troncos de tráfego de alta prioridade sempre possam ser roteados por caminhos relativamente favoráveis dentro de um ambiente de serviços diferenciados. Preempção também pode ser usada para implementar várias políticas de restauração priorizadas seguidas de eventos de falha. O atributo de preempção pode ser usado para especificar quatro modos de preempção para um tronco de tráfego: i) habilitado a fazer preempção, ii) não-habilitado a fazer preempção ou, iii) habilitado a sofrer preempção, e iiii) não-habilitado a sofrer preempção. Um tronco habilitado a fazer preempção pode atuar em troncos de tráfego de menor prioridade. Um tráfego especificado como não habilitado a sofrer preempção não pode ser afetado por qualquer outro tronco, não importando suas prioridades relativas. Um tronco de tráfego designado como

habilitado a sofrer preempção pode sofrê-lo por troncos de prioridade mais altos que estejam habilitados a fazerem preempções.

Preempção não é considerada um atributo obrigatório debaixo do modelo de serviço melhor-esforço da Internet atual, embora ele seja útil. Porém, em um cenário de serviços diferenciados, a necessidade para preempção fica mais forte. Além disso, nas emergentes arquiteturas ópticas de "internetworking", onde um pouco das funções de proteção e restauração podem ser migradas da camada óptica para elementos de rede de dados (como "Gigabit" e "Terabit" LSRs) para reduzir custos, podem ser usadas estratégias de preempção para reduzir o tempo de restauração de troncos de tráfego de alta prioridade sob condições de falha.

3.4.10. Atributo de Capacidade de Recuperação

O atributo de capacidade de recuperação determina o comportamento de um tronco de tráfego sob condições de falha, ou seja, quando uma falha acontece ao longo do caminho pelo qual um tronco de tráfego percorre os seguintes problemas básicos precisam ser enviados debaixo de tais circunstâncias: (1) descoberta de falha, (2) notificação de falha, (3) recuperação e restauração do serviço. Para tal, uma implementação de MPLS terá que incorporar mecanismos para resolver estes assuntos. Várias políticas de recuperação podem ser especificadas para troncos de tráfego cujos caminhos estabelecidos são acometidos de falhas.

3.4.11. Atributo de Policiamento

O atributo de policiamento determina as ações que deveriam ser tomadas pelos protocolos subjacentes quando um tronco de tráfego fica fora das especificações (non-compliant), ou seja, quando um tronco de tráfego excede seu contrato especificado nos parâmetros de tráfego. Geralmente, atributos de policiamento podem indicar se um tronco de tráfego, funcionalmente diminuído, terá sua taxa limitada, marcada, ou simplesmente encaminhada sem qualquer ação de policiamento. Se o policiamento for usado, então adaptações de algoritmos estabelecidos como o GCRA do ATM Forum podem ser usado para executar esta função.

Policiamento é necessário em muitos casos operacionais, mas é bastante indesejável em alguns outros. Em geral, é normalmente preferível manter a ordem no ponto de ingresso de uma rede e minimizar o policiamento mantendo-o dentro da ordem do núcleo, exceto quando restrições ditarem o contrário. Da perspectiva de engenharia de tráfego, é preferível poder habilitar ou desabilitar administrativamente policiamento de tráfego para cada tronco de tráfego.

3.5. Roteamento Baseado em Restrições

Um aspecto importante da engenharia de tráfego é a preocupação com os procedimentos e mecanismos de suporte empregados para calcular um caminho pela topologia de rede de um domínio. Uma vez que um caminho satisfatório é calculado, um protocolo de sinalização é usado para estabelecer uma LSP ao longo daquele caminho, e o tráfego que satisfaz uma determinada FEC é enviado pelo LSP. Em geral, computação do caminho para um LSP visa satisfazer um conjunto de exigências associadas com o LSP e podem levar em consideração um conjunto de restrições impostas por políticas administrativas e o estado prevalecente da rede que normalmente relaciona dados de topologia e disponibilidade de recursos. Computação de um caminho modelado que satisfaz o conjunto arbitrário de restrições é chamado roteamento baseado em restrições ou (*Contraint Based Routing*) **[XIA99]**. Caminhos calculados para um LSP baseados em restrições podem diferir do menor caminho, que normalmente seria determinado pelo IGP de um domínio.

A engenharia de tráfego manipula os parâmetros associados com roteamento baseado em restrições para atingir objetivos específicos de desempenho baseado nas políticas e nas preferências de um provedor de serviço. A computação de um caminho adequado às restrições é computacionalmente caro. Por isto, algoritmos online, que são usados para este propósito, tendem a ser heurísticas que produzem resultados razoáveis, porém sub-ótimos em tempo real. Em certos casos, explícita intervenção humana poderia ser necessária na forma de caminhos modelados, que são criados offline, e então inicializados por configuração administrativa.

Devido ao fato de que o componente humano não escala bem com tamanho da rede e não possua uma resposta previsível, seria bom automatizar a computação do caminho, o máximo possível, para os casos comuns. No evento em que os casos comuns não são suficientes em longo prazo para alcançar objetivos de engenharia de tráfego, as exigências declaradas neste capítulo são suficientemente flexíveis de forma que, complexidade adicional na computação do caminho para que possa ser incorporado no futuro. Isto é possível, porque toda a computação de caminho é executada pelo originador do LSP (o head-end). Como tal, a interoperabilidade é restrita à distribuição de informações no IGP e aos mecanismos básicos de protocolos de sinalização.

3.5.1. Requisitos de Arquitetura

As exigências de arquitetura consistem em dois componentes básicos: (1) um processo de roteamento baseado em restrições que é implementado nos LSRs, que podem servir como head-end para LSPs, e (2) um conjunto de mecanismos para disseminação e manutenção de informação da topologia de estados. Informação da topologia de estados pode ser disseminada estendendo um IGP de forma que este carregue detalhes adicionais relativos ao estado da rede. Como o IGP é somente usado para transporte de informação sobre a rede para o head-end, os atributos do IGP são um tanto restrito.

A outra categoria relevante de IGPs é a classe de protocolos *link-state* (protocolos de estado de ligação). Há dois protocolos em uso: OSPF **[MOY98]** e IS-IS. Em particular, protocolos *link-state* é a única classe conhecida de protocolos que são capazes de disseminar toda a topologia. Protocolos *link-state* são particularmente úteis para aplicações de engenharia de tráfego, porque eles podem ser facilmente estendidos para distribuir informação adicional relativa ao estado da rede.

A idéia é adicionar novos parâmetros ao IGP, de forma que ele possa distribuir informações adicionais sobre a rede. Daí a opção será por um IGP extensível. Além disso, por razões práticas, o mecanismo de extensão deve ser *Backward Compatible*, isto é, as implementações existentes têm que continuar funcionando após as mudanças necessárias para suportar engenharia de tráfego. Especificamente deve dar-se preferência por um determinado domínio IGP que permaneça operacional, enquanto povoado com dispositivos híbridos, sendo que apenas um subconjunto suporta as extensões de engenharia de tráfego.

Felizmente, o OSPF e o IS-IS satisfazem esta exigência. O protocolo OSPF **[MOY98]** provê um LSA (*Link State Advertisement*) opaco que pode ser usado para levar informações arbitrárias. LSAs opacos são ignorados por sistemas que não reconhecem seus conteúdos. O IS-IS codifica toda informação nos seus pacotes *link-state* em duplas descritas por um tipo, duração e valor TLVs (*Type-Length-Value)*. TLVs que não são reconhecidos por um sistema são ignorados. Isto torna possível adicionar informações para cada protocolo de uma maneira compatível com versões anteriores.

Protocolos *link-state* têm uma negligência inerente quando usados para engenharia de tráfego. Devido ao fato de que protocolos *link-state* distribuem grandes bancos de dados de informação e tentam fazer de uma maneira oportuna, o tamanho da área de distribuição da base

de dados deve necessariamente ser restringido. As limitações de escalabilidade dos algoritmos de *flooding* utilizados pelos protocolos *link-state* implicam que um limite máximo (*upper bound)* deve ser imposto na quantia de informação da topologia de estados que pode ser distribuída para engenharia de tráfego.

Considerações adicionais pertencem aos eventos de gatilho que fazem com que a informação da topologia de estados seja distribuída sem freqüência relativa de tais distribuições. Resolver o problema de controle da distribuição da informação da topologia de estados, que é inerente aos protocolos *link-state*, vai além do âmbito deste trabalho. Porém, é necessária uma solução que venha prover engenharia de tráfego em todo um domínio largo junto com um IGP hierárquico.

3.5.2. Requisitos de Distribuição de Dados

Além da exigência por distribuir a topologia básica de um domínio com engenharia de tráfego, é necessário que o IGP transporte outra informação sobre a rede. Muito desta informação provê uma granularidade mais fina sobre os *enlaces* na rede.

Para determinar se as exigências de largura de banda de um ISP serão acomodadas em um dado enlace, é necessário saber a sua largura de banda disponível naquele enlace. Conhecimento prévio da largura de banda já consumida no enlace também é bastante útil. Se os administradores do domínio impuserem uma restrição na proporção de banda de um enlace, que pode ser reservada, informações pertinentes a essa reserva também devem ser distribuídas.

3.5.2.1. Banda Máxima do Enlace

Trata-se da banda máxima disponível em um enlace. A quantia de banda normalmente é determinada pelo tipo físico do enlace ou por parâmetros de modelagem de tráfego em interfaces NBMA.

3.5.2.2. Reserva Máxima de Banda

Trata-se da quantia máxima de banda que pode ser reservada em uma interface. Note que isto pode diferir da banda máxima, porque os administradores podem escolher dedicar só parte da banda do enlace para engenharia de tráfego.

3.5.2.3. Interface Endereço IP

Para enlaces ponto-a-ponto, um protocolo deve poder distribuir o endereço IP do sistema do outro lado do enlace. Para enlaces com vários acessos, um protocolo tem que anunciar o endereço IP de sua interface no enlace de vários acessos. Estes endereços são necessários para o roteamento baseado em restrições especificar o caminho no Objeto de Rota Explícita (ERO), que é usado através de protocolos de sinalização tal como o RSVP **[BRA97]**.

3.5.2.4. Atributo de Recurso de Classe

Os atributos de classe de recurso para o enlace são usados através de roteamento baseado em restrições como meio de determinar quais enlaces são aceitáveis para um dado LSP, baseado em uma política administrativa configurada.

3.5.2.5. Reserva de Banda

Trata-se da banda atual reservada na interface. Como a engenharia de tráfego suporta preempção e múltiplos níveis de prioridade, sistemas remotos precisam conhecer a banda reservada em uma base por prioridade. Assim, um protocolo deve poder comunicar a quantia de banda reservada por cada nível de prioridade. Uma vez mais, se a interface for de vários acessos *full-duplex*, é necessário distribuir informações sobre a quantia atual de banda reservada para ambos os lados (de entrada e de saída) da interface.

3.5.2.6. Multiplicador de Alocação Máxima

O multiplicador de alocação máxima (MAM) de um recurso é um atributo configurável administrativamente que determina a proporção do recurso que está disponível para alocação de troncos de tráfego. Este atributo é principalmente aplicável a banda de um enlace. Porém, também pode ser aplicado a recursos de buffer em LSRs. O conceito de MAM é análogo aos conceitos de assinatura e reserva em redes Frame Relay e ATM.

Os valores de MAM podem ser escolhidos de forma que um recurso possa ser sublocado ou superalocado. Um recurso é dito sublocado se as demandas agregadas de todos os troncos (como expresso nos parâmetros do tronco) que podem ser alocadas a ele forem sempre menores que a capacidade do recurso. Um recurso é dito ser superalocado se as demandas agregadas de todos os troncos de tráfego que lhes são alocados, excedem a capacidade do recurso.

Sublocação pode ser usada para unir a utilização de recursos, entretanto, a situação sob o MPLS é mais complexa do que em esquemas de comutação de circuitos, pois debaixo de MPLS, alguns fluxos podem ser roteados por protocolos (*hop-by-hop)* convencionais (também através de caminhos explícitos) sem consideração às restrições de recurso. Superalocação pode ser usada para tirar proveito das características estatísticas de tráfego de forma a implementar políticas mais eficientes de alocação de recurso. Em particular, em superalocação, pode ser usada em situações onde os picos de demanda entre troncos de tráfego não coincidam no tempo.

3.5.2.7. Atributo de Classe de Recurso

Atributos de classes de recurso são parâmetros atribuídos administrativamente que expressam alguma noção de classe para recursos. Atributos de classe de recurso podem ser vistos como cores atribuídas a recursos, tal que, o conjunto de recursos com a mesma cor pertence conceitualmente à mesma classe. Atributos de classe de recurso podem ser usados para implementar uma variedade de políticas. Os recursos chaves de interesse aqui são os enlaces. Quando aplicados a enlaces, o atributo de classe de recurso se torna um aspecto dos parâmetros de *link state*.

O conceito de atributos de classe de recurso é uma abstração poderosa. De uma perspectiva de engenharia de tráfego, podem ser usados para implementar muitas políticas no que diz respeito à otimização de desempenho orientada a tráfego e o recurso. Especificamente, atributos de classe de recurso podem ser usados para: aplicar políticas uniformes para um conjunto de recursos que não precisam estar na mesma região topológica, e especificar a preferência relativa de conjuntos de recursos para alocação de caminhos de troncos de tráfego, explicitamente restringir a colocação de troncos de tráfego para subconjuntos específicos de recursos.

3.5.2.8. Políticas de Inclusão/Exclusão Generalizadas.

Trata-se de reforçar políticas de retenção de tráfego local, ou seja, políticas que buscam conter tráfego local dentro de regiões topológicas específicas da rede. Adicionalmente, podem ser usados atributos de classe de recurso para propósitos de identificação. Em geral, um recurso pode ter atribuído a si mais de um atributo de classe de recurso. Por exemplo, a todos os enlace OC-48, em uma determinada rede, podem ser atribuídos atributos de classe de recurso distintos. Podem ser atribuídos aos subconjuntos de enlaces OC-48, que existem com um determinado domínio de

abstração da rede, atributos de classe de recurso adicionais para implementar políticas de contenção específicas, ou para arquitetar a rede de uma maneira desejada.

3.6. Enquadramento para Serviços com Garantias

3.6.1. Definição de Qualidade de Serviço – QoS

Qualidade de Serviço (QoS – *Quality of Service*), numa rede de comunicação de dados, é um conceito que exprime a capacidade que a rede tem de oferecer e garantir diversos tipos de contratos de utilização da sua infra-estrutura.

Normalmente a expressão QoS é utilizada para classificar redes que oferecem e garantem determinados serviços, como as redes de comutação de circuitos, por exemplo, em oposição ao que sucede em outro tipo de redes, como são normalmente as redes de comutação de pacotes, onde o serviço é designado por "melhor esforço" (*best-effort*) ou ASAP (*As Soon As Possible*) "tão cedo quanto possível", termos estes que são usados como antônimos de QoS.

A título de exemplo, e como referências, podemos apresentar dois casos diretamente opostos no que toca às garantias de QoS oferecidas: a rede telefônica (que usa comutação de circuitos) e as redes IP (que usam comutação de pacotes).

Enquanto que nas redes telefônicas é reservado um canal com uma largura de banda fixa entre dois pontos extremos, na rede IP não existe qualquer reserva de largura de banda, sendo usada toda a que estiver disponível, que poderá eventualmente ser insuficiente para as necessidades de uma aplicação qualquer, num determinado momento.

No que diz respeito às redes telefônicas e em particular aos percursos adaptados por um canal telefônico, não existe buffer nos nós de comutação, para que não sofram atrasos além dos que são impostos pelo próprio meio físico de transmissão. Pelo contrário, nas redes IP, os pacotes podem ficar retidos por tempo indeterminado nos buffers, podendo eventualmente sofrer elevados atrasos, que diferem, inclusive, de um pacote para os seguintes. Os pacotes podem mesmo ser descartados se num determinado roteador não houver mais espaço disponível nos buffers.

O preço a pagar pelas garantias de QoS está no sub-aproveitamento das infra-estruturas instaladas, que será potencialmente maior em relação a uma rede que não ofereça quaisquer garantias, como é o caso das redes IP. Enquanto que na rede telefônica cada chamada faz a reserva de um circuito de utilização exclusiva, seja ou não transmitida informação (voz

geralmente), numa rede IP, toda a largura de banda é possível ser utilizada até no limite do congestionamento da rede.

Em alguma parte entre estes dois extremos situam-se as necessidades das aplicações distribuídas. Cabe, para isso, à rede de comunicação, definir os serviços e respectivas características postos à disposição das aplicações. Numa tentativa de sintetização, podemos separar o problema da realização duma rede capaz de oferecer garantias de QoS em três partes:

- Definição dos serviços oferecidos, definição dos parâmetros dos contratos a estabelecer com as aplicações;
- Infra-estrutura de suporte aos serviços oferecidos;
- Fiscalização e imposição dos contratos estabelecidos com as aplicações;

Naturalmente estas três partes não são independentes entre si, mas a sua separação reduz consideravelmente a complexidade de construir uma rede com QoS.

Nesta dissertação é dada especial ênfase à segunda destas três partes, à concepção de uma infra-estrutura que permita garantir QoS.

3.6.2. Parâmetros de QoS

Quando as aplicações estabelecem contratos com a rede ou, dito de outra forma, quando estabelecem reservas de recursos, têm que especificar um conjunto de métricas cujo significado seja perfeitamente conhecido por todas as partes envolvidas na comunicação. A esse conjunto de métricas chamamos "parâmetros de QoS". Idealmente, o conjunto de parâmetros de QoS suportados por uma rede deve constituir um sistema ortogonal e completo, i.e., os parâmetros devem ser totalmente independentes entre si, mas quando combinados de forma arbitrária devem cobrir todo o espectro de especificação de serviços que façam sentidos.

Na tabela 3.2, estão representados alguns dos parâmetros de QoS mais comuns. Estes parâmetros não obedecem necessariamente aos critérios que acabamos de referir. Cada aplicação, ao iniciar um contrato com a rede (se este existir, evidentemente) especifica ou não, um ou mais destes parâmetros. A Tabela 3.3 mostra, para um conjunto de aplicações, quais os parâmetros que faria sentido considerar. Note-se que a escolha dos parâmetros de QoS que se decide considerar numa arquitetura é bastante importante, porque um algoritmo de encaminhamento não pode suportar requisitos de QoS que não possam ser representados convenientemente pelos parâmetros de QoS existentes.

Tabela 2 - Parâmetros de QoS.

Parâmetro de QoS	Descrição
Largura de banda disponível/Maximo	Capacidade disponível/máxima de uma ligação física.
Atraso médio/Maximo	Tempo médio/Maximo que leva para atravessar uma ligação física.
Jiiter	Variação no atraso sentido pelas células ou pacotes dentro do mesmo fluxo de dados.
Taxa de Perdas	Percentagem da informação (células, pacotes, dígitos binários, ...) que se perde na transmissão.

Tabela 3 - Parâmetros de QoS por aplicações.

Aplicação	Parâmetros de QoS
Ftp	Largura de banda
Telnet	Atraso, Jiiter
http	Largura de banda
Telefone	Largura de banda, Atraso, Jiiter, Taxa de perdas
Vídeo a pedido	Largura de banda, Atraso, Jiiter, Taxa de perdas

4

Roteamento de Tráfego Adaptativo Baseado em Caminho Mínimo em Redes MPLS

Introdução – 4.1. Função de Roteamento – 4.2. Método de Roteamento Caminho Mínimo Estático – 4.3. Roteamento Adaptativo – 4.4. Controle de Congestionamento – 4.5. Método de Roteamento Proposto – 4.6. Avaliação dos Serviços Oferecidos.

O aumento do tráfego e a conseqüente saturação das redes de telecomunicações têm forçado os ISPs e operadoras a buscarem novas alternativas capazes de potencializar suas malhas físicas. Um dos principais problemas a ser resolvido no nível de rede é a escolha do caminho que os pacotes devem seguir para interconexão das estações de origem e de destino. Neste capítulo, apresentamos uma proposta de roteamento que usufrui da funcionalidade suporte ao roteamento pelo destino do MPLS, para garantir um bom desempenho da rede e da aplicação que nela estiver trafegando.

4.1. Função de Roteamento

A camada de rede do modelo ISO/OSI apresenta, como principal função, o encaminhamento dos pacotes trocados entre duas entidades, oferecendo uma comunicação fim-a-fim. Durante a trajetória, os pacotes sofrerão uma série de saltos, e a decisão sobre o caminho a utilizar é feita no nível de camada de rede, podendo levar, ou não, em conta a situação da rede do ponto de vista do tráfego de informação. Pode-se separar os diferentes algoritmos de roteamento em duas principais classes: os algoritmos adaptativos e não adaptativos (rota fixa). Os métodos de roteamento não adaptativos não levam em consideração a situação do tráfego da rede, fazendo o roteamento estático; já os adaptativos o fazem considerando as modificações de topologia da rede e do tráfego real, e podem ser roteamento dinâmico. A implementação do roteamento exige uma

estrutura de dados que informe os possíveis caminhos e seus custos, a fim de que se possa decidir sobre o melhor caminho.

No roteamento não adaptativo, uma vez criada a tabela de roteamento, não é mais alterada. As rotas são fixas e os caminhos alternativos são tomados somente em caso de falhas. Esse método tem a vantagem de ser bastante simples, mas em geral, leva à má utilização dos meios de comunicação, se o tráfego da rede não for regular e bastante conhecido. Normalmente, as técnicas adaptativas resultam em melhor desempenho da rede, ao custo de uma maior complexidade de implementação.

No roteamento adaptativo, a rota é escolhida de acordo com a carga da rede. Nas tabelas de rotas, são mantidas, por exemplo, informações sobre o tráfego, o número de pacotes perdidos em um determinado caminho, e que são consultadas para a escolha do caminho mais curto e com menor congestionamento. As tabelas devem ser periodicamente atualizadas, de vários modos:

- No modo isolado, a atualização é realizada com base nas filas de mensagens para os diversos caminhos e outras informações locais;
- No modo distribuído, cada nó envia periodicamente aos outros nós, as informações locais sobre a carga da rede. Essas informações são utilizadas para o cálculo da nova tabela;
- No modo centralizado, é comum existir na rede um ou mais Centros de Controle de Roteamento (CCR). Periodicamente, cada nó envia as informações de status para a CCR. Esta, coleta todas estas informações e, conhecendo o comportamento global da rede, determina o melhor caminho a ser seguido. Essas informações são utilizadas pelo ponto central para o cálculo das novas tabelas, as quais são enviadas aos roteadores e demais nós.

Existem várias formas de medir o "comprimento" do caminho: comprimento do enlace, velocidade, largura de faixa do enlace, se o enlace é seguro ou não, atraso estimado ou alguma combinação deles. O custo pode incluir estimativas de tráfego médio esperado para uma dada hora em um dado dia, estimativas de tráfego no enlace, ocupação de buffer e condições de erro no enlace.

Existem muitos algoritmos de roteamento. A seguir discutiremos o Algoritmo de Roteamento através do Caminho mais Curto, Estático.

4.2. Roteamento Caminho Mínimo Estático

No método de roteamento caminho mínimo estático **[FLO62]**, para cada par (s, d) de nós origem e destino, a rota escolhida é aquela que possui o menor número de saltos. Chamaremos esta métrica de comprimento de rota. Desta forma, os pacotes utilizarão sempre a mesma rota para uma certa origem e um certo destino.

O método roteamento caminho mínimo estático (Figura 12) pode ser implementado tanto na abordagem de roteamento na origem como no roteamento feito em cada nó. Quando um nó *x* recebe um pacote $P(s,d)$, o algoritmo seleciona o nó vizinho *y* tal que obtenha-se o mínimo custo de roteamento, expresso em cumprimento de rota a partir da seguinte expressão:

$$y = \arg \min_{z \in N(x)} l(z,d)$$

Equação 1 – Expressão determinadora do mínimo custo de roteamento.

onde *z* é um nó vizinho do nó *x* e *N(x)* é o conjunto de todos os vizinhos do nó *x*; $l(z,d)$ representa o custo deste vizinho eleito para o destino *d*, achando-se o custo mínimo dentre os vizinhos pelo operador *arg* (operador que fornece o valor do parâmetro sendo modificado para encontrar o mínimo.

Este algoritmo não possui uma política de roteamento que permita um bom desempenho para redes reais com mudanças devido aos seguintes fatores:

- Ocorrem mudanças na topologia, nó e enlace (caem ou são ativados);
- Uma vez que a carga de pacotes na rede é alta, as rotas mais utilizadas serão inundadas com pacotes, ou seja, serão enviados mais pacotes do que os enlaces podem suportar ou que os nós conseguem processar, causando:
    - atrasos no tempo de transporte dos pacotes, devido às grandes filas de espera;
    - congestionamento nesses nós, fazendo com que as filas fiquem cheias, podendo haver perda de pacotes;
    - o uso dos recursos da rede não é otimizado, uma vez que a maior parte do tráfego é roteado através de poucos nós, ficando outros com pouca atividade.

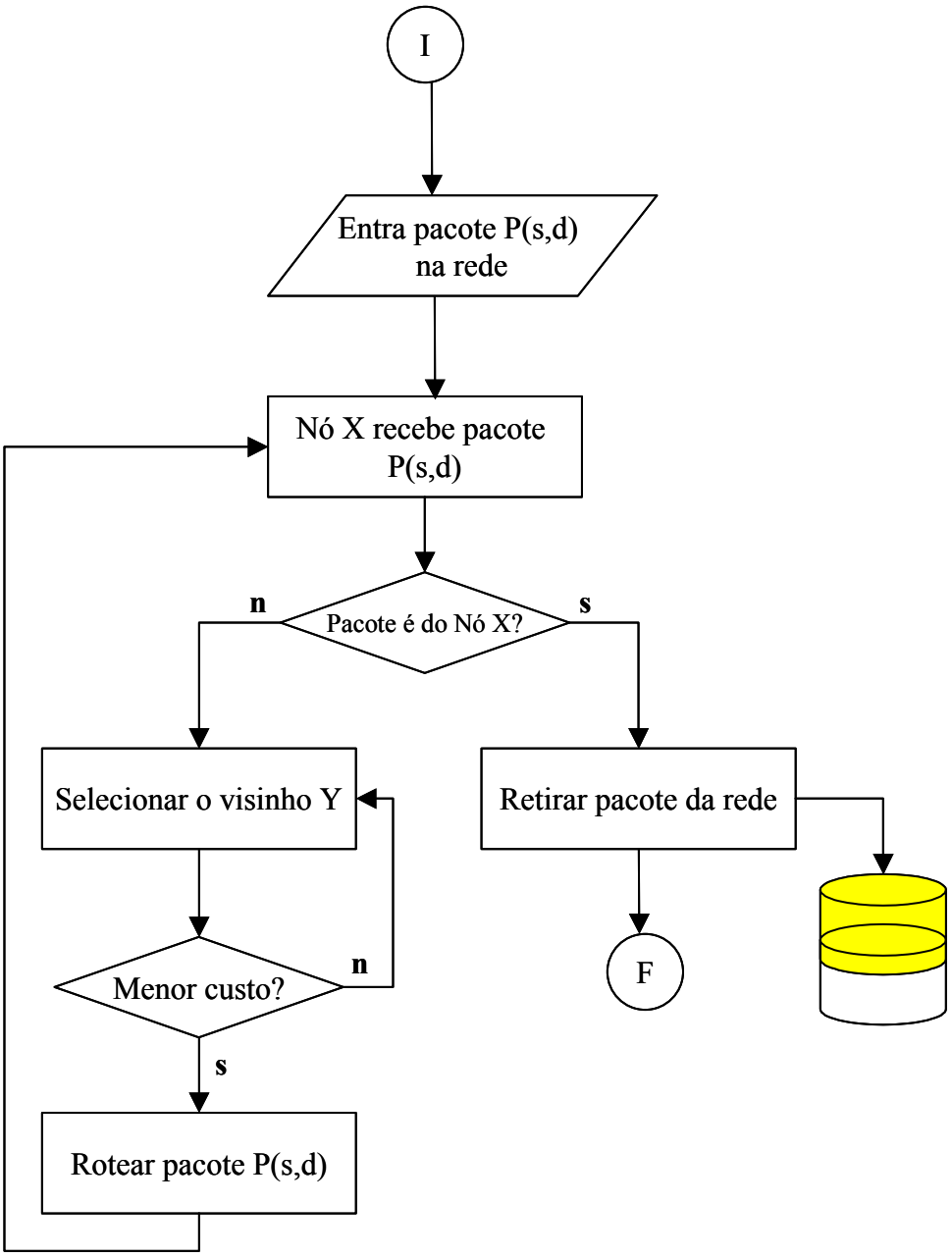


Figura 12 - Método Roteamento Caminho Mínimo Estático.

4.3. Roteamento Adaptativo

O método de roteamento caminho mínimo proporciona uma política de roteamento ótima quando o fluxo de pacotes na rede é baixo, porém, se a carga na rede aumentar, os nós intermediários terão problemas nas rotas mais utilizadas, recebendo mais pacotes do que podem processar. Este fato, leva ao aumento nas filas desses nós, acrescentando o tempo de roteamento. Embora o método de roteamento caminho mínimo seja simples de implementar em uma

arquitetura de rede, pode não suportar o comportamento dinâmico em uma rede de comunicação real. A opção é aplicar algoritmos que possuam adaptabilidade às mudanças do estado da rede e não apresentem exigências de conhecimento atualizado do estado total da rede, mas de conhecimento local do estado da rede.

O motivo para utilizar algoritmos adaptativos tem por princípio: quando o tráfego de pacotes aumenta nas rotas mais exploradas, o desempenho da rede pode ter quedas. Neste caso, deve-se encontrar rotas alternativas que, embora tendo maior comprimento, ofereça tempos menores de roteamento. Assim como a carga da rede, o padrão de tráfego e a topologia da rede sofrem mudanças, e por conseguinte a política de roteamento deve mudar também.

4.4. Controle de Congestionamento

Segundo Tanenbaum **[TAN97]**, a performance diminui quando há um número demasiado de pacotes presentes em uma parte da sub-rede. Essa situação é chamada de congestionamento. Quando o número de pacotes depositados na sub-rede pelos *hops* está dentro de sua envergadura de transporte, eles são todos entregues (exceto alguns que sofram com erros de transmissão) em número proporcional ao número enviado. Entretanto, quando o tráfego aumenta, os roteadores já não os suporta e começam a perder pacotes. Isso tende a piorar o desempenho. São situações de tráfego pesado, onde o desempenho diminui e pode ocorrer de quase nenhum pacote ser entregue.

4.5. O Método de Roteamento Proposto

Tivemos a oportunidade de observar no capítulo 3, que a Engenharia de Tráfego e o MPLS propiciam uma melhor utilização dos recursos da rede. O método de roteamento que estamos propondo, por estar fundamentado na Engenharia de Tráfego e no MPLS, apresenta uma implementação mais simples, quando comparado com MATE **[ELW01]** ou PASTE **[LI98],** que devido à complexidade em testá-los, tomamos a decisão de não utilizá-los. Com o Algoritmo de Roteamento Adaptativo Baseado em Caminho Mínimo em Rede de Computadores com Roteadores com MPLS, selecionaremos enlaces para formar dinamicamente os LSPs para as aplicações que ingressam na topologia MPLS**,** e com isso, resolveremos o segundo problema apresentado na seção 2 do capítulo 3, que é alocação ineficiente de recursos de rede.

4.5.1. Representação da Rede

Definimos uma rede de comunicação como sendo um grafo orientado G(V, E) onde V = $\{v_0, v_1, ... v_n\}$ é um conjunto de vértices representando os pontos de roteamento e E = $\{(e_i, e_j):e_i, e_j \in E, i < j\}$ é um conjunto de arcos representando os enlaces entre os nós de roteamento. Associado a cada arco está um número não negativo C_{ij} designado por custo do nó *i* para o nó *j*. Definimos também um percurso orientado como sendo uma seqüência ordenada de nós, desde uma origem "*o*" até um destino "*d"*: P_{od} = (o, p, q, r, ..., v, d)

Com objetivo de aproximar o máximo possível do cenário real, trabalharemos sempre com processos de geração de tráfego e controle totalmente dinâmicos.

4.5.2. Algoritmo de Roteamento Adaptativo Baseado em Caminho Mínimo em Rede de Computadores com Roteadores com MPLS

Foi adotado como principal objetivo do algoritmo, minimizar congestionamento em um domínio de rede MPLS com alto tráfego e, com isso, suavizar as perdas de pacotes. Alem disso, estabelecemos um procedimento para selecionar uma rota não congestionada para a aplicação que vai ingressar na rede (Figura 13).

Na implementação, o algoritmo foi composto de três módulos: o primeiro módulo é responsável pela seleção de um caminho com o menor custo através do algoritmo Dijkstra **[DIJ59] [AHU93]**, o segundo módulo é responsável pelo estabelecimento do LSP entre uma origem e um destino, na situação em que ocorre uma requisição de conexão e o terceiro módulo, tem como objetivo, fazer a correção de custo nos arcos, utilizando as informações através do tráfego da rede e do tráfego que solicita ingresso.

O algoritmo de roteamento adaptativo baseado em caminho mínimo é descrito a seguir.

O estado total da rede é armazenado em uma matriz de custos adjacentes $C = \lfloor C_{ij} \rfloor$, onde:

$C_{ii} = 0$, não existe custo para enviar um pacote ao mesmo lugar;

$C_{ij} = \infty$, se,

- Não existe um enlace *l* entre *i* e *j*;
- Enlace *l* entre *i* e *j* caiu;
- Nó *i* e/ou o nó *j* caíram.

$(C_{ij} \neq \infty) e (C_{ij} \neq 0)$, será calculado pela função de atualização abaixo;

A atualização do custo nos arcos é feita utilizando a expressão $C_{ij}^{t+1} = C_{ij}^{t} * f(\alpha, \beta, CT_{ij}, PB_{ij}^{t}, B^{\gamma})$ onde C_{ij}^{t} é o custo associado ao arco *l* no instante *t*, *f* é uma função de atualização do custo que tem como finalidade penalizar os enlaces com tráfegos sobrecarregados, CT_{ij} é a capacidade do enlace *l* em bytes, PB_{ij}^{t} é a quantidade de pacotes no buffer que alimenta o enlace *l* no instante *t* em bytes, B^{γ} é a taxa de geração de pacotes da aplicação γ em bytes, e α e $\beta \in \Re$ são os fatores de calibração da função de atualização. Assim, C_{ij} armazena o custo de enviar o pacote do nó *i* até seus vizinhos considerando a taxa de geração de pacotes, a quantidade de pacotes na fila e a capacidade do enlace. Com relação à banda residual, considera-se que uma vez existindo pacotes no buffer é porque neste instante o enlace esta sendo utilizado por completo.

Suponha que $S_h(i,j)$ seja custo total para um caminho de comprimento *h*, desde o nó *i* até o nó *j* e suponha que $P_h(i,j)$ seja próprio caminho com *h* nós intermediários. Então, se $S_h(i,j) = \infty$ e $P_h(i, j) = (\ )$, significa que não existe caminho nenhum de comprimento *h* entre o nó *i* e o nó *j*. Usando esta notação, o seguinte algoritmo pode ser usado para encontrar o melhor caminho entre dois nós *i* e *j* na rede. O peso entre o par de nós *i* e *j* é fornecido pelo valor C_{ij} na matriz de custos.

Módulo CM – CAMINHO MÍNIMO

1. **Inicializar**
   - $S_1(i,j) = C_{ij}$ para todos os nós i e j $\Rightarrow$ custo para comprimento 1;
   - $P_1(i,j) = (i,j)$, se C_{ij} é finito, do contrário, $P_1(i,j) = (\ )$;
   - Índice $h = 2$.
2. **Enquanto h < N fazer a) e b).**

   a) Para cada par (i,j) de nós:

   - Encontrar o melhor comprimento $\hat{k}$ de forma tal que:
     $$\hat{k} = \arg\min_k \left\{ S_{\frac{h}{2}}(i,k) + S_{\frac{h}{2}}(k,j) \right\};$$
   - Atualizar Sh (i, j) como segue:
     $$S_h(i,j) = S_{\frac{h}{2}}\left(i,\hat{k}\right) + S_{\frac{h}{2}}\left(\hat{k},j\right);$$
   - Atualizar P1(i,j):
     - Se $S_h(i,j) < S_{\frac{h}{2}}(i,j)$ então
       $$P_h(i,j) = concat\left[P_{\frac{h}{2}}\left(i,\hat{k}\right), P_{\frac{h}{2}}\left(\hat{k},j\right) \right];$$
     - Se $S_h(i,j) \geq S_{\frac{h}{2}}(i,j)$ então
       $$P_h(i,j) = P_{\frac{h}{2}}(i,j);$$

   b) $h = 2 \times h$**.**

Módulo CC – CONTROLE DE CONEXÃO

3. **Efetivar a conexão;**
4. **Fazer a correção do custo nos arcos que compõem o LSP através do Módulo AC;**

Módulo AC – ATUALIZAR CUSTO

5. **Atualizar custo dos enlaces através da expressão** $C_{ij}^{t+1} = C_{ij}^{t} * f\left(\alpha, \beta, CT_{ij}, PB_{ij}^{t}, B^{\gamma}\right)$**.**

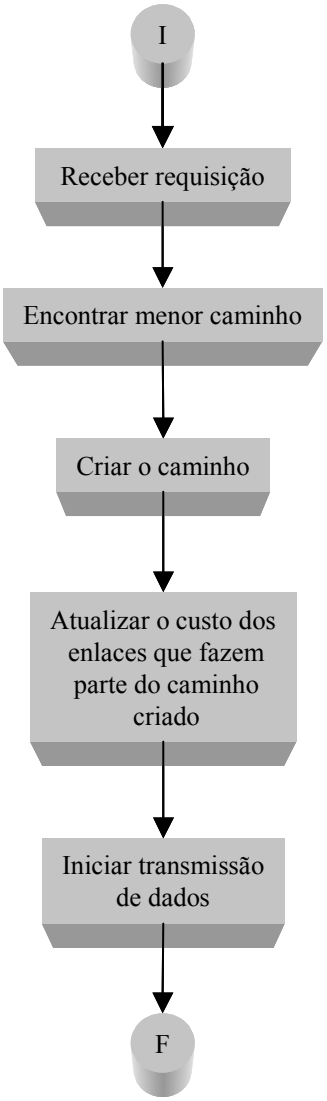


Figura 13 - Método de Roteamento Caminho Mínimo Adaptativo na Conexão.

4.6. Avaliação dos Serviços Oferecidos

A partir do momento em que uma aplicação estabelece um contrato de prestação de serviços com uma rede é suposto que o contrato será cumprido por ambas as partes. A aplicação compromete-se a não gerar tráfego que viole esse contrato, enquanto que a rede tem que garantir que o tráfego da aplicação será entregue dentro dos limites impostos para os parâmetros de QoS considerados.

Naturalmente, para conseguir cumprir os contratos que estabelece, a rede tem que fazer uma gestão criteriosa dos seus recursos, tendo que, eventualmente, negar a sua utilização a certas aplicações. Por outro lado, por motivos de viabilidade econômica, é preciso otimizar a utilização dos recursos disponíveis. Cabe então ao conjunto de mecanismos que garantem a QoS,

determinar o ponto de operação da rede, levando sempre em consideração estes dois objetivos oposto (cumprimento de contratos x otimização da utilização dos recursos).

A tarefa de avaliar estes mecanismos pode ser feita de acordo com os seguintes critérios:

- Taxa de ocupação – Fornece, em cada instante, qual é a capacidade das ligações físicas que está sendo usada em relação à capacidade máxima instalada;
- Taxa da rede – Corresponde ao número de bits por segundo que a rede consegue entregar às aplicações;
- Percentagem de perdas – Refere-se ao número médio de pacotes perdidos, descartados nos buffers, em relação ao número de pacotes que entram na rede.

É possível ainda definir muitos outros critérios, alguns dos quais consideravelmente mais complexos. Nós só enfocaremos os aspectos que são requeridos para implementação de interoperabilidade, isto é, o conjunto de mecanismos usados para disseminação de informação da topologia de estado.

Vale salientar, como teremos ocasião de ver nos resultados experimentais obtidos que, segundo estes critérios, o desempenho da rede está fortemente dependente do comportamento das aplicações.

Como foi especificado no item 4.7 do capítulo 3 a seleção de caminho genérico e manutenção de atributos definem as regras de seleção de rota tomada por um tronco de tráfego, como também, as regras para manutenção de caminhos já estabelecidos. O nosso método definirá os caminhos administrativamente auxiliando os operadores de rede. Neste estudo não trabalharemos com exigência de recurso ou restrições associadas com um tronco de tráfego, então o método proposto se baseará na topologia que será usada para selecionar os caminhos.

A administração de caminho consiste em todos os aspectos que pertencem à manutenção de caminhos percorridos por troncos de tráfego. Em alguns contextos operacionais, é desejável que uma implementação de MPLS possa se reconfigurar dinamicamente para adaptar a alguma noção de mudança em estado de sistema. Adaptabilidade e capacidade de recuperação são aspectos de administração dinâmica de caminhos.

5

Simulação e Resultados

Introdução – 5.1. Ambiente de teste – 5.2. Resultados.

Neste capítulo são descritos e discutidos as simulações realizadas e seus resultados através dos métodos *Shortest Path First* e o proposto no Capítulo 4.

5.1. Ambiente de Teste

5.1.1. Sobre a Ferramenta de Testes

Para fazer a validação experimental dos algoritmos de encaminhamento em uma rede MPLS, foi utilizado o simulador de rede ns-2.1b7a – *Network Simulator* versão 2.1b7a **[NS201]**. O NS-2 é um simulador de evento discreto desenvolvido em Berkely UC, escrito em C++ e OTcl e possui um modulo chamado MNS (MPLS Network Simulator) **[MNS01]**. Ele permite testar os métodos: troca de rótulos, LDP, CR-LDP, como um LSP é criado e destruído. Desde que a funcionalidade MPLS do simulador é bastante modular, gratuito, tem o código fonte disponível para possíveis alterações e rico em documentações, elegemo-los para realizar a tarefa desejada.

5.1.2. Topologia

As topologias usadas nas simulações podem ser vistas nas Figuras 14 e 15. A escolha destas topologias foi baseada na tentativa de aproximar-se de um cenário onde fosse possível aplicar o método de roteamento adaptativo, baseado em caminho mínimo em tempo real e comprovar sua funcionalidade, comparada com o método *Shortest Path First*.

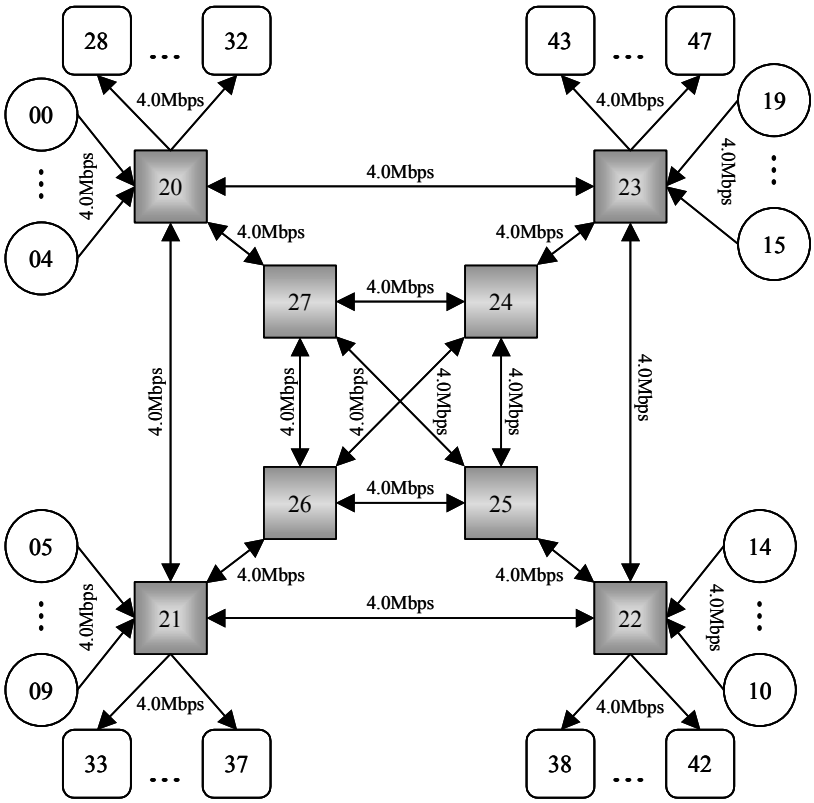


Figura 14 - Topologia I com 48 nós.

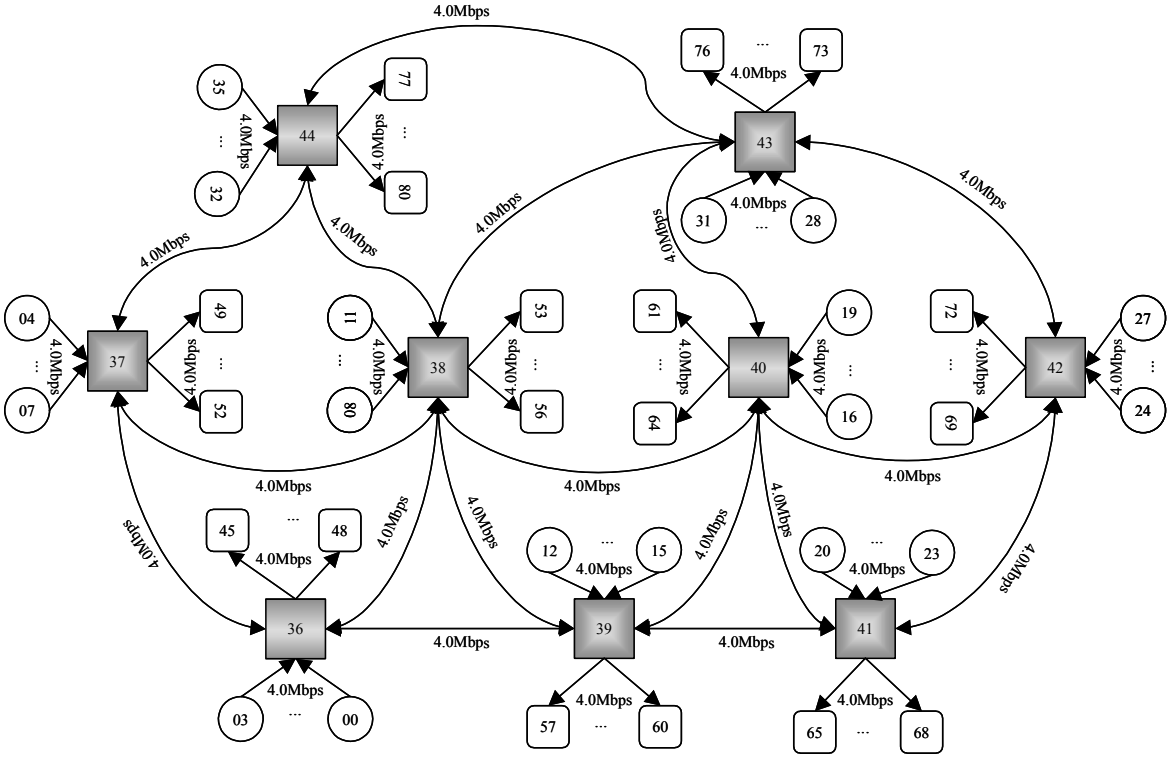


Figura 15 - Topologia II com 81 nós.

5.1.3. Cenários

Para a topologia I, os LERs (*Label Edge Router*) são 20, 21, 22 e 23 e os LSRs (*Label Switched Path*) são 24, 25, 26 e 27. Já na topologia II, os roteadores são LERs e LSRs ao mesmo tempo. Criadas as topologias, são definidos outros parâmetros, em particular, quantos e quais circuitos virtuais devem ser estabelecidos na rede, quais fontes de tráfego vão ser utilizadas e a matriz de custos, aqui definida para as topologias I e II nas Tabelas 4 e 5.

Quanto ao número de circuitos virtuais, estes foram tomados como iguais aos números de nós fontes existentes na rede. Para cada rede, utilizamos dois tipos de fontes de transmissão VBR (*Variable Bit Rate*) com tráfego obedecendo à distribuição exponencial e CBR (*Constant Bit Rate*) e foram simetricamente distribuídas na periferia da rede, possuindo mesma taxa média de transmissão Tb.

Tabela 4 - Matriz de custos da topologia I.

	LER 20	LER 21	LER 22	LER 23	LSR 24	LSR 25	LSR 26	LSR 27
LER 20	0	CUSTO	∞	CUSTO	∞	∞	∞	CUSTO
LER 21	CUSTO	0	CUSTO	∞	∞	∞	CUSTO	∞
LER 22	∞	CUSTO	0	CUSTO	∞	CUSTO	∞	∞
LER 23	CUSTO	∞	CUSTO	0	CUSTO	∞	∞	∞
LSR 24	∞	∞	∞	CUSTO	0	CUSTO	CUSTO	CUSTO
LSR 25	∞	∞	∞	∞	CUSTO	0	CUSTO	CUSTO
LSR 26	∞	CUSTO	∞	∞	CUSTO	CUSTO	0	CUSTO
LSR 27	CUSTO	∞	∞	∞	CUSTO	CUSTO	CUSTO	0

Tabela 5 - Matriz de custos da topologia II.

	LER 36	LER 37	LER 38	LER 39	LER 40	LER 41	LER 42	LER 43	LER 44
LER 36	0	CUSTO	CUSTO	CUSTO	∞	∞	∞	∞	∞
LER 37	CUSTO	0	CUSTO	∞	∞	∞	∞	∞	CUSTO
LER 38	CUSTO	CUSTO	0	CUSTO	CUSTO	∞	∞	CUSTO	CUSTO
LER 39	CUSTO	∞	CUSTO	0	CUSTO	CUSTO	∞	∞	∞
LER 40	∞	∞	CUSTO	CUSTO	0	CUSTO	CUSTO	CUSTO	∞
LER 41	∞	∞	∞	CUSTO	CUSTO	0	CUSTO	∞	∞
LER 42	∞	∞	∞	∞	CUSTO	CUSTO	0	CUSTO	∞
LER 43	∞	∞	CUSTO	∞	CUSTO	∞	CUSTO	0	∞
LER 44	∞	CUSTO	CUSTO	∞	∞	∞	∞	CUSTO	0

Para a topologia I, os fluxos partem dos nós 00 - 19 com destino aos 28 – 47, de acordo com a ordem apresentada na Tabela 6. A estrutura da topologia II foi simulada com os fluxos partindo dos nós 00 – 35, com destino aos 45 – 80, de acordo com a ordem apresentada na Tabela 7. Todos os pacotes utilizam o UDP (*User Datagram Protocol*) como protocolo da camada de transporte, possuem um comprimento de 572 bytes e recebem um único rótulo de 4 bytes do MPLS. É ainda importante destacarmos os seguintes aspectos relativos aos cenários que construímos:

- Todos os enlaces obedecem ao mesmo dimensionamento, 4.0Mbps com 3ms de tempo de propagação e os LSR (*Label Switching Router*) possuem filas do tipo FIFO (*First-In-First-Out*) para 100 pacotes ou 57.200 bytes.

Tabela 6 - Tráfego da Topologia I simulada.

Aplicação	<origem, destino>
CBR	<00,38> <02,40> <04,42> <06,44> <08,46> <10,28> <12,30> <14,32> <16,34> <18,36>
VBR	<01,39> <03,41> <05,43> <07,45> <09,47> <11,29> <13,31> <15,33> <17,35> <19,37>

Para a topologia I, cada fonte iniciou sua transmissão em um intervalo de tempo próprio, desde t = 0,30 min até t = 6,60 min. A partir deste instante, todas as fontes transmitiram simultaneamente pela rede durante um intervalo de 2 minutos. Então, as fontes terminaram sua transmissão na mesma ordem em que começaram a transmitir, desde t = 9,22 min até t = 15,60 min.

Tabela 7 - Tráfego da Topologia II simulada.

Aplicação	<origem, destino>
CBR	<00,79> <02,77> <04,75> <06,73> <08,71> <10,69> <12,67> <14,65> <16,47> <18,45> <20,59> <22,57> <24,55> <26,53> <28,51> <30,49> <32,63> <34,61>
VBR	<01,80> <03,78> <05,76> <07,74> <09,72> <11,70> <13,68> <15,66> <17,48> <19,46> <21,60> <23,58> <25,56> <27,54> <29,52> <31,50> <33,64> <35,62>

Para a topologia II, cada fonte iniciou sua transmissão em um intervalo de tempo próprio, desde t = 0,70 min até t = 18,20 min. A partir deste instante, todas as fontes transmitiram

simultaneamente pela rede durante um intervalo de 15,30 minutos. Então, as fontes terminaram sua transmissão na mesma ordem em que começaram a transmitir, desde t = 16,00 min até t = 33,50 min.

5.2. Resultados

Os resultados estatísticos que vamos apresentar, dividem-se em três grupos diferentes. Dois dos grupos contêm dados relativos à avaliação de diversos parâmetros exclusivos da rede de dados. São eles a taxa de perdas de pacotes e tráfego oferecido médio.

A função de atualização de custo utilizada é expressa por $f(\alpha,\beta,CT_l PB_l^t, B^\gamma) = \exp\left(\alpha\left(PB_l^t / CT_l\right)\right) * \exp\left(\beta\left(B^\gamma / CT_l\right)\right)$, onde α e β são parâmetros que podem ser ajustados pelo administrador. O comportamento de $f(\alpha,\beta,CT_l PB_l^t, B^\gamma)$ em função das aplicações está ilustrado na Figura 16.

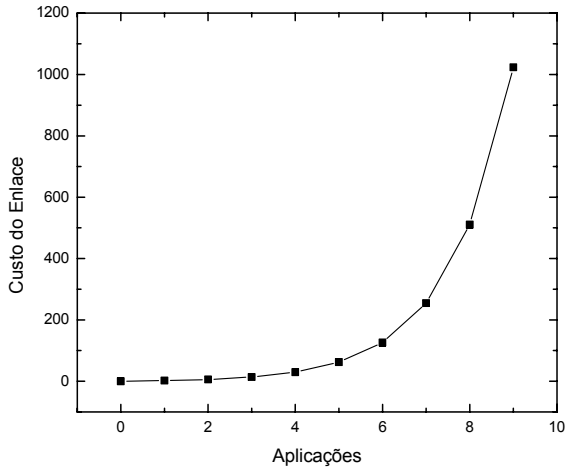


Figura 16 - Custo em função do tráfego gerado pela função custo.

A Figura 17 mostra a evolução temporal da perda média de pacotes para Tb = 0.75, 1.00, 1.25 e 1,5 Mbits/s, para o algoritmo SPF e para o algoritmo proposto. O máximo de perdas ocorre entre os instantes t = 9 e t = 10 min, pouco posterior ao instante em que as primeiras fontes deixaram de transmitir. Isto indica que a perda de pacotes para estas fontes é relativamente baixa e contribui de forma a diminuir a média global de pacotes perdidos.

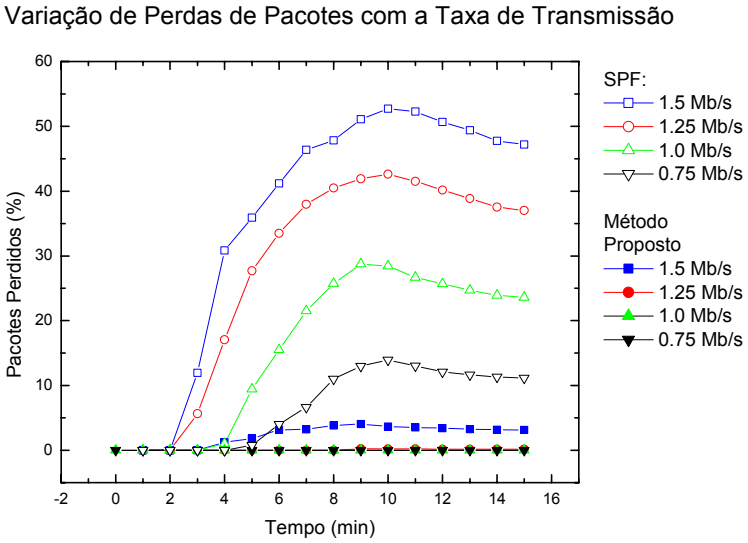
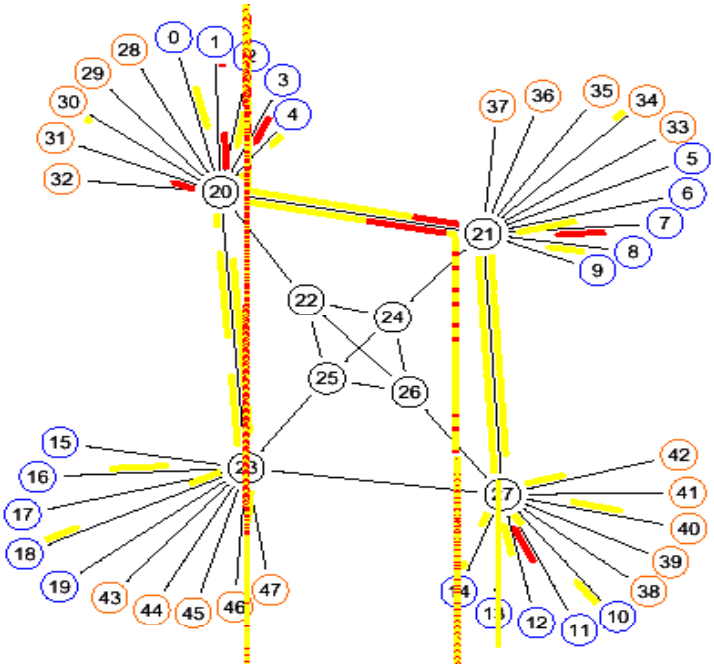
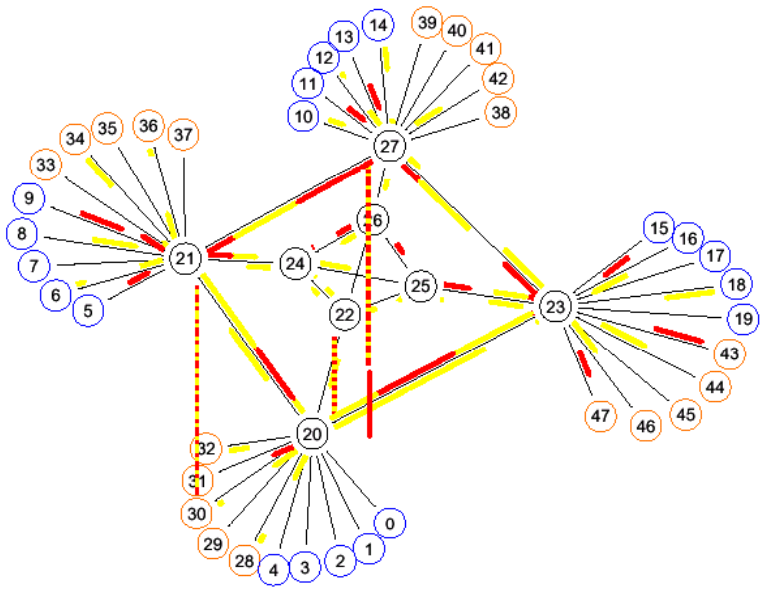


Figura 17 - Evolução Temporal da Perda Média de Pacotes na Topologia I, para diferentes taxas de transmissão, Tb. O algoritmo proposto é comparado com o SPF.

Observa-se que o algoritmo de SPF apresenta perdas que variam, com o aumento de Tb, de 10 a 50% ao final das transmissões. Por outro lado, o algoritmo proposto somente apresenta perdas apreciáveis para Tb = 1.5 Mbits/s. A Figura 18, ilustram a origem desta diferença de desempenhos, evidenciando que os LSPs criados pelo SPF concentram-se nas mesmas rotas e que, conforme o esperado, o algoritmo proposto distribui os LSPs por todas as rotas das redes.



(a)



(b)

Figura 18 - Transmissão e Perda de Pacotes Durante o Período de Maior Tráfego na Topologia I: (a) com o algoritmo SPF e (b) com o algoritmo proposto.

A análise individual da perda de pacotes por nó, para Tb = 1.50 Mbits/s, também enfatiza o bom desempenho do método proposto. De acordo com a Figura 19, a menor perda de pacotes experimentada por um nó com SPF foi de 10%, ao passo que o método proposto proporcionou que 85% dos nós experimentassem perdas inferiores a esta. Ainda no caso deste método, observa-se que o nó com o pior desempenho, Nó 37, possui uma perda média de pacotes relativamente alta, de 34%.

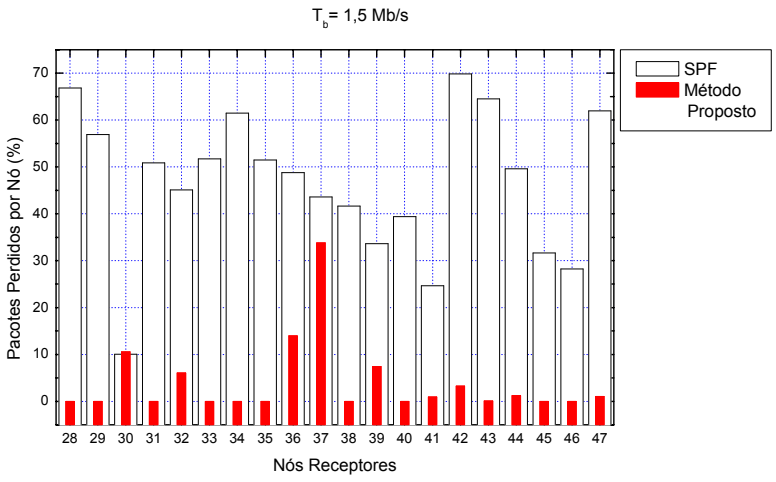


Figura 19 - Número Relativo de Pacotes Perdidos em Cada Nó de Recepção na Topologia I.

A Figura 20 mostra que o Nó 37 tem um surto de perda de pacotes entre os instantes t = 7 e t = 9 min, entre os quais, o tráfego pela rede é máximo. De fato, a observação da dinâmica da transferência de pacotes, permitida pelo NS-2, revela que entre estes instantes os LERs 21 e 23 passam por intensa atividade.

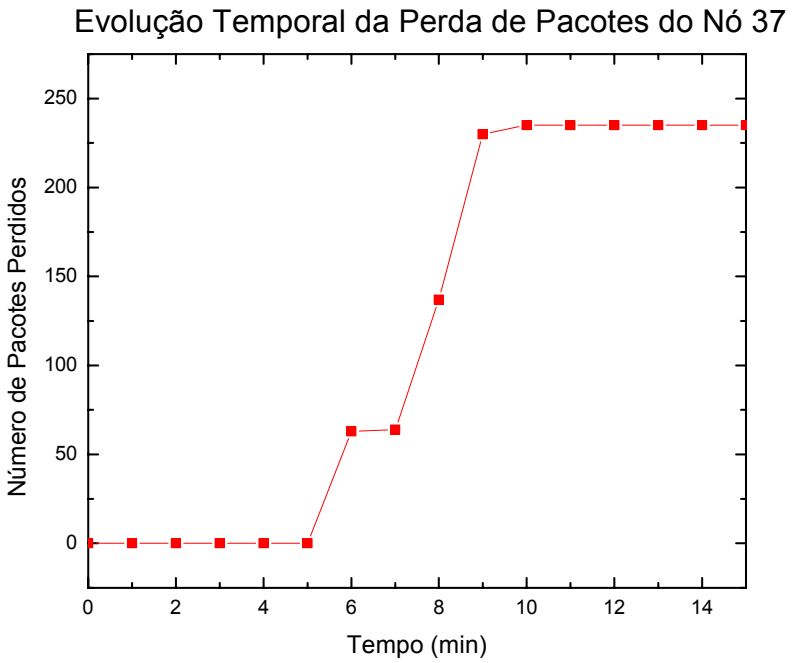
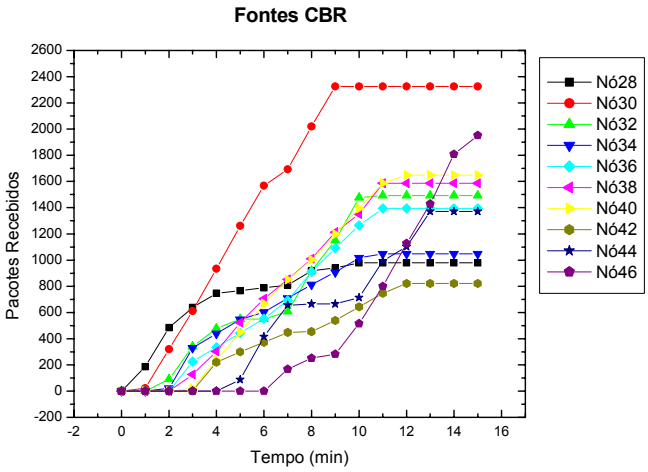
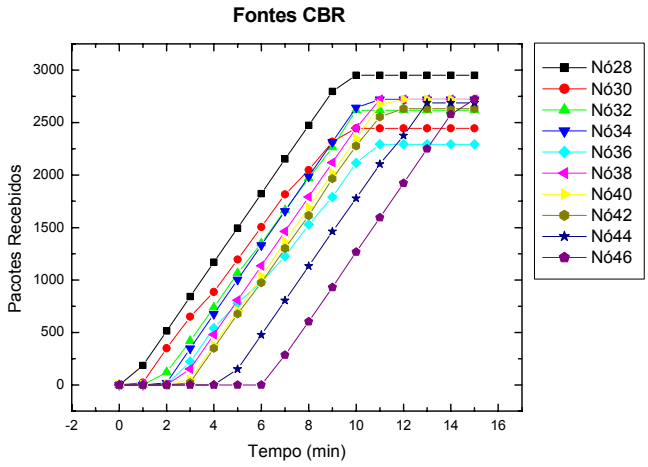


Figura 20 - Evolução Temporal do Número Absoluto de Pacotes Perdidos pelo Nó 37 da Topologia I.

Os gráficos da Figura 21 exibem o comportamento do tráfego CBR. Observa-se que, no caso de roteamento SPF, há a ocorrência de patamares na evolução temporal do número de pacotes recebidos em cada nó. Isto evidencia a presença de períodos de perdas de pacotes geradas nos intervalos de congestionamento da rede. A utilização do algoritmo proposto reduz significativamente o número destes patamares, tornando o número de pacotes recebidos diretamente proporcional ao tempo de transmissão.



(a)

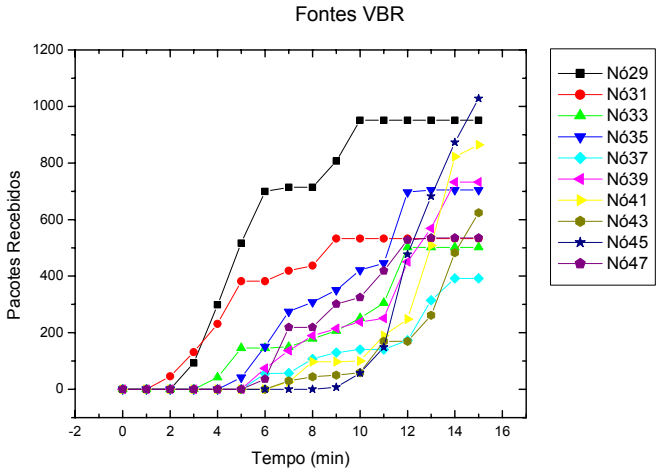


(b)

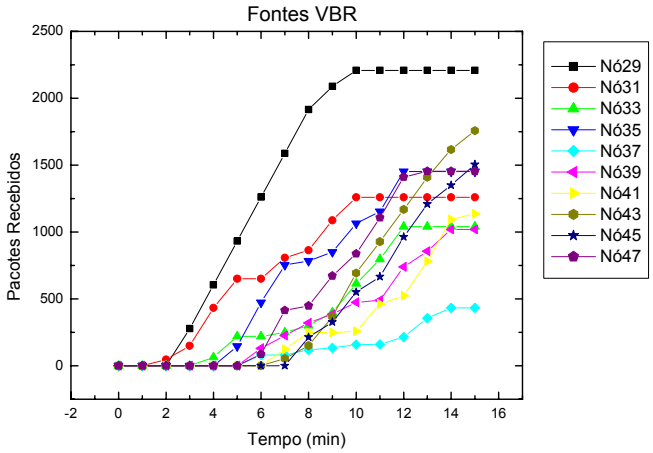
Figura 21 - Evolução temporal do número absoluto de pacotes CBR recebidos na Topologia I: (a) com o método SPF e (b) com o método proposto.

De fato, as fontes CBR parecem não sofrer perdas apreciáveis de pacotes com a utilização do algoritmo proposto.

No caso das fontes VBR, a Figura 22 ilustra que o método proposto introduziu uma mudança quantitivamente expressiva em relação ao método SPF e o desempenho do método proposto também é superior para este tipo de tráfego.



(a)



(b)

Figura 22 - Evolução temporal do número absoluto de pacotes VBR recebidos na Topologia I: (a) com o método SPF e (b) com o método proposto.

A Figura 23 mostra a evolução temporal da perda média de pacotes para Tb = 1.00, 1.25, 1.50 e 2.00 Mbits/s na topologia II, para o algoritmo SPF e para o algoritmo proposto. Observa-se que o algoritmo de SPF manteve com altas perdas que variam, com o aumento de Tb, de 10 a 40% ao final das transmissões. Por outro lado, o algoritmo proposto somente apresenta perdas apreciáveis para Tb = 2.00 Mbits/s.

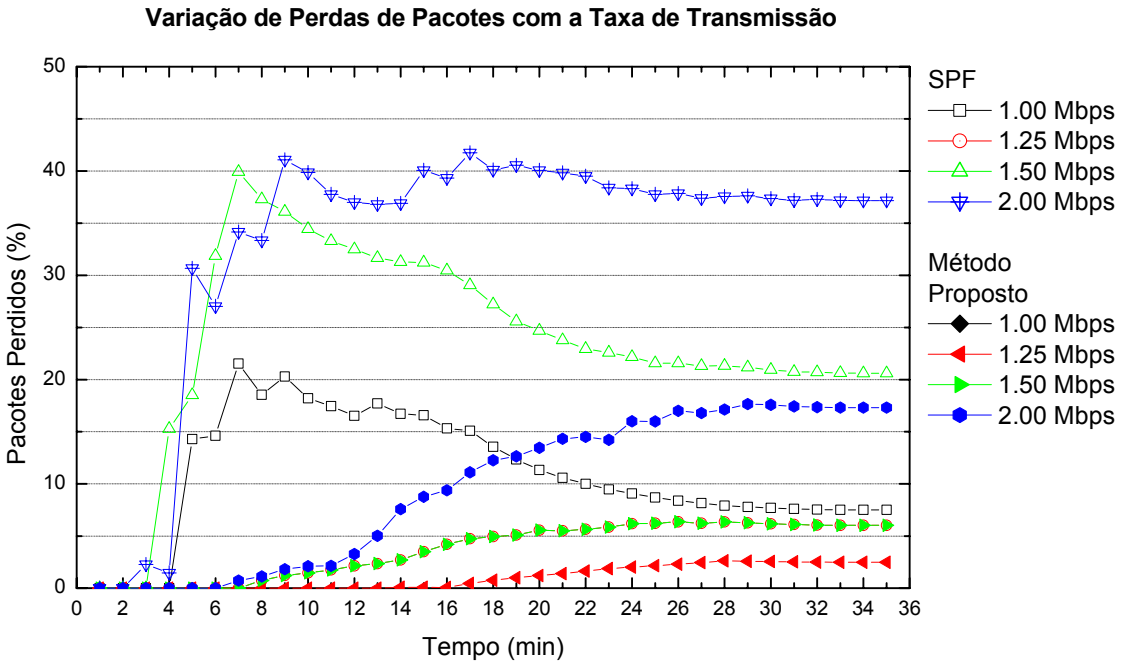


Figura 23 - Evolução Temporal da Perda Média de Pacotes na Topologia II, para diferentes taxas de transmissão, Tb. O algoritmo proposto é comparado com o SPF.

A Figura 24 mostra que o método tem um número de perda de pacotes superior ao SPF em alguns receptores localizados no interior e periferia da rede. De acordo com a Figura 25, mostra que 53% dos receptores não experimentarão perda de pacotes, 14% foi igual, 19% inferior e 14% superior, ao passo que o método SPF proporcionou que 100% dos nós experimentassem perdas. Deve-se ressaltar que cinco nós, sendo eles 45, 53, 55, 56 e 69 tiveram um péssimo desempenho, devido ao alto tráfego interno da rede gerado pelos LER 30 e 40. Assim, a análise individual da perda de pacotes por nó para Tb = 1,50 Mbps da topologia II, também mostra o bom desempenho na média do método proposto.

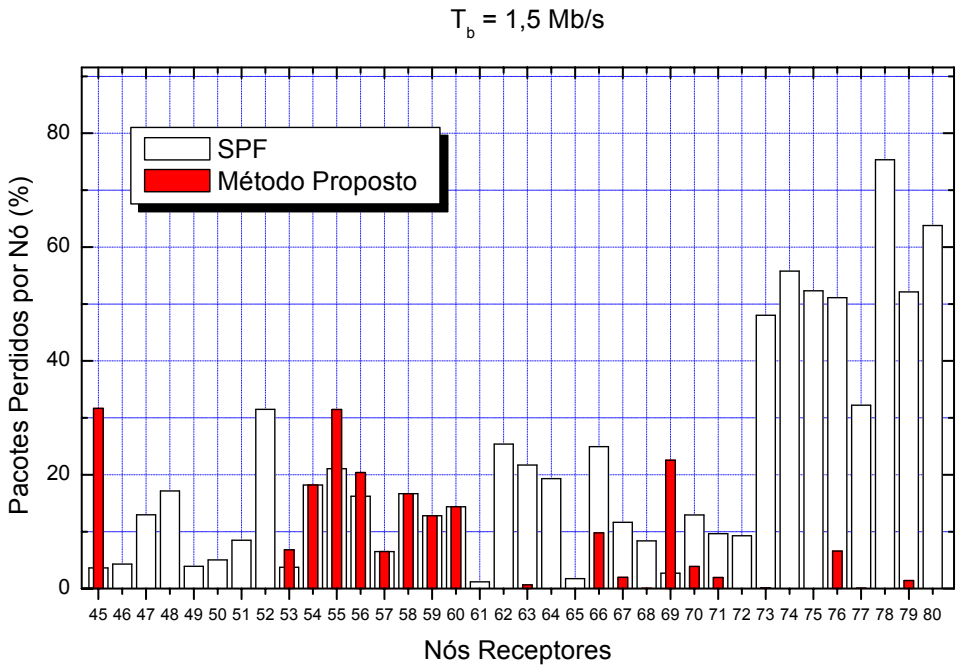


Figura 24 - Número Relativo de Pacotes Perdidos em Cada Nó de Recepção na Topologia II.

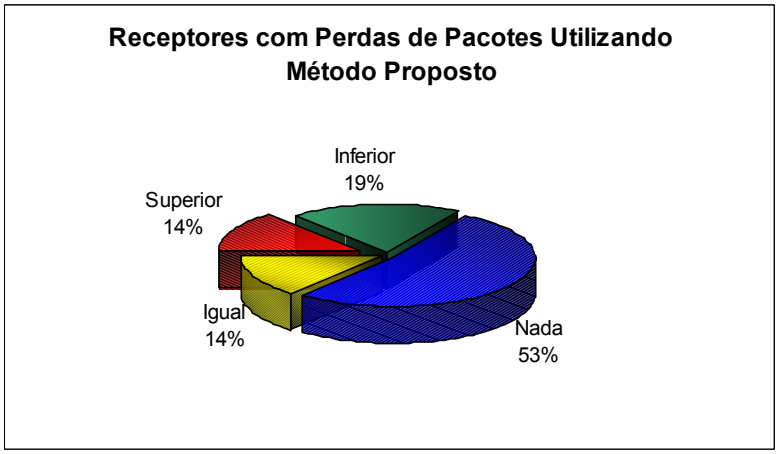


Figura 25 - Número Relativo de Receptores da Topologia II com Perdas de Pacotes Utilizando o Método Proposto.

Os gráficos da Figura 26 exibem o comportamento do tráfego CBR na topologia II. Observa-se que, no caso de roteamento SPF, permanece a ocorrência de patamares na evolução temporal do número de pacotes recebidos em cada nó. Isto continua evidenciando a presença de períodos de perdas de pacotes geradas nos intervalos de congestionamento da rede. A utilização

do algoritmo proposto reafirmou a redução do número destes patamares, tornando o número de pacotes recebidos diretamente proporcional ao tempo de transmissão.

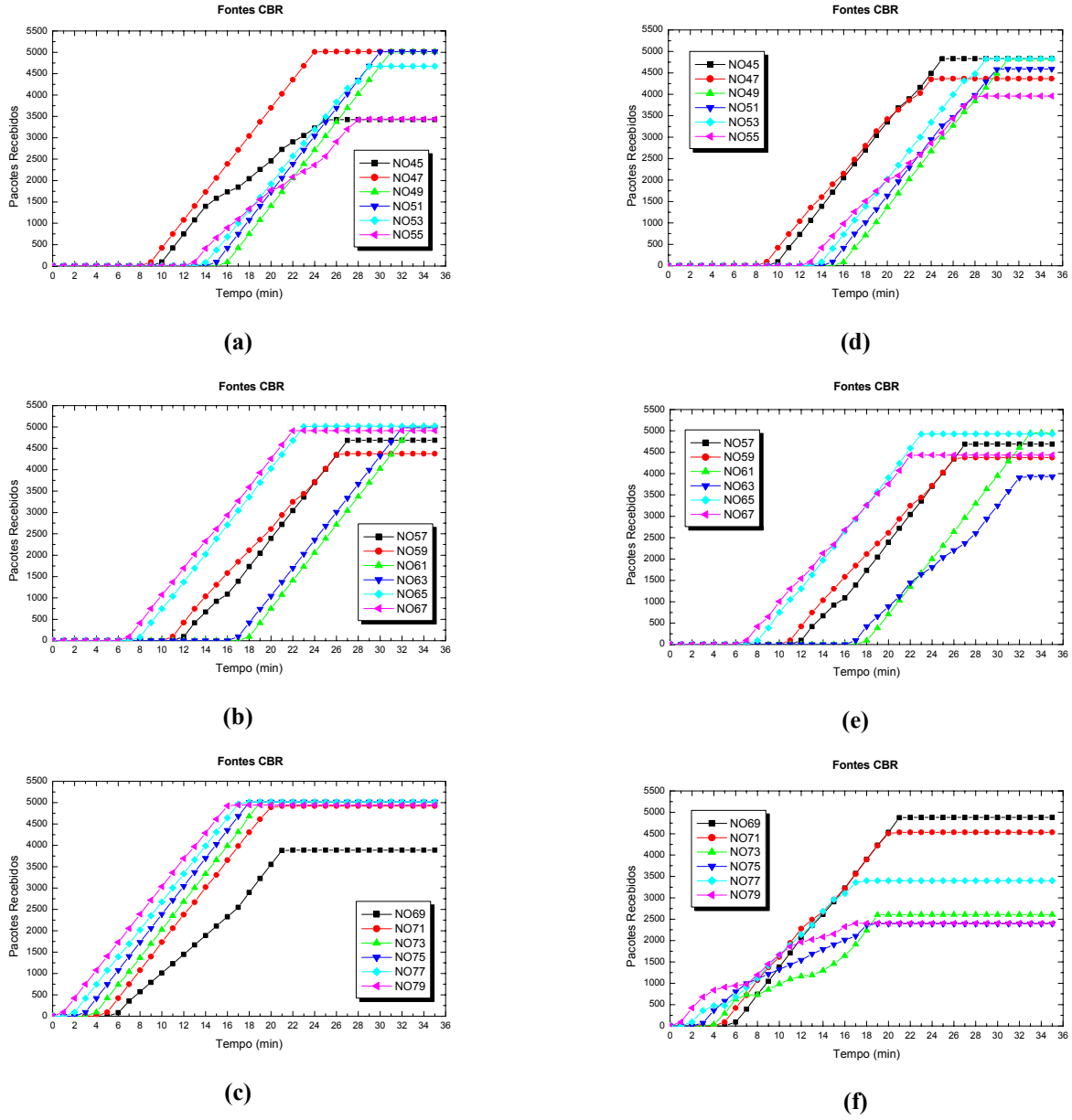


Figura 26 - Evolução temporal do número absoluto de pacotes CBR recebidos na Topologia II: (a, b, c) com o método proposto e (d, e, f) com o método SPF.

No caso das fontes VBR, a Figura 27 ilustra que o método proposto confirma a introdução de uma mudança quantitivamente expressiva em relação ao método SPF e o desempenho do método proposto também é superior para este tipo de tráfego.

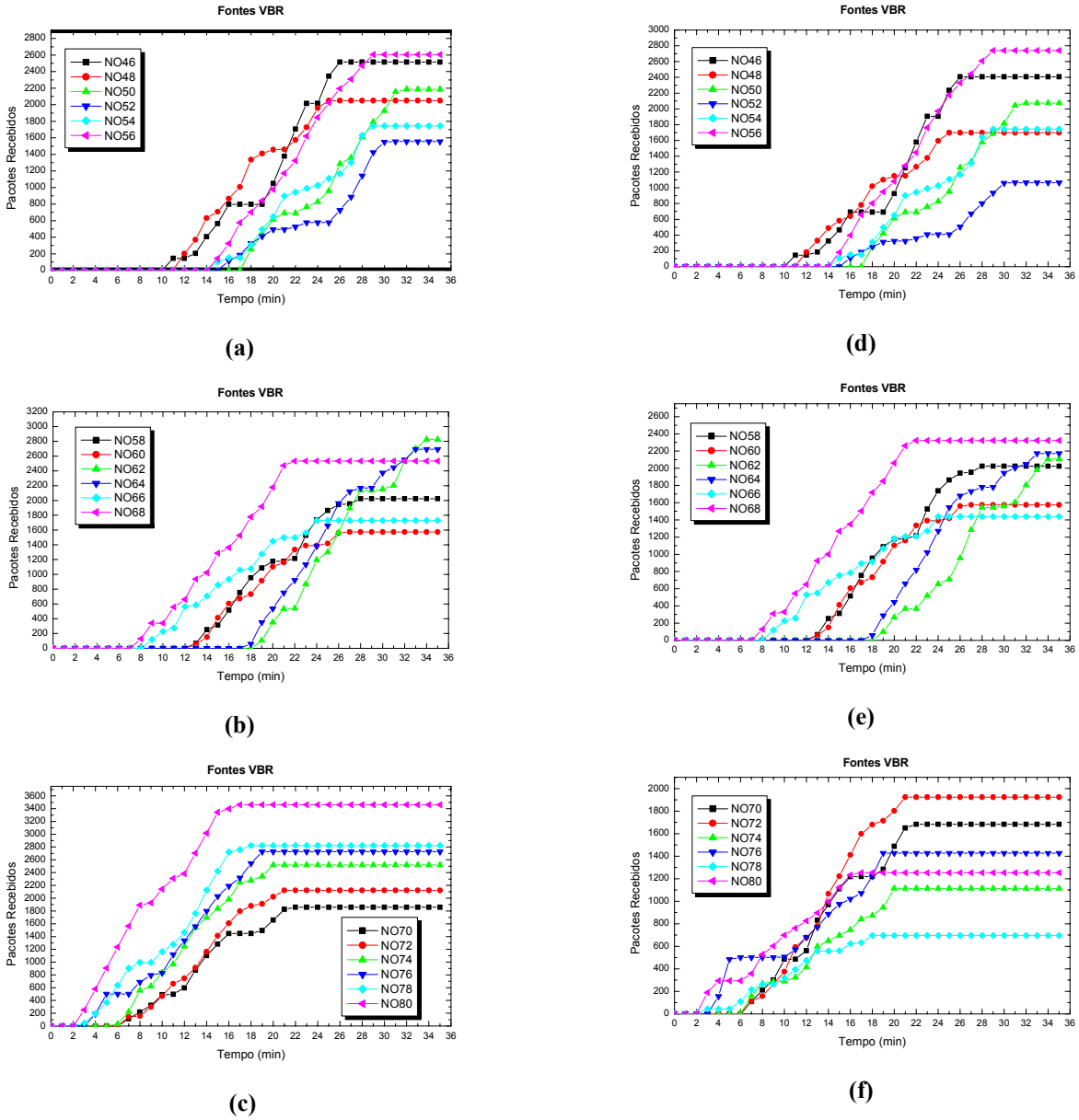


Figura 27 - Evolução temporal do número absoluto de pacotes VBR recebidos na Topologia II: (a, b, c) com o método proposto e (d, e, f) com o método SPF.

6

Conclusões e Trabalhos Futuros

Introdução – 6.1. Conclusões – 6.2. Sugestões para Trabalhos Futuros.

Neste capítulo, serão apresentadas as conclusões da pesquisa bem como algumas propostas para desenvolvimento de novos trabalhos.

6.1. Conclusões

O nosso objetivo principal era desenvolver um método de roteamento que maximizasse a utilização dos recursos de uma rede e minimizasse a perda de pacotes de uma aplicação que ingresse na rede através do suporte ao roteamento pelo destino do MPLS. Estes objetivos foram atingidos com êxito.

Para os problemas apresentados, foi proposto um método de roteamento adaptativo baseado em caminho mínimo em uma rede de computadores com roteadores com MPLS. Mais especificamente, foi proposto um algoritmo que procura selecionar as rotas menos congestionadas nos eventos de solicitação de conexão.

Através de simulações com a ferramenta NS-2/MNS, foram implementados os modelos e o funcionamento de uma rede MPLS submetida ao tráfego de várias fontes. A taxa de perda de pacotes foi o parâmetro de qualidade de serviço utilizado para avaliar se a proposta é eficiente.

Os resultados de simulações, utilizando a ferramenta NS-2/MNS, mostraram que o método proposto tem um comportamento aceitável, dentro do enfoque de engenharia de tráfego, na direção de otimizar os recursos da rede e de manter as qualidades de serviço.

6.2. Sugestões para Trabalhos Futuros

Diversos trabalhos ainda podem ser feitos para tornar mais robusto o método proposto. Dentre eles podemos destacar:

- Testar o algoritmo mais exaustivamente, considerando diferentes tipos de fontes;
- Incluir características preditivas quanto à engenharia de tráfego, através de informações estatísticas de rede e de tráfegos, com inclusão de medidas de probabilidades de perdas;
- Introdução de medição estatística de atrasos de transmissão baseada nos estados da rede, com conseqüente procedimento de decisão envolvendo perdas de pacotes versus atraso de transmissão;

Referências Bibliográficas

[AHU93] AHUJA, R.K. et al. – Network flows: theory, algorithms, and applications. Prentice-Hall. 1993.

[AND01] ANDERSSON, L. et al. – RFC 3036: LDP Specification. January 2001.

[AWD01] AWDUCHE, D. et al. – Internet Draft draft-ietf-mpls-rsvp-lsp-tunnel-08.txt: RSVP-TE: Extensions to RSVP for LSP Tunnels. February 2001.

[AWD99] AWDUCHE, D. et al. – RFC 2702: Requirements for Traffic Engineering Over MPLS. September 1999.

[BRA97] BRADEN, R. et al. – RFC 2205: Resource ReSerVation Protocol (RSVP). September 1997.

[CON01] CONTA, A. et al. – RFC 3034: Use of Label Switching on Frame Relay Networks Specification. January 2001.

[DAV01] DAVIE, B. et al. – RFC 3035: MPLS using LDP and ATM VC Switching. January 2001.

[DEE98] DEERING, S.; HINDEN, R. – RFC 2460: Internet Protocol, Version 6 (IPv6) Specification. December 1998.

[DIJ59] DIJKSTRA, E. – A Note on Two Problems in Connection with Graphs. Numerical Mathematics, v.1, p.269-271, October 1959.

[ELW01] ELWALID, A. et al. – MATE: MPLS Adaptive Traffic Engineering. IEEE INFOCOM, p. 1300 – 1309, 2001.

[FLO62] FLOYD, R. W. – Algorithm 97: Shortest Path, Communications of the ACM, v.5, n.6, p.345, June 1962.

[HÄG98] HÄGÄRD, G; WOLF, M. – Multiprotocol Label Switching in ATM Networks. Ericsson Review, The Telecommunications Technology Journal, n.1, 1998.

[LI98] LI, T.; REKHTER, Y. – RFC 2430: A Provider Architecture for Differentiated Services and Traffic Engineering. October 1998.

[MAG01] MAGALHÃES, M. F.; CARDOZO, E. – Introdução à Comutação IP por Rótulos Através de MPLS. Anais do SBRC 2001 – 19º Simpósio Brasileiro de Redes de Computadores, Cap.3, Maio 2001.

[MAL98] MALKIN, G. – RFC 2453: RIP Version 2. November 1998.

[MNS01] AHN, G.; WOOJIK, C. – MPLS Network Simulator, Department of Computer Engineering, Chungnam National University, Korea - http://www.raonet.com/ - http://www.ce.cnu.ac.kr/

[MOY98] MOY, J. – RFC 2328: OSPF Version 2. April 1998.

[MUT00] MUTHUKRISHNAN, K.; MALIS, A. – RFC 2917: A Core MPLS IP VPN Architecture. September 2000.

[NS01] NS-2 – The Network Simulator, http://www.isi.edu/nsnam/ns/

[REK01] REKHTER, Y. et al. – RFC 3107: Carrying Label Information in BGP-4. May 2001.

[REK95] REKHTER, Y.; LI, T. – RFC 1771: A Border Gateway Protocol 4 (BGP-4). March 1995.

[REK97] REKHTER, Y. et al. – RFC 2105: Tag Switching Architecture Overview. February 1997.

[ROS01] ROSEN, E. et al. – RFC 3031: Multiprotocol Label Switching Architecture. January 2001.

[TAN97] TANENBAUM, A. S. – Redes De Computadores. 3.ed. Ed.Campus. 1997.

[THO01] THOMAS, B.; GRAY, E. – RFC 3037: LDP Applicability. January 2001.

[XIA99] XIAO, X.; LI, L. M. – Internet QoS: a big picture. IEEE Network, v.13, n.2, p.8-18, March/April 1999.

Anexo I

Neste anexo estão as listagens dos códigos fontes, composto por três arquivos, para a simulação do cenário II deste trabalho. O principal (start.tcl) traz o conteúdo do arquivo primário que irá desencadear os eventos da simulação. O cenário da topologia foi representado como cenario.tcl e os monitores dos enlaces ficaram no monitores.tcl. Porem, não foi colocado no anexo o código fonte que trata a rotina de caminho mínimo representada por *spf-dijkstra*.

I.1. Arquivo Principal (start.tcl) da Topologia II

```
set ns [new Simulator]
$ns rtproto DV
set nf [open cena01.nam w]
$ns namtrace-all $nf
proc finish {} {
    global ns nf nl nr file
    $ns flush-trace
    close $nf
    puts " - flushing delay & loss information"
    for {set i 0} {$i < 36} {incr i} {
      flush $nl($i)
      flush $nr($i)
      close $nl($i)
      close $nr($i)
    }
    exec nam cena01.nam &
    exit 0
}
source "cenario.tcl"
source "variaveis.tcl"
source "monitores.tcl"
for {set i 0} {$i < 9} {incr i} {
    set a LSR($i)
    for {set j [expr $i+1]} {$j < 9} {incr j} {
      set b LSR($j)
      eval $ns LDP-peer $$a $$b
      }
    set m [eval $$a get-module "MPLS"]
    $m enable-reroute "new"
}
$ns ldp-request-color       blue
$ns ldp-mapping-color       red
$ns ldp-withdraw-color      pink
$ns ldp-release-color       orange
$ns ldp-notification-color  yellow
set to data-driven
Classifier/Addr/MPLS set control_driven_ 1
```

```
Classifier/Addr/MPLS enable-on-demand
Classifier/Addr/MPLS enable-ordered-control
[$LSR(0) get-module "MPLS"] enable-control-driven
[$LSR(0) get-module "MPLS"] enable-data-driven
Agent/LDP set trace_ldp_ 1
Classifier/Addr/MPLS set trace_mpls_ 1
$ns use-scheduler List
proc attach-const-traffic { node sink size burst rate col } {
    global ns
    set udp [new Agent/UDP]
    $ns attach-agent $node $udp
    set traffic [new Application/Traffic/CBR]
    $traffic set packetSize_ $size
    $traffic set burst_time_ $burst
    $traffic set rate_ $rate
    $traffic set class_ $col
    $udp     set fid_ $col
    $traffic attach-agent $udp
    $ns connect $udp $sink
    return $traffic
}
proc attach-expoo-traffic { node sink size burst idle rate col } {
    global ns
    set udp [new Agent/UDP]
    $ns attach-agent $node $udp
    set traffic [new Application/Traffic/Exponential]
    $traffic set packetSize_ $size
    $traffic set burst_time_ $burst
    $traffic set idle_time_ $idle
    $traffic set rate_ $rate
    $traffic set class_ $col
    $udp     set fid_ $col
    $traffic attach-agent $udp
    $ns connect $udp $sink
    return $traffic
}
set k 36
for {set i 0} {$i < 36} {incr i} {
    set sink($i) [new Agent/LossMonitor]
    $ns attach-agent $node($k) $sink($i)
    set k [expr $k + 1]
}
set Tba 3145728.0 ; # 1048576.0 1310720.0 1572864.0 2097152.0 3145728.0
set Tb  3.0Mb ; # 1.0Mb 1.25Mb 1.5Mb 2.0Mb 3.0Mb
set cbr0  [attach-const-traffic $node(0)  $sink(34) 572 0.5 $Tb 5]
set cbr1  [attach-const-traffic $node(2)  $sink(32) 572 0.5 $Tb 5]
set cbr2  [attach-const-traffic $node(4)  $sink(30) 572 0.5 $Tb 5]
set cbr3  [attach-const-traffic $node(6)  $sink(28) 572 0.5 $Tb 5]
set cbr4  [attach-const-traffic $node(8)  $sink(26) 572 0.5 $Tb 5]
set cbr5  [attach-const-traffic $node(10) $sink(24) 572 0.5 $Tb 5]
set cbr6  [attach-const-traffic $node(12) $sink(22) 572 0.5 $Tb 5]
set cbr7  [attach-const-traffic $node(14) $sink(20) 572 0.5 $Tb 5]
set cbr8  [attach-const-traffic $node(16) $sink(2)  572 0.5 $Tb 5]
set cbr9  [attach-const-traffic $node(18) $sink(0)  572 0.5 $Tb 5]
set cbr10 [attach-const-traffic $node(20) $sink(14) 572 0.5 $Tb 5]
set cbr11 [attach-const-traffic $node(22) $sink(12) 572 0.5 $Tb 5]
set cbr12 [attach-const-traffic $node(24) $sink(10) 572 0.5 $Tb 5]
```

```
set cbr13 [attach-const-traffic $node(26) $sink(8)  572 0.5 $Tb 5]
set cbr14 [attach-const-traffic $node(28) $sink(6)  572 0.5 $Tb 5]
set cbr15 [attach-const-traffic $node(30) $sink(4)  572 0.5 $Tb 5]
set cbr16 [attach-const-traffic $node(32) $sink(18) 572 0.5 $Tb 5]
set cbr17 [attach-const-traffic $node(34) $sink(16) 572 0.5 $Tb 5]
set vbr0  [attach-expoo-traffic $node(1)  $sink(35) 572 0.5 0.5 $Tb 3]
set vbr1  [attach-expoo-traffic $node(3)  $sink(33) 572 0.5 0.5 $Tb 3]
set vbr2  [attach-expoo-traffic $node(5)  $sink(31) 572 0.5 0.5 $Tb 3]
set vbr3  [attach-expoo-traffic $node(7)  $sink(29) 572 0.5 0.5 $Tb 3]
set vbr4  [attach-expoo-traffic $node(9)  $sink(27) 572 0.5 0.5 $Tb 3]
set vbr5  [attach-expoo-traffic $node(11) $sink(25) 572 0.5 0.5 $Tb 3]
set vbr6  [attach-expoo-traffic $node(13) $sink(23) 572 0.5 0.5 $Tb 3]
set vbr7  [attach-expoo-traffic $node(15) $sink(21) 572 0.5 0.5 $Tb 3]
set vbr8  [attach-expoo-traffic $node(17) $sink(3)  572 0.5 0.5 $Tb 3]
set vbr9  [attach-expoo-traffic $node(19) $sink(1)  572 0.5 0.5 $Tb 3]
set vbr10 [attach-expoo-traffic $node(21) $sink(15) 572 0.5 0.5 $Tb 3]
set vbr11 [attach-expoo-traffic $node(23) $sink(13) 572 0.5 0.5 $Tb 3]
set vbr12 [attach-expoo-traffic $node(25) $sink(11) 572 0.5 0.5 $Tb 3]
set vbr13 [attach-expoo-traffic $node(27) $sink(9)  572 0.5 0.5 $Tb 3]
set vbr14 [attach-expoo-traffic $node(29) $sink(7)  572 0.5 0.5 $Tb 3]
set vbr15 [attach-expoo-traffic $node(31) $sink(5)  572 0.5 0.5 $Tb 3]
set vbr16 [attach-expoo-traffic $node(33) $sink(19) 572 0.5 0.5 $Tb 3]
set vbr17 [attach-expoo-traffic $node(35) $sink(17) 572 0.5 0.5 $Tb 3]
for {set i 0} {$i < 36} {incr i} {
    set bw($i) 0
    set totalpkt($i) 0
    set pktlost($i) 0
    set pktrcv($i) 0
}
proc record {} {
    global totalpkt pktlost pktrcv bw ns sink file nl nr steptime
    set time 0.005
    set now [$ns now]
    for {set i 0} {$i < 36} {incr i} {
      $sink($i) instvar nlost_ npkts_ bytes_
      set bw($i)      $bytes_
      set pktlost($i) [expr $pktlost($i) + $nlost_]
      set pktrcv($i)  [expr $pktrcv($i)  + $npkts_]
      set bw($i) [expr $bw($i) / 200]
      puts $nl($i)   "$now $pktlost($i)"
      puts $nr($i)   "$now $pktrcv($i)"
      set totalpkt($i) [expr $totalpkt($i) + $npkts_]
      set bytes_ 0
      set nlost_ 0
      set npkts_ 0
      flush $nl($i)
      flush $nr($i)
    }
    set nexttime [expr $now + $steptime]
    $ns at $nexttime "record"
}
proc recv-pkts {} {
    global totalpkt pktlost
    flush stdout
    puts "=========================================================="
    puts "=                                                        ="
    puts "=========================================================="
```

```
    puts " "
    for {set i 0} {$i < 36} {incr i} {
      puts " O <NO [expr 45 + $i]> recebeu $totalpkt($i)  Perdeu $pktlost($i)
pacotes"
    }
    puts " "
    puts "======================================================="
}
proc Engenharia_Trafego {aplic ing egr flbyte fec LSPid} {
    global ns ordem fator rota MEC CUSTMEC rota_explicita k B Blink Tba LSR
    set now [$ns now]
    set rota ""
    if {$ing == 36} {
      set x $LSR(0)
    } elseif {$ing == 37} {
      set x $LSR(1)
    } elseif {$ing == 38} {
      set x $LSR(2)
    } elseif {$ing == 39} {
      set x $LSR(3)
    } elseif {$ing == 40} {
      set x $LSR(4)
    } elseif {$ing == 41} {
      set x $LSR(5)
    } elseif {$ing == 42} {
      set x $LSR(6)
    } elseif {$ing == 43} {
      set x $LSR(7)
    } else {
      set x $LSR(8)
    }
    if {$egr == 36} {
      set y $LSR(0)
    } elseif {$egr == 37} {
      set y $LSR(1)
    } elseif {$egr == 38} {
      set y $LSR(2)
    } elseif {$egr == 39} {
      set y $LSR(3)
    } elseif {$egr == 40} {
      set y $LSR(4)
    } elseif {$egr == 41} {
      set y $LSR(5)
    } elseif {$egr == 42} {
      set y $LSR(6)
    } elseif {$egr == 43} {
      set y $LSR(7)
    } else {
      set y $LSR(8)
    }
    for {set i 0} {$i < $ordem} {incr i} {
      for {set j 0} {$j < $ordem} {incr j} {
          set aux [concat $CUSTMEC($i,$j)/]
          set rota [concat $rota$aux]
      }
    }
    set lsr_inicio [expr $ing - 35]
```

```
    set lsr_fim    [expr $egr - 35]
    set rota_explicita [[$x get-module MPLS] spf-dijkstra $lsr_inicio $lsr_fim
$rota $ordem $fator]
    #Conta o número de nós que possui o roteamento
    set conta_no_rot [string length $rota_explicita]
    set j 0
    # transforma caracter em número
    set j 0
    for {set i 0} {$i < $conta_no_rot} {incr i} {
      set caminho($i) ""
    }
    for {set i 0} {$i < $conta_no_rot} {incr i} {
      set cam [string index $rota_explicita $i]
      if {$cam != "_"} {
          set caminho($j) [concat $caminho($j)$cam]
      } else {
          set j [expr $j + 1]
      }
    }
    puts "$now - aplicacao $aplic"
    for {set i 0} {$i < $j} {incr i} {
      set l [expr $caminho($i) - 36]
      set c [expr $caminho([expr $i + 1]) - 36]
      puts "l = $l - $caminho($i)"
      puts "c = $c - $caminho([expr $i + 1])"
      set CUSTMEC($l,$c) [expr $CUSTMEC($l,$c) * [expr [expr round([expr
exp([expr $k * [expr $MEC($l,$c) / $Blink]])])] * [expr round([expr exp([expr
$B * [expr $Tba / $Blink]])])]]]
    }

    $ns at $now "[$y get-module MPLS] ldp-trigger-by-withdraw $fec -1"
    $ns at $now "[$x get-module MPLS] make-explicit-route $fec $rota_explicita
$LSPid -1"
    $ns at [expr $now + 0.09] "[$x get-module MPLS] flow-erlsp-install $fec -1
$LSPid"
}
$ns at 0.0 "record"
$ns at 0.0 "monitore_3637" ; $ns at 0.0 "monitore_3736"
$ns at 0.0 "monitore_3638" ; $ns at 0.0 "monitore_3836"
$ns at 0.0 "monitore_3639" ; $ns at 0.0 "monitore_3936"
$ns at 0.0 "monitore_3738" ; $ns at 0.0 "monitore_3837"
$ns at 0.0 "monitore_3744" ; $ns at 0.0 "monitore_4437"
$ns at 0.0 "monitore_3839" ; $ns at 0.0 "monitore_3938"
$ns at 0.0 "monitore_3840" ; $ns at 0.0 "monitore_4038"
$ns at 0.0 "monitore_3843" ; $ns at 0.0 "monitore_4338"
$ns at 0.0 "monitore_3844" ; $ns at 0.0 "monitore_4438"
$ns at 0.0 "monitore_3940" ; $ns at 0.0 "monitore_4039"
$ns at 0.0 "monitore_3941" ; $ns at 0.0 "monitore_4139"
$ns at 0.0 "monitore_4041" ; $ns at 0.0 "monitore_4140"
$ns at 0.0 "monitore_4042" ; $ns at 0.0 "monitore_4240"
$ns at 0.0 "monitore_4043" ; $ns at 0.0 "monitore_4340"
$ns at 0.0 "monitore_4142" ; $ns at 0.0 "monitore_4241"
$ns at 0.0 "monitore_4243" ; $ns at 0.0 "monitore_4342"
$ns at 0.0 "monitore_4344" ; $ns at 0.0 "monitore_4443"
$ns at 0.5 "Engenharia_Trafego cbr0 36 44 572 79 1079"
$ns at 0.7 "$cbr0 start"
$ns at 1.5 "Engenharia_Trafego cbr1 36 44 572 77 1077"
```

```
$ns at 1.7 "$cbr1 start"
$ns at 2.5 "Engenharia_Trafego cbr2 37 43 572 75 1075"
$ns at 2.7 "$cbr2 start"
$ns at 3.5 "Engenharia_Trafego cbr3 37 43 572 73 1073"
$ns at 3.7 "$cbr3 start"
$ns at 4.5 "Engenharia_Trafego cbr4 38 42 572 71 1071"
$ns at 4.7 "$cbr4 start"
$ns at 5.5 "Engenharia_Trafego cbr5 38 42 572 69 1069"
$ns at 5.7 "$cbr5 start"
$ns at 6.5 "Engenharia_Trafego cbr6 39 41 572 67 1067"
$ns at 6.7 "$cbr6 start"
$ns at 7.5 "Engenharia_Trafego cbr7 39 41 572 65 1065"
$ns at 7.7 "$cbr7 start"
$ns at 8.5 "Engenharia_Trafego cbr8 40 36 572 45 1045"
$ns at 8.7 "$cbr8 start"
$ns at 9.5 "Engenharia_Trafego cbr9 40 36 572 43 1043"
$ns at 9.7 "$cbr9 start"

$ns at 10.5 "Engenharia_Trafego cbr10 41 39 572 59 1059"
$ns at 10.7 "$cbr10 start"
$ns at 11.5 "Engenharia_Trafego cbr11 41 39 572 57 1057"
$ns at 11.7 "$cbr11 start"
$ns at 12.5 "Engenharia_Trafego cbr12 42 38 572 55 1055"
$ns at 12.7 "$cbr12 start"
$ns at 13.5 "Engenharia_Trafego cbr13 42 38 572 53 1053"
$ns at 13.7 "$cbr13 start"
$ns at 14.5 "Engenharia_Trafego cbr14 43 37 572 51 1051"
$ns at 14.7 "$cbr14 start"
$ns at 15.5 "Engenharia_Trafego cbr15 43 37 572 49 1049"
$ns at 15.7 "$cbr15 start"
$ns at 16.5 "Engenharia_Trafego cbr16 44 40 572 63 1063"
$ns at 16.7 "$cbr16 start"
$ns at 17.5 "Engenharia_Trafego cbr17 44 40 572 61 1061"
$ns at 17.7 "$cbr17 start"
$ns at 1.0 "Engenharia_Trafego vbr0 36 44 572 80 1080"
$ns at 1.2 "$vbr0 start"
$ns at 2.0 "Engenharia_Trafego vbr1 36 44 572 78 1078"
$ns at 2.2 "$vbr1 start"
$ns at 3.0 "Engenharia_Trafego vbr2 37 43 572 76 1076"
$ns at 3.2  "$vbr2 start"
$ns at 4.0 "Engenharia_Trafego vbr3 37 43 572 74 1074"
$ns at 4.2 "$vbr3 start"
$ns at 5.0 "Engenharia_Trafego vbr4 38 42 572 72 1072"
$ns at 5.2 "$vbr4 start"
$ns at 6.0 "Engenharia_Trafego vbr5 38 42 572 70 1070"
$ns at 6.2 "$vbr5 start"
$ns at 7.0 "Engenharia_Trafego vbr6 39 41 572 68 1068"
$ns at 7.2 "$vbr6 start"
$ns at 8.0 "Engenharia_Trafego vbr7 39 41 572 66 1066"
$ns at 8.2 "$vbr7 start"
$ns at 9.0 "Engenharia_Trafego vbr8 40 36 572 64 1064"
$ns at 9.2 "$vbr8 start"
$ns at 10.0 "Engenharia_Trafego vbr9 40 36 572 46 1046"
$ns at 10.2 "$vbr9 start"
$ns at 11.0 "Engenharia_Trafego vbr10 41 39 572 44 1044"
$ns at 11.2 "$vbr10 start"
$ns at 12.0 "Engenharia_Trafego vbr11 41 39 572 58 1058"
```

```
$ns at 12.2 "$vbr11 start"
$ns at 13.0 "Engenharia_Trafego vbr12 42 38 572 56 1056"
$ns at 13.2 "$vbr12 start"
$ns at 14.0 "Engenharia_Trafego vbr13 42 38 572 54 1054"
$ns at 14.2 "$vbr13 start"
$ns at 15.0 "Engenharia_Trafego vbr14 43 37 572 52 1052"
$ns at 15.2 "$vbr14 start"
$ns at 16.0 "Engenharia_Trafego vbr15 43 37 572 50 1050"
$ns at 16.2 "$vbr15 start"
$ns at 17.0 "Engenharia_Trafego vbr16 44 40 572 64 1064"
$ns at 17.2 "$vbr16 start"
$ns at 18.0 "Engenharia_Trafego vbr17 44 40 572 62 1062"
$ns at 18.2 "$vbr17 start"
$ns at 16.0 "$cbr0 stop"  ; $ns at 17.0 "$cbr1 stop"
$ns at 18.0 "$cbr2 stop"  ; $ns at 19.0 "$cbr3 stop"
$ns at 20.0 "$cbr4 stop"  ; $ns at 21.0 "$cbr5 stop"
$ns at 22.0 "$cbr6 stop"  ; $ns at 23.0 "$cbr7 stop"
$ns at 24.0 "$cbr8 stop"  ; $ns at 25.0 "$cbr9 stop"
$ns at 26.0 "$cbr10 stop" ; $ns at 27.0 "$cbr11 stop"
$ns at 28.0 "$cbr12 stop" ; $ns at 29.0 "$cbr13 stop"
$ns at 30.0 "$cbr14 stop" ; $ns at 31.0 "$cbr15 stop"
$ns at 32.0 "$cbr16 stop" ; $ns at 33.0 "$cbr17 stop"
$ns at 16.5 "$vbr0 stop"  ; $ns at 17.5 "$vbr1 stop"
$ns at 18.5 "$vbr2 stop"  ; $ns at 19.5 "$vbr3 stop"
$ns at 20.5 "$vbr4 stop"  ; $ns at 21.5 "$vbr5 stop"
$ns at 22.5 "$vbr6 stop"  ; $ns at 23.5 "$vbr7 stop"
$ns at 24.5 "$vbr8 stop"  ; $ns at 25.5 "$vbr9 stop"
$ns at 26.5 "$vbr10 stop" ; $ns at 27.5 "$vbr11 stop"
$ns at 28.5 "$vbr12 stop" ; $ns at 29.5 "$vbr13 stop"
$ns at 30.5 "$vbr14 stop" ; $ns at 31.5 "$vbr15 stop"
$ns at 32.5 "$vbr16 stop" ; $ns at 33.5 "$vbr17 stop"
$ns at 35.0 "recv-pkts"
$ns at 35.2 "finish"
$ns run
```

I.2. Arquivo Cenário (cenario.tcl) da Topologia II

```
for {set i 0} {$i < 36} {incr i} {
    set node($i) [$ns node]
    $node($i) color blue
}
$ns node-config -MPLS ON
for {set i 0} {$i < 9} {incr i} {
    set LSR($i)  [$ns node]
}
$ns node-config -MPLS OFF
for {set i 36} {$i < 72} {incr i} {
    set node($i) [$ns node]
    $node($i) color orange
}
$ns duplex-link $LSR(0) $LSR(1) 4.0Mb 3ms DropTail
$ns queue-limit $LSR(0) $LSR(1) 100
$ns duplex-link-op $LSR(0) $LSR(1) queuePos 0.5
$ns duplex-link $LSR(0) $LSR(2) 4.0Mb 3ms DropTail
$ns queue-limit $LSR(0) $LSR(2) 100
$ns duplex-link-op $LSR(0) $LSR(2) queuePos 0.5
$ns duplex-link $LSR(0) $LSR(3) 4.0Mb 3ms DropTail
```

```
$ns queue-limit $LSR(0) $LSR(3) 100
$ns duplex-link-op $LSR(0) $LSR(3) queuePos 0.5
$ns duplex-link $LSR(1) $LSR(2) 4.0Mb 3ms DropTail
$ns queue-limit $LSR(1) $LSR(2) 100
$ns duplex-link-op $LSR(1) $LSR(2) queuePos 0.5
$ns duplex-link $LSR(1) $LSR(8) 4.0Mb 3ms DropTail
$ns queue-limit $LSR(1) $LSR(8) 100
$ns duplex-link-op $LSR(1) $LSR(8) queuePos 0.5
$ns duplex-link $LSR(2) $LSR(3) 4.0Mb 3ms DropTail
$ns queue-limit $LSR(2) $LSR(3) 100
$ns duplex-link-op $LSR(2) $LSR(3) queuePos 0.5
$ns duplex-link $LSR(2) $LSR(4) 4.0Mb 3ms DropTail
$ns queue-limit $LSR(2) $LSR(4) 100
$ns duplex-link-op $LSR(2) $LSR(4) queuePos 0.5
$ns duplex-link $LSR(2) $LSR(7) 4.0Mb 3ms DropTail
$ns queue-limit $LSR(2) $LSR(7) 100
$ns duplex-link-op $LSR(2) $LSR(7) queuePos 0.5
$ns duplex-link $LSR(2) $LSR(8) 4.0Mb 3ms DropTail
$ns queue-limit $LSR(2) $LSR(8) 100
$ns duplex-link-op $LSR(2) $LSR(8) queuePos 0.5
$ns duplex-link $LSR(3) $LSR(4) 4.0Mb 3ms DropTail
$ns queue-limit $LSR(3) $LSR(4) 100
$ns duplex-link-op $LSR(3) $LSR(4) queuePos 0.5
$ns duplex-link $LSR(3) $LSR(5) 4.0Mb 3ms DropTail
$ns queue-limit $LSR(3) $LSR(5) 100
$ns duplex-link-op $LSR(3) $LSR(5) queuePos 0.5
$ns duplex-link $LSR(4) $LSR(5) 4.0Mb 3ms DropTail
$ns queue-limit $LSR(4) $LSR(5) 100
$ns duplex-link-op $LSR(4) $LSR(5) queuePos 0.5
$ns duplex-link $LSR(4) $LSR(6) 4.0Mb 3ms DropTail
$ns queue-limit $LSR(4) $LSR(6) 100
$ns duplex-link-op $LSR(4) $LSR(6) queuePos 0.5
$ns duplex-link $LSR(4) $LSR(7) 4.0Mb 3ms DropTail
$ns queue-limit $LSR(4) $LSR(7) 100
$ns duplex-link-op $LSR(4) $LSR(7) queuePos 0.5
$ns duplex-link $LSR(5) $LSR(6) 4.0Mb 3ms DropTail
$ns queue-limit $LSR(5) $LSR(6) 100
$ns duplex-link-op $LSR(5) $LSR(6) queuePos 0.5
$ns duplex-link $LSR(6) $LSR(7) 4.0Mb 3ms DropTail
$ns queue-limit $LSR(6) $LSR(7) 100
$ns duplex-link-op $LSR(6) $LSR(7) queuePos 0.5
$ns duplex-link $LSR(7) $LSR(8) 4.0Mb 3ms DropTail
$ns queue-limit $LSR(7) $LSR(8) 100
$ns duplex-link-op $LSR(7) $LSR(8) queuePos 0.5
set f 0
set d 36
for {set l 0} {$l < 9} {incr l} {
    set k 0
    while {$k < 4} {
      $ns duplex-link $node($f) $LSR($l) 4.0Mb 3ms DropTail
      $ns queue-limit $node($f) $LSR($l) 100
      $ns duplex-link-op $node($f) $LSR($l) queuePos 0.5
      $ns duplex-link $node($d) $LSR($l) 4.0Mb 3ms DropTail
      $ns queue-limit $node($d) $LSR($l) 100
      $ns duplex-link-op $node($d) $LSR($l) queuePos 0.5
      set k [expr $k + 1]
      set f [expr $f + 1]
```

```
      set d [expr $d + 1]
    }
}
```

I.3. Arquivo Variáveis (variáveis.tcl) da Topologia II

```
set tcl_precision 17
$ns color 1 green
$ns color 2 orange
$ns color 3 red
$ns color 4 brown
$ns color 5 yellow
for {set i 0} {$i < 36} {incr i} {
    set f [expr 36 + $i]
    set nl($i)   [open lost$f.dat w]
    set nr($i)   [open rcv$f.dat w]
    set file($i) [open f$f.dat w]
}
set lsr_inicio 0
set IniLink 0
set lsr_fim 0
set rota ""
set ordem 9
set fator 35
set steptime 0.3
set launchTime 0.5
for {set i 0} {$i != $ordem} {incr i} {
    for {set j 0} {$j != $ordem} {incr j} {
      set CUSTMEC($i,$j) 0
      set MEC($i,$j) 0.0
    }
}
set CUSTMEC(0,1) 1 ; set CUSTMEC(1,0) 1
set CUSTMEC(0,2) 1 ; set CUSTMEC(2,0) 1
set CUSTMEC(0,3) 1 ; set CUSTMEC(3,0) 1
set CUSTMEC(1,2) 1 ; set CUSTMEC(2,1) 1
set CUSTMEC(1,8) 1 ; set CUSTMEC(8,1) 1
set CUSTMEC(2,3) 1 ; set CUSTMEC(3,2) 1
set CUSTMEC(2,4) 1 ; set CUSTMEC(4,2) 1
set CUSTMEC(2,7) 1 ; set CUSTMEC(7,2) 1
set CUSTMEC(2,8) 1 ; set CUSTMEC(8,2) 1
set CUSTMEC(3,4) 1 ; set CUSTMEC(4,3) 1
set CUSTMEC(3,5) 1 ; set CUSTMEC(5,3) 1
set CUSTMEC(4,5) 1 ; set CUSTMEC(5,4) 1
set CUSTMEC(4,6) 1 ; set CUSTMEC(6,4) 1
set CUSTMEC(4,7) 1 ; set CUSTMEC(7,4) 1
set CUSTMEC(5,6) 1 ; set CUSTMEC(6,5) 1
set CUSTMEC(6,7) 1 ; set CUSTMEC(7,6) 1
set CUSTMEC(7,8) 1 ; set CUSTMEC(8,7) 1
set rota_explicita ""
set k 2.0 ; set B 2.0 ; set Blink 4194304.0 ; set steptime 1.0
```

I.4. Arquivo Monitores (monitores.tcl) da Topologia II

```
set fmon(0)  [open f3637.dat w]
set qmon(0)  [$ns monitor-queue $LSR(0) $LSR(1) $fmon(0)]
set fmon(1)  [open f3736.dat w]
set qmon(1)  [$ns monitor-queue $LSR(1) $LSR(0) $fmon(1)]
set fmon(2)  [open f3638.dat w]
set qmon(2)  [$ns monitor-queue $LSR(0) $LSR(2) $fmon(2)]
set fmon(3)  [open f3836.dat w]
set qmon(3)  [$ns monitor-queue $LSR(2) $LSR(0) $fmon(3)]
set fmon(4)  [open f3639.dat w]
set qmon(4)  [$ns monitor-queue $LSR(0) $LSR(3) $fmon(4)]
set fmon(5)  [open f3936.dat w]
set qmon(5)  [$ns monitor-queue $LSR(3) $LSR(0) $fmon(5)]
set fmon(6)  [open f3738.dat w]
set qmon(6)  [$ns monitor-queue $LSR(1) $LSR(2) $fmon(6)]
set fmon(7)  [open f3837.dat w]
set qmon(7)  [$ns monitor-queue $LSR(2) $LSR(1) $fmon(7)]
set fmon(8)  [open f3744.dat w]
set qmon(8)  [$ns monitor-queue $LSR(1) $LSR(8) $fmon(8)]
set fmon(9)  [open f4437.dat w]
set qmon(9)  [$ns monitor-queue $LSR(8) $LSR(1) $fmon(9)]
set fmon(10) [open f3839.dat w]
set qmon(10) [$ns monitor-queue $LSR(2) $LSR(3) $fmon(10)]
set fmon(11) [open f3938.dat w]
set qmon(11) [$ns monitor-queue $LSR(3) $LSR(2) $fmon(11)]
set fmon(12) [open f3840.dat w]
set qmon(12) [$ns monitor-queue $LSR(2) $LSR(4) $fmon(12)]
set fmon(13) [open f4038.dat w]
set qmon(13) [$ns monitor-queue $LSR(4) $LSR(2) $fmon(13)]
set fmon(14) [open f3843.dat w]
set qmon(14) [$ns monitor-queue $LSR(2) $LSR(7) $fmon(14)]
set fmon(15) [open f4338.dat w]
set qmon(15) [$ns monitor-queue $LSR(7) $LSR(2) $fmon(15)]
set fmon(16) [open f3844.dat w]
set qmon(16) [$ns monitor-queue $LSR(2) $LSR(8) $fmon(16)]
set fmon(17) [open f4438.dat w]
set qmon(17) [$ns monitor-queue $LSR(8) $LSR(2) $fmon(17)]
set fmon(18) [open f3940.dat w]
set qmon(18) [$ns monitor-queue $LSR(3) $LSR(4) $fmon(18)]
set fmon(19) [open f4039.dat w]
set qmon(19) [$ns monitor-queue $LSR(4) $LSR(3) $fmon(19)]
set fmon(20) [open f3941.dat w]
set qmon(20) [$ns monitor-queue $LSR(3) $LSR(5) $fmon(20)]
set fmon(21) [open f4139.dat w]
set qmon(21) [$ns monitor-queue $LSR(5) $LSR(3) $fmon(21)]
set fmon(22) [open f4041.dat w]
set qmon(22) [$ns monitor-queue $LSR(4) $LSR(5) $fmon(22)]
set fmon(23) [open f4140.dat w]
set qmon(23) [$ns monitor-queue $LSR(5) $LSR(4) $fmon(23)]
set fmon(24) [open f4042.dat w]
set qmon(24) [$ns monitor-queue $LSR(4) $LSR(6) $fmon(24)]
set fmon(25) [open f4240.dat w]
set qmon(25) [$ns monitor-queue $LSR(6) $LSR(4) $fmon(25)]
set fmon(26) [open f4043.dat w]
set qmon(26) [$ns monitor-queue $LSR(4) $LSR(7) $fmon(26)]
```

```
set fmon(27) [open f4340.dat w]
set qmon(27) [$ns monitor-queue $LSR(7) $LSR(4) $fmon(27)]
set fmon(28) [open f4142.dat w]
set qmon(28) [$ns monitor-queue $LSR(5) $LSR(6) $fmon(28)]
set fmon(29) [open f4241.dat w]
set qmon(29) [$ns monitor-queue $LSR(6) $LSR(5) $fmon(29)]
set fmon(30) [open f4243.dat w]
set qmon(30) [$ns monitor-queue $LSR(6) $LSR(7) $fmon(30)]
set fmon(31) [open f4342.dat w]
set qmon(31) [$ns monitor-queue $LSR(7) $LSR(6) $fmon(31)]
set fmon(32) [open f4344.dat w]
set qmon(32) [$ns monitor-queue $LSR(7) $LSR(8) $fmon(32)]
set fmon(33) [open f4443.dat w]
set qmon(33) [$ns monitor-queue $LSR(8) $LSR(7) $fmon(33)]
proc monitore_3637 {} {
    global ns MEC qmon fmon k Blink integ steptime
    set now [$ns now]
    set i 0
    set j 1
    $qmon(0) instvar size_ pkts_ parrivals_ barrivals_ pdepartures_
bdepartures_ pdrops_ bdrops_
    set MEC($i,$j) $size_
    puts $fmon(0) "$now $pdrops_"
    $qmon(0) set pdrops_ 0
    $qmon(0) set barrivals_ 0
    set nexttime [expr $now + $steptime]
    $ns at $nexttime "monitore_3637"
}
proc monitore_3736 {} {
    global ns MEC qmon fmon k Blink integ steptime
    set now [$ns now]
    set i 1
    set j 0
    $qmon(1) instvar size_ pkts_ parrivals_ barrivals_ pdepartures_
bdepartures_ pdrops_ bdrops_
    set MEC($i,$j) $size_
    puts $fmon(1) "$now $pdrops_"
    $qmon(1) set pdrops_ 0
    $qmon(1) set barrivals_ 0
    set nexttime [expr $now + $steptime]
    $ns at $nexttime "monitore_3736"
}
proc monitore_3638 {} {
    global ns MEC qmon fmon k Blink integ steptime
    set now [$ns now]
    set i 0
    set j 2
    $qmon(2) instvar size_ pkts_ parrivals_ barrivals_ pdepartures_
bdepartures_ pdrops_ bdrops_
    set MEC($i,$j) $size_
    puts $fmon(2) "$now $pdrops_"
    $qmon(2) set pdrops_ 0
    $qmon(2) set barrivals_ 0
    set nexttime [expr $now + $steptime]
    $ns at $nexttime "monitore_3638"
}
```

```
proc monitore_3836 {} {
    global ns MEC qmon fmon k Blink integ steptime
    set now [$ns now]
    set i 2
    set j 0
    $qmon(3) instvar size_ pkts_ parrivals_ barrivals_ pdepartures_
bdepartures_ pdrops_ bdrops_
    set MEC($i,$j) $size_
    puts $fmon(3) "$now $pdrops_"
    $qmon(3) set pdrops_ 0
    $qmon(3) set barrivals_ 0
    set nexttime [expr $now + $steptime]
    $ns at $nexttime "monitore_3836"
}
proc monitore_3639 {} {
    global ns MEC qmon fmon k Blink integ steptime
    set now [$ns now]
    set i 0
    set j 3
    $qmon(4) instvar size_ pkts_ parrivals_ barrivals_ pdepartures_
bdepartures_ pdrops_ bdrops_
    set MEC($i,$j) $size_
    puts $fmon(4) "$now $pdrops_"
    $qmon(4) set pdrops_ 0
    $qmon(4) set barrivals_ 0
    set nexttime [expr $now + $steptime]
    $ns at $nexttime "monitore_3639"
}
proc monitore_3936 {} {
    global ns MEC qmon fmon k Blink integ steptime
    set now [$ns now]
    set i 3
    set j 0
    $qmon(5) instvar size_ pkts_ parrivals_ barrivals_ pdepartures_
bdepartures_ pdrops_ bdrops_
    set MEC($i,$j) $size_
    puts $fmon(5) "$now $pdrops_"
    $qmon(5) set pdrops_ 0
    $qmon(5) set barrivals_ 0
    set nexttime [expr $now + $steptime]
    $ns at $nexttime "monitore_3936"
}
proc monitore_3738 {} {
    global ns MEC qmon fmon k Blink integ steptime
    set now [$ns now]
    set i 1
    set j 2
    $qmon(6) instvar size_ pkts_ parrivals_ barrivals_ pdepartures_
bdepartures_ pdrops_ bdrops_
    set MEC($i,$j) $size_
    puts $fmon(6) "$now $pdrops_"
    $qmon(6) set pdrops_ 0
    $qmon(6) set barrivals_ 0
    set nexttime [expr $now + $steptime]
    $ns at $nexttime "monitore_3738"
}
```

```
proc monitore_3837 {} {
    global ns MEC qmon fmon k Blink integ steptime
    set now [$ns now]
    set i 2
    set j 1
    $qmon(7) instvar size_ pkts_ parrivals_ barrivals_ pdepartures_
bdepartures_ pdrops_ bdrops_
    set MEC($i,$j) $size_
    puts $fmon(7) "$now $pdrops_"
    $qmon(7) set pdrops_ 0
    $qmon(7) set barrivals_ 0
    set nexttime [expr $now + $steptime]
    $ns at $nexttime "monitore_3837"
}
proc monitore_3744 {} {
    global ns MEC qmon fmon k Blink integ steptime
    set now [$ns now]
    set i 1
    set j 8
    $qmon(8) instvar size_ pkts_ parrivals_ barrivals_ pdepartures_
bdepartures_ pdrops_ bdrops_
    set MEC($i,$j) $size_
    puts $fmon(8) "$now $pdrops_"
    $qmon(8) set pdrops_ 0
    $qmon(8) set barrivals_ 0
    set nexttime [expr $now + $steptime]
    $ns at $nexttime "monitore_3744"
}
proc monitore_4437 {} {
    global ns MEC qmon fmon k Blink integ steptime
    set now [$ns now]
    set i 8
    set j 1
    $qmon(9) instvar size_ pkts_ parrivals_ barrivals_ pdepartures_
bdepartures_ pdrops_ bdrops_
    set MEC($i,$j) $size_
    puts $fmon(9) "$now $pdrops_"
    $qmon(9) set pdrops_ 0
    $qmon(9) set barrivals_ 0
    set nexttime [expr $now + $steptime]
    $ns at $nexttime "monitore_4437"
}
proc monitore_3839 {} {
    global ns MEC qmon fmon k Blink integ steptime
    set now [$ns now]
    set i 2
    set j 3
    $qmon(10) instvar size_ pkts_ parrivals_ barrivals_ pdepartures_
bdepartures_ pdrops_ bdrops_
    set MEC($i,$j) $size_
    puts $fmon(10) "$now $pdrops_"
    $qmon(10) set pdrops_ 0
    $qmon(10) set barrivals_ 0
    set nexttime [expr $now + $steptime]
    $ns at $nexttime "monitore_3839"
}
```

```
proc monitore_3938 {} {
    global ns MEC qmon fmon k Blink integ steptime
    set now [$ns now]
    set i 3
    set j 2
    $qmon(11) instvar size_ pkts_ parrivals_ barrivals_ pdepartures_
bdepartures_ pdrops_ bdrops_
    set MEC($i,$j) $size_
    puts $fmon(11) "$now $pdrops_"
    $qmon(11) set pdrops_ 0
    $qmon(11) set barrivals_ 0
    set nexttime [expr $now + $steptime]
    $ns at $nexttime "monitore_3938"
}
proc monitore_3840 {} {
    global ns MEC qmon fmon k Blink integ steptime
    set now [$ns now]
    set i 2
    set j 4
    $qmon(12) instvar size_ pkts_ parrivals_ barrivals_ pdepartures_
bdepartures_ pdrops_ bdrops_
    set MEC($i,$j) $size_
    puts $fmon(12) "$now $pdrops_"
    $qmon(12) set pdrops_ 0
    $qmon(12) set barrivals_ 0
    set nexttime [expr $now + $steptime]
    $ns at $nexttime "monitore_3840"
}
proc monitore_4038 {} {
    global ns MEC qmon fmon k Blink integ steptime
    set now [$ns now]
    set i 4
    set j 2
    $qmon(13) instvar size_ pkts_ parrivals_ barrivals_ pdepartures_
bdepartures_ pdrops_ bdrops_
    set MEC($i,$j) $size_
    puts $fmon(13) "$now $pdrops_"
    $qmon(13) set pdrops_ 0
    $qmon(13) set barrivals_ 0
    set nexttime [expr $now + $steptime]
    $ns at $nexttime "monitore_4038"
}
proc monitore_3843 {} {
    global ns MEC qmon fmon k Blink integ steptime
    set now [$ns now]
    set i 2
    set j 7
    $qmon(14) instvar size_ pkts_ parrivals_ barrivals_ pdepartures_
bdepartures_ pdrops_ bdrops_
    set MEC($i,$j) $size_
    puts $fmon(14) "$now  $pdrops_"
    $qmon(14) set pdrops_ 0
    $qmon(14) set barrivals_ 0
    set nexttime [expr $now + $steptime]
    $ns at $nexttime "monitore_3843"
}
```

```
proc monitore_4338 {} {
    global ns MEC qmon fmon k Blink integ steptime
    set now [$ns now]
    set i 7
    set j 2
    $qmon(15) instvar size_ pkts_ parrivals_ barrivals_ pdepartures_
bdepartures_ pdrops_ bdrops_
    puts $fmon(15) "$now  $pdrops_"
    $qmon(15) set pdrops_ 0
    $qmon(15) set barrivals_ 0
    set nexttime [expr $now + $steptime]
    $ns at $nexttime "monitore_4338"
}
proc monitore_3844 {} {
    global ns MEC qmon fmon k Blink integ steptime
    set now [$ns now]
    set i 2
    set j 8
    $qmon(16) instvar size_ pkts_ parrivals_ barrivals_ pdepartures_
bdepartures_ pdrops_ bdrops_
    set MEC($i,$j) $size_
    puts $fmon(16) "$now $pdrops_"
    $qmon(16) set pdrops_ 0
    $qmon(16) set barrivals_ 0
    set nexttime [expr $now + $steptime]
    $ns at $nexttime "monitore_3844"
}

proc monitore_4438 {} {
    global ns MEC qmon fmon k Blink integ steptime
    set now [$ns now]
    set i 8
    set j 2
    $qmon(17) instvar size_ pkts_ parrivals_ barrivals_ pdepartures_
bdepartures_ pdrops_ bdrops_
    set MEC($i,$j) $size_
    puts $fmon(17) "$now $pdrops_"
    $qmon(17) set pdrops_ 0
    $qmon(17) set barrivals_ 0
    set nexttime [expr $now + $steptime]
    $ns at $nexttime "monitore_4438"
}
proc monitore_3940 {} {
    global ns MEC qmon fmon k Blink integ steptime
    set now [$ns now]
    set i 3
    set j 4
    $qmon(18) instvar size_ pkts_ parrivals_ barrivals_ pdepartures_
bdepartures_ pdrops_ bdrops_
    set MEC($i,$j) $size_
    puts $fmon(18) "$now $pdrops_"
    $qmon(18) set pdrops_ 0
    $qmon(18) set barrivals_ 0
    set nexttime [expr $now + $steptime]
    $ns at $nexttime "monitore_3940"
}
```

```
proc monitore_4039 {} {
    global ns MEC qmon fmon k Blink integ steptime
    set now [$ns now]
    set i 4
    set j 3
    $qmon(19) instvar size_ pkts_ parrivals_ barrivals_ pdepartures_
bdepartures_ pdrops_ bdrops_
    set MEC($i,$j) $size_
    puts $fmon(19) "$now $pdrops_"
    $qmon(19) set pdrops_ 0
    $qmon(19) set barrivals_ 0
    set nexttime [expr $now + $steptime]
    $ns at $nexttime "monitore_4039"
}
proc monitore_3941 {} {
    global ns MEC qmon fmon k Blink integ steptime
    set now [$ns now]
    set i 3
    set j 5
    $qmon(20) instvar size_ pkts_ parrivals_ barrivals_ pdepartures_
bdepartures_ pdrops_ bdrops_
    set MEC($i,$j) $size_
    puts $fmon(20) "$now $pdrops_"
    $qmon(20) set pdrops_ 0
    $qmon(20) set barrivals_ 0
    set nexttime [expr $now + $steptime]
    $ns at $nexttime "monitore_3941"
}
proc monitore_4139 {} {
    global ns MEC qmon fmon k Blink integ steptime
    set now [$ns now]
    set i 5
    set j 3
    $qmon(21) instvar size_ pkts_ parrivals_ barrivals_ pdepartures_
bdepartures_ pdrops_ bdrops_
    set MEC($i,$j) $size_

    puts $fmon(21) "$now $pdrops_"
    $qmon(21) set pdrops_ 0
    $qmon(21) set barrivals_ 0
    set nexttime [expr $now + $steptime]
    $ns at $nexttime "monitore_4139"
}
proc monitore_4041 {} {
    global ns MEC qmon fmon k Blink integ steptime
    set now [$ns now]
    set i 4
    set j 5
    $qmon(22) instvar size_ pkts_ parrivals_ barrivals_ pdepartures_
bdepartures_ pdrops_ bdrops_
    set MEC($i,$j) $size_
    puts $fmon(22) "$now  $pdrops_"
    $qmon(22) set pdrops_ 0
    $qmon(22) set barrivals_ 0
    set nexttime [expr $now + $steptime]
    $ns at $nexttime "monitore_4041"
}
```

```
proc monitore_4140 {} {
    global ns MEC qmon fmon k Blink integ steptime
    set now [$ns now]
    set i 5
    set j 4
    $qmon(23) instvar size_ pkts_ parrivals_ barrivals_ pdepartures_
bdepartures_ pdrops_ bdrops_
    set MEC($i,$j) $size_
    puts $fmon(23) "$now  $pdrops_"
    $qmon(23) set pdrops_ 0
    $qmon(23) set barrivals_ 0
    set nexttime [expr $now + $steptime]
    $ns at $nexttime "monitore_4140"
}
proc monitore_4042 {} {
    global ns MEC qmon fmon k Blink integ steptime
    set now [$ns now]
    set i 4
    set j 6
    $qmon(24) instvar size_ pkts_ parrivals_ barrivals_ pdepartures_
bdepartures_ pdrops_ bdrops_
    set MEC($i,$j) $size_
    puts $fmon(24) "$now  $pdrops_"
    $qmon(24) set pdrops_ 0
    $qmon(24) set barrivals_ 0
    set nexttime [expr $now + $steptime]
    $ns at $nexttime "monitore_4042"
}
proc monitore_4240 {} {
    global ns MEC qmon fmon k Blink integ steptime
    set now [$ns now]
    set i 6
    set j 4
    $qmon(25) instvar size_ pkts_ parrivals_ barrivals_ pdepartures_
bdepartures_ pdrops_ bdrops_
    set MEC($i,$j) $size_
    puts $fmon(25) "$now  $pdrops_"
    $qmon(25) set pdrops_ 0
    $qmon(25) set barrivals_ 0
    set nexttime [expr $now + $steptime]
    $ns at $nexttime "monitore_4240"
}
proc monitore_4043 {} {
    global ns MEC qmon fmon k Blink integ steptime
    set now [$ns now]
    set i 4
    set j 7
    $qmon(26) instvar size_ pkts_ parrivals_ barrivals_ pdepartures_
bdepartures_ pdrops_ bdrops_
    set MEC($i,$j) $size_
    puts $fmon(26) "$now  $pdrops_"
    $qmon(26) set pdrops_ 0
    $qmon(26) set barrivals_ 0
    set nexttime [expr $now + $steptime]
    $ns at $nexttime "monitore_4043"
}
```

```
proc monitore_4340 {} {
    global ns MEC qmon fmon k Blink integ steptime
    set now [$ns now]
    set i 7
    set j 4
    $qmon(27) instvar size_ pkts_ parrivals_ barrivals_ pdepartures_
bdepartures_ pdrops_ bdrops_
    set MEC($i,$j) $size_
    puts $fmon(27) "$now  $pdrops_"
    $qmon(27) set pdrops_ 0
    $qmon(27) set barrivals_ 0
    set nexttime [expr $now + $steptime]
    $ns at $nexttime "monitore_4340"
}
proc monitore_4142 {} {
    global ns MEC qmon fmon k Blink integ steptime
    set now [$ns now]
    set i 5
    set j 7
    $qmon(28) instvar size_ pkts_ parrivals_ barrivals_ pdepartures_
bdepartures_ pdrops_ bdrops_
    set MEC($i,$j) $size_
    puts $fmon(28) "$now  $pdrops_"
    $qmon(28) set pdrops_ 0
    $qmon(28) set barrivals_ 0
    set nexttime [expr $now + $steptime]
    $ns at $nexttime "monitore_4142"
}
proc monitore_4241 {} {
    global ns MEC qmon fmon k Blink integ steptime
    set now [$ns now]
    set i 6
    set j 5
    $qmon(29) instvar size_ pkts_ parrivals_ barrivals_ pdepartures_
bdepartures_ pdrops_ bdrops_
    set MEC($i,$j) $size_
    puts $fmon(29) "$now  $pdrops_"
    $qmon(29) set pdrops_ 0
    $qmon(29) set barrivals_ 0
    set nexttime [expr $now + $steptime]
    $ns at $nexttime "monitore_4241"
}

proc monitore_4243 {} {
    global ns MEC qmon fmon k Blink integ steptime
    set now [$ns now]
    set i 6
    set j 7
    $qmon(30) instvar size_ pkts_ parrivals_ barrivals_ pdepartures_
bdepartures_ pdrops_ bdrops_
    set MEC($i,$j) $size_
    puts $fmon(30) "$now  $pdrops_"
    $qmon(30) set pdrops_ 0
    $qmon(30) set barrivals_ 0
    set nexttime [expr $now + $steptime]
    $ns at $nexttime "monitore_4243"
}
```

```
proc monitore_4342 {} {
    global ns MEC qmon fmon k Blink integ steptime
    set now [$ns now]
    set i 7
    set j 6
    $qmon(31) instvar size_ pkts_ parrivals_ barrivals_ pdepartures_
bdepartures_ pdrops_ bdrops_
    set MEC($i,$j) $size_
    puts $fmon(31) "$now  $pdrops_"
    $qmon(31) set pdrops_ 0
    $qmon(31) set barrivals_ 0
    set nexttime [expr $now + $steptime]
    $ns at $nexttime "monitore_4342"
}
proc monitore_4344 {} {
    global ns MEC qmon fmon k Blink integ steptime
    set now [$ns now]
    set i 7
    set j 8
    $qmon(32) instvar size_ pkts_ parrivals_ barrivals_ pdepartures_
bdepartures_ pdrops_ bdrops_
    set MEC($i,$j) $size_
    puts $fmon(32) "$now  $pdrops_"
    $qmon(32) set pdrops_ 0
    $qmon(32) set barrivals_ 0
    set nexttime [expr $now + $steptime]
    $ns at $nexttime "monitore_4344"
}
proc monitore_4443 {} {
    global ns MEC qmon fmon k Blink integ steptime
    set now [$ns now]
    set i 8
    set j 7
    $qmon(33) instvar size_ pkts_ parrivals_ barrivals_ pdepartures_
bdepartures_ pdrops_ bdrops_
    set MEC($i,$j) $size_
    puts $fmon(33) "$now  $pdrops_"
    $qmon(33) set pdrops_ 0
    $qmon(33) set barrivals_ 0
    set nexttime [expr $now + $steptime]
    $ns at $nexttime "monitore_4443"
}
```