PAULO ROBERTO DA CUNHA ESTANTE

PROPOSTA DE MÉTODO DE ENGENHARIA DE TRÁFEGO COM PROTEÇÃO DE CAMINHOS PARA REDES MPLS

Dissertação apresentada ao Programa de Pós-Graduação em Informática Aplicada da Pontifícia Universidade Católica do Paraná como requisito parcial para obtenção do título de Mestre em Informática Aplicada.

CURITIBA

2008

PAULO ROBERTO DA CUNHA ESTANTE

PROPOSTA DE MÉTODO DE ENGENHARIA DE TRÁFEGO COM PROTEÇÃO DE CAMINHOS PARA REDES MPLS

Dissertação apresentada ao Programa de Pós-Graduação em Informática Aplicada da Pontifícia Universidade Católica do Paraná como requisito parcial para obtenção do título de Mestre em Informática Aplicada.

Área de Concentração: *Engenharia de Tráfego*
Orientador: Prof. Dr. Edgard Jamhour

CURITIBA
2008

Estante, Paulo Roberto da Cunha

Proposta de Método de Engenharia de Tráfego com Proteção de Caminhos para Redes MPLS. Curitiba, 2008. 81p.

Dissertação – Pontifícia Universidade Católica do Paraná. Programa de Pós-Graduação em Informática Aplicada.

1. MPLS 2. Engenharia de Tráfego 3. Proteção de Caminho 4. Algoritmos Genéticos. I.Pontifícia Universidade Católica do Paraná. Centro de Ciências Exatas e de Tecnologia. Programa de Pós-Graduação em Informática Aplicada

Esta página deve ser reservada à ata de defesa e termo de aprovação que serão fornecidos pela secretaria após a defesa da dissertação e efetuadas as correções solicitadas.

Dedico a todos sonhadores que batalham pela realização de seus objetivos.

Agradecimentos

Agradeço em especial ao Prof. Edgard Jamhour que durante todo o mestrado sempre esteve à disposição, tendo excepcional dedicação e paciência para me auxiliar em todas as fases.

A minha família que sempre me ajudou e apoiou em todos os meus projetos de vida.

A minha esposa pelo apoio, paciência e dedicação em todas as horas.

Aos meus colegas de trabalho, e a empresa Siemens por me viabilizar e dar condições de cursar o programa de Mestrado.

E a todos que de uma forma ou de outra contribuíram para que este trabalho fosse realizado.

Sumário

Lista de Figuras

Lista de Tabelas

Lista de Símbolos

Σ	somatório
A	conjunto de arcos do grafo
a_j	arco
G	grafo (estrutura da rede)
lsp_k	solução candidata
M	número máximo de vértices
N	número máximo de arcos
P_k	caminho principal (seqüência de arcos)
R_k	caminho de recuperação (seqüência de arcos)
T	vetor de tráfego
t_k	demanda de tráfego
V	conjunto de vértices do grafo

Lista de Abreviaturas

ATM	Asynchronous Transfer Mode
BE	Best-Effort
CoS	Class of Service
CSPF	Constrained Shortest Path First
DSCP	Differentiated Services Codepoint
ERO	Explicit Route Object
FEC	Forwarding Equivalence Class
FF	Fixed Filter
IETF	Internet Engineering Task Force
IGP	Interior Gateway Protocol
IP	Internet Protocol
ISP	Internet Service Provider
IS-IS	Intermediate System to Intermediate System
LDP	Label Distribution Protocol
LSP	Label Switched Path
MIP	Mixed-Integer Programming
MPLS	Multi-Protocol Label Switching
MTU	Maximum Transmission Unit
NGN	Next Generation Networks
OSPF	Open Shortest Path First
PHP	Penultimante Hop Popping
PoP	Point of Presence
PPVPN	Provider Provisioned VPN
QoS	Quality of Service
RED	Random Early Detection
RFC	Request for Comments

RSVP	Resource Reservation Protocol
SAL	Simulated Allocation
SAN	Simulated Annealing
SE	Shared Explicit
SLA	Service Level Agreement
SPF	Shortest Path First
TE	Traffic Engineering
TED	Traffic Engineering Database
TLV	Type Length Value
TOS	Type of Service
TTL	Time to Live
UHP	Ultimate Hop Popping
WF	Wildcard Filter

Resumo

Este trabalho apresenta um método de otimização *off-line* para determinação de LSPs (*Label Switched Paths*) próxima ao ótimo para redes MPLS com suporte a proteção de caminho. O objetivo deste método é determinar caminhos principais e de recuperação para múltiplas demandas concorrentes sujeitas a restrições de banda, atraso e proteção de caminho. A proteção de caminho é provida para o caso de falhas simples de enlace ou de nó. Esta proteção visa acomodar LSPs afetados por estes eventos de falha sem causar sobre-provisionamento (e congestionamento) de enlaces. Usando um algoritmo de k-caminhos mais curtos modificado, nós modelamos o problema de planejamento de LSPs como um problema de busca, o qual é resolvido utilizando um algoritmo genético.

Palavras-Chave: MPLS, Engenharia de tráfego, Proteção de caminho.

Abstract

This work presents an off-line optimization approach for determining near-optimal LSPs (label Switched Paths) for MPLS-based networks with support to path-protection. The objective of the method proposed in this paper is to determine working and recovery paths for multiples demands subjected to capacity, delay and path protection constraints. The path protection is provided for the demands to be still accommodated in the case of a single link or node failure without oversubscribing any link. By using a modified k-shortest path algorithm, we model the LPSs planning problem as a search problem, which is solved using a genetic algorithm approach.

Keywords: MPLS, Traffic-Engineering, Path-Protection.

Capítulo 1

Introdução

Para empresas de telecomunicações, instalar e manter diversas redes distintas para prover todos os seus serviços se torna algo mais oneroso que manter apenas uma rede. Com a atual convergência de serviços para a rede IP, um problema enfrentado por estas empresas é a otimização dos seus recursos para atender todos os serviços usando uma mesma planta. Esses serviços geralmente possuem requerimentos de recursos e SLAs (*Service Level Agreements*) diferentes entre si, o que dificulta um compartilhamento adequado da rede.

Como o IP é uma rede comutada por pacotes sem conexão, os pacotes são tratados por ordem de chegada e não há controle que reserve ou garanta recursos de rede por fluxo. Para resolver esta questão da qualidade de serviço (QoS), existem dois modelos propostos pelo *Internet Engineering Task Force* (IETF), o modelo DiffServ [RFC2475] e o IntServ [RFC1633]. Enquanto o DiffServ tem como objetivo a agregação de fluxos distintos de mesmo tipo para tratamento igual (priorização por classes), o IntServ provê reserva de recursos fim-a-fim por fluxo. Para a sinalização e alocação dos recursos no modelo IntServ se utiliza o protocolo RSVP (*Resource Reservation Protocol*) [RFC2205].

Utilizando-se o RSVP-TE [RFC3209] (extensões para engenharia de tráfego do RSVP) juntamente com o protocolo MPLS (*Multiprotocol Label Switching*) [RFC3031] pode-se criar circuitos virtuais (*Label Switched Path*s ou “LSPs”), em cima da rede IP (que não necessariamente seguem o caminho mais curto do protocolo de roteamento), visando controlar a utilização dos recursos. Com o MPLS também é possível utilizar mecanismos de proteção para o tráfego em situações de problemas na rede.

1.1 Motivação

Mesmo com o uso combinado do MPLS e do RSVP-TE para engenharia de tráfego, existe um problema em aberto que é a determinação da capacidade, para prever (testar logicamente) se a rede suporta todos os LSPs desejados simultaneamente. Para o provisionamento de LSPs através do RSVP-TE, existem dois métodos: dinâmico ou explícito. Para o método dinâmico, o RSVP-TE utiliza as extensões de engenharia de tráfego do IGP (*Interior Gateway Protocol*) para o cálculo do caminho; enquanto no método explícito o administrador de rede entra manualmente com o caminho a ser utilizado. As extensões de engenharia de tráfego dos IGPs permitem que sejam calculados e utilizados outros múltiplos caminhos além do melhor caminho.

Normalmente o que ocorre é que após o provisionamento de um LSP no roteador de entrada (utilizando o método dinâmico), o administrador de rede verifica se foi possível ou não alocar recursos para o LSP. Este problema está diretamente relacionado com a escolha dos caminhos que os LSPs deverão utilizar. Como cada novo caminho é dado pelos recursos disponíveis no momento do cálculo (imediatamente antes da sinalização), podem ocorrer situações onde a ordem do provisionamento impede que LSPs com maior necessidade de banda não sejam atendidos. A figura 1.1 mostra um exemplo desta situação: uma rede que deve transportar 4 LSPs com requerimentos de banda distintos entre os roteadores 1 e 4 utilizando enlaces GigabitEthernet. Imagine que primeiramente foram apenas provisionados dinamicamente os LSPs de 300, 400 e de 500Mbps. Na hora de provisionar o LSP de 600Mbps, a rede não possui um caminho com 600Mbps, embora disponha de 800Mbps livres somando os dois caminhos.

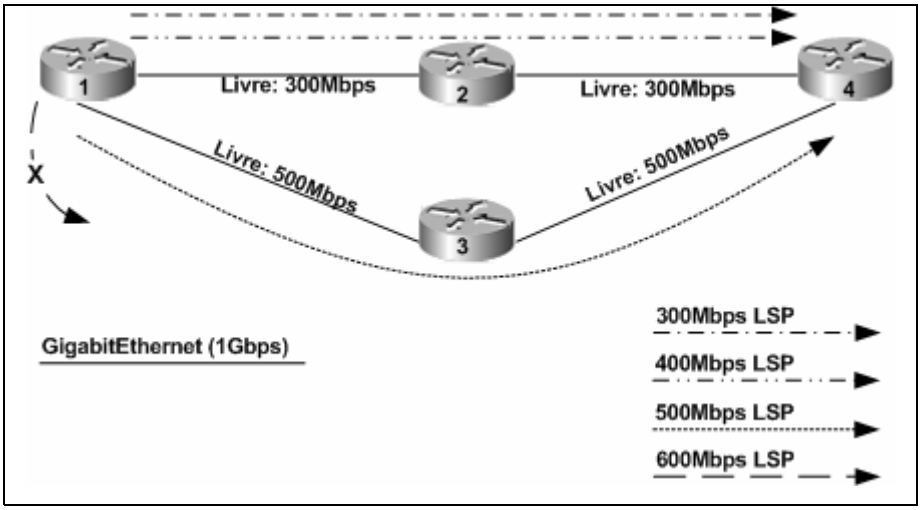

FIGURA 1.1: EXEMPLO DE PROBLEMA DE ALOCAÇÃO DE RECURSOS

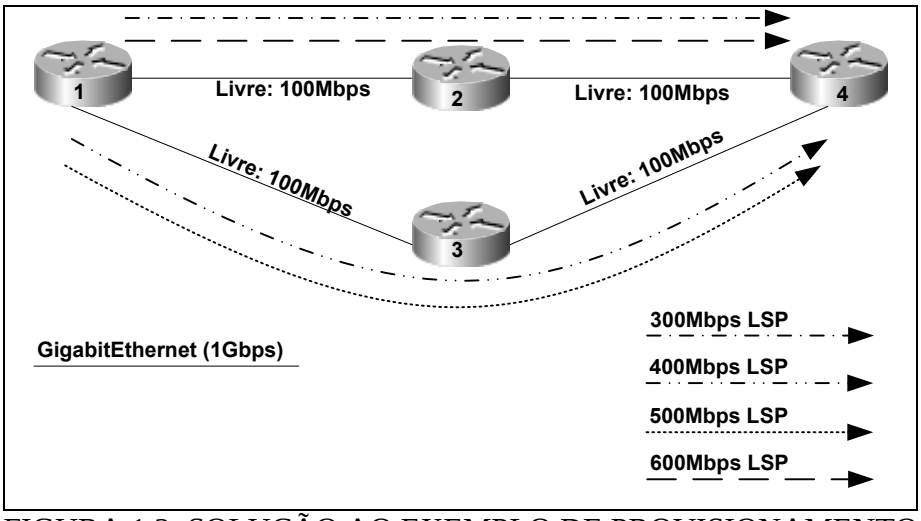

FIGURA 1.2: SOLUÇÃO AO EXEMPLO DE PROVISIONAMENTO

Uma possível solução para a alocação dos LSPs do exemplo da figura 1.1 está mostrado na figura 1.2. Tendo previamente toda a demanda e a estrutura da rede, foi possível calcular caminhos atendendo todos os LSPs. Contudo, deve-se provisionar estes túneis de forma explícita para contornar o problema da alocação dinâmica da figura 1.1.

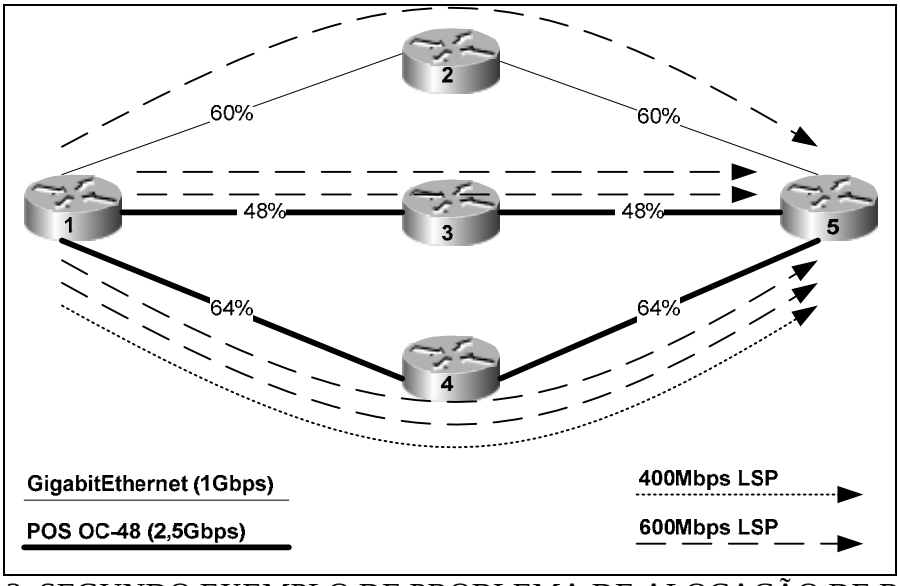

FIGURA 1.3: SEGUNDO EXEMPLO DE PROBLEMA DE ALOCAÇÃO DE RECURSOS

Outro importante problema é a questão dos mecanismos de proteção para contorno de falhas, para que o tráfego seja afetado da menor forma possível durante um problema de rede.

Para contornar falhas, a rede deve ter recursos disponíveis ociosos para transportar o tráfego afetado quando um enlace ou nó da rede estiver indisponível. Ao ocorrer uma falha, os LSPs afetados tem que ser sinalizados novamente utilizando caminhos alternativos, e também podem sofrer o mesmo problema de escassez de recursos dado a ordem de provisionamento (utilizando o método dinâmico). Esta situação é exemplificada na figura 1.3, onde uma segunda rede tem que transportar 5 LSPs de 600M e 1 de 400M entre o roteador 1 e o roteador 5. A alocação dos caminhos demonstrada na figura 1.3 representa a melhor distribuição de tráfego possível para este cenário de teste (visando equalizar a distribuição da carga e de recursos livres).

Imagine que após a alocação dos LSPs ocorra uma falha no roteador 3 que o impeça de encaminhar pacotes (e.g. *crash* de sistema, falta de energia elétrica, etc). Todos os LSPs que o utilizam como trânsito devem buscar outro caminho para o transporte dos pacotes. Neste segundo exemplo dois LSPs de 600M são afetados e devem ser re-sinalizados por caminhos alternativos. Dada a disposição da figura 1.3, apenas é possível alocar um destes LSPs, via o roteador 4. Note que ainda existe 700M disponível para transporte entre os roteadores 1 e 5, mas como não há 600M em um único caminho, não há como transportar o segundo LSP de 600M afetado pela falha. A figura 1.4 ilustra esta situação.

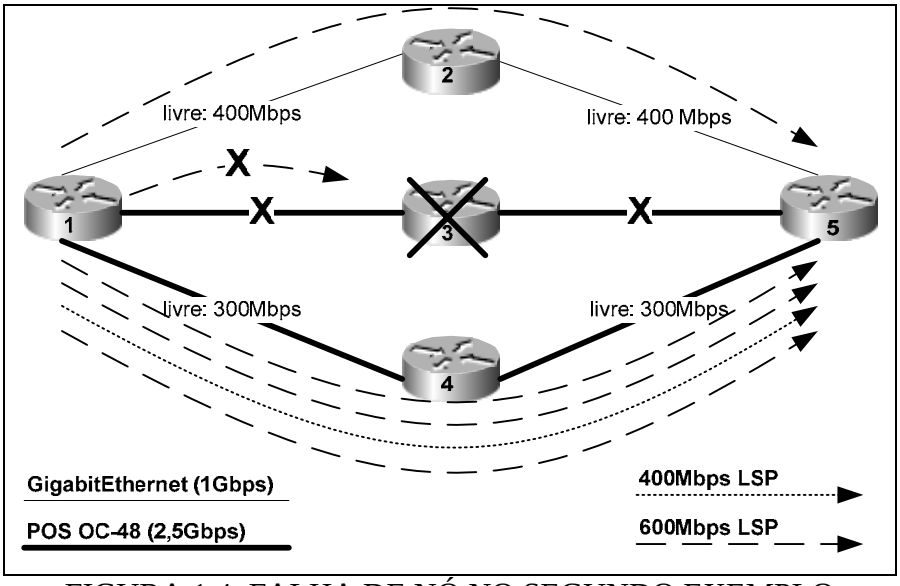

FIGURA 1.4: FALHA DE NÓ NO SEGUNDO EXEMPLO

Neste segundo exemplo percebemos que apenas distribuir tráfego o mais igualmente o possível entre os enlaces pode não ser suficiente para contornar situações de falhas de nó ou

de enlace (no exemplo foi representada a queda do nó 3, mas a queda de qualquer um dos dois enlaces do roteador 3 resulta no mesmo problema de alocação).

O RSVP-TE oferece recurso de preempção para tentar realocar recursos em situações de falta de banda livre para alocação de um dado LSP. Para empregar este mecanismo, deve-se utilizar as extensões de engenharia de tráfego do IGP e atribuir prioridades a cada um dos LSPs. Utilizando preempção, um LSP com maior prioridade pode “derrubar” um LSP de menor prioridade visando “roubar” os recursos.

Para o exemplo da figura 1.4, o LSP de 400Mbps deve ter prioridade menor do que o LSP de 600Mbps que não foi alocado após a falha. Nesta situação, o LSP de 600Mbps derrubaria o LSP de 400Mbps e seria alocado no caminho 1-4-5, fazendo com que o LSP de 400Mbps procurasse um caminho alternativo (via roteadores 1-2-5).

Contudo, este mecanismo não é recomendado por causar disrupção no tráfego. Outro problema deste mecanismo é a dificuldade de se atribuir prioridades adequadamente aos LSPs. Para que haja preempção na figura 1.4, o LSP de 400Mbps deve ter prioridade inferior ao de 600Mbps que não foi alocado. O mecanismo de preempção tem comportamento imprevisível, variável de acordo com as prioridades e a ordem de provisionamento e/ou preempção.

Uma outra estratégia para o problema da figura 1.4 seria não balancear a carga e deixar recursos disponíveis em alguns enlaces mais que em outros para re-rotear o tráfego durante uma falha. A figura 1.5 mostra uma possível solução para o problema utilizando esta estratégia, onde a queda simples de qualquer nó ou de enlace da topologia permite a criação de caminhos alternativos. Note que a distribuição de tráfego entre as interfaces não está balanceada.

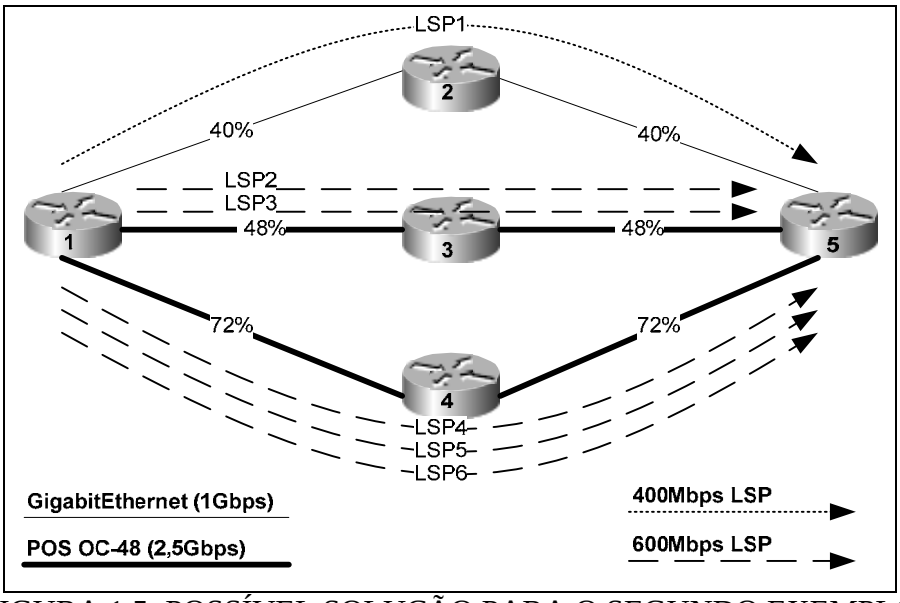

FIGURA 1.5: POSSÍVEL SOLUÇÃO PARA O SEGUNDO EXEMPLO

Esta estratégia não sofre os problemas de imprevisibilidade e de disrupção de tráfego causados pelo mecanismo de preempção.

1.2 Proposta

Este trabalho propõe um método *centralizado* para engenharia de tráfego em redes MPLS, com suporte a proteção de caminho. Com esta proposta pode-se validar se uma determinada topologia suporta todos os fluxos previstos para serem transportados por ela, com os respectivos caminhos principais (também conhecidos como primários, ou operacionais) e de proteção (também conhecidos como secundários, ou de recuperação). O método busca caminhos de proteção de forma que (sempre que possível) qualquer queda simples de enlace ou de nó afete minimamente a rede, no que se refere a congestionamento nos enlaces.

1.3 Organização

Esta dissertação de mestrado é composta por 7 capítulos, seguidos da conclusão:

Capítulo 1 – Introdução: Apresenta a motivação e o objetivo do trabalho.

Capítulo 2 – MPLS: Apresenta a arquitetura do protocolo MPLS e seus protocolos de sinalização.

Capítulo 3 – RSVP-TE: Descreve o funcionamento do protocolo RSVP e suas extensões para engenharia de tráfego.

Capítulo 4 – Métodos de Otimização: Apresenta métodos de otimização que se aplicam ao problema de alocação de recursos para fluxos simultâneos.

Capítulo 5 – Estado da Arte: Apresenta propostas para a resolução do problema.

Capítulo 6 – Proposta: Apresenta a proposta em detalhes, da formulação matemática ao método de procura.

Capítulo 7 – Resultados Obtidos: Apresenta resultados obtidos com o método proposto em cenários de teste.

Conclusão: Conclui o trabalho realizado, e aponta trabalhos futuros.

Capítulo 2

MPLS

2.1 Introdução

O MPLS [RFC3031] foi inicialmente proposto como um protocolo para tornar o mecanismo de busca por rotas mais rápido. Posteriormente, o MPLS passou a ser visto como um importante instrumento para engenharia de tráfego da rede.

Este capítulo começa com a exposição do objetivo original do MPLS, seguido pela descrição dos principais aspectos da sua arquitetura, utilizações, e dos seus protocolos de sinalização.

2.2 Motivação

Em redes sem conexão como as redes IP, a cada salto do pacote o roteador em questão tem que realizar individualmente uma nova busca pela melhor rota (a mais específica) para o destino. Esta busca pela melhor rota é feita através de uma tabela, e pode demorar tempo proporcional ao número de entradas (rotas) nesta tabela.

Dado este fato, temos que o atraso no encaminhamento de um pacote dentro uma rede está relacionado ao número de saltos e ao número de rotas presentes nas tabelas dos roteadores, além dos tempos de fila e de propagação dos pacotes nos enlaces.

A proposta do MPLS foi a de criar classes equivalentes de encaminhamento (*Forwarding Equivalence Cla*ss, ou “FECs”), onde um roteador de borda ao receber um determinado pacote faz uma busca em sua tabela de rotas e o associa a uma FEC. Esta busca é realizada apenas neste primeiro roteador, o qual classifica o pacote com uma etiqueta (entre as camadas de enlace e de rede) antes de enviar o pacote para a nuvem de roteadores de trânsito.

Esta etiqueta inserida representa um roteador destino do pacote (*egress*), o qual será o nó final da FEC e será responsável pela remoção da etiqueta e entrega final do pacote ao destino.

O resultado final é a substituição das rotas por FECs nos roteadores trânsito, criando-se um caminho único e unidirecional (também chamado de túnel) entre roteadores de borda. Estes túneis são denominados LSPs (*Label Switched Paths*). Uma única FEC pode representar diversas rotas cujo destino de trânsito seja um mesmo roteador remoto da rede. Por agregar diversas rotas para um mesmo destino em uma única FEC, a tabela de FECs tende a ser consideravelmente menor que a tabela de rotas, e a decisão por caminhos em roteadores trânsito torna-se um processo mais rápido.

Outra característica do MPLS é o fato da etiqueta inserida isolar a camada 3 do *backbone* do provedor, o qual permite que outros tipos de protocolos (e.g. IPX, *Appletalk*, SNA, etc) sejam transportadas em cima de uma rede IP, até mesmo o encapsulamento de quadros camada 2 (e.g. Frame-Relay, ATM, etc).

Em resumo, o MPLS procura tornar as redes IP parecidas com as redes ATM no que se refere à comutação: a rede de comutação de pacotes torna-se uma rede comutada de circuitos virtuais. Cada circuito MPLS agrega diversos prefixos, e o roteamento se torna mais eficiente.

2.3 Cabeçalho MPLS

A etiqueta utilizada para a comutação dos pacotes MPLS (denominada "*shim header*") é de tamanho 32 bits e tem a sua estrutura demonstrada na figura 2.1.

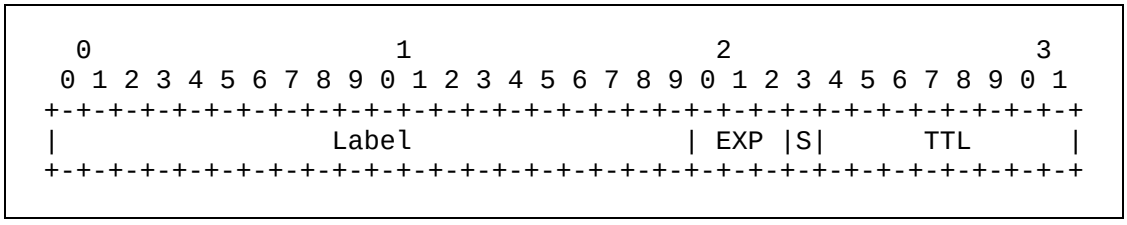

FIGURA 2.1: MPLS SHIM HEADER

A função dos campos são:

- Label (20 bits): etiqueta, utilizado como identificador da FEC, e tem significado local;
- EXP (3 bits): bits experimentais, tipicamente usados para classe de serviço (CoS);

- S (1 bit): "*Bottom of Stack*", indica se a etiqueta é a última da pilha;
- TTL (8 bits): "*Time to Live*" campo similar ao do cabeçalho IP, podendo ser o mesmo TTL do datagrama IP original, ou iniciado com 255 na entrada do túnel (quando o LSP trabalha em modos de não propagação ou de não decremento de TTL no destino para o cabeçalho de camada 3).

Este cabeçalho é inserido entre os cabeçalhos de camada de enlace e da camada de rede, conforme demonstrado na figura 2.2. Dado a este fato, o MPLS é comumente citado como sendo uma tecnologia de camada 2,5.

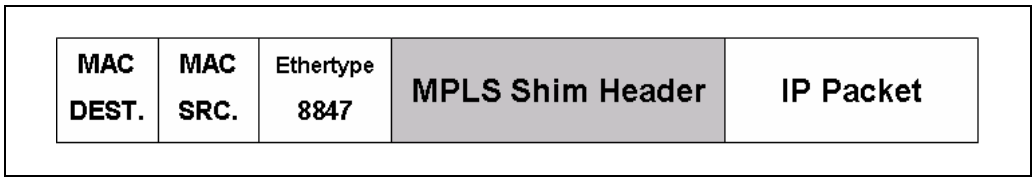

FIGURA 2.2: CABEÇALHO MPLS EM UM QUADRO ETHERNET

2.3.1 Operações

Existem três operações que são realizadas com as etiquetas MPLS:

- PUSH: inserção de uma ou mais etiquetas entre os cabeçalhos de camada 2 e 3.
- SWAP: troca de uma etiqueta por outra.
- POP: remoção de uma ou mais etiquetas do pacote.

Estas operações serão detalhadas no item 2.6.

2.3.2 Valores da Etiqueta

Pode-se ter um total de 1.048.575 valores diferentes para o *label*, onde os valores de 0 a 15 são reservados pelo IETF. Os *labels* definidos são:

- Label 0 ("*IPv4 Explicit Null*"): o roteador que recebe este pacote deve realizar a operação POP, em seguida realizando a procura pela rota IPv4 para encaminhar o pacote.
- Label 1 ("*Router Alert*"): função análoga à do IPv4 [RFC2113], onde o roteador trânsito em questão deverá fazer inspeção no pacote com este valor no *label*.

- Label 2 (“*IPv6 Explicit Null*”): o roteador que recebe este pacote deve realizar a operação POP, em seguida realizando a procura pela rota IPv6 para encaminhar o pacote.
- Label 3 (“*Implicit Null*”): este label é apenas sinalizado, e nunca deve aparecer em um cabeçalho MPLS. Serve para sinalizar túnel PHP (*Penultimate Hop Popping*, descrito no item 2.4.2), onde o penúltimo roteador deve realizar a operação POP.

2.4 Comutação

O primeiro roteador do túnel realiza uma operação PUSH com a etiqueta MPLS no pacote IP (propagando a informação de TTL do pacote IP para o cabeçalho MPLS), e envia o pacote para a respectiva interface de saída sinalizada. A partir do segundo roteador, haverá uma troca de etiquetas (operação SWAP) até o roteador final do túnel (modo UHP, *Ultimate Hop Popping*) ou até o penúltimo (modo PHP), o qual realizará a operação POP no pacote.

A cada salto também é decrementado o valor do campo TTL (time to live), de forma análoga ao IP, utilizado para prevenção de *loop* infinito dos pacotes. O roteador que realiza a operação POP no pacote também retorna (quando túnel não foi sinalizado para trabalhar em modo “no-propagate-ttl” ou “no-decrement-ttl”) o TTL do pacote MPLS para o pacote IP.

2.4.1 Ultimate Hop Popping

No tipo *Ultimate Hop Popping* (UHP), as etiquetas MPLS são comutadas até o último roteador do túnel, sendo esta última etiqueta sinalizada com o valor zero para IPv4, como ilustrado na figura 2.3.

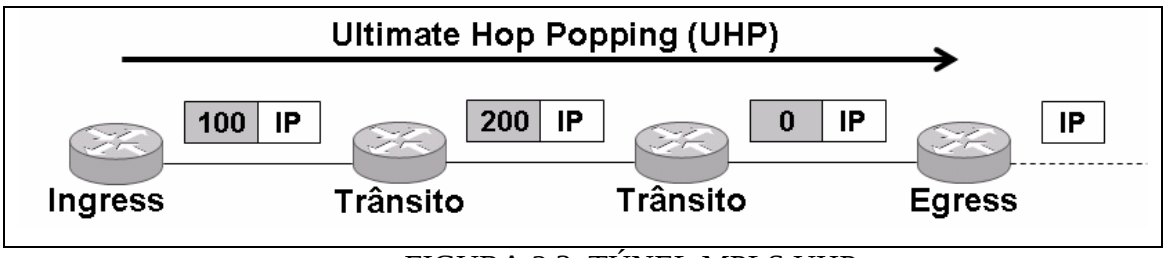

FIGURA 2.3: TÚNEL MPLS UHP

2.4.2 Penultimate Hop Popping

Já em um túnel *Penultimate Hop Popping* (PHP), quem faz a remoção da etiqueta é o penúltimo roteador do túnel. Para realizar esta operação, o último roteador do túnel deve

sinalizar o valor de etiqueta 3 (embora este valor nunca seja de fato utilizado na etiqueta). Este tipo de túnel é útil para roteadores egressos que não conseguem fazer duas consultas (a primeira de etiqueta MPLS e a segunda do cabeçalho IP) mantendo velocidade de linha ("*wire-speed*"). Este tipo de túnel está ilustrado na figura 2.4.

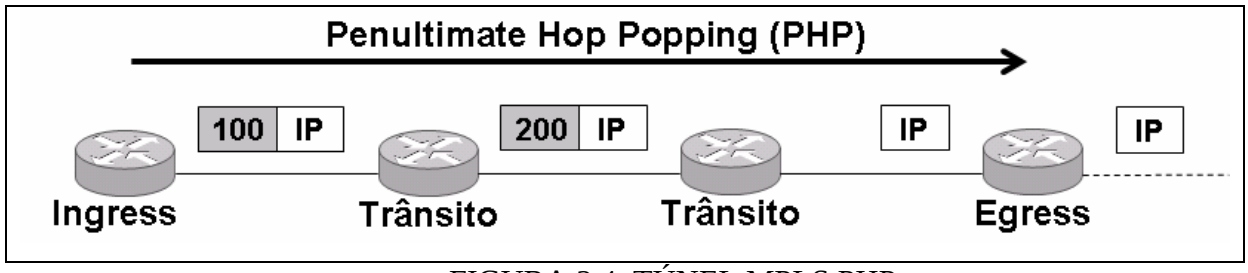

FIGURA 2.4: TÚNEL MPLS PHP

2.5 Protocolos de Sinalização

Para se ter as informações de etiquetas para cada FEC, pode-se definir manualmente nos roteadores (definindo valor da etiqueta e interface de entrada, e respectiva etiqueta e interface de saída) ou se utilizar de protocolos de sinalização que automatizam o processo. Os protocolos de sinalização provêm meios para que dois roteadores troquem anúncios de etiquetas para as FECs entre si e possibilitem o encaminhamento dos pacotes de um ponto ao outro na rede. Os dois protocolos utilizados para esta tarefa são o *Label Distribution Protocol* (LDP) e o *Resource Reservation Protocol* (RSVP-TE).

No protocolo LDP (*Label Distribution Protocol*) [RFC3036], cada roteador anuncia seus prefixos com suas respectivas etiquetas para cada um dos seus vizinhos. Cada roteador vizinho que recebe esta mensagem a processa e novamente a re-anuncia para o resto do domínio LDP, podendo nestes novos anúncios atribuir novos valores às etiquetas recebidas (as etiquetas têm significância local). O resultado final deste processo é todo o domínio LDP tendo etiquetas para todos os destinos. Cada roteador apenas instala a FEC cuja interface de saída é a mesma que a eleita pelos protocolos de IGP (i.e. OSPF, IS-IS), como medida para prevenção de *loops*.

O protocolo RSVP-TE [RFC3209] ao contrário do LDP, apenas realiza a formação de vizinhança e a sinalização dos túneis quando necessário (i.e. novo túnel configurado em um roteador). O processo de sinalização se dá através de mensagens que primeiramente

estabelecem o caminho (na ida, com mensagens PATH), e somente depois reservam recursos e sinalizam as respectivas etiquetas salto a salto (na volta, com mensagens RESV).

Além do RSVP-TE, existe também o protocolo CR-LDP (*Constrained Based Label Distribution Protocol*) [RFC3212] que permite engenharia de tráfego (ambos os protocolos foram propostos pelo IETF). Dado o desinteresse dos fabricantes no protocolo CR-LDP, o IETF decidiu focar no RSVP-TE [BRA03] como protocolo de sinalização para LSPs.

2.6 Conclusão

Desde o advento do MPLS, mesmo com o constante aumento no número de prefixos da Internet, houve grande avanço na arquitetura de processamento dos roteadores que reduziram drasticamente o tempo de procura por rotas.

Embora atualmente o MPLS não seja utilizado tanto pelo seu atrativo inicial da rapidez de comutação, ele está sendo altamente empregado nos *backbones* das empresas de telecomunicações para prover engenharia de tráfego (junto ao protocolo RSVP-TE).

Este trabalho será desenvolvido baseado no conceito dos circuitos virtuais estabelecidos via MPLS (em conjunto com o protocolo RSVP-TE, apresentado no capítulo 3) provendo engenharia de tráfego para *backbones* IP.

Capítulo 3

RSVP-TE

3.1 Introdução

O RSVP (*Resource Reservation Protocol*) [RFC2205] foi inicialmente concebido para ser um protocolo de reserva de recurso para *hosts* (usuários finais). Dado o tamanho e a imensa quantidade de computadores concetados a Internet, ele não foi amplamente implementado da maneira inicialmente planejada (reservas individuais por fluxo).

O RSVP-TE [RFC3209] é composto pelo protocolo RSVP com extensões para engenharia de tráfego. As extensões providas no RSVP-TE permitem controle sobre os recursos da rede para a formação dos túneis MPLS.

Este capítulo aborda a apresentação geral da arquitetura de sinalização e de engenharia de tráfego do RSVP-TE, e os mecanismos de proteção para túneis MPLS.

3.2 Arquitetura

O RSVP reserva recursos para fluxos, onde cada fluxo consiste em uma sessão identificável entre dois pontos (fonte e origem).

3.2.1 Estilo de Reservas

Para a reserva existem 3 estilos, que variam dependendo da seleção da fonte. O estilo de uma reserva caracteriza a contagem de recursos a serem reservados, podendo ser:

- FF (*Fixed Filter*): Uma reserva por fonte.
- SE (*Shared Explicit*): Fontes explicitamente identificadas que compartilham a mesma sessão.

- WF (*Wildcard Filter*): Uma reserva para todas as fontes da mesma sessão, tipicamente usado em ambientes *Multicast*.

Nesta proposta o estilo de reserva utilizado será o *Fixed Filter* (FF), dado que não haverá compartilhamento de recursos entre as reservas.

3.2.2 Sinalização

A sinalização do RSVP se dá através de um conjunto de mensagens e objetos padronizados. As principais mensagens utilizadas pelo protocolo são:

- **PATH** – Usada para sinalizar e manter reservas.
- **RESV** – Usada para reservar recursos, em resposta a uma mensagem PATH.
- **PATHERR** – Enviada por um receptor de uma mensagem PATH que detecta um erro. Esta mensagem não destrói a reserva.
- **RESVERR** – Enviada por um receptor de uma mensagem RESV que detecta um erro. Esta mensagem não destrói a reserva.
- **PATHTEAR** – Usada para sinalizar a remoção de um caminho. Esta mensagem não destrói a reserva.
- **RESVTEAR** – sinalização para remoção da reserva. Esta mensagem destrói a reserva.
- **RESVCONF** – Enviada por um receptor de uma mensagem RSV, confirmando a reserva.
- **ACK** – Confirmação de mensagens (quando solicitado) [RFC2961].
- **HELLO** – Usado para detecção de falha de nó [RFC3209].

Cada mensagem é composta por um ou mais objetos que descrevem as características da sinalização em questão. Os principais objetos utilizados pelo protocolo são:

- **ADSPEC** – Define características e tipo do serviço (carga controlada, serviço garantido).
- **FLOWSPEC** – Define os aspectos de QoS da reserva (tipo de serviço, token-bucket, etc.).
- **TSPEC** – Características do tráfego (token-bucket, MTU, entre outros).

- **ERROR** – Informa a descrição do erro ocorrido.
- **EXPLICIT ROUTE –** Permite a especificação do caminho.
- **FILTER** – Define a fonte da sessão.
- **STYLE** – Especifica o estilo da reserva.

Destes objetos o de principal relevância para a proposta é o "*Explicit Route*", o qual permite a criação de uma reserva especificando o caminho fim a fim, seguidos do FLOWSPEC e do TSPEC, que especificam os requerimentos da reserva.

3.2.3 Estabelecimento de Caminho

Uma sessão RSVP é estabelecida através de mensagens PATH que são enviadas a partir do nó de ingresso, salto a salto, até o nó de egresso. A rota tomada pode ser baseada na visão do IGP, ou feita a partir das extensões para engenharia de tráfego (apresentada no item 3.3). Uma terceira possibilidade é ser especificado o caminho a ser tomado, utilizando-se o objeto "*Explicit Route*".

A mensagem PATH tipicamente contém os objetos: *Session; Hop; Time-Values; SessionAttribute; Sender Template; Sender-Tspec; Adspec; Explicit Route; Label Request; Record Route*.

Uma vez que as mensagens PATH chegam até o roteador de egresso, as reservas começam a serem realizadas salto a salto no sentido inverso, através das mensagens RESV.

Tipicamente uma mensagem RESV contêm os objetos: *Session; Hop; Time-Values; Style; Flowspec; Filter-Spec; Label; Record Route*.

A figura 3.1 demonstra o processo de formação de uma reserva passo a passo (índices das mensagens representam a ordem de ocorrência das mesmas). No exemplo apenas é demonstrado a troca das mensagens sem especificação dos objetos.

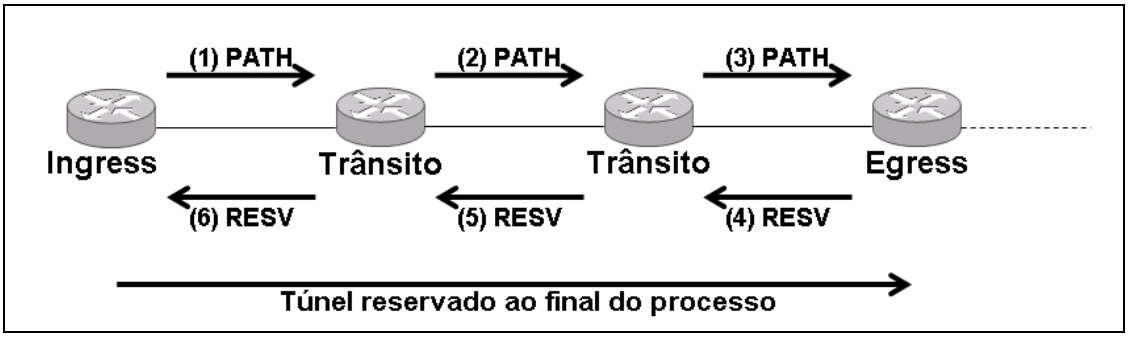

FIGURA 3.1: FORMAÇÃO DE UMA RESERVA RSVP

Em reposta a uma mensagem RESV, o roteador pode confirmar esta com uma mensagem RESVCONF, quando solicitado (parâmetro opcional).

3.2.4 Manutenção de Caminho

Os recursos dos túneis são mantidos reservados pelo re-envio periódico das mensagens PATH e RESV. A periodicidade destes re-envios é definida através do objeto "*Time-Values*". Uma vez que qualquer nó do caminho detecte que não recebeu a mensagem PATH dentro do valor sinalizado no objeto, ele inicia o processo de remoção da reserva.

3.2.5 Finalização de Caminho

Quando ocorre um erro, ou o nó de ingresso decide remover a reserva da sessão, é utilizada a mensagem de PATHTEAR, e ocorre processo análogo ao de estabelecimento do túnel. As mensagens PATHTEAR são enviadas salto a salto até o nó de egresso da reserva, uma vez que este último roteador recebe a mensagem, ele envia salto a salto uma mensagem RESVTEAR em direção ao nó ingresso da sessão. Apenas a mensagem RESVTEAR remove a reserva.

3.3 Engenharia de Tráfego

Por engenharia de tráfego entende-se a manipulação do caminho que determinados fluxos de pacotes devem tomar baseados em certas características além do caminho mais curto.

Para realizar engenharia de tráfego o RSVP-TE permite que caminhos sejam alocados dinamicamente (através de extensões apresentadas nos próximos itens) ou através de rotas explícitas (computadas de forma centralizada).

No caso da alocação dinâmica de caminhos, a tomada de decisão leva em conta restrições de enlaces, especificando tipos de enlaces permitidos ou proibidos de serem utilizados para uma dada demanda. Esta informação das características de enlace é descrita como grupos-administrativos ou "cores", e permite ao administrador configurar determinado enlace com um dado grupo-administrativo. Um exemplo deste caso é marcar todos os enlaces de menor latência e/ou taxas de erro no grupo "ouro", e nos LSPs de tráfego de usuários

prioritários (clientes com contratos de serviço com requerimentos finos de SLA) especificar que os caminhos apenas pode utilizar enlaces “ouro”.

Uma versão modificada do algoritmo SPF (*Shortest Path First*) chamada CSPF (*Constrained Shortest Path First*) realiza esta cálculo. Além da informação dos grupos-administrativos, também existe a informação de prioridade (utilizada pelo mecanismo de preempção, explicado no item 3.4.1) Estas informações adicionais são armazenadas na TED (*Traffic Engineering Database*) e utilizadas pelo algoritmo CSPF localmente em cada roteador. Para distribuição destas informações são necessárias adições aos protocolos de IGP para que estes advirtam informações de engenharia de tráfego para os enlaces para o domínio.

Já para a utilização de rotas explícitas, o administrador ao criar um LSP define os nós intermediários do caminho que o LSP deve atravessar do nó de ingresso ao nó de egresso.

3.3.1 Extensões do IS-IS para engenharia de Tráfego

O IS-IS [RFC1195] é um protocolo de IGP do tipo “*link-state*” (adverte informações sobre topologia da rede e não propriamente as rotas, deixando a cargo de cada roteador calcular os caminhos) baseado em um conjunto de mensagens fixas com campos variáveis que carregam as informações da rede (chamados “TLVs” – *Type Length Value*). O IS-IS permite roteamento hierárquico, segmentando a rede em dois níveis.

Extensões para o IS-IS foram adicionadas [RFC3784], onde foi proposto um novo TLV (de identificação de suporte a engenharia de tráfego do roteador que o adverte, TLV #134) além de adições ao existente TLV #22 (“*Extended IS Reachability*”).

Para as informações de engenharia de tráfego dos enlaces, foram adicionados sub-campos ao TLV #22, contendo informações sobre banda, grupos administrativos, prioridades, etc. A tabela 3.1 descreve os itens adicionados.

Tipo	Descrição
3	Grupo Administrativo (cor)
6	Endereço IPv4 da interface
8	Endereço IPv4 vizinho
9	Máxima banda do enlace
10	Banda reservável do enlace
11	Banda disponível para reserva do enlace

TABELA 3.1: SUB-TLVS ADICIONADOS AO TLV #22

3.3.2 Extensões do OSPF para engenharia de Tráfego

O OSPF [RFC2328] também é um protocolo de IGP do tipo *link-state*, composto por mensagens de campos variáveis. Os campos de informação que carregam as informações da topologia são denominados LSAs (“*Link State Advertisement”*).

De forma análoga ao IS-IS, extensões foram feitas ao OSPF [RFC3630] para anúncios de informação para engenharia de tráfego, carregando os mesmos tipos de informação. Este LSA (tipo 10) tem como escopo a área local, não sendo propagado para outras áreas. A figura 3.2 apresenta a estrutura do LSA tipo 10.

A estrutura do LSA para engenharia de tráfego é similar ao LSA tipo 2, onde:

- **Instance**: Número atribuído pelo roteador que realiza o anúncio (significância local);
- **LS-Type**:
  - Tipo ‘1’ tem a mesma finalidade do TLV#134 do IS-IS (item 3.3.1);
  - Tipo ‘2’ tem os mesmos objetos dos sub-TLVs #22 (item 3.3.1).
- **Traffic Engineering Type / Length / Value**: descrevem os atributos do LSA sendo anunciados.

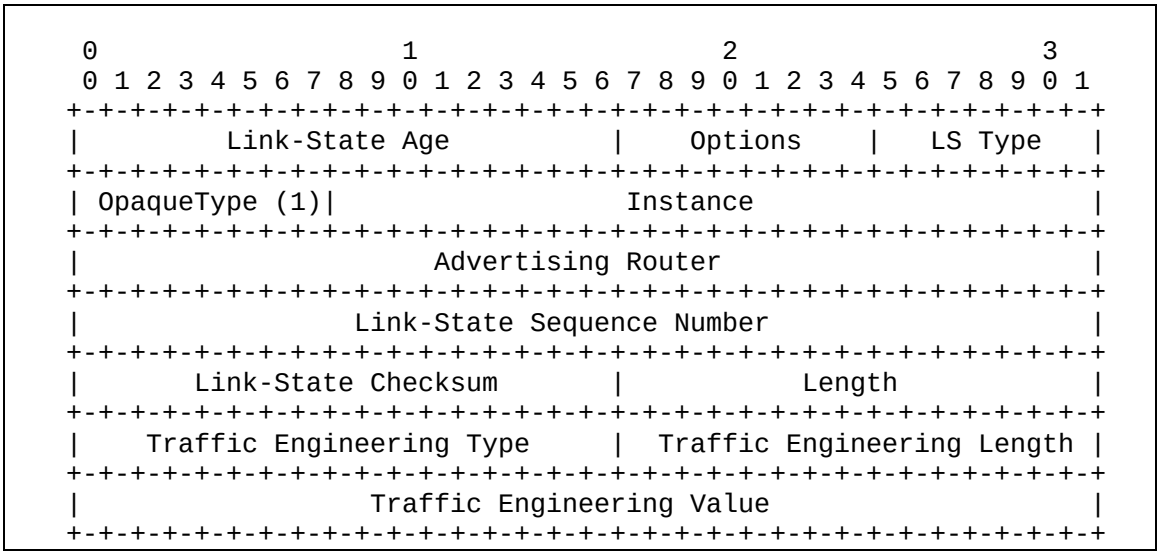

FIGURA 3.2: LSA TIPO 10

3.3.3 Engenharia de Tráfego com Rotas Explícitas

Existem dois tipos de rotas: "*Loose*" e "*Strict*" [RFC3209]. Quando a rota é "*Strict*", significa que os saltos intermediários da origem ao destino devem ser rigorosamente respeitados, não podendo haver qualquer salto intermediário não previamente especificado entre dois nós quaisquer inseridos no caminho. Já a rota "*Loose*" define saltos intermediários (pontos obrigatórios para a passagem do LSP), porém a rota (e número de saltos) fica livre entre dois nós especificados como "*Loose*"

A alocação dinâmica de caminhos embora prática sob o ponto de vista de administração, não é globalmente ótimo (conforme demonstrado no capítulo 1.1) por não levar em conta os outros fluxos para o cálculo dos caminhos (considera apenas as taxas livres dos enlaces). Outra desvantagem é que as informações de engenharia de tráfego carregadas pelos IGPs são locais à área (OSPF) ou nível (IS-IS) que o roteador se encontra, o que inviabiliza LSPs de serem calculados dentro de um mesmo domínio quando os roteadores de ingresso e egresso estão em áreas/níveis distintos. Além disso, há um acréscimo de processamento dos protocolos de IGP pelos roteadores pela necessidade da transmissão destas informações adicionais.

Por estas razões, a utilização do recurso de rotas explícitas, embora haja uma maior complexidade administrativa, tem as seguintes vantagens:

1. Permite que LSPs sejam sinalizados entre roteadores de ingresso e egresso que se encontram em áreas distintas
2. Poupa processamento dos roteadores pela não necessidade do uso das extensões de tráfego dos protocolos de IGP.
3. Pode prover resultados globalmente ótimos realizando o cálculo de forma centralizada.

3.4 Mecanismos de Proteção

Quando um LSP fica indisponível devido a uma falha (de algum nó ou enlace intermediário do caminho), o tráfego transportado por ele acaba sendo encaminhado via o roteamento do IGP. Essa situação pode ser crítica quando algum enlace estoura a sua capacidade de transmissão e o provedor não é capaz de cumprir os níveis de serviço (SLAs) acordados com seus clientes.

Para contornar esse tipo de situação, foram criados alguns mecanismos de proteção para os LSPs.

Os mecanismos de proteção a redes MPLS podem ser classificados em duas vertentes: proteção de caminho e proteção local [HUA02].

Proteção de caminho são mecanismos fim-a-fim (do nó de ingresso ao de egresso) de proteção ao LSP, onde os caminhos operacionais e de proteção devem ser preferencialmente disjuntos (i.e. não compartilham enlaces ou nós trânsito). Neste mecanismo sempre o nó de ingresso sinaliza o caminho de proteção. Este método é considerado lento dado o tempo que leva para o nó de ingresso perceber a falha e sinalizar o novo caminho.

Alternativamente, nos mecanismos de proteção local, um segmento de proteção pode ser sinalizado por um nó trânsito, imediatamente adjacente à falha, reduzindo o tempo de percepção de falha e de contorno. Este mecanismo, porém tem a desvantagem de levar a alocações de recurso sub-ótima após uma falha, devido a não levar em conta os caminhos utilizados por fluxos concorrentes.

3.4.1 Proteção de Caminho

Este tipo de proteção utiliza LSPs com caminhos alternativos (conhecidos como caminhos de proteção ou secundários) permitindo que no caso do caminho principal (primário) estar indisponível, o tráfego pode ser transportado por outro caminho.

Para a escolha de um caminho de proteção pode-se utilizar os mesmos mecanismos de escolha dos caminhos primários, através de CSPF ou de rota explícita.

Estes caminhos de proteção podem ser previamente sinalizados (apenas sendo usado durante falhas do caminho principal), ou serem sinalizados e utilizados somente após a queda do caminho principal.

Em conjunto com os caminhos principais, pode-se utilizar o protocolo BFD (*Bidirectional Forwarding Detection*) para diminuir o tempo de detecção de falha. O BFD é um protocolo baseado em mensagens (*keep-alives)* para detecção de queda de vizinhos. Extensões foram desenvolvidas [AGG07] para o protocolo também ser utilizado para verificar conectividade fim-a-fim em túneis MPLS. Este mecanismo permite a detecção de quedas com temops na ordem de mili-segundos. Para o MPLS, este protocolo opera trocando mensagens de *keep-alives* entre o roteador de ingresso e o roteador de egresso. Estes keep-alives são enviados dentro do LSP e realizam um teste de conectividade fim-a-fim.

Outra opção de proteção de caminho pode ser atingida com preempção através do uso das extensões de tráfego dos IGPs, estabelecendo prioridades aos túneis MPLS. No caso de

uma queda na rede em que um dado LSP não consiga ser re-alocado com os recursos disponíveis no momento, o LSP em questão pode desfazer outros LSPs com menor prioridade para “roubar” seus recursos. Caso não seja possível computar um caminho utilizando enlaces com banda disponível do nó de ingresso ao nó de egresso, o roteador de ingresso computa um caminho levando em conta os recursos das reservas de menor prioridade feitas nos enlaces intermediários. Se for possível liberar recursos suficientes dos LSPs de menor prioridade, o novo LSP será sinalizado.

Todavia o mecanismo de preempção não é recomendado, pois causa disrupção de serviço e não tem comportamento previsível, além de não garantir que haja contorno para qualquer falha.

3.4.2 Proteção Local

Os dois mecanismos de proteção local mais utilizados atualmente são o *Fast-Reroute* e LSPs de proteção de enlace.

3.4.2.1 Fast-Reroute

O *Fast-Reroute* [RFC4090] é um mecanismo de proteção local temporário que cria desvios ao longo de um caminho primário (principal) de um LSP. Estes desvios são previamente sinalizados (após o estabelecimento do caminho principal) e não necessariamente comportam a reserva de banda alocada ao caminho primário. Cada caminho de desvio retorna ao caminho principal assim que possível, na tentativa de diminuir o número de estados RSVP-TE na rede (os estados de conexões RSVP-TE consomem recursos de processamento nos roteadores para a manutenção dos caminhos, devido a constante troca de mensagens). A figura 3.3 ilustra um exemplo de um LSP com proteção *Fast-Reroute*: ao longo do caminho primário são criados LSPs de desvio aos próximos nós do caminho entre os roteadores de ingresso e de egresso. Estes LSPs de desvio somente serão utilizados quando ocorrer uma falha de enlace ou de nó ao longo do caminho primário.

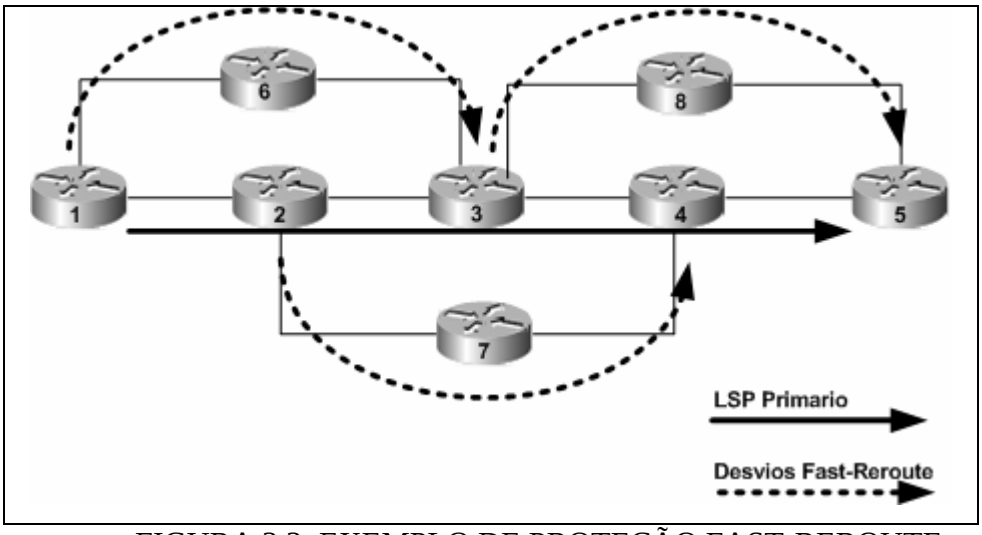

FIGURA 3.3: EXEMPLO DE PROTEÇÃO FAST-REROUTE

O roteador intermediário que detectar uma falha (PLR – "*Point of Local Repair*") irá comutar o tráfego (trocando a etiqueta MPLS) para o LSP de desvio para contornar a falha. Em paralelo, este roteador irá enviar uma mensagem de erro ao roteador de ingresso para que o mesmo sinalize um novo caminho, pois como o desvio não garante os recursos do LSP, ele somente deve ser utilizado durante um curto período de tempo enquanto o novo LSP não está sinalizado.

Para o funcionamento do *Fast-Reroute*, é necessário o uso das extensões para engenharia de tráfego do IGP, para que o roteador de ingresso possa calcular caminhos alternativos no domínio RSVP-TE.

Embora com este mecanismo haja rápido contorno da falha (tipicamente abaixo de 1 segundo), há a criação de 'n' LSPs de desvio ao longo da rede para cada caminho primário ('n' é o número de roteadores intermediários de um caminho primário).

3.4.2.2 Proteção de enlace

O esquema de proteção de enlace é similar ao *Fast-Reroute* (proteção local) no que se refere aos LSPs de desvio. Porém, este tipo de proteção utiliza outro objeto de sinalização RSVP-TE (objeto FAST REROUTE), e ao invés de o desvio contornar um nó, o desvio apenas contorna um dado enlace configurado pelo usuário como passível a sofrer proteção. Outra diferença é a questão do encaminhamento quando uma falha é detectada, para comutar o tráfego o PLR empilha mais uma etiqueta MPLS para utilizar o desvio, ao invés de utilizar a operação de SWAP como no *Fast-Reroute*.

3.5 Conclusão

Embora os mecanismos de proteção local sejam mais rápidos para detecção e contorno de falhas, estes métodos não são globalmente ótimos. O método proposto nesta dissertação visa obter caminhos de proteção evitando que alguma falha resulte em enlaces sobrecarregados (globalmente ótimo). Para se obter menores valores de tempo para convergência no caso de falhas, pode-se utilizar em conjunto com este projeto o BFD para MPLS.

Capítulo 4

Métodos de Otimização

4.1 Introdução

Problemas de otimização são formulados como um conjunto de equações (ou inequações) que descrevem o comportamento de um determinado sistema em conjunto com uma função objetivo a ser maximizada (ou minimizada) na busca pela melhor solução.

O problema de alocação de recursos para fluxos simultâneos pode ser formulado (ou aproximado) como um problema de otimização via um conjunto de sistemas de equação lineares [PIO04]. Devido ao excessivo número de variáveis e restrições que podem existir em um cenário de uma rede de grande porte, a resolução do problema via programação linear pode levar muito tempo.

Para se obter uma solução aproximada ao problema, pode-se utilizar métodos heurísticos, os quais buscam uma resposta aproximada ao problema pesquisando apenas parte do universo de soluções.

Por não retornarem a solução ótima ao problema, deve-se estabelecer critérios de paradas para estes métodos, que tipicamente são atrelados ao fator “tempo” (e.g. tempo computacional, número de iterações sem melhoria na qualidade da resposta, etc.).

Neste capítulo, serão revisados alguns dos principais métodos heurísticos que podem ser aplicados ao problema de alocação de recursos para fluxos simultâneos.

4.2 Busca Local

Os algoritmos de busca local podem ser utilizados em situações onde se busque uma solução maximizando ou minimizando um determinado critério em um conjunto finito de soluções candidatas.

Um algoritmo de busca local começa em uma solução candidata e iterativamente se move para uma solução candidata "vizinha". O conceito de vizinhança [AAR03] é definido através de uma função que estabeleça uma relação entre duas soluções.

Para realizar este movimento entre uma solução candidata e outra, o algoritmo busca qual a solução vizinha melhor minimiza/maximiza a função objetivo para efetuar o movimento.

4.3 Arrefecimento Simulado

Os algoritmos de arrefecimento simulado ("*Simulated Annealing*") [NAH89] são utilizados para resolução de problemas combinatoriais [SAI99]. Estes algoritmos fazem analogia ao processo de fundição de metais, onde primeiramente se eleva a temperatura do sólido a tal modo que os metais se fundam (átomos se movimentando livremente), e lentamente se resfria este material de modo a se organizar estes átomos (soluções) em uma estrutura uniforme com energia mínima.

Analogamente a estes processos, os algoritmos de arrefecimento simulado se movem entre soluções candidatas (utilizando o conceito de vizinhança dos métodos de busca local), onde uma solução vizinha aleatória é escolhida em função de uma variável 'T' (temperatura). Quanto maior for o valor de 'T', maior será a componente aleatória da nova solução vizinha a ser escolhida (diminuindo o risco de mínimo local, possibilitando um movimento para uma uma direção onde inicialmente se deteriore a qualidade da solução). A variável 'T' vai diminuindo a cada iteração ("resfriando") até um valor próximo de zero, onde se termina o processo.

4.4 Pesquisa Tabu

Os algoritmos de pesquisa tabu [GLO97] são similares aos de busca local, com a adição de estruturas que memorizam atributos das soluções já visitadas (e não as próprias soluções visitadas), as quais são consultadas antes do algoritmo mover de uma para outra

solução candidata vizinha. Elementos da lista tabu são apenas inválidos por um número determinado de iterações.

O problema do mínimo local é contornado com a modificação da estrutura de soluções sendo procurada, buscando explorar novas regiões do espaço de soluções através da lista tabu.

4.5 Algoritmos Genéticos

Os algoritmos genéticos realizam uma busca em uma dada população de cromossomos (soluções candidatas) pelos melhores indivíduos (cromossomos) que minimizam/maximizam a função objetivo. Este método utiliza duas operações (recombinação e mutação) que ocorrem durante o ciclo evolutivo (otimização). Algoritmos genéticos combinam a troca de informações (características) com a sobrevivência dos melhores indivíduos (classificados através da função objetivo) [SAI99].

4.5.1 Ciclo Evolutivo

As iterações (evolução) do método são dadas a partir da geração aleatória de uma população inicial de tamanho ‘n’, a cada ciclo sendo selecionados os melhores indivíduos. A partir destes melhores indivíduos, são gerados novos cromossomos para substituir os menos aptos (que pior minimizam/maximizam a função objetivo).

O término do ciclo evolutivo pode ser dado através da localização de uma resposta com determinada qualidade (ou erro), ou através de tempo computacional. O critério por tempo computacional é o mais comum, e impõe limite a execução do algoritmo.

4.5.2 Recombinação

A produção dos novos indivíduos é dada através das operações de recombinação e de mutação. A operação de recombinação é feita pela criação de dois novos cromossomos que herdam características de dois cromossomos “pais”. A recombinação usa como base os melhores indivíduos encontrados, visando produzir novos indivíduos que otimizem ainda mais a resposta (função custo).

4.5.3 Mutação

Como a recombinação apenas utiliza informação dos cromossomos pais, limita-se a procura ao conjunto de características que já estão presentes na população.

Para o método explorar novas regiões de soluções, é necessário inserir aleatoriamente novas características, fazendo com que o método aumente a região de soluções procuradas.

A operação de mutação altera um ou mais genes de determinado cromossomo estocasticamente para qualquer valor possível dentro do conjunto de elementos. Com esta operação, além de se aumentar a região explorada, também se contorna o problema de uma solução ficar “presa” em um mínimo local.

4.5.4 Pseudo-Código

Abaixo um pseudo-código de um algoritmo genético (adaptado de [WIK08a]):

1. Criação da População inicial (aleatória)
2. Repetir:
    1. Classificação (“Ranking”) dos indivíduos de acordo com a função custo
    2. Seleção dos melhores indivíduos para reprodução
    3. Criação da nova geração através da operação de recombinação (usando os melhores indivíduos), substituindo os piores indivíduos
    4. Mutação nos indivíduos
3. Até critério de parada

4.6 Conclusão

Como apontado em [ERB02], o problema de alocação de recursos concorrentes pode levar muito tempo se resolvido por sistemas de equações lineares em cenários (topologias) grandes. Para o projeto em questão será utilizado um algoritmo genético, conforme outros trabalhos similares já utilizaram (vide [ERI01], [MUL02], [MUL04]) com resultados satisfatórios.

Capítulo 5

Estado da Arte

5.1 Introdução

Neste capítulo serão apresentadas algumas propostas para engenharia de tráfego para redes IP discutidas na literatura. O universo de propostas de otimização para redes IP é bastante vasto. A fim de facilitar a apresentação, apenas os trabalhos mais diretamente relacionados com essa proposta serão apresentados nessa seção. O critério para seleção das propostas segue a nomenclatura definida pelo IETF em [RFC3272], sendo que todas as propostas discutidas nesse capítulo seguem os seguintes critérios:

- *Offline*: método utilizado quando não há necessidade de mudanças em tempo-real na topologia, empregado especialmente para busca das melhores soluções em espaços com muitas soluções;
- Centralizado: os planos de roteamento são estabelecidos por uma unidade central;
- Informação Global: as informações de recursos e demanda de toda a rede estão disponíveis para cálculos e futuro provisionamento.

5.2 Modelos Puramente IGP

Embora esta categoria de modelos não utilize túneis MPLS para engenharia de tráfego, estas propostas são utilizadas como base para as propostas híbridas (discutidas no item 5.4), e são historicamente as primeiras tentativas de engenharia de tráfego em redes IP.

A engenharia de tráfego destes métodos se baseia na escolha de métricas de interface nos anúncios do IGP, para influenciar os caminhos pelos quais os fluxos devem seguir, aproveitando melhor os recursos de rede.

5.2.1 Inverso da Capacidade

Um dos primeiros métodos proposto [CIS97] foi a de utilizar como métrica o inverso da capacidade do enlace, ao invés de se utilizar métrica unitária para cada um dos enlaces (contagem de saltos). O resultado desta operação é que enlaces com menor capacidade terão métricas maiores, diminuindo a probabilidade de serem utilizados (preferência por caminhos com enlaces de maior capacidade).

5.2.2 Modelo utilizando Busca Local

Em [FOR00] é proposto um algoritmo de busca local para determinar métricas para as interfaces visando balancear a carga nos enlaces dos nós, a partir de uma matriz de demanda e do grafo com a estrutura da rede. Para cada nova iteração, é modificada apenas a métrica de um elemento do vetor que contém a métrica de todos os arcos (interfaces), e recalculada a função objetivo.

Como função objetivo, se utilizou minimizar uma função somatório de todos os arcos, onde o valor de cada arco é dado a partir de sua utilização, possuindo valores fixos inteiros para determinadas faixas de utilização (e.g. se utilização entre 0 e 1/3, custo 1; se entre 1/3 e 2/3, custo 3).

Como contorno ao problema de mínimo local, a cada vez que uma solução candidata não é melhorada em 300 iterações, o algoritmo “perturba” a mesma, adicionando um valor aleatório entre -2 e 2 (igualmente distribuídos) em 10% das métricas do vetor, fazendo com que outras regiões de soluções sejam exploradas.

Como a cada nova solução devem-se recalcular todos os fluxos (computacionalmente oneroso), foi adotado um mecanismo para apenas recalcular os caminhos de menor custo dos fluxos cujos arcos sofreram alteração entre uma solução e outra.

Foram realizados testes utilizando uma topologia de um provedor norte-americano, comparando os resultados do método com os resultados do problema sendo resolvido através de programação linear. Os resultados obtidos com o método ficaram 2% abaixo do

dos resultados do problema via programação linear, e pelo menos 13% superior a outros métodos (como métrica unitária, inverso da capacidade, "*random-ospf*"). O problema é "NP-hard" e, portanto, a aproximação via a heurística de busca local que foi feita é mais rápida para cenários de redes grandes.

5.2.3 Modelo utilizando Algoritmo Genético Híbrido

Em [MUL02] a estratégia adotada foi a de um algoritmo genético híbrido, onde o vetor com possíveis soluções (cromossomo) é representado por valores inteiros significando as métricas a serem utilizadas em cada um dos enlaces. A partir da matriz de demanda, as soluções são avaliadas computando Dijkstra e realizando os somatórios resultantes dos fluxos em cada enlace. Em paralelo ao *loop* de evolução das soluções, é realizada uma busca heurística procurando o último melhor cromossomo que melhora o objetivo, o qual permanecerá na população para o próximo *loop*.

Como função objetivo, o algoritmo busca minimizar a carga nos enlaces, enquanto prefere soluções que contenham menos alterações quando comparada com as métricas correntes dos enlaces (que geralmente resultam em rotas de menor caminho).

Os resultados obtidos no cenário de teste apresentado distribuíram o tráfego de melhor forma que os esquemas de métrica inverso da capacidade e métrica unitária.

5.3 Modelos Híbridos IGP+MPLS

Os modelos híbridos IGP+MPLS buscam utilizar o roteamento via IGP o tanto quanto possível, apenas provisionando túneis MPLS quando a otimização das métricas do IGP é incapaz de melhorar a utilização dos recursos de rede.

5.3.1 Modelo utilizando Algoritmo Genético

O modelo híbrido apresentado em [MUL04] cria túneis MPLS usando três cenários de atalhos ("*shortcuts*" [RFC3906]), onde se utiliza um dado túnel MPLS para transporte de determinada parte do caminho, não sendo fim a fim, visando o reaproveitamento de túneis para diversos fluxos.

Como tratativa para atraso máximo, é adicionada uma condição de número máximo de saltos e atraso que um LSP pode ter (global, os mesmos valores máximo de número de saltos e atraso se aplicam indiscriminadamente a todos os LSPs). O algoritmo genético

utilizado neste método é basicamente uma versão modificada do apresentado em [MUL02], onde inicialmente se estabelecem os LSPs candidatos a uso (‘n’ caminhos mais curtos que atendem ao limite de saltos e atraso) para cada uma das demandas entre dois nós da rede. Como representação para o algoritmo genético, cada LSP para cada demanda é indexado, e nos cromossomos (vetor solução), cada posição se refere a um LSP a ser usado para a respectiva demanda, ou ‘0’ no caso de o fluxo seguir o IGP e não utilizar nenhum túnel MPLS. Com a lista dos LSPs candidatos e as demandas, é criada uma população inicial de soluções candidatas, e então é realizado o processo evolutivo em busca da melhor solução.

Como função objetivo, busca-se minimizar a utilização da rede e também o número de LSPs a serem empregados. A procura termina após 150 iterações ou após 70 iterações sem melhorias. O método é aplicado a uma rede de pesquisa Alemã (G-WiN), cujos resultados demonstrados aumentam o atraso médio do tráfego, porém, reduzem a utilização média dos enlaces.

Por utilizar valores máximos de atraso e número de saltos igual para todos os caminhos, este método não permite uma engenharia de tráfego fina, pois é incapaz de prover priorização de recursos de acordo com o tipo de tráfego (serviço) que cada caminho visa atender.

5.3.2 Modelo utilizando Programação Linear

O método híbrido apresentando em [LAH05] utiliza programação linear modelando a rede como um grafo não direcionado, resolvendo uma função objetivo composta por três parâmetros:

1. Minimizar congestionamento dos enlaces;
2. Minimizar utilização de recursos (i.e. preferir caminhos com menos número de saltos);
3. Reduzir custo administrativo (número de LSPs).

A partir da estrutura da rede e da matriz de demanda, o método utiliza uma lista pré-calculada de todos os LSPs possíveis para cada demanda, fazendo a seleção do melhor caminho de acordo com a funçao objetivo.

O cálculo do sistema de equações lineares foi feito utilizando MATLAB em conjunto com um *software* de código aberto para resolução de programas lineares. O

método foi testado utilizando como entrada a estrutura de rede de um provedor estadunidense (com 19 nós e 66 enlaces), utilizando uma matriz de demanda com valores fictícios. Os resultados foram primeiramente feitos utilizando cada função objetivo individualmente, e então testado uma soma ponderada dos termos de modo a se ter a melhor utilização da rede.

5.3.3 Modelo utilizando Arrefecimento Simulado

Outro modelo híbrido é apresentando em [SKI06], o qual propõe um método baseado em heurística Arrefecimento Simulado para computar um conjunto de LSPs, otimizando duas funções objetivo. A primeira função objetivo busca minimizar a carga no enlace mais utilizado, e a segunda busca o balanceamento de carga entre todos os enlaces. O método é aplicado usando o esquema de atalho ("*shortcut*") tal qual proposto em [MUL04].

O algoritmo utiliza uma lista de túneis candidatos pré-calculados para cada par fonte-destino, escolhendo um número máximo de saltos e um número máximo de caminhos distintos para cada LSP. A partir de uma matriz de demanda, o algoritmo realiza a busca pela melhor solução (impondo um teto ao número de LSPs utilizados na rede visando diminuir complexidade administrativa), gerando uma solução inicial com túneis aleatórios da lista de túneis candidatos. O algoritmo então procura uma nova solução na vizinhança da anterior, substituindo um LSP por iteração.

Como problema de desempenho do algoritmo, foi apontado a avaliação da função objetivo, a qual a cada nova solução tinha que recalcular a carga em todos os enlaces de todos os nós. Como contorno, foi criada uma matriz para armazenar as modificações entre uma solução e outra, fazendo com que o novo cálculo da função objetivo fosse realizado apenas somando ou subtraindo as diferenças entre as duas soluções.

Nos resultados demonstrados, usando como base uma rede de aproximadamente 20 roteadores e 40 enlaces, apenas 12 LSPs foram necessários para realizar balanceamento de carga satisfatório entre os enlaces. Em relação à comparação entre as duas funções objetivo, não houve diferença considerável, pois elas obtiveram distribuição de carga nos enlaces de forma igual para números de LSPs superiores a 10.

5.4 Modelos puramente MPLS

A engenharia de tráfego destes métodos se baseia na escolha de túneis para o transporte de todos os fluxos, sendo o esquema de IGP utilizado indiferente para a resolução do problema.

5.4.1 Modelo utilizando Programação Inteira Mista

O método em [ERB03] apresenta um estudo teórico sobre a relação entre as diversas componentes da função objetivo, utilizando programação inteira mista. O método é testado em uma rede fictícia com 10 roteadores (90 demandas). Para o atraso máximo, o método somente aceita LSPs com no máximo 3 saltos.

A partir da demanda e da estrutura da rede foram estudados os efeitos dos componentes da função objetivo em separado, divididos em subproblemas. As três componentes estudadas foram: balanceamento de carga nos enlaces, minimização do atraso dos túneis, e minimização da divisão de fluxos em caminhos distintos.

O trabalho concluiu que escolher um baixo atraso como objetivo implica em um balanceamento de carga menos eficiente (e vice-versa). Relativo ao terceiro objetivo, quanto mais caminhos distintos são alocados, melhor é o balanceamento e menor é o atraso médio do tráfego.

Como trabalhos futuros (também apontando em um artigo anterior [ERB02]), os autores consideraram o desenvolvimento de um método heurístico para ser adaptado à programação inteira mista que foi desenvolvida, para resolver o problema em cenários de redes grandes.

5.5 Modelos puramente MPLS com mecanismos de proteção

A engenharia de tráfego destes métodos se baseia na escolha de túneis para o transporte de todos os fluxos, com mecanismos de proteção à falhas.

5.5.1 Método com Proteção Local

Em [MEL03] é proposto um método em tempo real e descentralizado para cálculo de LSPs usando o mecanismo de proteção local. O algoritmo proposto visa proteger tráfego com requerimentos finos de atraso (i.e. tráfego tipo EF) agregados em um domínio *Diffserv*.

O método reduz a alocação de recursos requerida para obter a proteção, assumindo que a qualquer momento, no máximo 1 falha de enlace irá ocorrer na rede. Neste caso, diferentes caminhos de proteção (que protegem enlaces distintos) não serão utilizados simultaneamente.

Por ser descentralizado, o método exige que adições de código sejam feitas nos sistemas operacionais dos roteadores usados, o que dificulta a implementação do método.

5.5.2 Método com Proteção de Caminho

Em [GIA04] é apresentado um método heurístico centralizado *offline* que utiliza o mecanismo de proteção de caminho. O método apresentado consiste em um algoritmo de 3 fases, visando proteção à queda de enlace. A primeira fase calcula os caminhos operacionais utilizando as melhores rotas disponíveis, a segunda fase calcula os caminhos de proteção (utilizando as sobras de recursos da primeira fase). Como as fases I e II apenas utilizam os melhores caminhos, a solução obtida pode ser não-ótima em termos de número de demandas servidas. É proposta então uma terceira fase onde um algoritmo guloso é utilizado para degradar os caminhos operacionais e de proteção visando acomodar mais demandas previamente não atendidas.

5.6 Conclusão

O método proposto nesta dissertação emprega estratégias similares a outros trabalhos, como a utilização de um algoritmo de k-caminhos mais curtos, uma função custo não linear para obter balanceamento de carga e um algoritmo genético para resolver o cálculo dos caminhos como um problema de busca.

Entretanto, o método difere dos trabalhos apresentados por calcular caminhos dos LSPs sobre duas perspectivas: proteção de caminho e engenharia de tráfego. Dos trabalhos estudados, a proposta de [GIA04] é a mais similar encontrada, porém segue uma abordagem distinta, pois nossa abordagem não reserva banda para os caminhos de proteção.

Nosso método segue um caminho similar a [MEL03], onde a condição de uma falha simples permite realizar proteção de caminho sem desperdício de recursos.

Capítulo 6

Projeto

6.1 Introdução

Neste capítulo será apresentado o projeto para engenharia de tráfego de redes MPLS em detalhes. O projeto foi desenvolvido usando como base os trabalhos já realizados na área apresentados no capítulo anterior, com a adição da tolerância a falhas (de enlaces e de nós) através de caminhos de proteção. Notadamente os artigos que mais influenciaram a estrutura deste projeto foram os apresentados por [FOR00], [MUL04] e [ERB03].

Em [FOR00] foi apresentada um função objetivo visando diminuir o congestionamento através da penalização progressiva do custo dos enlaces de acordo com a sua carga. Como conclusão e trabalhos futuros apresentado em [ERB03], foi apontado a necessidade de trabalho com métodos heurísticos para cenários de redes grandes. O artigo de [MUL04] influenciou na estruturação do problema codificado na forma de um algoritmo genético.

O projeto utiliza o conceito de caminho de proteção, onde o mesmo será computado para ser disjunto em nó (sempre que possível) ou disjunto em enlace. No caso de ser disjunto a nó, o caminho de proteção garante que a queda de apenas 1 enlace ou de apenas 1 nó (com todos seus respectivos enlaces) não deixa o tráfego sem encaminhamento. Caso não seja possível calcular (devido a topologia da rede) um caminho de proteção disjunto em nó, será calculado um caminho disjunto em enlaces. Neste segundo caso somente suportará a queda de enlace simples na topologia, caso um nó intermediário de um dado LSP caia, o LSP não fica ativo. As reservas feitas serão do tipo FF (*Fixed Filter*), onde não há compartilhamento de recursos entre diferentes LSPs.

Este capítulo inicia apresentando a estrutura geral do algoritmo, seguido da formulação do problema e do cálculo dos caminhos. O capitulo termina apresentando o algoritmo genético utilizado, seguido da conclusão.

6.2 Estrutura

O algoritmo para o cálculo da otimização dos recursos de rede tem como entradas:

- Topologia da rede
- Demandas de tráfego

Como saídas, o algoritmo retorna:

- LSPs com caminhos operacionais e de proteção definidos
- Taxa de alocação de reserva do RSVP-TE para cada um dos enlaces na situação de rede sem falhas
- Situações de falhas simples de enlace ou de nó que causam congestionamento na rede

6.2.1 Tipos de Solução

A solução retornada pelo algoritmo pode ser uma dos três seguintes casos:

- Caso I: O algoritmo foi capaz de localizar caminhos primários e de proteção para todas as demandas entradas sem exceder a capacidade de nenhum enlace nas situações de nenhuma queda de enlace e de queda simples de enlace ou de nó.
- Caso II: O algoritmo foi capaz de encontrar caminhos primários sem estourar nenhuma capacidade de enlace, porém alguma(s) situação(ões) de queda simples de enlace ou de nó levam a rede a ter congestionamento em determinados enlaces. Neste caso, o algoritmo retorna as situações críticas de falha e suas respectivas conseqüências.
- Caso III: O algoritmo não foi capaz de encontrar um conjunto de caminhos primários com a estrutura de rede disponível de forma a acomodar todas as demandas sem estourar a capacidade de nenhum enlace. Neste caso, o algoritmo pode ser executado novamente, removendo demandas uma a uma até conseguir uma resposta tipo II.

6.3 Formulação do Problema

Neste trabalho, nós modelamos a topologia de rede como um grafo. O grafo com a topologia, denotado por **G** é definido como um conjunto de vértices **V** e um conjunto de arcos **A**. As equações abaixo expressam as relações.

$$\mathbf{G} = (\mathbf{V}, \mathbf{A}) \quad (6.3.1)$$

$$\mathbf{V} = \{v_i \mid i = 1,2, ..., N\} \quad (6.3.2)$$

$$\mathbf{A} = \{a_j \mid j = 1,2, ..., M\} \quad (6.3.3)$$

Onde N é o número de nós (vértices) e M é o número de enlaces (arcos). O enlace a_j conectando nós adjacentes *i* e *k* é usualmente denotado por:

$$a_j = (v_i, v_k) \quad (6.3.4)$$

Cada enlace a_j , ($a_j \in A$), tem os seguintes atributos: capacidade c_j (expresso em Mbps) e latência l_j (expresso em ms). Um requerimento de tráfego t_k entre os nós v_i e v_j é representado por uma tupla contendo o requerimento de banda b_{ij} (correspondendo a demanda a ser transportada da origen *i* ao destino *j*) e o máximo atraso d_{ij} (latência) fim-a-fim (expresso em ms).

$$t_k = (b_{ij}, d_{ij}) \quad (6.3.5)$$

O vetor de tráfego **T** contém os requerimentos ponto-a-ponto entre os pares de nós origem e destinos (OD).

$$\mathbf{T} = \{ t_k = (b_{ij}, d_{ij}) \mid i, j \in [1,N] \} = \{t_1 ... t_k ... t_D\} \quad (6.3.6)$$

onde: D = número de demandas de tráfego

O problema consiste em determinar os caminhos ótimos dos LSPs (principal e de proteção) correspondendo a cada requerimento de tráfego t_k. Os caminhos dos LSPs correspondendo aos $t_k = (b_{ij}, d_{ij})$ requerimentos é denotado como:

$$lsp_k = (P_k, R_k) \tag{6.3.7}$$

Os caminhos P_k e R_k correspondem a seqüência de arcos, e pode ser representada como:

$$P_k = \left(a_{w1} \quad \ldots \quad a_{wj} \quad \cdots \quad a_{wJ}\right) \quad wj \in [1, M] \tag{6.3.8}$$

$$R_k = \left(a_{r1} \quad \ldots \quad a_{rj} \quad \cdots \quad a_{rJ}\right) \quad rj \in [1, M] \tag{6.3.9}$$

A seqüência P_k e R_k devem ser disjuntas, i.e., elas não contêm nenhum enlace (ou nó) transito em comum, i.e.,

$$a_j \in P_k \Rightarrow a_j \notin R_k \wedge a_j \in R_k \Rightarrow a_j \notin P_k \quad \forall j \in [1, M] \quad \forall k \in [1, D] \tag{6.3.10}$$

Seja **lsp** = $\{lsp_k \mid k=1..D\}$ uma solução candidata correspondendo a todas as demandas do vetor de demandas **T**, definido em (6.3.6). Uma solução LSP é considerada possível se:

- A carga resultante do tráfego em cada enlace não ultrapassa a capacidade de transmissão.
- O atraso fim-a-fim correspondente a um caminho LSP não excede o atraso máximo tolerado pelo tráfego.
- Em caso de falha simples de enlace ou nó, a comutação dos LSPs afetados para os caminhos de proteção não resulta em sobrecarga da capacidade de nenhum enlace.

Um problema de otimização pode ser definido em termos de uma função custo f_c(**lsp**). Para uma solução candidata **lsp**, f_c(**lsp**) representa o “nível de rejeição” da solução correspondente. Para uma solução **lsp** ser válida ela precisa satisfazer as restrições do problema em questão. Seja **LSP** o espaço das soluções possíveis que satisfazem as restrições do problema, o problema de otimização consiste em achar o elemento **lsp**=**lsp***$\in$ **LSP** que minimiza a função f_c(**lsp**). Matematicamente, ela pode ser expressa como:

$$\mathbf{lsp}^* \in \mathbf{LSP} \quad \text{e} \quad f_c(\mathbf{lsp}^*) = \inf_{\mathbf{lsp} \in \mathbf{LSP}} f_c(\mathbf{lsp}) \tag{6.3.11}$$

Neste trabalho, nós consideramos uma função multi-objetivo que impõe as seguintes condições:

a) Penaliza soluções candidatas que levam a sobrecarga em enlaces, considerando a situação que a rede não contém nenhuma falha de nó ou de enlace (apenas os caminhos principais são usados). Isto é útil para acomodar novas demandas de tráfego sem modificar as já estabelecidas. Quando a carga atribuída a um enlace supera a sua capacidade, a solução correspondente é severamente penalizada. Como notado em [FOR00], o balanceamento de carga pode ser alcançado pela adoção de uma função convexa não linear. Esta componente da função f_c é denotada como f_{ca}, e definida como:

$$\mathrm{f}_{\mathrm{ca}}(\mathbf{lsp}) = \sum_{j=1}^{M} \gamma(\rho_j) \tag{6.3.12}$$

onde: $$y(\rho_j) = \begin{cases} \rho_j{}^3 & \text{for } 0 \le \rho_j \le 1 \\ 1000 \cdot \rho_j{}^3 & \text{for } 1 < \rho_j \end{cases}$$

e: ρ_j = índice de ocupação do enlace a_j

b) Penaliza soluções candidatas que, após uma falha de enlace, resultam em excesso de carga em pelo menos um dos enlaces da topologia. A carga do tráfego é analisada considerando que todos os LSPs afetados pela falha do enlace em questão são comutados para os respectivos caminhos de proteção. Esta componente da função f_c é denotada como f_{cb}, e definida como:

$$\mathrm{f}_{\mathrm{cb}}(\mathbf{lsp}) = \frac{1}{M} \sum_{k=1}^{M} f_{cfailure}(\mathbf{lsp}, k) \tag{6.3.13}$$

$$\mathrm{f}_{\mathrm{cfailure}}(\mathbf{lsp}, k) = \sum_{j=1,\, j \neq k}^{M} \gamma(\rho_j) \tag{6.3.14}$$

onde: $f_{cfailure}$ é o custo da função quando o k-ésimo enlace falha

c) Penaliza soluções candidatas que, após uma falha nó, resultam em excesso de carga em pelo menos um dos enlaces da topologia. A carga do tráfego é analisada considerando que todos os LSPs afetados pela falha do enlace em questão são comutados para os respectivos caminhos de proteção, e os LSP que eram ingresso ou egresso no nó que falhou são removidos. Esta componente da função f_c é denotada como f_{cc}, e definida como:

$$f_{cc}(\mathbf{lsp}) = \frac{1}{N}\sum_{k=1}^{N} f_{cnfailure}(\mathbf{lsp}, k) \tag{6.3.15}$$

$$f_{cnfailure}(\mathbf{lsp}, k) = \sum_{j=1,\, j \notin arcs(v_k)}^{M} \gamma(\rho_j) \tag{6.3.16}$$

onde: $f_{cnfailure}$ é o custo da função quando o k-ésimo nó falha

d) Apenas evitar congestionamento na rede através de melhor distribuição de tráfego leva a caminhos mais longos, o que resulta em consumo extra de banda [LAH05]. Assim, a função objetivo penaliza soluções candidatas onde o caminho utilizado pelo LSP principal não corresponde ao caminho ótimo (i.e., o caminho determinado utilizando o algoritmo de Dijkstra no caso de nenhuma falha). Esta componente da função f_c é denotada como f_{cd}, e definida como:

$$f_{cd}(\mathbf{lsp}) = \frac{1}{D}\sum_{k=1}^{D} f_{ref\_dijkstra}(lsp_k) \tag{6.3.17}$$

$$f_{ref\_dijkstra}(lsp_k) = \frac{\text{delay}(lsp_k)}{\text{delay}(lsp_{k,dijkstra})} \tag{6.3.18}$$

onde: delay(lsp_k) é o atraso do LSP candidato para a demanda t_k

e delay(lsp_k,dijkstra) é o atraso considerando o caminho ótimo

A função custo atribuída à solução lsp é então definida como:

$$f_c(\mathbf{lsp}) = f_{ca}(\mathbf{lsp}) + \alpha \cdot f_{cb}(\mathbf{lsp}) + \beta \cdot f_{cc}(\mathbf{lsp}) + \chi \cdot f_{cd}(\mathbf{lsp}) \tag{6.3.19}$$

Os parâmetros α, β e χ são fatores de ponderação, e podem ser utilizados para determinarem a importância relativa entre as componentes da função custo.

6.4 Cálculo dos LSPs Candidatos

A diversidade de caminhos dos LSPs é um ponto importante para o algoritmo, já que constitui os elementos individuais das combinações do universo de respostas possíveis. Se este cálculo fornecer poucas (ou nenhuma) opções de caminhos para um determinado fluxo, o tráfego deste fluxo ficará atrelado obrigatoriamente a um determinado conjunto de enlaces.

O algoritmo de cálculo de caminhos (principal e de proteção) tenta, primeiramente, computar caminhos que sejam disjuntos em enlaces, podendo tolerar a queda simples de um nó transito qualquer. Na impossibilidade de se computar um caminho com esta característica, o algoritmo tenta computar um caminho disjunto em enlaces, onde não tolerará a queda simples de determinados nós trânsito, mas tolerará a queda simples de qualquer enlace da topologia.

6.4.1 Caminhos Tolerantes a queda de nó simples

Os múltiplos caminhos computados para cada demanda são obtidos através do uso de um método de k-caminhos mais curtos. Este método é basicamente uma pequena alteração no algoritmo Dijkstra, que após a primeira execução (quando localiza o caminho mais curto), é executado novamente, removendo nós intermediários do caminho mais curto localizado. Os enlaces são removidos um a um para este cálculo.

Para cada caminho principal encontrado, o mesmo processo é executado para achar diversidade nos respectivos caminhos de proteção, porém com todos os nós intermediários do caminho principal removidos.

Em algumas situações, apenas a remoção de um nó por vez no mesmo caminho não é suficiente para prover caminhos distintos. Isto acontece se a solução alternativa é disjunta em nós em relação a melhor solução. Para superar este problema, após todos os nós do melhor caminho terem sido removidos, o algoritmo inicia uma segunda rodada, onde são removidos simultaneamente nós do melhor e do segundo melhor caminho encontrados. Isto assegura que, se a topologia permitir, pelo menos três caminhos principais distintos sejam encontrados para cada demanda.

Note que este processo pode gerar diferentes LSPs candidatos que possuem mesmo caminho principal, apenas diferindo nos caminhos de proteção. Também ocorre de LSPs candidatos terem caminhos principais diferentes, mas caminhos de proteção iguais.

A métrica usada no algoritmo de Dijkstra é o atraso de propagação (e de roteamento/comutação) do enlace. Um caminho candidato é apenas considerado como válido se a soma de todos os atrasos de enlaces não ultrapassar o máximo valor tolerado pela demanda. Caso o atraso não seja um problema para uma determinada topologia, esta métrica pode ser definida como uma simples contagem de saltos. Também são excluídos antes do cálculo de caminho para cada demanda os enlaces que não comportam a largura de banda que a demanda requer.

Ao calcularmos os LSPs candidatos através do algoritmo apresentado nesta seção para o cenário apresentado na introdução (Figura 1.3), obtivemos a lista de candidatos conforme a figura 6.1. Note que os LSPs não são compartilhados por demandas diferentes, cada demanda individual tem seus próprios LSPs. A lista de candidatos da figura 6.1 representa apenas uma das demandas entre os nós 1 e 5.

Candidato 0	Primário	1 - 2 - 5
	Proteção	1 - 3 - 5
Candidato 1	Primário	1 - 2 - 5
	Proteção	1 - 4 - 5
Candidato 2	Primário	1 - 3 - 5
	Proteção	1 - 2 - 5
Candidato 3	Primário	1 - 3 - 5
	Proteção	1 - 4 - 5
Candidato 4	Primário	1 - 4 - 5
	Proteção	1 - 2 - 5
Candidato 5	Primário	1 - 4 - 5
	Proteção	1 - 3 - 5

FIGURA 6.1 – EXEMPLO DE CANDIDATOS

6.4.2 Caminhos Tolerantes a queda de enlace simples

O cálculo dos caminhos tolerantes a queda simples de enlace é análogo ao cálculo dos caminhos tolerantes a queda simples de nós. A única diferença é que ao invés de se remover nós intermediários e executar novamente o algoritmo de Dijkstra, se remove enlaces.

6.4.3 Permutação de Caminhos

Notou-se que durante a execução do cálculo dos caminhos alguns pares de candidatos não eram encontrados de forma inversa, relativamente aos caminhos principal e de proteção, i.e., para um candidato “x” composto por um caminho principal “a” e caminho de proteção disjunto “b”, não havia um candidato “y” composto por um caminho principal “b” e caminho de proteção “a”.

Como os caminhos “a” e “b” são disjuntos, e têm os mesmos nós de ingresso e de egresso, definiu-se uma função que criava um LSP candidato invertendo os caminhos “a” e “b” de cada indivíduo encontrado, se o mesmo não foi encontrado pelo algoritmo.

6.5 Algoritmo Genético

O algoritmo genético utilizado para buscar a solução que minimiza a função custo entre as soluções candidatas segue as bases apresentadas na seção 4.5, sendo os detalhes discutidos nos próximos subitens.

6.5.1 Representação dos Cromossomos

Para a representação dos indivíduos foi utilizado um esquema similar ao apresentado em [MUL04]. Cada indivíduo é representado como um cromossomo, onde cada posição dos “genes” corresponde a uma determinada demanda. O conteúdo de cada gene representa qual LSP candidato o indivíduo está utilizando para atender a respectiva demanda.

A figura 6.2 ilustra um exemplo de um cromossomo (cenário da Figura 1.3).

	LSP candidatos para cada t (i,,j) ∈ T					
	Dmnd.0 cand. 5	Dmnd.1 cand. 5	**Dmnd.2 cand. 5**	Dmnd.3 cand. 5	Dmnd.4 cand. 5	Dmnd.5 cand. 5
	Dmnd.0 cand. 4	Dmnd.1 cand. 4	Dmnd.2 cand. 4	Dmnd.3 cand. 4	Dmnd.4 cand. 4	Dmnd.5 cand. 4
	Dmnd.0 cand. 3	**Dmnd.1 cand. 3**	Dmnd.2 cand. 3	Dmnd.3 cand. 3	Dmnd.4 cand. 3	Dmnd.5 cand. 3
	Dmnd.0 cand. 2	Dmnd.1 cand. 2	Dmnd.2 cand. 2	Dmnd.3 cand. 2	Dmnd.4 cand. 2	**Dmnd.5 cand. 2**
	Dmnd.0 cand. 1	Dmnd.1 cand. 1	Dmnd.2 cand. 1	Dmnd.3 cand. 1	Dmnd.4 cand. 1	Dmnd.5 cand. 1
	Dmnd.0 cand. 0	Dmnd.1 cand. 0	Dmnd.2 cand. 0	**Dmnd.3 cand. 0**	**Dmnd.4 cand. 0**	Dmnd.5 cand. 0
indivíduo	**5**	**3**	**5**	**0**	**0**	**2**

FIGURA 6.2 – EXEMPLO DE INDIVÍDUO

6.5.2 Classificação dos Indivíduos

Os indivíduos são classificados de acordo com a função custo (equação 6.3.19). Os que possuem menores valores são utilizados como "pais" para a operação de recombinação, enquanto os tem maior custo são removidos pelos novos cromossomos "filhos" da operação de recombinação.

Como parâmetros variáveis do método relativo à classificação, temos:

- Taxa de cromossomos pais
- Taxa de cromossomos filhos (igual a taxa de cromossomos pais devido a recombinação uniforme, explicada no item 6.5.3).

6.5.3 Operação de Recombinação

Seguindo a taxa de cromossomos pais, os mesmos são utilizados para a geração dos novos cromossomos que irão substituir os cromossomos da geração anterior com pior função custo. A recombinação utilizada foi a uniforme, onde cada cromossomo gerado herda características de ambos os cromossomos pais com igual probabilidade. Este método é provavelmente o mais utilizado para recombinação, pois é eficiente em identificar, herdar e proteger genes comuns e ao mesmo tempo recombinar genes não tão comuns ([SYW89], [PAG99] e [FAL99] apud [HUX07]). Cada operação de recombinação gera dois novos filhos, um sendo o complemento do outro nos valores de genes herdados dos pais.

6.5.4 Operação de Mutação

Após a operação de recombinação, todos os indivíduos podem sofrer mutação em seus valores de genes. A mutação auxilia o método a não convergir para um mínimo local. Apenas os indivíduos pais são poupados da mutação. Este conceito de elitismo evita que bons indivíduos (geradores de filhos) sofram uma mutação degenerativa [HAU98].

Como parâmetro variável do método relativo à classificação, temos:

- Taxa de mutação: valor percentual que o produto dos genes com os indivíduos não pais pode sofrer mutação.

Como exemplo, caso hajam 100 indivíduos, 50 demandas e taxa de mutação de 2%, o total de genes que sofrem mutação é igual a 100*50*0,02 = 100 genes (aleatoriamente sorteados entre os indivíduos).

Durante os testes, notou-se que para cenários pequenos (tal qual o cenário descrito no item 7.2 do próximo capítulo), um valor baixo da taxa de mutação (abaixo de 2%) pode induzir o algoritmo a ficar preso em uma região de mínimo local. Em contrapartida, em cenários grandes (como o cenário do item 7.3), notou-se que valores acima de 0,5% dificultaram a convergência da resposta (i.e. pequenas melhorias na resposta após a operação de recombinação desapareciam após a operação de mutação).

6.5.5 Critério de Parada

Por este método de otimização ser um método estocástico, deve-se atrelar um critério de parada. Este critério geralmente é baseado em termos de tempo computacional, mas também pode-se estabelecer critérios baseados na qualidade da solução encontrada.

Neste projeto, foram adotados dois parâmetros para tratar desta questão:

- Gerações sem Melhoria
- Máximo de Gerações

Enquanto o primeiro interrompe o ciclo evolutivo (operação de busca) após “n” gerações onde não houve melhorias, o segundo estabelece um tempo máximo computacional, impondo um limite do tempo de execução.

Para cenários pequenos (como exposto no item 7.2), o método pode convergir rapidamente para a melhor resposta e ter que rodar muitas gerações mais desnecessariamente. Neste caso, o primeiro parâmetro interrompe a busca. Para cenários grandes (como apresentado no item 7.3), notou-se que havia constante melhoria na qualidade da resposta, que ao longo do tempo iria decaindo percentualmente (o método encontrava respostas que melhoravam a melhor resposta anterior em muito pouco). Neste segundo caso, tornou-se necessário interromper o ciclo evolutivo em certo tempo.

6.6 Conclusão

O problema atacado pelo projeto foi o de alocação de recursos para túneis MPLS concorrentes com proteção a quedas simples (de enlace ou de nós).

O projeto foi estruturado como um algoritmo genético para buscar a melhor resposta para um problema de otimização de variáveis múltiplas. O método de otimização

visa encontrar uma resposta mais rapidamente que um método de força-bruta ou de um sistema de equações lineares com grande número de variáveis.

O algoritmo genético implementado neste projeto possui diversos parâmetros configuráveis, onde para cenários diferentes, existe a possibilidade de melhorar o desempenho de convergência. Estas melhoras variam de caso para caso, e devem ser testadas de acordo com a topologia em questão.

Capítulo 7

Resultados Obtidos

7.1 Introdução

Neste capítulo serão apresentados alguns resultados de testes realizados com o algoritmo proposto neste trabalho, sob o ponto de vista funcional e de desempenho.

Todos os testes foram executados em uma máquina Linux com processador AMD Opteron 2,2GHz.

7.2 Teste Funcional

O primeiro teste realizado foi baseado no segundo problema apresentado no capítulo 1.2 (cenário das figuras 1.3 a 1.5). O cenário consiste em 5 LSPs de 600Mbps e 1 LSP de 400Mbps a serem transportados do roteador 1 ao 5 (utilizando enlaces GigabitEthernet e POS OC-48 entre 3 roteadores de trânsito) com proteção a todas as falhas simples de enlace sem estouro de capacidade. Os parâmetros utilizados foram:

- $\alpha = 1$
- $\beta = 1$
- $\chi = 0,1$
- Taxa de mutação = 2%
- Tamanho da população = 50
- Taxa de cromossomos pais = 20%

O algoritmo rodou 54 gerações, sendo o critério de parada 50 gerações sem melhoria na reposta. O tempo total de execução foi de 0,045 segundo, incluindo o tempo de computação dos caminhos e da resposta (processo evolutivo). A figura 7.1 mostra a solução encontrada pelo problema, que nesse caso, foi do tipo I .

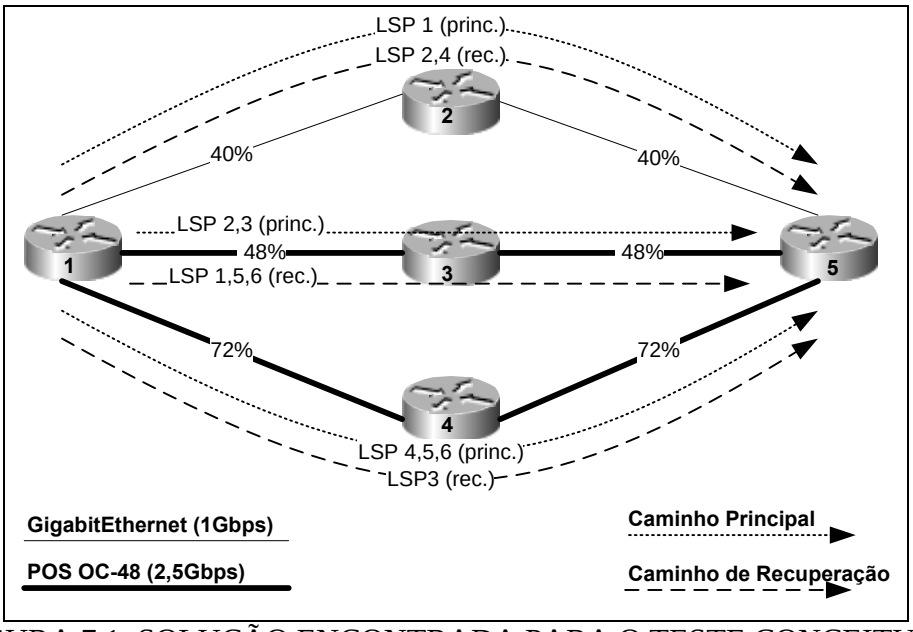

FIGURA 7.1: SOLUÇÃO ENCONTRADA PARA O TESTE CONCEITUAL

7.3 Testes de Desempenho

Os testes de desempenho foram feitos utilizando uma topologia baseada em um *backbone* de um provedor americano (figura 7.2), com 30 roteadores e 46 enlaces (todos sendo considerados com velocidade de 10Gbps). Na mesma topologia foram feitos testes diferentes apenas variando parâmetros nas simulações.

Para estes testes, os critérios de parada utilizados foram 50 gerações sem melhoria, ou 600 gerações de evolução.

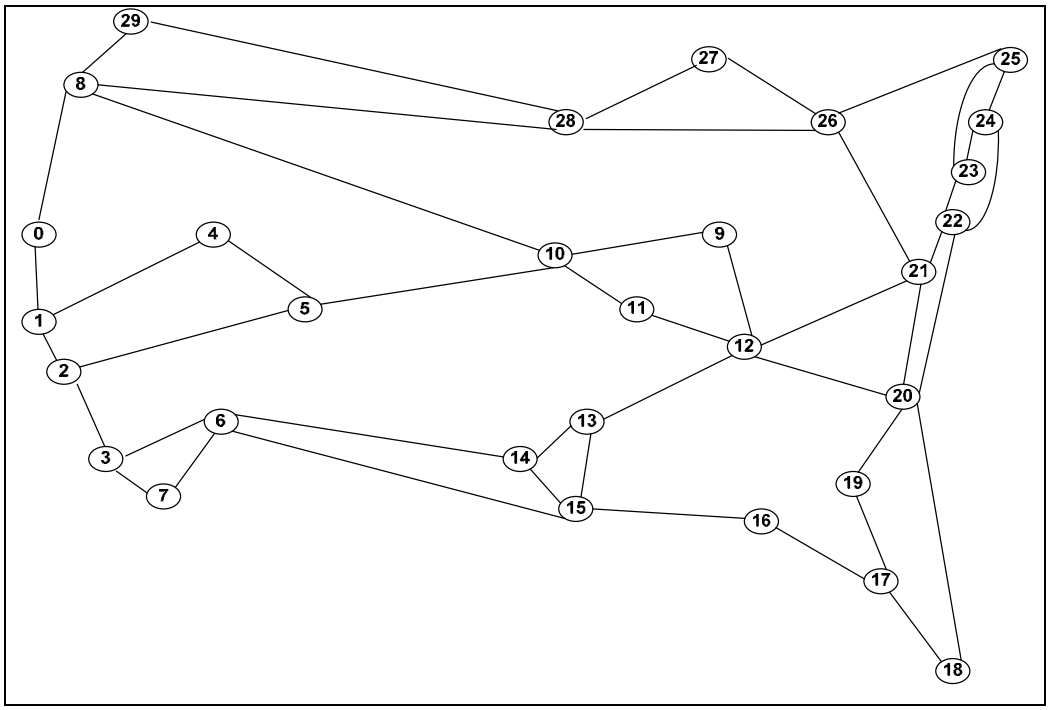

FIGURA 7.2: TOPOLOGIA DE TESTE BASEADA NO BACKBONE DE UM PROVEDOR AMERICANO

7.3.1 Carga Full-Mesh de 100M

O primeiro teste nesta topologia foi realizado simulando *full-mesh* de LSPs de 100M entre cada par de roteadores da topologia.

Os parâmetros utilizados foram:

- $\alpha = 0,25$
- $\beta = 0,75$
- $\chi = 0,001$
- Taxa de mutação = 0,5%
- Tamanho da população = 50
- Taxa de cromossomos pais = 20%

Para este cenário o algoritmo computou 7308 LSPs candidatos em 0,91 segundos. Em média 8,4 candidatos por demanda.

Neste teste o algoritmo parou após 600 gerações, conseguindo uma resposta tipo I para o problema (todos os LSPs foram alocados, e nenhuma falha de enlace ou nó simples causa congestionamento em nenhum outro enlace). O tempo total de execução do

programa foi de aproximadamente 14 minutos. A figura 7.3 demonstra as taxas de alocação de cada enlace (na situação de nenhuma falha na rede). A taxa média de alocação foi de 39,25% (3.925Mbps), com desvio padrão de 1.493Mbps.

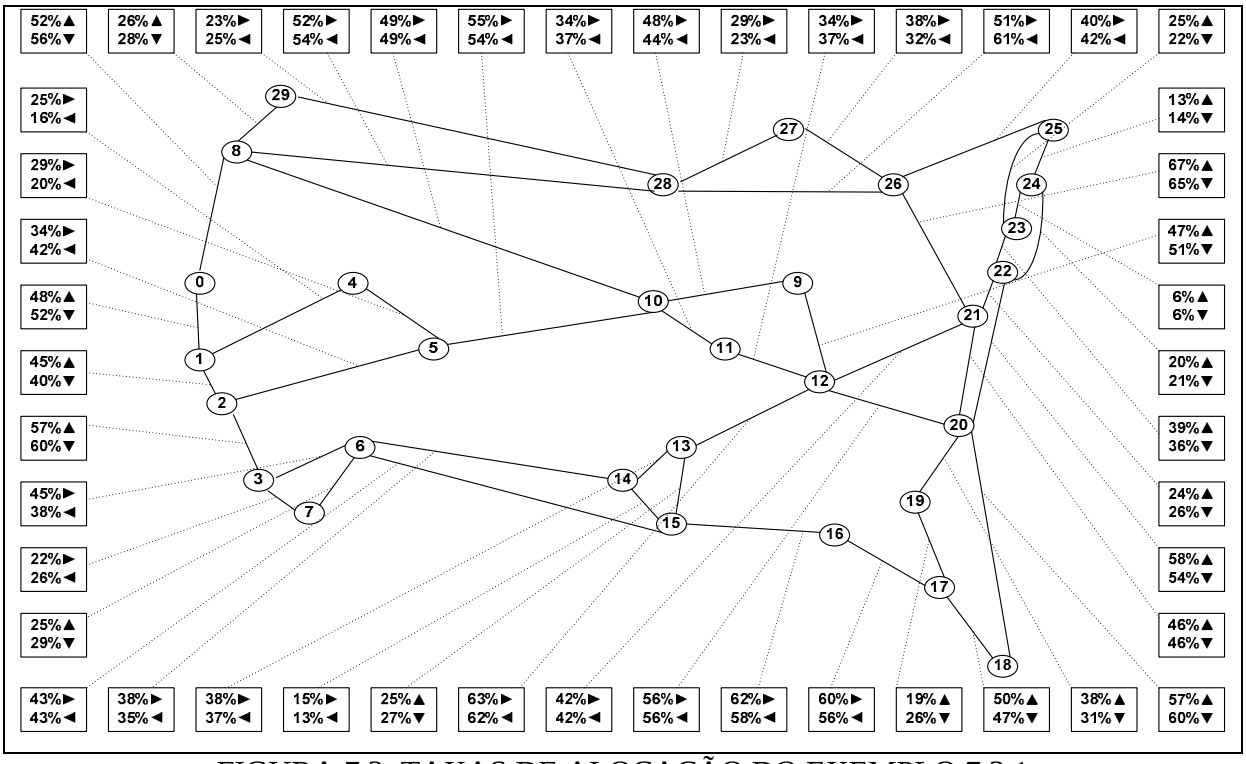

FIGURA 7.3: TAXAS DE ALOCAÇÃO DO EXEMPLO 7.3.1

7.3.2 Carga Full-Mesh de 100M com 60.000 gerações

Este segundo teste foi realizado com os mesmos parâmetros do item 7.3.1, apenas alterando o critério de parada do método para 60.000 gerações ao invés de 600. O tempo total de execução do programa foi de aproximadamente 23h30min. A figura 7.4 demonstra as taxas de alocação de cada enlace (na situação de nenhuma falha na rede). A taxa média de alocação foi de 36,5% (3.650Mbps), com desvio padrão de 1.196Mbps.

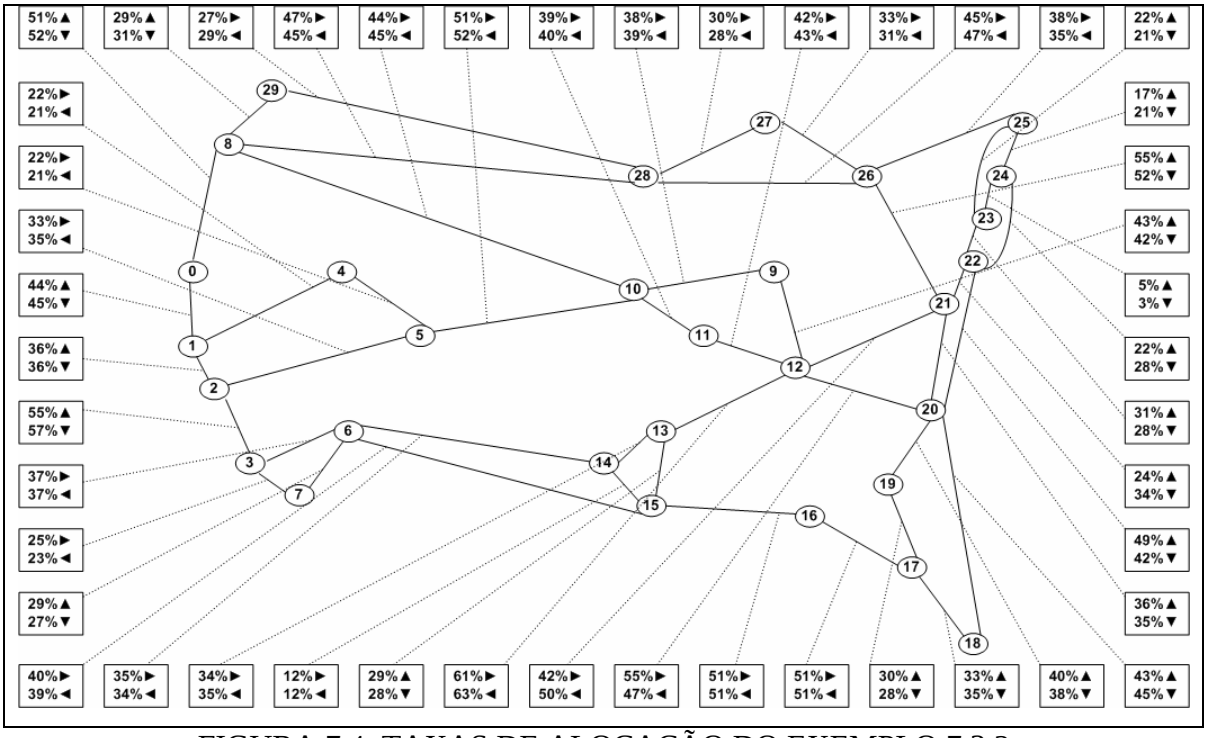

FIGURA 7.4: TAXAS DE ALOCAÇÃO DO EXEMPLO 7.3.2

	Item 7.3.1 com 600 gerações	**Item 7.3.2 com 60.000 gerações**
Carga média nos enlaces (Mbps)	3.925	3.650
Desvio padrão (Mbps)	1.493	1.196

TABELA 7.1 – COMPARAÇÃO DE RESULTADOS

7.3.3 Carga Full-Mesh de 100M apenas balanceamento de carga

Este segundo teste foi realizado com os mesmos parâmetros do item 7.3.1, exceto:

- $\alpha = 0$
- $\beta = 0$
- $\chi = 0$

Este teste foi realizado com o intuito de comparar a taxa de alocação dos enlaces utilizando apenas a componente de balanceamento de carga, com o exemplo anterior (que considerava as componentes de falha e de atraso); não levando em conta as situações de falha que causam congestionamento. A média de alocação neste exemplo foi de 35,69%

(3.569Mbps), com desvio padrão de 1.392Mbps. A figura 7.5 demonstra as taxas de alocação, e a tabela 7.2 compara com os resultados do item 7.3.1.

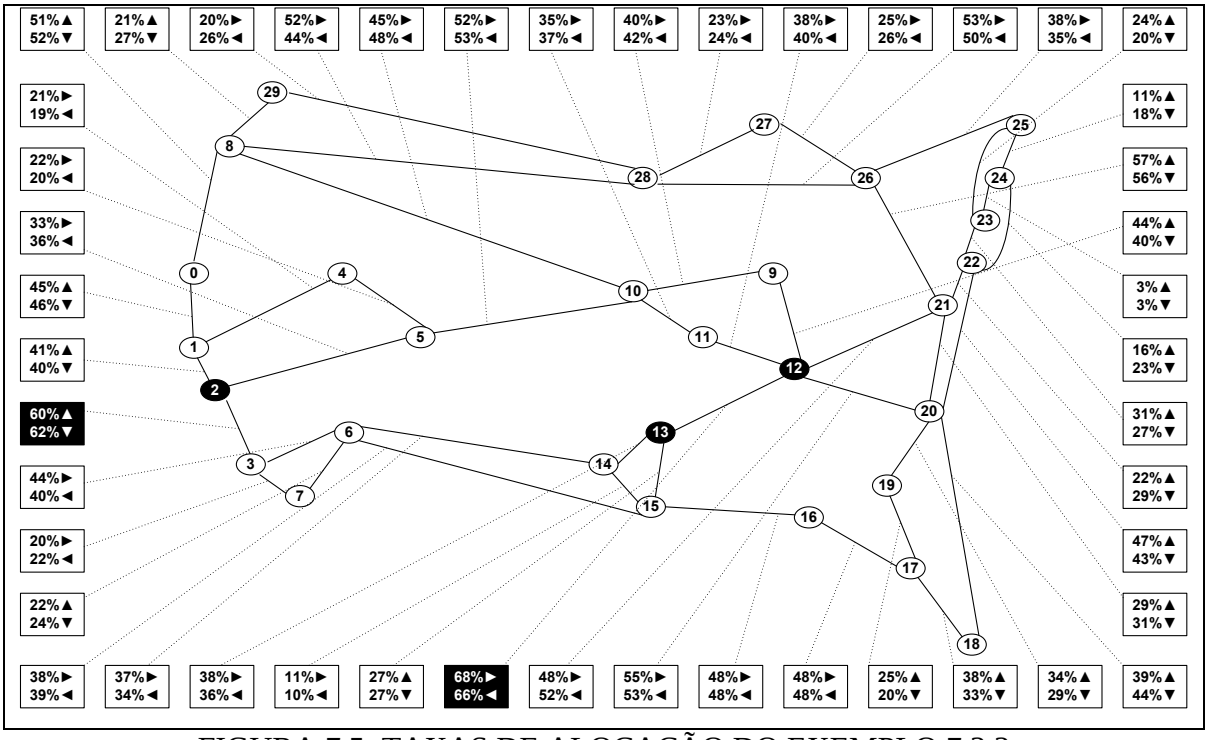

FIGURA 7.5: TAXAS DE ALOCAÇÃO DO EXEMPLO 7.3.3

	Item 7.3.1	**Item 7.3.3 (apenas balanc. de carga)**
Carga média nos enlaces (Mbps)	3.925	3.569
Desvio padrão (Mbps)	1.493	1.392
Num. de falhas de enlace que causam congest.	0	2
Num. de falhas de nó que causam congest.	0	3

TABELA 7.2 – COMPARAÇÃO DE RESULTADOS

As situações de falha para a resposta encontrada no caso de apenas balanceamento de carga são:

- Quando o enlace 2-3 (e 3-2) falha:
  - o enlace 12-13 fica 114% reservado
  - o enlace 13-12 fica 114% reservado

- Quando o enlace 12-13 (e 13-12) falha:
  - o enlace 2-3 fica 117% reservado
  - o enlace 3-2 fica 109% reservado

- Quando o nó 2 falha:
  - o enlace 12-13 fica 108% reservado
  - o enlace 13-12 fica 109% reservado

- Quando o nó 12 falha:
  - o enlace 0-1 fica 102% reservado
  - o enlace 2-3 fica 134% reservado
  - o enlace 3-2 fica 133% reservado
  - o enlace 3-6 fica 101% reservado
  - o enlace 8-0 fica 101% reservado
  - o enlace 8-28 fica 115% reservado
  - o enlace 26-28 fica 105% reservado
  - o enlace 28-8 fica 112% reservado
  - o enlace 28-26 fica 108% reservado

- Quando o nó 13 falha:
  - o enlace 2-3 fica 101% reservado

7.3.4 Carga Full-Mesh de 110M

Este cenário de teste utilizou os mesmo parâmetros do teste 7.3.1, sendo a única diferença nas demandas, ao invés de 100M para cada LSP, foi utilizado 110M. Os tempos de execução foram aproximadamente os mesmos do item 7.3.1, porém foi encontrada uma resposta tipo II, onde algumas falhas de enlace ou de nó causam congestionamentos na rede.

As situações de falha para a resposta encontrada são:

- Quando o enlace 2-3 (e 3-2) falha:
  - o enlace 12-13 fica 101,2% reservado

- Quando o enlace 12-13 (e 13-12) falha:
  - o enlace 2-3 fica 111,1% reservado

- Quando o enlace 12-20 (e 20-12) falha:
  - o enlace 2-3 fica 103,4% reservado
  - o enlace 3-2 fica 101,2% reservado

- Quando o nó 12 falha:
  - o enlace 2-3 fica 129,8% reservado
  - o enlace 3-2 fica 129,8% reservado
  - o enlace 8-28 fica 112,2% reservado
  - o enlace 28-26 fica 111,1% reservado

- Quando o nó 20 falha:
  - o enlace 2-3 fica 110% reservado
  - o enlace 3-2 fica 110% reservado
  - o enlace 15-16 fica 110% reservado
  - o enlace 16-15 fica 110% reservado

A figura 7.6 demonstra a distribuição de tráfego na rede. Os enlaces e nós em negrito representam as falhas que causam congestionamentos. A média de alocação neste exemplo foi de 47,82% (4.782Mbps), com desvio padrão de 2.061Mbps.

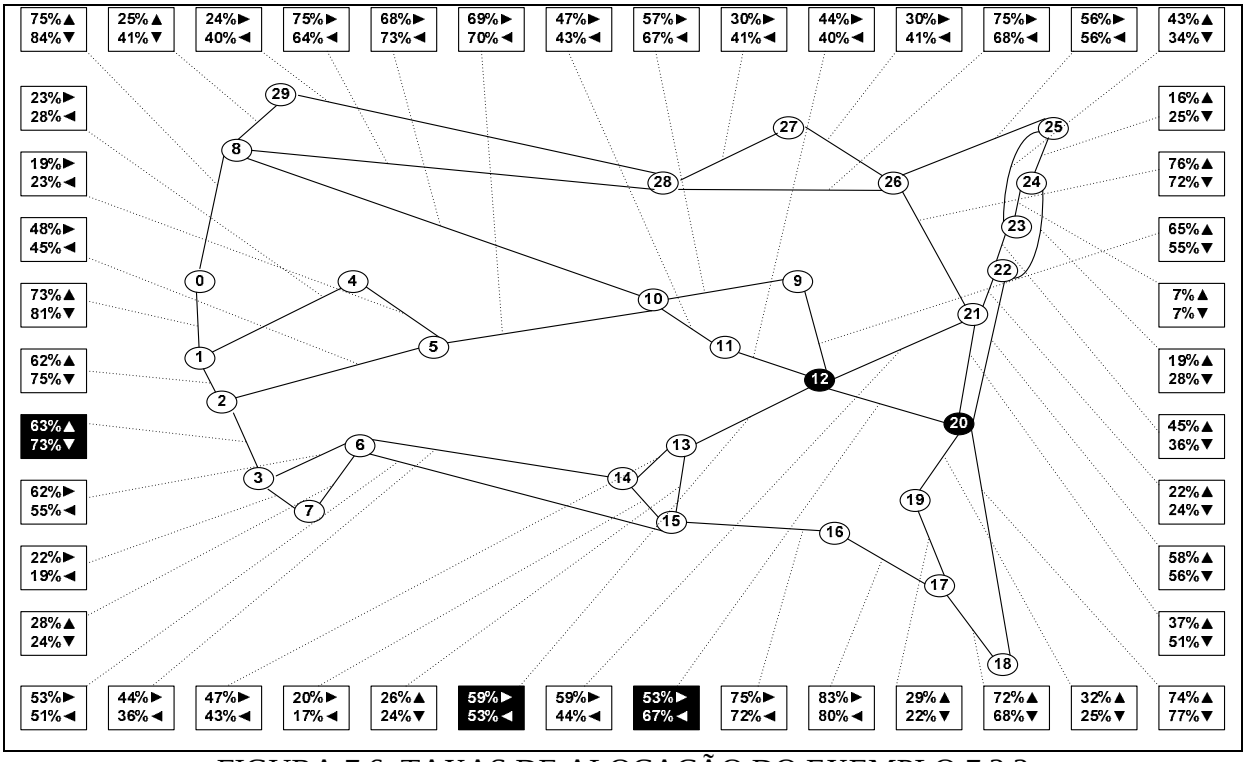

FIGURA 7.6: TAXAS DE ALOCAÇÃO DO EXEMPLO 7.3.3

7.3.5 Carga Full-Mesh de 110M com 60.000 gerações

Este teste foi realizado com os mesmos parâmetros do item 7.3.4, apenas alterando o critério de parada do método para 60.000 gerações ao invés de 600. O tempo total de execução do programa foi de aproximadamente 23h30min. A figura 7.7 demonstra as taxas de alocação de cada enlace (na situação de nenhuma falha na rede). A taxa média de alocação foi de 41,07% (4.107Mbps), com desvio padrão de 1.312Mbps.

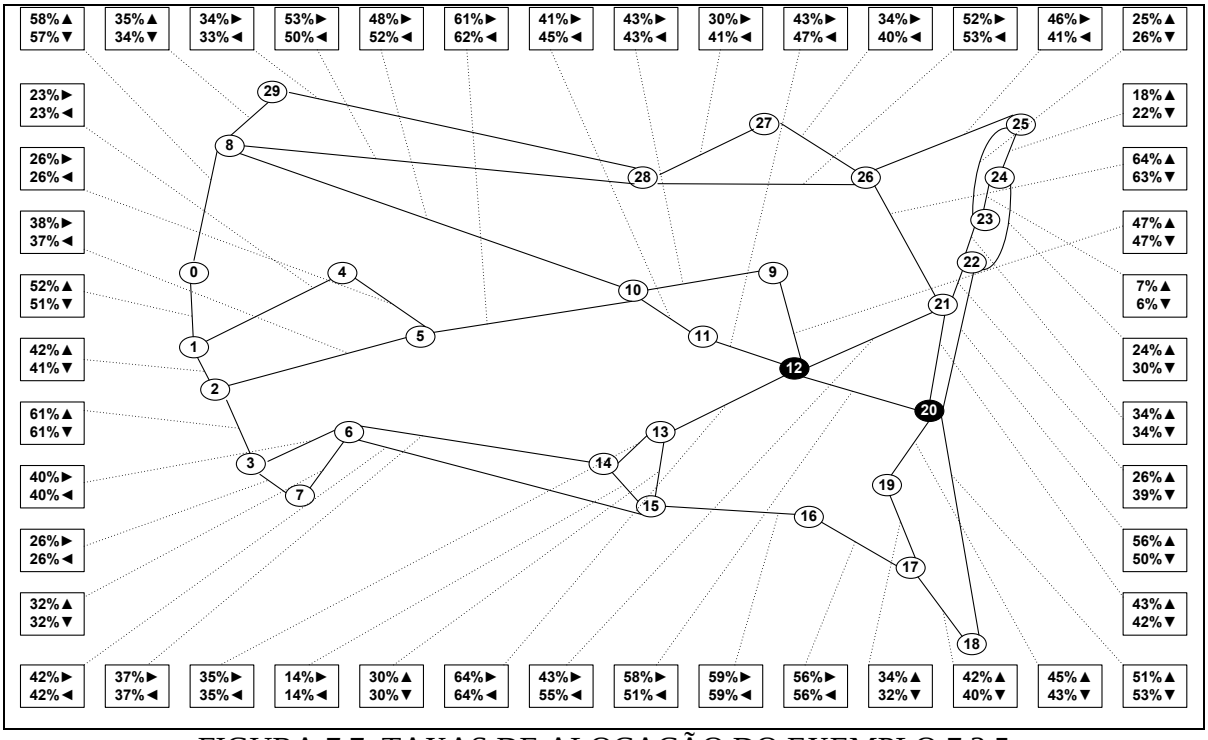

FIGURA 7.7: TAXAS DE ALOCAÇÃO DO EXEMPLO 7.3.5

A reposta encontrada também foi uma resposta tipo II, onde algumas falhas de nó causam congestionamentos na rede.

As situações de falha para a resposta encontrada são:

- Quando o nó 12 falha:
  - o enlace 2-3 fica 114,4% reservado
  - o enlace 3-2 fica 107,8% reservado
  - o enlace 16-17 fica 105,6% reservado

- Quando o nó 20 falha:
  - o enlace 2-3 fica 110% reservado
  - o enlace 3-2 fica 110% reservado
  - o enlace 15-16 fica 110% reservado
  - o enlace 16-15 fica 110% reservado

A tabela 7.3 compara os resultados do teste do item 7.3.4 com o teste do item 7.3.5.

	Item 7.3.4 com 600 gerações	**Item 7.3.5 com 60.000 gerações**
Carga média nos enlaces (Mbps)	4.782Mbps	4.107Mbps
Desvio padrão (Mbps)	2061Mbps	1312Mbps
Num. de falhas de enlace que causam congest.	3	0
Num. de falhas de nó que causam congest.	2	2

TABELA 7.3 – COMPARAÇÃO DE RESULTADOS

Conclusão

Este trabalho apresentou um método para determinar LSPs ótimos para redes MPLS. O método de otimização empregado utiliza uma função multi-objetivo que permite ao usuário balancear (ponderar) o objetivo de proteção com os objetivos de balanceamento de carga e de minimização de recursos utilizados.

Cenários de teste indicaram que o método convergiu para um resultado esperado e que o mesmo pode ser empregado em cenários grandes com relativo baixo tempo computacional. Em uma simulação com uma rede com aproximadamente 900 LSPs, o método foi capaz de localizar uma resposta em menos de 15 minutos.

Uma análise dos trabalhos relacionados e também das tecnologias disponíveis para o MPLS nos permite aportar diversos trabalhos futuros. Primeiramente, o método pode ser aprimorado para prover proteção para maior número de falhas de enlaces ou nós. Nesse caso, seria importante avaliar a utilização de outros métodos heurísticos para a resolução do problema combinatorial (a fim de comparação de eficiência computacional/convergência).

Neste trabalho, o problema de proteção foi tratado apenas para o caso de geração de rotas explícitas, sem utilizar os recursos de proteção de caminhos oferecidos pelo RSVP-TE. Todavia, também seria possível utilizar o método para calcular as prioridades de retenção e preempção dos caminhos de maneira a induzir que o RSVP-TE faça a proteção dinâmica de caminhos e ao mesmo tempo, busque objetivos de otimização globais.

Referências Bibliográficas

[AAR03] AARTS, E; LENSTRA, J. *Local Search in combinatorial optimization.* Princeton University Press, 2003.

[AGG07] AGGARWAL, R; KOMPELLA, K; NADEAU, T; SWALLOW, G. BFD For MPLS LSPs. IETF Internet draft, 2007.

[BRA03] BRADNER, S. https://www.ietf.org/IESG/LIAISON/IETF-RSVP-TE.txt , ITU-T Study Group 13, 2003.

[CIS97] CISCO. *Network Protocols Configuration Guide - Configuring OSPF*, http://www.cisco.com/univercd/cc/td/doc/product/software/ios113ed/113ed_cr/np1_c/1cospf.pdf , 1997.

[ERB02] ERBAS, S; MATHAR, R. *An Off-line Traffic Engineering Model for MPLS Networks*. 27th Annual IEEE Conference on Local Computer Networks, 2002.

[ERB03] ERBAS, S; ERBAS, C. *A Multiobjective Off-line Routing Model for MPLS Networks*, Proc. of the 18th International Teletraffic Congress, 2003.

[ERI01] ERICSSON, M; RESENDE, M; PARDALOS, P. *A Genetic Algorithm for the Weight Setting Problem in OSPF Routing*. AT&T Research, 2001.

[FAL99] FALKENAUER, E. *The worth of uniform crossover*. Proceedings of the 1999 Congress on Evolutionary Computation, vol. 1, pp. 782, CEC99, USA, 1999.

[FOR00] FORTZ, B; THORUP, M. *Internet Traffic Engineering by Optimizing OSPF Weights*, INFOCOM, 2000.

[GIA04] GIANSANTE, E., IOVANNA, P., ORIOLO, G., PASCALI, F.,.ROMAGNOLI, A. AND SABELLA, R. *Offline Protection Algorithm in a MPLS-based scenario*, Workshop on Traffic Engineering, Protection and Restoration for NGI, 2004.

[GLO97] GLOVER, F; LAGUNA, M. *Tabu Search*. Kluwer, MA, 1997.

[HAU98] HAUPT, R; HAUPT, S. *Practical Genetic Algorithms.* John Wiley & Sons, 1998, p. 41-42.

[HUA02] HUANG, C; SHARMA, V; OWENS, K; MAKAN, S. *Building reliable MPLS networks using a path protection mechanism*. IEEE Communication Magazine, pp. 156-162, March 2002.

[HUX07] HU, X; PAOLO, E. *An Efficient Genetic Algorithm with Uniform Crossover for the Multi-Objective Airport Gate Assignment Problem*. IEEE Congress on Evolutionary Computation, 2007.

[LAH05] LAHOUD, S; TEXIER, G; TOUTAIN, L. *Off-line Flow Allocation for Traffic Engineering in MPLS Networks*. LANMAN Greece, 2005.

[MEL03] MÉLON, L; BLANCHY, F; LEDUC, G. *Decentralized Local Backup LSP Calculation with Efficient Bandwidth Sharing*. IEEE ICT 2003.

[MUL02] MULYANA, E; KILLAT, U. *A Hybrid Genetic Algorithm Approach for OSPF Weight Setting Problem*. 2nd Polish-German Teletraffic Symposium, 2002.

[MUL04] MULYANA, E; KILLAT, U. *Optimization of IP Networks in Various Hybrid IGP/MPLS Routing Schemes*. 3rd Polish-German Teletraffic Symposium, 2004.

[NAH89] NAHAR, S; SAHNI, S; SHRAGOWITZ. *Simulated annealing and combinatorial optimization.* International Journal of Computer Aided VLSI Design, 1989.

[PAG99] PAGE, J; POLI, P; LANGDON, W. *Smooth uniform crossover with smooth point mutation in genetic programming: A preliminary study*. Genetic Programming. Proceedings of EuroGP'99, Sweden, 1999.

[PIO04] PIÓRO, M; MEDHI, D. *Routing, Flow, and Capacity Design in Communication and Computer Networks*. Elsevier publishing, 2004.

[RFC1195] CALLON, R. *Use of OSI IS-IS for Routing in TCP/IP and Dual Environments.* IETF RFC 1195, 1990.

[RFC1633] BRADEN, R; CLARK, D; SHENKER, S. Integrated Services in the Internet Architecture: an Overview. IETF RFC 1633, 1994.

[RFC2113] KATZ, D. *IP Router Alert Option*. IETF RFC 2113, 1997.

[RFC2205] BRADEN, R; ZHANG, L; BERSON, S; HERZOG, S; JAMIN, S. Resource ReSerVation Protocol (RSVP) -- Version 1 Functional Specification. IETF RFC 2205, 1997.

[RFC2328] MOY, J.*OSPF Version 2.* IETF RFC 2328, 1998.

[RFC2475] BLAKE, S; BLACK, D; CARLSON, M; DAVIES, E; WANG, Z; WEISS, W. An Architecture for Differentiated Services. IETF RFC 2475, 1998.

[RFC2961] BERGER, L; GAN, D; SWALLOW G; PAN, P; TOMMASI, F; MOLENDINI, S. *RSVP Refresh Overhead Reduction Extensions*. IETF RFC 2961, 2001.

[RFC3031] ROSEN, E; VISWANATHAN, A; CALLON, R. Multiprotocol Label Switching Architecture. IETF RFC 3031, 2001.

[RFC3036] ANDERSSON, L; DOOLAN, P; FELDMAN, N; FREDETTE, A; THOMAS, B. *LDP Specification*. RFC 3036, IETF, 2001.

[RFC3209] AWDUCHE, D; BERGER, L; GAN, D; LI, T; SRINIVASAN, V; SWALLOW, G. *RSVP-TE: Extensions to RSVP for LSP Tunnels*. IETF RFC 3209, 2001.

[RFC3212] JAMOUSSI, B; ANDERSON, L; CALLON, R; DANTU, R; WU, L; DOOLAN, P; WORSTER, T; FELDMAN, N; FREDETTE, A; GIRISH, M; GRAY, E; HEINANEN, J; KILTY, T; MALIS, A. *Constraint-Based LSP Setup using LDP*. IETF RFC 3212, 2002.

[RFC3272] AWDUCHE, D; CHIU, A; ELWALID, A; WIDJAJA, I; XIAO, X. *Overview and Principles of Internet Traffic Engineering*. IETF RFC 3272, 2002.

[RFC3630] KATZ, D; KOMPELLA, K; YEUNG, D. *Traffic Engineering (TE) Extensions to OSPF Version 2*. IETF RFC 3630, 2003.

[RFC3784] SMIT, H. *Intermediate System to Intermediate System (IS-IS) Extensions for Traffic Engineering (TE)*. IETF RFC 3784, 2004.

[RFC3906] SHEN, N; SMIT, H. *Calculating Interior Gateway Protocol (IGP) Routes Over Traffic Engineering Tunnels*. IETF RFC 3906, 2004.

[RFC4090] PAN, P; SWALLOW, G; ATLAS; A. *Fast Reroute Extensions to RSVP-TE for LSP Tunnels*. IETF RFC 4090, 2005.

[SAI99] SAIT, S; YOUSSEF, H. *Iterative Computer Algorithms with applications in Engineering*. IEEE Computer Society, 1999.

[SKI, 06] SKIVÉE, F; BALON, S; LEDUC, G. A *scalable heuristic for hybrid IGP/MPLS traffic engineering – Case study on an operational network*. ICON '06. 14th IEEE International Conference on Networks, 2006.

[SYW89] SYWERDA, G. *Uniform crossover in genetic algorithms*. Proceedings of the 3rd International Conference on Genetic Algorithms, USA, 1989.

[WIK08a] WIKIPEDIA. *Genetic Algorithms.* Disponível em http://en.wikipedia.org/wiki/Genetic_algorithm , Acesso em: 3/3/2008.