



**UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ
CAMPUS CURITIBA**

GERÊNCIA DE PESQUISA E PÓS-GRADUAÇÃO

**PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA
ELÉTRICA E INFORMÁTICA INDUSTRIAL - CPGEI**

FERNANDO BARRETO

**ESQUEMA DE CAMINHOS EMERGENCIAIS
RÁPIDOS PARA AMENIZAR PERDAS DE
PACOTES**

TESE DE DOUTORADO

**CURITIBA
AGOSTO-2008**

UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA
E INFORMÁTICA INDUSTRIAL

TESE
apresentada à UTFPR
para obtenção do título de

DOUTOR EM CIÊNCIAS

por

FERNANDO BARRETO

ESQUEMA DE CAMINHOS EMERGENCIAIS RÁPIDOS
PARA AMENIZAR PERDAS DE PACOTES

Banca Examinadora:

Presidente e Orientador:

Prof. Dr. Luiz Nacamura Junior **UTFPR**

Co-orientador:

Prof. Dr. Emilio Carlos Gomes Wille **UTFPR**

Examinadores:

Prof. Dra. Anelise Munaretto Fonseca **UTFPR**

Prof. Dr. Walter Godoy Jr **UTFPR**

Prof. Dr. Marcelo Eduardo Pellenz **PUC-PR**

Prof. Dr. Antônio Tadeu Azevedo Gomes **LNCC**

Curitiba, agosto de 2008.

FERNANDO BARRETO

**ESQUEMA DE CAMINHOS EMERGENCIAIS RÁPIDOS PARA
AMENIZAR PERDAS DE PACOTES**

Tese apresentada ao Programa de Pós-Graduação em Engenharia Elétrica e Informática Industrial da Universidade Tecnológica Federal do Paraná, como requisito parcial para a obtenção do título de “Doutor em Ciências” – Área de Concentração: Telemática.

Orientador: Prof. Dr. Luiz Nacamura Junior

Co-orientador: Prof. Dr. Emilio Carlos Gomes Wille

Curitiba
2008

Agradecimentos

Agradeço a meus pais, por darem apoio, incentivo, suporte financeiro desde o início dessa longa jornada acadêmica. Aos meus irmãos por ter dado todo apoio nos momentos difíceis. A minha esposa Ana por tudo que ela representa na minha vida além de compreender o tempo empenhado no desenvolvimento dessa tese.

A todos os meus amigos que proporcionaram ajuda e momentos de descontração necessários para aliviar o estresse decorrente da elaboração desta tese.

Ao meu orientador, por toda a paciência e ajuda na elaboração desse documento. Sempre guiando quando possível no desenvolvimento deste trabalho.

Ao professor Emilio por ter ajudado a esclarecer vários detalhes das formulações dessa tese e ajuda nesse documento.

Aos professores da Coordenação de Informática da UTFPR Ponta Grossa pela amizade e momentos de descontração.

Aos professores do campus Apucarana da UTFPR por me liberar em algumas ocasiões para finalização dessa tese.

Resumo

Os *backbones* IP utilizam protocolos de roteamento do tipo estado do enlace para definir as rotas corretamente. Em situações de mudança na topologia, como uma falha, esses protocolos necessitam de um tempo para reagir e encontrar novas rotas. Durante esse tempo, as rotas ficam instáveis com alta taxa de pacotes perdidos e queda na confiabilidade do *backbone*. Esse trabalho propõe uma abordagem pró-ativa denominada *Esquema de Caminhos Emergenciais Rápidos* para auxiliar o protocolo de roteamento OSPF a reduzir a taxa de pacotes perdidos durante esse período. Essa abordagem realiza cálculos reutilizando a base de informações de roteamento do OSPF para gerar esses caminhos emergenciais, que são representados na tabela de encaminhamento através de marcas. Essas marcas são utilizadas então para guiar corretamente os pacotes no contorno de uma falha. Essa abordagem é avaliada em representações de topologias artificiais e reais, e também em simulação para analisar qual o ganho obtido na redução de pacotes perdidos em relação ao OSPF original. A abordagem desenvolvida demonstra resultados bastante satisfatórios em termos de extensão dos caminhos de recuperação utilizados e quantidade de informações extras adicionadas na tabela de encaminhamento em relação à abordagem concorrente.

Abstract

Title: Fast Emergency Paths Schema to Soften Packet Loss

IP network backbones use link state routing protocols to find correct routes. In face of a topology change, e.g. a failure, these protocols need some time to react to it in order to find new routes. During this time, the routes become unstable, causing high packet loss rate and depreciation of backbone reliability. This work presents a proactive approach named *Fast Emergency Paths Schema* to help the OSPF routing protocol during the convergence period in order to reduce packet loss rate. The approach conducts calculations reusing the routing information base of the OSPF in order to generate these emergency paths, which are represented in the forwarding table as marks. These are then used to guide the packets to correctly bypass a failure. The approach is evaluated by using various artificial and real topologies, and a simulation is also implemented in order to analyze the packet loss rate reduction in relation to the original OSPF. The analysis yielded satisfactory results in terms of the extension of the recovery paths used and of the amount of extra information added to the forwarding table in relation to the concurrent approach.

Sumário

1 INTRODUÇÃO	1
1.1 Objetivos	3
1.2 Estrutura da Tese.....	4
2 ABORDAGENS DE ROTEAMENTO IP	5
2.1 Introdução	5
2.2 Protocolo de Roteamento em Redes de Interconexão	6
2.2.1 Plano de Controle.....	7
2.2.2 Plano de Dados	9
2.3 Protocolos de Roteamento em Redes IP	10
2.4 Interior Gateway Protocols	11
2.4.1 Open Shortest Path First – OSPF.....	14
2.4.2 Período de Convergência do OSPF.....	16
2.4.3 Implicações do Período de Convergência	19
2.4.4 Redução do Período de Convergência do OSPF.....	21
2.4.4.1 Detecção de falha: <i>Hello</i> e Mecanismos de <i>Hardware</i>	21
2.4.4.2 Redução do Tempo de Notificação de Falha, <i>spf-interval</i> e <i>SPF</i>	22
2.4.4.3 Atualização da FIB	23
2.4.5 Viabilidade do Período de Convergência ser < 1s	23
2.5 Conclusão.....	24
3 FALHAS EM BACKBONES IP E SOLUÇÕES EXISTENTES	25
3.1 Introdução	25
3.2 Características das Falhas em Backbones IP.....	25
3.3 Abordagens Reativas.....	27
3.4 Abordagens Pró-Ativas	29
3.4.1 Abordagem <i>Equal Cost Multi-Path</i> – ECMP.....	30
3.4.2 Abordagem <i>Loop-Free Alternates</i> – LFA.....	30
3.4.3 Abordagem <i>Failure Insensitive Routing</i> – FIR.....	32
3.4.4 Abordagem <i>Multiple Routing Configuration</i> – MRC	33
3.4.5 Abordagem Not-Via	37
3.4.6 Outras Abordagens Pró-Ativas Relacionadas	40
3.5 Sumário das Abordagens em Redes IP	42

3.6	Conclusão.....	46
4	ESQUEMA DE CAMINHOS EMERGENCIAIS RÁPIDOS – E-CER	47
4.1	Módulo: CER_pró-ativa.....	49
4.1.1	Condições para o Correto Funcionamento da CER_pró-ativa	50
4.1.2	Geração de Caminhos Alternativos.....	51
4.1.3	Nível ECMP.....	55
4.1.4	Nível LFA.....	56
4.1.5	Nível SIG.....	57
4.1.6	Exemplificação dos Níveis	58
4.1.7	Formulação Geral da CER_pró-ativa.....	60
4.1.8	Extensão Adicionada na FIB: CER_Marca/IR	64
4.1.9	CER_Signal e CER_pró-ativa-ext	68
4.2	Módulo: CER_EDif	70
4.2.1	CER_EDif: Particularidades em Redes IPv4 e IPv6.....	73
4.2.2	Observações sobre a Execução do CER_EDif.....	75
4.3	Conclusão.....	76
5	IMPLEMENTAÇÃO E AVALIAÇÃO DO E-CER	77
5.1	Introdução	77
5.2	Implementação do módulo CER_pró-ativa.....	77
5.2.1	Algoritmo CER_pró-ativa.....	78
5.2.2	Implementação da CER_pró-ativa em Java	79
5.2.3	Implementação da CER_pró-ativa no J-SIM	80
5.3	Implementação do módulo CER_EDif	81
5.3.1	Algoritmo e Implementação da CER_EDif no J-SIM	82
5.4	Análise Experimental.....	83
5.4.1	Cenário de Testes e Simulação	84
5.4.2	Parâmetros dos Testes e Simulação	85
5.4.3	Análise da Extensão Total dos Caminhos de Recuperação (E-CER x Not-Via).....	86
5.4.4	Análise do Total de Informações Extras Adicionadas na FIB (E-CER x Not-Via)	89
5.4.5	Avaliação do E-CER em Execução no Simulador	92
5.5	Análise Geral do E-CER	95
5.6	Conclusão.....	96
6	CONCLUSÃO E TRABALHOS FUTUROS	97
6.1	Conclusão.....	97
6.2	Trabalhos Futuros.....	101
7	REFERÊNCIAS BIBLIOGRÁFICAS	102

APÊNDICES**8 SIMULADOR JAVASIM (J-SIM)****108****9 TOPOLOGIAS****111**

Lista de Figuras

Figura 2.1 Backbone de um domínio interligando vários fluxos de rede	5
Figura 2.2 Domínio datagrama IP	11
Figura 2.3 Exemplo de uma FIB	12
Figura 2.4 Plano de Controle e Dados do IGP – Estado do Enlace	13
Figura 2.5 Tempo de Recuperação de um nó OSPF.....	17
Figura 2.6 Loop de Roteamento	20
Figura 3.1 Exemplo de um SRLG	26
Figura 3.2 Exemplo LFA.....	31
Figura 3.3 Exemplo de problema com a abordagem FIR.....	33
Figura 3.4 Exemplo de uma geração de camada utilizando MRC.....	35
Figura 3.5 Endereços Not-Via.....	38
Figura 3.6 Exemplo de caminho mais extenso com Not-Via	39
Figura 4.1 Topologia Exemplo para Identificação dos Níveis da CER_pró-ativa.....	59
Figura 4.2 CER_Marca.....	64
Figura 4.3 Adição da CER_Marca/IR na FIB	68
Figura 4.4 Fluxograma do Encaminhamento IP Padrão	70
Figura 4.5 Fluxograma CER_EDif.....	71
Figura 5.1 N° de Saltos Necessários para Alcançar os NDs (Servint, Qwest, AT&T, AT Home)	86
Figura 5.2 N° de Saltos Necessários para Alcançar os NDs (NFSNET, Abilene, GEANT2, AGIS)	87
Figura 5.3 N° de Saltos Necessários para Alcançar os NDs (CAIS, Sprint, BRITE, Level3)	87
Figura 5.4 Exemplo do Uso de Recursos na FIB por Topologia (E-CER x Not-Via)	92
Figura 5.5 Topologia GEANT2.....	92
Figura A.1 Visão geral da componente Nó no J-SIM.....	109
Figura A.2 Interface TCL para o simulador J-SIM	110
Figura B.1 Abilene	111
Figura B.2 AGIS.....	112
Figura B.3 AT Home.....	112
Figura B.4 AT&T	113
Figura B.5 BRITE 32 Nós e 64 Enlaces	113
Figura B.6 CAIS.....	114
Figura B.7 GEANT2	114
Figura B.8 Level 3.....	115

Figura B.9 NFSNET	115
Figura B.10 Qwest.....	116
Figura B.11 Servint	116
Figura B.12 Sprint	117

Lista de Tabelas

Tabela 3.1 Sumário das Abordagens de Reroteamento em Redes IP	44
Tabela 4.1 Notação Utilizada na Descrição dos Níveis e na Formulação Geral.....	52
Tabela 4.2 CAs e CERs para a Topologia Exemplo.....	59
Tabela 5.1 Algoritmo CER_pró-ativa	78
Tabela 5.2 Algoritmo CER_EDif	82
Tabela 5.3 Número Médio de Nós nos Caminhos de Recuperação por Topologia	89
Tabela 5.4 Representação Usada pelas Abordagens E-CER e Not-Via na FIB.....	90
Tabela 5.5 Quantidade de Informações Adicionadas na FIB por Topologia.....	91
Tabela 5.6 % Pacotes Perdidos durante o Período de Convergência (200ms).....	94
Tabela 5.7 % Pacotes Perdidos do Tráfego de Fundo (500ms)	95

Lista de Abreviaturas

E-CER	Esquema de Caminhos Emergências Rápidos
CER	Caminho Emergencial Rápido
FIB	Forwarding Information Base
RIB	Routing Information Base
RED	Random Early Detection
ECMP	Equal Cost MultiPath
LFA	Loop-Free Alternates
SIG	Signal (Multi-Hop)
MRC	Multiple Routing Configuration
MPLS	MultiProtocol Label Switching
QoS	Quality of Service
IGP	Interior Gateway Protocol
BGP	Border Gateway Protocol
AS	Autonomous System
IP	Internet Protocol
IR	Interface de Rede
SPF	Shortest Path First
OSPF	Open Shortest Path First
LSA	Link State Advertisement
BFD	Bidirectional Forwarding Detection
SRLG	Shared Risk Link Group
MTU	Maximum Transfer Unit

Capítulo 1

INTRODUÇÃO

Atualmente, cada vez mais usuários procuram ter acesso à Internet e, em consequência disso, há uma crescente quantidade de informações trafegando nas redes e de novas aplicações que exigem mais qualidade, robustez e confiabilidade na infra-estrutura de rede. Dentre essas novas aplicações, as de uso comercial, negócios e aplicações VoIP são as que mais exigem confiabilidade na infra-estrutura de rede. Entende-se por confiável a infra-estrutura de rede, tanto em *hardware* como *software*, capaz de manter a troca de informações mesmo em casos de falhas ou de períodos de manutenção.

Estudos em topologias de rede reais revelam que falhas ou interrupções durante períodos de manutenção ocorrem freqüentemente em infra-estruturas de rede. Em torno de 80% delas são causadas por falhas transitórias de um único enlace ou roteador em um curto espaço de tempo (não mais que poucos minutos, geralmente por manutenção de *hardware*), e 50% dessas falhas transitórias duram no máximo 1 minuto [IYER et al, 2003].

Uma infra-estrutura de rede é composta de vários roteadores/*switches* que ao todo formam um *backbone*. Um *backbone* geralmente pertence a um domínio de gerência responsável por determinar qual a métrica adotada para identificar uma rota e encaminhar os pacotes dentro do *backbone*. As rotas são definidas pelos *Interior Gateway Protocols* (IGP), que utilizam de métricas para encontrar as rotas e assim distribuir os tráfegos passantes dentro da topologia do *backbone* em que ele atua. Em redes puramente IP os protocolos de roteamento do tipo estado do enlace são predominantes e a principal métrica utilizada para cálculo das rotas é a soma de custos dos enlaces. O principal representante desse tipo de protocolo é o *Open Shortest Path First* (OSPF) versão 2 [MOY, 1998], sendo esse desde sua versão 1 o protocolo recomendado pela IETF [GROSS, 1992] para ser utilizado como IGP de *backbones* IP.

O protocolo OSPF é reconhecido como reativo, pois ao detectar uma mudança na

topologia necessita de um período denominado “convergência” para que todos os roteadores do *backbone* reconheçam a mudança e encontrem novas rotas corretas para se adequar à mesma. Porém, isso acarreta um determinado tempo que implica em instabilidade na rede (*loops* de roteamento) e altas taxas de perdas de pacotes por não ter uma rota correta em tempo hábil, principalmente em *backbones* com taxas de transmissão elevadas. Essa ocorrência afeta diretamente a confiabilidade da infra-estrutura do *backbone* de rede.

Durante esse período de convergência, melhores formas de distribuição de tráfego são necessárias para auxiliar os protocolos de roteamento a reduzir a instabilidade e a alta taxa de pacotes perdidos. Como a grande maioria das falhas é de um único componente de rede [IANNACCONE et al, 2004] [IYER et al, 2003] [MARKOPOULOU et al, 2004] [WATSON et al, 2003], várias pesquisas foram desenvolvidas na área: [WANG and CROWCROFT, 1990], [NARVAEZ, 2000], [FORTZ and THORUP, 2002], [LIU and REDDY, 2004], [NUCCI et al, 2003], [SCHOLLMEIER et al, 2003], [LEE et al, 2004], [ZHONG et al, 2005], [ATLAS and ZININ, 2008], [BRYANT et al, 2008], [KVALBEIN et al, 2006], [ATLAS, 2006] e [BRYAN et al, 2005]. As quatro primeiras abordagens são reativas ([WANG and CROWCROFT, 1990], [NARVAEZ, 2000], [FORTZ and THORUP, 2002] e [LIU and REDDY, 2004]), sendo que as abordagens [NUCCI et al, 2003], [SCHOLLMEIER et al, 2003], [LEE et al, 2004], [ZHONG et al, 2005], [ATLAS and ZININ, 2008], [BRYANT et al, 2008], [KVALBEIN et al, 2006], [ATLAS, 2006] e [BRYAN et al, 2005] são pró-ativas calculando antecipadamente, isto é, antes da ocorrência de falhas existem rotas que possibilitam contornar as falhas. Porém, a abordagem mais promissora observada é a Not-Via [BRYANT et al, 2008] que consegue auxiliar o OSPF em 100% das falhas de um único componente. No entanto, essa abordagem gera caminhos de recuperação para os pacotes mais extensos que o necessário, não considera possíveis congestionamentos causados pelo desvio, não considera a ocorrência de múltiplas falhas independentes o que causa mais instabilidade na rede, utiliza recursos excessivos na tabela de encaminhamento para realizar o desvio dos pacotes e depende da técnica de encapsulamento.

O presente trabalho procura estabelecer uma metodologia dentro do *backbone* disponibilizando uma solução eficiente para auxiliar o protocolo OSPF. Porém, essa abordagem pode ser estendida para os demais protocolos do tipo estado do enlace IGP. O auxílio visa aliviar a ocorrência de instabilidades na rede geradas pela alta taxa de perdas

de pacotes por não ter uma rota correta a tempo e pelos *loops* de roteamento. Essa metodologia resulta em um esquema nomeado *Esquema de Caminhos Emergências Rápidos* (E-CER), que procura previamente disponibilizar caminhos de recuperação para conseguir desviar rapidamente os fluxos de tráfego comprometidos durante o período de reação do OSPF. O E-CER é separado em dois módulos: *CER_Pró-ativa* e *CER_EDif*. A *CER_pró-ativa* é responsável por calcular os *Caminhos Emergenciais Rápidos* e definir uma representação dos mesmos através de uma marca na tabela de encaminhamento. A *CER_EDif* adiciona um recurso ao encaminhamento IP padrão para utilizar essa marca nos pacotes quando houver uma falha adjacente e realizar um encaminhamento baseado na marca quando for necessário. O E-CER busca alcançar uma alta confiabilidade e robustez dentro do *backbone* sem influenciar a estabilidade do protocolo de roteamento previamente configurado.

1.1 Objetivos

O objetivo principal desse trabalho é desenvolver uma metodologia pró-ativa de roteamento nomeada *Esquema de Caminhos Emergenciais Rápidos* (E-CER) para auxiliar os protocolos de roteamento a reduzir a taxa de perdas de pacotes e a instabilidade existente durante o período de convergência. Além disso, o E-CER deve reduzir o caminho de recuperação percorrido pelos pacotes desviados de uma falha até o nó destino em relação à abordagem Not-Via, permitir a detecção de ocorrência de múltiplas falhas independentes para que um novo desvio de pacotes não seja acionado, reduzir a quantidade de informações adicionadas na tabela de encaminhamento dos roteadores em relação ao Not-Via e ser independente do processo encapsulamento para desviar os pacotes da falha.

Como objetivos específicos relacionados:

- Definição de uma abordagem para calcular os caminhos emergenciais reutilizando a base de dados do estado do enlace do OSPF;
- Definição de uma abordagem de troca de mensagens adicional entre os roteadores com OSPF para manutenção dos caminhos emergenciais e a definição de uma modificação no plano de encaminhamento dos roteadores para reconhecer os pacotes quando estes estiverem sendo desviados;
- Avaliação da abordagem proposta com uso de simuladores e da qualidade da

solução gerada comparada à abordagem Not-Via;

1.2 Estrutura da Tese

A estrutura deste documento está organizada em capítulos. O Capítulo 2 descreve as abordagens de roteamento, as estruturas internas dos roteadores e realiza uma descrição detalhada do protocolo OSPF e do período de convergência. O Capítulo 3 realiza um levantamento das falhas que ocorrem em *backbones*, seus problemas e as abordagens existentes para auxiliar as abordagens de roteamento. O Capítulo 4 descreve a proposta E-CER, revelando os seus módulos integrantes. O Capítulo 5 apresenta uma implementação do E-CER e uma avaliação do mesmo. O Capítulo 6 apresenta uma conclusão do trabalho e possíveis trabalhos futuros.

Capítulo 2

ABORDAGENS DE ROTEAMENTO IP

2.1 Introdução

As áreas de interligação das redes de computadores de um mesmo domínio, nomeadas *backbones*, são compostas por vários roteadores/*switches* (nós) interligados nas mais diversas topologias físicas. Essas estruturas têm a capacidade de encaminhar pacotes de dados de um fluxo de rede, utilizando um protocolo roteável, i.e. IP, de um nó qualquer origem até qualquer rede de destino remota que um nó pertence. Os pacotes de dados pertencentes a um fluxo de rede podem conter desde informações de texto até imagens ou sons.

A Figura 2.1 ilustra os nós do *backbone* e as setas indicam os vários fluxos, gerados pelas redes de computadores interconectadas entrando e saindo do *backbone*. Esses fluxos são encaminhados nó a nó no interior do *backbone* entre as redes de origem e destino para que consigam se comunicar.

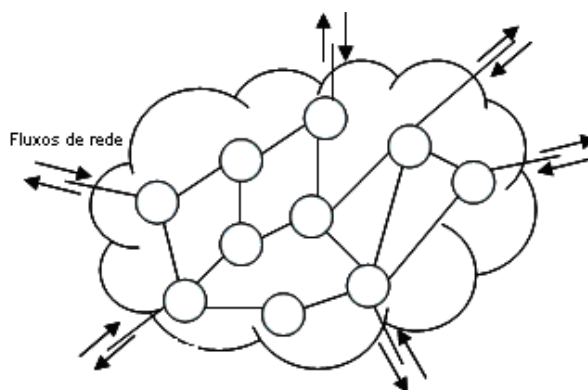


Figura 2.1 Backbone de um domínio interligando vários fluxos de rede

Cada nó do *backbone* realiza essa tarefa de encaminhamento graças às informações

mantidas pelas **rotinas/algoritmos de roteamento**, que se encarregam de encontrar qual a melhor rota para um determinado destino dos pacotes.

Este capítulo apresenta conceitos gerais de roteamento, um detalhamento das rotinas de roteamento e encaminhamento bem como uma separação dessas rotinas em dois planos de execução. A seguir, é apresentada uma conceituação dos protocolos de roteamento em redes IP tendo como foco o funcionamento do protocolo OSPF. Uma descrição dos passos do período de convergência do OSPF são apresentados, bem como os problemas que esse período gera na rede. O capítulo finaliza com as principais soluções que regulam a configuração do OSPF padrão para reduzir o tamanho dos períodos de convergência do OSPF.

2.2 Protocolo de Roteamento em Redes de Interconexão

Em uma rede de interconexão o roteamento é um processo que visa encontrar o melhor caminho entre uma origem e um destino de tráfego da rede. Este melhor caminho, encontrado pelo roteamento, é denominado de rota. O processo de roteamento torna possível aos usuários em partes remotas da rede o acesso a informações e serviços fornecidos pelos demais computadores ligados à rede.

As duas funções principais de um algoritmo de roteamento são [BERTSEKAS and GALLAGER, 1992]: a seleção de rotas entre pares de nó origem e destino e o encaminhamento correto de pacotes para seu destino, uma vez que as rotas foram determinadas. A construção de rotas se traduz na construção de tabelas de roteamento e de encaminhamento. A tabela de roteamento (*Routing Information Base* – RIB) e os algoritmos utilizados na manutenção desta tabela são estabelecidos no **plano de controle**, segundo [BASAK et al, 2002]. A tabela de encaminhamento (*Forwarding Information Base* – FIB) sintetizada a partir da RIB e os algoritmos de controle de admissão, gerenciamento das filas e de encaminhamento dos pacotes fazem parte do **plano de dados** [BASAK et al, 2002].

Tanto a RIB quanto a FIB possuem algumas particularidades nas suas estruturas tanto em redes com tecnologia não orientada à conexão, e.g. IP, quanto em redes orientadas a conexão, e.g. *MultiProtocol Label Switching* (MPLS) [ROSEN et al, 2001]. No entanto, este trabalho apresentará apenas a RIB e FIB referentes à tecnologia IP. A seguir,

os planos de controle e de dados são descritos abordando as diversas formas de algoritmos e de manipulação das tabelas RIB e FIB.

2.2.1 Plano de Controle

O plano de controle está relacionado com a manutenção da RIB e preenchimento da FIB além dos algoritmos de roteamento.

A RIB é uma tabela de roteamento mantida em um nó, que é preenchida com informações de roteamento do próprio nó e referentes aos demais nós da topologia do *backbone*. Essa tabela pode ser preenchida estaticamente pelo administrador de rede ou dinamicamente por um algoritmo de roteamento. Um algoritmo de roteamento define critérios para determinação de rotas a partir dessas informações de roteamento contidas na RIB e preenchem a FIB com uma síntese das rotas obtidas.

Um dos critérios mais simples utilizados pelos algoritmos de roteamento para determinação do melhor caminho entre uma origem e um destino é a contagem do número de *hops* (ou número de passos) existentes entre a origem e o destino. Neste caso, o melhor caminho é o que apresenta menor número de saltos. Esta estratégia pode ser adequada quando todos os enlaces apresentam a mesma taxa de transmissão, porém pode não ser a mais eficiente.

Critérios mais sofisticados de determinação de melhor caminho entre a origem e o destino costumam atribuir custos aos enlaces de rede baseando-se em fatores relacionados ao desempenho do enlace e ao planejamento de rede.

Com base em fatores relacionados ao desempenho do enlace, quanto mais ocioso encontra-se o enlace, menor será o custo atribuído a ele e assim maior será a probabilidade de que este enlace torne-se parte de uma rota para o destino de uma rede. Atribuindo custos menores para enlaces menos congestionados pode possibilitar um melhor uso da infraestrutura de rede reduzindo a perda de pacotes e/ou diminuindo a utilização de pontos críticos da rede.

Com base em planejamento de rede, o administrador de rede define os custos aos enlaces de acordo com um planejamento ou com alguma ferramenta de auxílio à distribuição desses custos a base de heurísticas para melhor distribuir uma matriz estimada de tráfego na topologia.

Os algoritmos também podem ser classificados como estáticos e dinâmicos. No roteamento estático as tabelas de roteamento são incorporadas ao nó apenas pelo

administrador de rede e permanecem imutáveis até uma nova intervenção do administrador. As rotas são fixas e novos caminhos são selecionados somente em caso de falhas de enlaces ou de equipamentos de rede pertencentes à rota fixa. Algoritmos de roteamento estáticos são simples de implementar, mas são vulneráveis a falhas de recurso (necessitam da atuação do administrador de rede), além de poder levar a uma má utilização dos recursos da rede (normalmente, o mesmo conjunto de enlaces/equipamentos é utilizado, enquanto outros caminhos possíveis são desprezados) .

No roteamento dinâmico o próprio nó atualiza via protocolo de roteamento a tabela de roteamento. Um algoritmo de roteamento dinâmico permite atualizar as tabelas de roteamento a fim de retratar mudanças ocorridas na topologia. Quando são capazes de retratar também mudanças da carga de tráfego na rede, com o objetivo de evitar a opção por caminhos mais congestionados, algoritmos de roteamento dinâmico são denominados algoritmos adaptativos.

Nos algoritmos de roteamento dinâmico, a cada alteração de topologia, informações de roteamento são trocadas entre os nós até que ocorra a atualização das entradas nas tabelas de roteamento em todos os nós. O tempo decorrido entre a alteração da topologia/condição da rede até a atualização das tabelas RIB e FIB é denominado de **período de convergência**. O período de convergência é dependente da topologia da rede e da velocidade de processamento dos roteadores, variando de décimos de segundo [ALAETTINOGLU et al, 2000] a vários minutos [BOUTREMANS et al, 2002]. Durante o período de convergência, a inconsistência das informações na RIB e na FIB pode causar instabilidade na rede com destinos inalcançáveis, problema na distribuição de tráfegos, descarte de pacotes no nó adjacente a uma falha (FIB não atualizada acarreta em encaminhamento para o recurso falho) e descarte por *loops* de roteamento [BASU and RIECKE, 2001].

Os algoritmos de roteamento dinâmicos podem ser classificados como baseados no **vetor de distância** ou baseados no **estado do enlace** [TANEMBAUM, 2003].

No roteamento baseado no vetor de distância cada nó envia para cada vizinho uma lista com as distâncias para cada destino. Por conseqüência, cada nó receberá de cada vizinho uma lista de distância para cada destino da rede. Cada nó descobre sua distância para cada destino da rede a partir das seguintes considerações ou do seguinte modo:

- Para nós vizinhos, a distância é previamente conhecida; e
- Para nós não vizinhos, a distância é iniciada com um valor maior que o esperado

(distância infinita) para qualquer destino da rede e com a troca de informações de roteamento acaba se ajustando ao valor de menor distância.

A partir das listas de distâncias recebidas, o nó calcula sua tabela de roteamento, com rotas para cada destino da rede, utilizando os menores valores das listas de distâncias. Apenas nós vizinhos são avisados a respeito de mudanças na distância para qualquer destino em particular. Devido a essa característica, o roteamento baseado no vetor de distância é em geral lento para divulgar as informações a todos os nós da topologia, o que aumenta seu período de convergência e ocasiona vários *loops* de roteamento.

No roteamento baseado no estado do enlace, os nós propagam para todos os demais (e não apenas para os vizinhos) somente as informações de roteamento referentes às rotas diretamente ligadas em suas interfaces (e não todas as informações de roteamento da RIB). Este tipo de roteamento, utilizado pelo protocolo OSPF, em vários *backbones* Internet, leva menos tempo para convergir.

Os algoritmos de roteamento podem determinar rotas baseados em **caminho único** ou em **múltiplos caminhos** [KESHAV, 1997]. No roteamento baseado em caminho único, um nó mantém apenas um caminho para cada destino. No roteamento baseado em múltiplos caminhos, um nó mantém um caminho principal e vários caminhos secundários para um destino. Se o caminho principal estiver indisponível por alguma razão, um caminho secundário é usado.

2.2.2 Plano de Dados

O plano de dados trata de operações realizadas em cada pacote: Encaminhamento dos pacotes, Gerenciamento/Disciplinas de Filas e a Classificação/Escalonamento de tráfego.

A FIB é uma tabela que contém informações sintetizadas dos caminhos calculados pelos algoritmos de roteamento disponíveis na RIB. As rotinas de encaminhamento utilizam a FIB para descobrir qual a rota que os pacotes devem seguir. Quando um pacote adentra em um nó, as rotinas de encaminhamento verificam o endereçamento do pacote e buscam por uma entrada na FIB respectiva ao endereçamento verificado. É a partir dessa entrada que as rotinas de encaminhamento identificam qual interface de rede deve ser usada para encaminhamento e enfileiram o pacote na fila de transmissão dessa interface.

O gerenciamento/disciplina de filas define uma política de gerenciamento de filas. A política de gerenciamento de fila tradicional utilizada pelos roteadores é a *Tail Drop*,

que consiste em descartar os pacotes a partir do momento que se atinge o limite da fila. Entretanto, a política de gerenciamento de fila recomendada pela IETF é a RED [BRADEN et al, 1998], a qual controla o tamanho médio da fila em um período de tempo, caso essa média aumente, a RED aumenta a probabilidade de descarte de pacotes que chegam, mantendo a fila com tamanho suficiente para comportar rajadas de tráfego e prevenindo congestionamentos. Para maiores informações sobre a RED ver [FLOYD and JACOBSON, 1993].

A classificação/escalonamento de tráfego permite um controle sobre os tráfegos passantes de forma a classificar e priorizar tráfegos previamente identificados com Qualidade de Serviço (QoS).

2.3 Protocolos de Roteamento em Redes IP

As infra-estruturas atuais de rede utilizam o protocolo *Internet Protocol* (IP). Uma das características deste protocolo é o seu endereçamento hierárquico. Segundo esta abordagem, a rede é subdivida em um conjunto de redes e nós que trocam informações de roteamento via protocolos de roteamento para interligar essas redes. Um conjunto de redes e roteadores pertencentes a um domínio de gerência de rede e que possuem um protocolo de roteamento comum é denominado *Autonomous System* – AS. Um AS pode ser dividido em áreas que trocam informações de roteamento com outras áreas dentro do mesmo AS através de nós de borda. Os nós de borda são nós que fazem parte dos elementos de uma área e que ainda conectam-se a uma outra área, fazendo parte assim de duas áreas, sendo que toda área deve se conectar à área principal por meio de um nó de borda pertencente a ambas as áreas.

Outra característica do protocolo IP é o fato deste não ser orientado a conexão, ou seja, ser do tipo *datagrama*. Essa característica permite que pacotes pertencentes a um mesmo fluxo sigam por rotas distintas até o seu destino. A especificação do protocolo IP não aborda a ordenação dos pacotes pertencentes a um mesmo fluxo, ficando esta atribuição delegada aos protocolos da camada de transporte [CLARK, 1988].

A Figura 2.2 ilustra um fluxo composto de 3 pacotes IP com um mesmo destino chegando à nuvem *datagrama* IP (utiliza-se no decorrer do texto o termo “redes IP” com o mesmo significado), por um nó de borda que serve de interface com outra rede IP. Na Figura 2.2 os números ilustrativos representam os segmentos da camada de transporte. À

medida que estes pacotes trafegam na rede eles podem seguir por caminhos diferentes chegando ao nó de borda de destino fora de ordem.

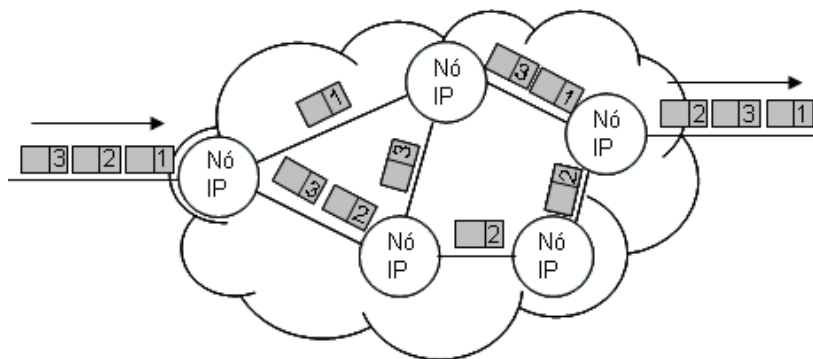


Figura 2.2 Domínio datagrama IP

Segundo a estrutura hierárquica de roteamento, em uma rede IP podem existir dois níveis de comunicação. Um nível que abrange a comunicação interna a um AS e outro externa ao AS. A comunicação interna utiliza-se dos protocolos de roteamento denominados de *Interior Gateway Protocol* (IGP). Por sua vez, os protocolos para comunicação externa, executados nos nós de borda de um AS são denominados de *Exterior Gateway Protocol* (EGP) [TANEMBAUM, 2003].

Como este trabalho está focado para redes IP com protocolos IGP do tipo estado do enlace, o decorrer desse capítulo irá focar esse tipo de protocolo.

2.4 Interior Gateway Protocols

Para realizar a tarefa de encaminhamento, cada nó da rede IP deve manter uma RIB e FIB atualizadas para encaminhar corretamente os pacotes até o seu destino (nó de borda ou *host* de destino).

A FIB em redes IPv4 pode ser representada basicamente pelos seguintes campos [MOY, 1998]: Endereço IP da rede de destino (4 bytes), Máscara de rede (4 bytes) e o Próximo Nó/Interface de Rede (*next-hop*) (4 bytes). A Figura 2.3 ilustra um esquema da FIB em tabela onde cada entrada dita como encaminhar um pacote de acordo com seu endereço IP de destino.

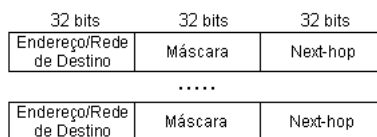


Figura 2.3 Exemplo de uma FIB

A decisão do encaminhamento toma como base o endereço de destino (IP) de cada pacote que entra consultando a FIB para saber qual o próximo nó/interface de saída a ser utilizado para encaminhar o pacote (ver Figura 2.4). Além do endereço IP, outras opções como métricas e distâncias administrativas podem ser utilizadas na decisão de encaminhamento. A cada nó que um pacote atravessa, ele é encaminhado para a interface correta e o cabeçalho do pacote IP é alterado: o campo *Time to Live* (TTL) é decrementado de uma unidade e o valor do *Checksum* do cabeçalho é recalculado.

No plano de controle, as rotinas de roteamento IGP atualmente utilizam-se de protocolos do tipo Estado do Enlace [GROSS, 1992] [TANEMBAUM, 2003]. Cada nó mantém essas informações armazenadas na RIB sob a forma de um grafo direcionado. Para que todos os nós tenham essas mesmas informações, cada nó cria uma tabela local (estado do enlace) basicamente contendo informações dos seus enlaces adjacentes, dos custos de cada enlace (valor associado e atribuído a essa interface) e nós vizinhos (passo (1) da Figura 2.4). Após sua criação, a rotina de roteamento deste roteador distribui sua tabela local para os demais nós do seu domínio IGP através de mensagens especiais de controle (passo (2)). Ao receber uma mensagem de controle, o nó armazena essa informação na RIB (passo (3)) e repassa para os outros roteadores (passo (4)). Desta forma, cada nó consegue montar a mesma base de dados contendo o estado do enlace de todos os nós da topologia em um grafo direcionado. Os vértices do grafo representam os nós e as arestas identificam o valor dos custos dos enlaces, sendo que o custo atribuído a ida de um enlace pode ser diferente do custo de volta. Cada nó monta esse grafo na RIB e através do cálculo dos caminhos mínimos (passo (5)), escolhe-se a menor soma dos custos dos enlaces (métrica utilizada para definir uma rota) do seu nó origem para cada nó destino. O algoritmo mais utilizado é a busca pelo menor caminho, i.e. *Shortest Path First* (SPF), ou algoritmo de Dijkstra [DIJKSTRA, 1959]. Após ter encontrado as rotas para os nós destino, uma síntese das rotas obtidas é preenchida na FIB (passo (6)). A Figura 2.4 ilustra todos esses passos na visão dos planos de dados e controle.

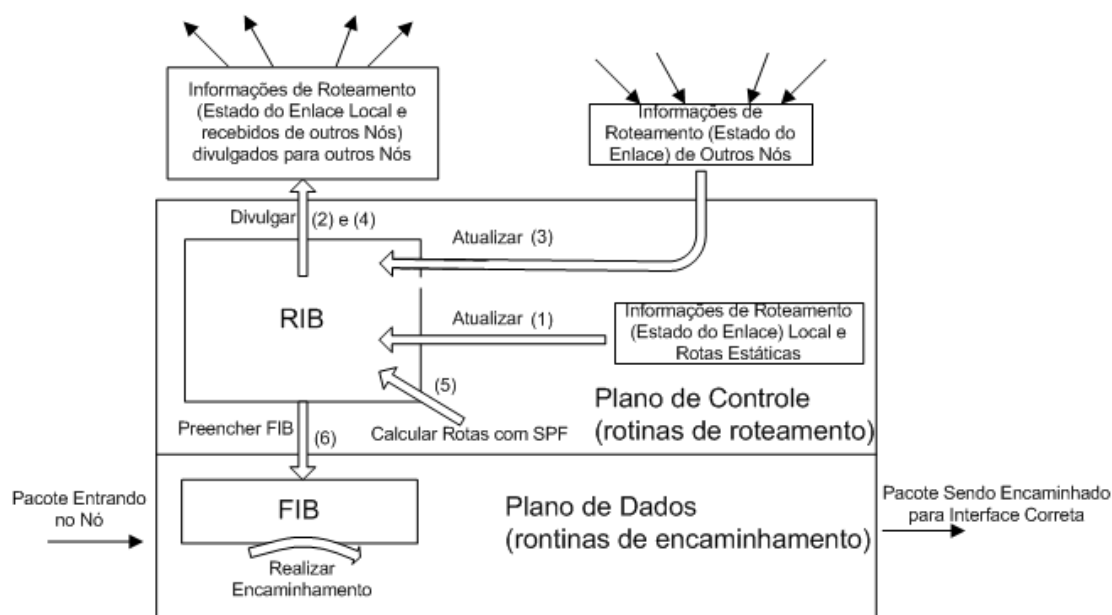


Figura 2.4 Plano de Controle e Dados do IGP – Estado do Enlace

Inicialmente, tentou-se utilizar na Internet rotinas de roteamento adaptativas ao tráfego ou sensíveis ao tráfego em seus protocolos de roteamento IGP de estado do enlace [MCQUILLAN et al, 1980] [KHANNA and ZINKY, 1989] [GLAZER and TROOPER, 1990]. Esses protocolos buscavam adaptar dinamicamente as rotas com base no monitoramento das filas de transmissão do nó. Entretanto, os estudos apresentados em [WANG and CROWCROFT, 1992] revelaram que esta abordagem era adequada em ambientes em que as condições de tráfego são leves e variam de maneira gradual. O aumento de tráfego decorrente do aumento da velocidade da rede e introdução de novas aplicações torna os padrões menos previsíveis, conseqüentemente o comportamento desses algoritmos de roteamento tornava-se instável. Além disso, a escolha da rota era feita apenas com o conhecimento da carga local, sem considerar a carga de tráfego dos enlaces distantes, conseqüentemente os resultados nem sempre contribuía para uma distribuição eficiente do tráfego. Conseqüentemente, o roteamento adaptativo puro torna-se de difícil aplicação prática. Os protocolos atuais utilizam uma abordagem quase estática [WANG and CROWCROFT, 1992], mantendo as métricas constantes por um período de tempo, podendo ser atualizadas caso ocorram mudanças significativas no estado da topologia de rede.

Além de seguir uma abordagem quase estática, os protocolos do tipo IGP de estado do enlace devem possuir algumas características necessárias [BAKER, 1995]:

- Reagir rapidamente às mudanças de topologia da rede;

- Prover rápida convergência sem *loops* de roteamento;
- Utilizar largura de banda mínima na troca de informações;
- Fornecer rotas de mesmo custo para permitir divisão do tráfego;

O protocolo IGP mais utilizado atualmente que obedece a essas características é o protocolo *Open Shortest Path First* (OSPF) [MOY, 1998], além de ser o protocolo IGP recomendado pela *Internet Engineering Task Force* (IETF) [GROSS, 1992] para redes IP.

2.4.1 Open Shortest Path First – OSPF

O OSPF é um protocolo de padrão aberto de estado do enlace muito divulgado na literatura. A versão mais referenciada na literatura é a de número 2 [MOY, 1998]. A sigla SPF desse protocolo indica que é independentemente da política adotada para atribuir custos aos enlaces, o menor caminho utilizado para calcular rotas é a menor soma dos custos dos enlaces. O OSPF possui algumas características, além do exigido em [BAKER, 1995], que o torna flexível o suficiente para se adaptar melhor à topologia em que está instalado [MOY, 1998]:

- O custo atribuído a cada enlace é um valor inteiro de 16 bits, ou seja [1,65535];
- Não há limite no custo máximo de uma rota;
- O administrador da rede é responsável pela definição dos custos dos enlaces;
- A rotina de roteamento permite múltiplos caminhos com *Equal Cost MultiPath* (ECMP), que tem como objetivo balancear carga dividindo igualmente vários fluxos para um mesmo nó destino entre menores caminhos de mesmo custo existentes.

As redes ou *hosts*, que utilizam o *backbone*, estão associadas a alguns nós de acesso (nós de borda) da topologia como um meio de transporte de pacotes entre redes diferentes. Esses nós de acesso são aqueles por onde ocorre a entrada/saída dos diversos fluxos de tráfego originados dessas redes/*hosts* associadas ao *backbone*. Cada rede conectada possui um prefixo de rede que a identifica, ou seja, um prefixo de rede comum a todos os fluxos pertencentes a essa rede. Portanto, um componente importante das informações do estado do enlace desses nós de acesso são os prefixos de rede que têm acesso direto por meio de suas interfaces de rede. Cada nó do *backbone* possui um ou mais prefixos de rede associado a cada enlace configurado, que identificam as redes anexadas. Essas informações

de prefixos de rede, os custos do enlaces, e as demais informações do estado do enlace desse nó são divulgadas pelas rotinas de roteamento do OSPF para todos os nós da topologia através de mensagens de controle nomeadas pacotes *Link State Advertisement* (LSA) [MOY, 1998]. Quando um nó do *backbone* receber um LSA, as rotinas de roteamento usam as informações do LSA para atualizar a *Link-State DataBase* (LSDB) [MOY, 1998] localizada na RIB. Quando esse nó receber todos os LSAs dos demais nós, é possível calcular o melhor caminho SPF para os demais nós do *backbone* (nó destino). As informações dos prefixos de rede são obtidas a partir dos LSAs anunciados de cada nó e já armazenadas na RIB. Portanto, um nó encontra na RIB as rotas para os prefixos de rede anunciados por um nó destino e, em seguida, essas informações são utilizadas para atualizar as entradas da FIB com um prefixo de rede destino, máscara de rede e o próximo nó/interface de rede que deve ser utilizado [MOY, 1998].

Dessa forma, um fluxo de pacotes para um mesmo prefixo de rede de destino tem cada um dos seus pacotes encaminhado com base nas entradas da FIB, de nó a nó, até alcançar o nó de acesso desse prefixo de rede que encaminha os pacotes no enlace respectivo. Como cada nó possui a mesma RIB atualizada, geram-se entradas na FIB desses nós de forma que o caminho seguido pelos pacotes é o menor caminho SPF da origem até o nó destino.

Como a atribuição de custos aos enlaces no domínio OSPF fica a cargo do administrador de rede, isto se torna um ponto de investigação para otimizar a distribuição dos custos dos enlaces. Essa otimização pode adequar uma dada topologia com o objetivo de comportar uma matriz de demanda de tráfego estipulada (fluxos de tráfego origem e destino obtidos a partir da análise e ou previsão de tráfego passante). Um exemplo desta abordagem é apresentado em [FORTZ and THORUP, 2000], onde o processo de otimização calcula os valores dos custos dos enlaces para alcançar o máximo desempenho da rede dado os recursos disponíveis, a matriz de demanda de tráfego e as restrições da capacidade dos enlaces. Porém, o máximo desempenho é sempre aproximado, pois em redes de dados reais existem muitas restrições. Um dos motivos é o cálculo do roteamento ótimo ser inviável na prática, pois os principais protocolos de roteamento, como o OSPF, realizam roteamento apenas pelo menor caminho e não tem suporte à livre distribuição e divisão de fluxos em caminhos de mesmo custo. Portanto, o resultado do roteamento ótimo geral acaba sendo utilizado apenas como um mero patamar ideal de comparação por esse tipo de pesquisas de roteamento.

Outra abordagem heurística simples, recomendada pela Cisco [CISCO, 2005], atribui custos aos enlaces como sendo inversamente proporcionais à largura de banda associada à interface de cada enlace. Quanto maior a largura de banda menor será o custo atribuído, favorecendo o seu uso como menor caminho. A equação 2.1 é a indicada em [CISCO, 2005]. O objetivo do numerador 10^9 é manter o valor inteiro do custo > 0 . Porém, em casos de enlaces mais rápidos esse valor deveria ser aumentado para evitar um número menor que 0 (não é suportado pelo OSPF), porém [CISCO, 2005] não define explicitamente essa possibilidade.

$$\text{custo_enlace} = 10^9 / \text{largura_banda_bps_do_enlace} \quad (2.1)$$

Ao distribuir os custos aos enlaces, há a possibilidade de existir mais de um caminho de mesmo custo entre dois nós. Nesse caso, o nó origem pode utilizar o recurso ECMP, que consiste em distribuir todos os fluxos para um mesmo nó destino em todas as rotas que contenham o mesmo custo de caminho para atingir esse nó. Dessa forma é possível obter um balanceamento de carga [MOY, 1998].

2.4.2 Período de Convergência do OSPF

Quando um nó detecta uma mudança no estado do enlace, e.g. uma falha de enlace/nó (mudança de estado não planejada) ou adição de novo enlace/nó (mudança de estado planejada), o protocolo OSPF adota uma abordagem reativa, ou seja, procura adaptar a configuração da rede às mudanças causadas pelo novo estado do enlace para garantir a redistribuição dos fluxos de tráfego na nova estrutura de topologia. O período decorrido entre a detecção da mudança de estado e a atualização das tabelas RIB e FIB de acordo com a nova topologia corresponde ao *período de convergência* do OSPF.

De forma geral, as etapas executadas por cada nó OSPF durante o período de convergência são: detecção da falha, a modificação e notificação dos novos estados do enlace (LSA), recálculo do SPF na RIB e o preenchimento da FIB [VASSEUR et al, 2004]. O tempo de execução destas etapas no OSPF depende da topologia da rede, da quantidade de nós existentes, da carga de tráfego, da quantidade de prefixos de rede existentes e da velocidade de processamento do nó.

A Figura 2.5 adaptada de [VASSEUR et al, 2004], ilustra esses passos que cada nó OSPF realiza ao detectar uma falha.

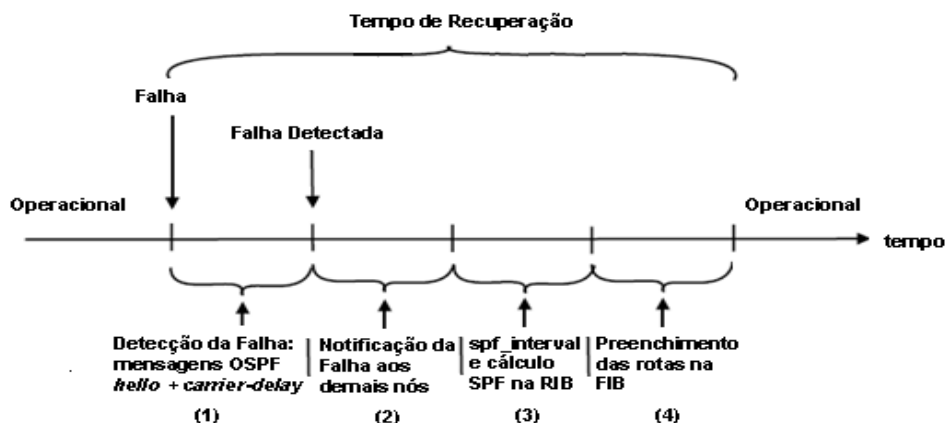


Figura 2.5 Tempo de Recuperação de um nó OSPF

Os nós adjacentes à uma falha normalmente detectam essa falha através de mensagens de controle OSPF ponto a ponto nomeadas *hello* (passo (1) da Figura 2.5). As mensagens do tipo *hello* são geradas localmente por cada nó para seus nós vizinhos a cada 10 segundos [MOY, 1998] (valor padrão, que pode ser alterado para o mínimo a cada 1 segundo [BASU and RIECKE, 2001]). Caso haja *timeout* nessas mensagens *hello* (entre 30 e 60 segundos [IANNACCONE et al, 2004]), as rotinas de roteamento do OSPF interpretam como um evento de falha do enlace adjacente. Alguns *hardwares* possuem temporizadores que evitam várias sinalizações de mudança de estado para rotinas de roteamento (*LinkDown*) causadas por oscilações no enlace. Os temporizadores atuam quando o enlace volta do estado inativo (falha) para o estado ativo, esperando então 10 segundos para sinalizar as rotinas de roteamento [FRANCOIS et al, 2005]. Porém, dependendo do equipamento utilizado, como em equipamentos CISCO, existe um temporizador em software em seu sistema denominado *carrier-delay*, que atrasa o processamento dos sinais de falha do *hardware* (útil quando o *hardware* não possui um temporizador corretamente implementado). A configuração padrão do *carrier-delay* de fábrica é 2 segundos para sinalizar as rotinas de roteamento sobre uma falha, e 12 segundos para sinalizar uma recuperação da falha [IANNACCONE et al, 2002].

Ao detectar e interpretar uma mudança no estado do enlace (falha ou evento *LinkDown*), as rotinas de roteamento atualizam as informações do estado do enlace local na RIB e notificam/propagam essa informação através de mensagens LSAs, sendo ilustrado no passo (2) da Figura 2.5. Essa notificação propaga o LSA a todos os seus nós vizinhos, que atualizam suas RIBs e repassam o LSA para seus próximos nós vizinhos de forma a inundar todos os nós do mesmo AS. O processo de atualizar a RIB e notificar o

LSA aos vizinhos pode levar de 10 a 50 milissegundos por nó [IANNACCONE et al, 2004], sem ainda considerar o atraso de propagação dos enlaces.

As rotinas de roteamento de um nó, ao receber uma mensagem LSA com alteração do estado do enlace, não executam o cálculo SPF imediatamente, pois podem existir outras mensagens LSA na rede, mas ainda não recebidas e processadas pelo nó. Caso esse cálculo fosse aplicado imediatamente, poderia acarretar vários cálculos SPF: um para cada mensagem LSA nova que seja recebida e processada. Por exemplo, para um nó detectar que uma falha seja do nó adjacente, esse nó deve receber as demais mensagens LSA (que indicam falha de enlace) de outros nós vizinhos ao nó falho para então atualizar sua RIB e identificar esse tipo de falha. Apenas quando receber todas as LSAs o nó deveria calcular o SPF, caso contrário haverá várias execuções SPF desnecessárias. Por esse motivo as rotinas de roteamento OSPF não executam o cálculo SPF imediatamente, mas sim após certo intervalo de tempo definido (*spf_interval*, que é definido por padrão em aproximadamente 5,5 segundos [IANNACCONE et al, 2004]) para poder receber os demais LSA. Após esse tempo, as rotinas de roteamento podem seguramente realizar o cálculo SPF apenas uma vez e atualizar a FIB. O tempo de execução do SPF também é um fator a ser considerado, pois necessita realizar o cálculo da árvore SPF inteira (necessita entre 100 e 400 milissegundos [IANNACCONE et al, 2004] dependendo da configuração e tamanho da topologia). Essa etapa é ilustrada no passo (3) da Figura 2.5.

O último passo (4) é o preenchimento/atualização das rotas na FIB. Nesse passo, em equipamentos do *backbone* Tier-1, em torno de 20 entradas são atualizadas na FIB por milissegundo [IANNACCONE et al, 2004]. Uma alteração no estado do enlace que afete milhares de entradas em uma FIB é regularmente encontrada em *backbones* [IANNACCONE et al, 2004]. Portanto, considerando esses valores, uma FIB com 10000 entradas afetadas levaria em torno de 500 milissegundos para ser preenchida.

Dentre todos esses passos, os demais nós da rede (não adjacentes à falha) possuem como tempo de recuperação somente a atualização da RIB com notificação aos demais nós, o *spf-interval* com o cálculo SPF na RIB, e o preenchimento da FIB (passos (2), (3) e (4) da Figura 2.5).

O estudo [IANNACCONE et al, 2002] realizou uma avaliação desses passos no *backbone* Sprint utilizando as configurações padrão do OSPF. Para um evento *LinkDown* detectado em um nó, obteve-se um tempo de convergência para esse nó entre 5,1 e 5,9 segundos (*carrier-delay* + *notificação* + *spf-interval* + *SPF*) e adicionando 1,5 segundos

de atualização na RIB/FIB, resulta em um total entre 6,6 e 7,4 segundos. Para um evento de *LinkUp* detectado, obteve-se um tempo de convergência entre 17,5 e 17,6 segundos (*carrier-delay* + *notificação* + *spf-interval* + *SPF*) e adicionando 1,5 segundos de atualização na RIB/FIB, resulta em um total entre 19,0 e 19,1 segundos. Dependendo das características da topologia e do tráfego utilizados, pode-se atingir até alguns minutos [BOUTREMANS, 2002] se utilizar detecção de falha por *software*.

Quando todos os nós terminarem a reação ao novo estado do enlace, chega-se então ao final do período de convergência do OSPF.

2.4.3 Implicações do Período de Convergência

Durante o período de convergência do OSPF, pode-se gerar um ambiente instável de roteamento devido às informações erradas ainda existentes na RIB/FIB de alguns nós, sendo uma consequência do tempo necessário para a execução de todos esses passos (processo reativo).

Em situações de falha, que são as maiores causadoras de problemas aos protocolos de roteamento [IYER et al, 2003], as informações erradas na RIB/FIB causam descarte de pacotes no nó adjacente à falha por não ter uma rota correta na FIB (a rota antiga continua encaminhando para o componente falho), e também por *loops* de roteamento causados por rotas não atualizadas na FIB dos demais nós da rede.

Para descrever essas duas ocorrências, utiliza-se a Figura 2.6, que ilustra uma topologia com os custos dos enlaces atribuídos. Supondo que os nós *a* e *c* possuem na FIB um registro para encaminhar pacotes destinados ao nó *c* e ao nó *a* respectivamente, e que utilizam o enlace *a-c* e *c-a* (que são o mesmo enlace com custos diferentes no estado do enlace de *a* e de *c*). Com uma falha no enlace *a-c* esse registro permanece nos nós detectores dessa falha (nó *a* e nó *c*) até que a RIB seja atualizada com a nova informação do estado da rede para então atualizar a FIB e conseguir encaminhar os pacotes corretamente para uma nova rota. Durante esse período de tempo, os pacotes que chegam ao nó *a* e ao nó *c* são descartados por ainda ter essa informação errada na FIB (procedimento reativo com tempos necessários: detecção, notificação, *spf-interval*, RIB, cálculo SPF, preenchimento FIB). Como o tempo de atualização da RIB e FIB pode acarretar até várias dezenas de segundos, uma considerável quantidade de pacotes é descartada principalmente na presença de enlaces mais rápidos.

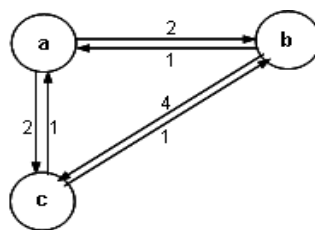


Figura 2.6 Loop de Roteamento

Um *loop* de roteamento é definido quando existem informações não atualizadas nas informações de estado do enlace (RIB e conseqüentemente a FIB) nos demais nós além do nó adjacente à falha. A geração de *loop* de roteamento está diretamente associada ao período de convergência, o qual acarreta várias centenas de milisegundos em um melhor caso, ou normalmente dezenas de segundos conforme visto na seção anterior.

Para exemplificar um *loop* de roteamento, considere ainda a Figura 2.6. O nó *a*, após atualizar sua RIB, deve informar os outros nós sobre a mudança no estado do enlace, sendo que *a* está encaminhando os pacotes destinados ao nó *c* agora para o nó *b* por um caminho de custo 6. Quando *b* recebe as informações para atualizar o estado do enlace, ele deve processar essas informações e atualizar a RIB e FIB (tempos necessários: notificação, *spf-interval*, cálculo SPF, preenchimento FIB). Entretanto, o nó *a* já está enviando os pacotes destinados a *c* via *b* (rota para *c* agora utiliza menor caminho *a-b-c*), e *b* ainda contém informações desatualizadas na RIB e FIB (rota para *c* utiliza ainda o menor caminho *b-a-c*). Portanto, *b* ainda envia os pacotes para *a*, o que ocasiona um *loop* de roteamento. O descarte por *loop* de roteamento ocorre quando o *TTL* do cabeçalho IP dos pacotes atinge o valor zero ou por congestionamentos gerados pelo *loop*.

Caso a falha no enlace *a-c* seja transitória (mesmo considerando o uso de temporizadores), esse problema pode ser agravado ao detectar duas ou mais mudanças no estado do enlace em menos de 1 minuto com dois ou mais períodos de convergência. Com essas mudanças próximas uma da outra, possibilita vários períodos de convergência, e conseqüentemente, mais descartes são realizados até que a rede fique estável.

O período de convergência é necessário ao OSPF, porém, em conseqüência disto, pode-se criar um ambiente com alta taxa de perda de pacotes além de várias redes com destinos temporariamente inalcançáveis. De acordo com os requisitos de qualquer protocolo de roteamento IGP [BAKER, 1995], é desejável que esse tipo ambiente instável seja o mínimo possível. Para tanto, algumas abordagens que visam a redução do período de convergência do OSPF são propostas.

2.4.4 Redução do Período de Convergência do OSPF

Algumas melhorias foram propostas por [IANNACCONE et al, 2004], [FRANCOIS et al, 2005], [ALAETTINOGLU et al, 2000], [KATZ and WARD, 2005] baseados nos estudos feitos sobre os fatores que influenciam o tempo de convergência do OSPF.

2.4.4.1 Detecção de falha: *Hello* e Mecanismos de *Hardware*

Para melhorar o desempenho do processo de detecção de uma falha, pode-se modificar o padrão de intervalo de mensagens *hello*, que está em 10 segundos, para a casa de décimos de segundo [ALAETTINOGLU et al, 2000]. Essa abordagem reduz consideravelmente o período de convergência, entretanto o fato de diminuir muito o intervalo dessas mensagens pode causar instabilidades na rede conforme apontado em [BASU and RIECKE, 2001] e [GOYAL et al, 2003], pois pode aumentar consideravelmente a quantidade de mensagens *hello* geradas, a carga de processamento dos nós, e a ocorrência de oscilações de rotas (causadas por frequentes *timeouts* das mensagens *hello*).

Como a geração de mensagem *hello* do OSPF é um processo pertencente à camada de rede, o seu uso para detecção de falhas de *hardware* possui um considerável *overhead* pelo fato de estar no plano de controle. O trabalho de [KATZ and WARD, 2005] propõe uma solução para esse problema através de um processo de detecção de falha com um protocolo nomeado *Bidirectional Forwarding Detection* (BFD). O BFD é um protocolo simples e rápido de *hello*, que tem como objetivo prover um baixo *overhead* na detecção de falhas independente dos protocolos da camada física e da camada de rede. Seu projeto está voltado para trabalhar no plano de encaminhamento utilizando serviços da camada de enlace diretamente, ou seja, podendo ser implementado diretamente na interface de rede.

Considerando-se os mecanismos de *hardware* avançado atualmente utilizados, como *Synchronous Optical Network* (SONET), *Synchronous Digital Hierarchy* (SDH) e *Gigabit Ethernet*, pode-se obter um processo de detecção mais rápido que o *hello*, i.e. menos que 20 milissegundos para sinalizar as rotinas de roteamento sobre uma falha no enlace [FRANCOIS et al, 2005]. Alguns desses *hardwares* estabelecem um período de contenção para sinalizar as rotinas de roteamento, pois tentam realizar uma correção por parte da camada física quando possível. Caso esse tempo se expire, o *hardware* sinaliza as rotinas de roteamento do OSPF normalmente para realizar a atualização das rotas.

2.4.4.2 Redução do Tempo de Notificação de Falha, *spf-interval* e SPF

A atualização da RIB em conjunto com a notificação de falha (LSAs para os demais nós) pode ser reduzida de 50 milissegundos para valores quase insignificantes (menor que 12 milissegundos), a partir do momento em se opte por divulgar os LSAs (notificação de falha aos demais nós) sempre antes de atualizar a RIB [FRANCOIS et al, 2005].

Outra forma de otimização é a redução do intervalo *spf-interval*. Porém, caso esse tempo seja muito curto, o nó pode executar várias vezes o SPF caso pacotes LSAs não processados cheguem após o intervalo *spf-interval*. Experimentos foram realizados em [FRANCOIS et al, 2005] diminuindo o valor de *spf-interval*, o que obteve sucesso reduzindo consideravelmente o tempo de convergência para falha de enlace. Porém para falha de nó, essa redução acarretou em um tempo maior de convergência (mas ainda menor que 1 segundo), sendo ocasionado por várias ocorrências de cálculo SPF causadas pela chegada de informações LSA de nós distantes na topologia. Ou seja, o cálculo SPF foi acionado antes que todos os pacotes LSA tivessem chegado, necessitando de novos recálculos. Por esse motivo que em vários equipamentos, o *spf-interval* padrão configurado, é de 5.5 segundos para garantir a chegada de todos os pacotes LSA e calcular o SPF.

Outra abordagem é a redução do número de iterações a serem feitas para calcular o SPF. Na presença de falhas, as informações do estado do enlace devem ser atualizadas, entretanto essas atualizações implicam apenas em algumas modificações no estado do enlace em relação à base anterior armazenada. Nesse caso, é possível realizar um cálculo diferenciado sobre a estrutura SPF antiga já calculada, nomeada *Incremental-SPF*. As pequenas alterações de falhas de enlaces ou nós que ocorrem influenciam uma pequena parte da árvore SPF [MCQUILLAN et al, 1978] [NARVAEZ, 2000], podendo, nesse caso, diminuir consideravelmente o tempo de cálculo para dezenas de milissegundo [FRANCOIS et al, 2005], sendo que esse tempo é diretamente dependente da quantidade de nós na rede. Essa abordagem, apesar de ser antiga, não era adotada nos equipamentos de rede devido à complexidade existente nesse algoritmo [IANNACCONE et al, 2004]. Entretanto, com o avanço nas tecnologias de memória e dos processadores, os equipamentos de rede modernos tornam a adoção dessa abordagem mais viável. Para maiores informações sobre o *Incremental-SPF*, ver em [MCQUILLAN et al, 1978] e [NARVAEZ, 2000].

2.4.4.3 Atualização da FIB

A redução do tempo de atualização da FIB é dependente da arquitetura do *software* e *hardware* utilizado nos nós de rede. O levantamento feito por [IANNACCONI et al, 2004] aponta que o hardware do *backbone* Sprint pode atualizar 20 entradas na FIB em 1 milissegundo aproximadamente. Outro trabalho estimou que, utilizando o *hardware* de roteador *Percentile-90*, o tempo de atualização de uma entrada na RIB e na FIB (soma desses dois tempos) pode alcançar em torno de 148 microssegundos [FRANCOIS et al, 2005]. Nesse último caso, supondo uma RIB e FIB com 5000 entradas, seriam necessários aproximadamente 740 milissegundos.

2.4.5 Viabilidade do Período de Convergência ser < 1s

Em uma simulação do período de convergência, dependendo da topologia e da qualidade dos equipamentos de rede, pode-se alcançar um tempo de convergência total (*carrier-delay* + *detecção* + *notificação* + *spf-interval* + *Incremental-SPF* + RIB/FIB) em torno de várias centenas de milissegundos [FRANCOIS et al, 2005]. Esse tempo reduzido apresentou resultados positivos para falha de enlace. Porém para falhas de nó a solução não se apresentou estável, pois dependendo da topologia, pode-se necessitar de um tempo adicional para receber todos os LSA dos nós adjacentes ao nó falho (*spf-interval*). Nesse caso, houve vários recálculos da árvore SPF devido ao tempo muito reduzido para receber todos os LSAs.

Por fim, observa-se que a avaliação [FRANCOIS et al, 2005] simula a redução do período de convergência para < 1 segundo, e que teoricamente torna-se possível em alguns ambientes ideais, porém pode ser ineficiente em caso de falhas de nó. O trabalho [FRANCOIS et al, 2005] ainda não considerou a ação dos temporizadores em ocasiões de falhas transitórias (~10 segundos) além de restringir várias configurações nos nós.

De forma geral, as soluções adotadas pelos protocolos de roteamento IP conseguem se adaptar a qualquer mudança na estrutura da rede, atuando reativamente para manter o objetivo da camada de rede do modelo Internet: conseguir se adaptar às falhas na estrutura da rede. Entretanto, considerando um tempo de convergência reduzido (várias centenas de milissegundos), ainda pode existir uma grande quantidade de pacotes perdidos principalmente em enlaces de mais alta velocidade e uma possibilidade de oscilações de rotas [BASU and RIECKE, 2001]. Mesmo com o tempo de convergência < 1 segundo, se um enlace OC-192 (9.6 Gbps) estiver em um estado de falha por 500 milissegundos e

assumindo um tamanho de pacotes de 1024 bytes, são descartados em torno de 500 mil pacotes devido a informações de rotas erradas apenas no nó detector de falha. Existe ainda a taxa de perdas que pode ocorrer nos demais nós por *loops* de roteamento, o que aumenta a instabilidade na rede.

2.5 Conclusão

Esse capítulo mostra em detalhes as características principais das arquiteturas dos nós de redes de interconexão de forma geral e voltados para a tecnologia IP. Descreve os algoritmos de roteamento tendo como foco o protocolo OSPF para o desenvolvimento desse trabalho. Aponta-se o período de convergência, detalhando as implicações do mesmo em específico no OSPF. Verificou-se a possibilidade da redução desse período melhorando as configurações de tempo dos passos necessários à convergência do OSPF, podendo atingir a casa de várias centenas de milissegundos. Entretanto, essas características foram realizadas em ambientes ideais com *hardwares* avançados e, mesmo assim, a reação do OSPF pode ainda ocasionar altas taxas de pacotes perdidos em enlaces mais rápidos mesmo com um período de convergência reduzido.

Capítulo 3

FALHAS EM BACKBONES IP E SOLUÇÕES EXISTENTES

3.1 Introdução

As falhas de componentes de rede (de enlaces e ou de nós) são ocorrências que surgem no dia a dia de operação dos *backbones*, sendo essa a maior causa dos problemas de congestionamentos em redes IP [IYER et al, 2003] [IANNACCONE et al, 2004].

Os protocolos de roteamento IGP de *backbones* devem reagir à presença dessas falhas para reacomodar os tráfegos passantes na topologia remanescente. Essa acomodação é feita na medida do possível, pois não se pode garantir que todo o tráfego passante seja acomodado sem a existência de perdas.

Este capítulo apresenta as características das falhas existentes em *backbones* IP tratadas pelas rotinas de roteamento do OSPF. São detalhadas também as principais abordagens alternativas que auxiliam as rotinas de roteamento do OSPF e de encaminhamento a contornar os problemas do período de convergência. O capítulo finaliza com uma discussão dessas soluções alternativas e com a identificação das abordagens com mais chances de sucesso.

3.2 Características das Falhas em Backbones IP

As falhas transitórias de componentes de rede nos *backbones* podem ser caracterizadas em curtas ou longas. As falhas transitórias curtas são decorrentes da manutenção planejada da rede e de ocorrências de eventos não previstos. As falhas ocasionadas pela atualização de software (*firmware*) dos roteadores, troca de interfaces de rede e de conectores de rede são exemplos de falhas curtas geradas pela manutenção

planejada da rede. As falhas transitórias curtas não previstas podem ser causadas pelo mau contato nos componentes da rede, interferência eletromagnética nos sinais ou falta de energia elétrica. Em geral, a duração dessas falhas é da ordem de poucos segundos a minutos.

As falhas transitórias longas são decorrentes de problemas de funcionamento de *hardware*, rompimento de enlaces e ocorrência de desastres naturais. A duração destas falhas é da ordem de horas ou até mesmo de meses. Observe que em algumas situações as falhas transitórias longas podem ser tratadas como falhas permanentes.

Outra característica das falhas transitórias nos *backbones* é que em um período curto de tempo a sua ocorrência é única, existindo uma baixa probabilidade de ocorrência de múltiplas falhas independentes [VASSEUR et al, 2004] [NUCCI et al, 2003].

Alguns trabalhos também abordam a ocorrência de múltiplas falhas dependentes em enlaces. Essas falhas que atingem um conjunto de enlaces, denominados de *Shared Risk Link Group* (SRLG), são decorrentes de uma única falha. Um duto com várias fibras óticas é um exemplo típico de SRLG. Nesta situação, a ocorrência de uma falha no duto faz com que todas as rotas (enlaces) sejam interrompidas.

A Figura 3.1 ilustra os enlaces *a-b* e *d-c* utilizando um mesmo duto físico e, portanto pertencem a um mesmo SRLG. Caso ocorra um rompimento desse duto, ambos os enlaces *a-b* e *d-c* irão apresentar falha em conjunto, ou seja, ao mesmo tempo.

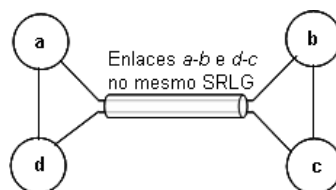


Figura 3.1 Exemplo de um SRLG

As rotinas de roteamento IGP identificam quais enlaces pertencem a qual(is) SRLG(s) através da troca de mensagens LSA. Algumas extensões ao OSPF, como a proposta em [KOMPELLA and REKHTER, 2005], permitem identificar nessas mensagens LSA quais enlaces pertencem a um SRLG mantendo informações suplementares na RIB. Através desta identificação as rotinas de roteamento podem reagir adequadamente a uma falha de enlace pertencente a um SRLG, uma vez que consegue calcular caminhos que contornem os demais enlaces pertencentes a esse SRLG.

Estudos empíricos realizados no *backbone Sprint*, apresentado em [IYER et al,

2003], comprovam a ocorrência de falhas transitórias de curto período de tempo. Esse estudo detecta a ocorrência de sobrecargas na rede (congestionamento), e em aproximadamente 80% dos casos, é decorrente de falhas transitórias de um único enlace ou nó, sendo que em 50% desses casos as falhas duram menos de um minuto. Um fator importante apresentado nesse estudo é que para acomodar o tráfego da rede após a ocorrência de falhas, os enlaces são contingenciados em 50% de sua capacidade durante o funcionamento normal.

Os resultados apresentados em [IANNACCONE et al, 2004] e [MARKOPOULOU et al, 2004] obtidos no mesmo *backbone Sprint* reforçam a caracterização de falhas obtida por [IYER et al, 2003]. Esse estudo concluiu também que a duração das falhas é variável sendo que 46% têm duração inferior a 1 minuto e 81% duram menos de 10 minutos. Outra conclusão importante é que 20% das falhas são decorrentes de manutenção planejada e afetam apenas um componente da rede. Em torno de 70% do restante das falhas observadas (56% do total) são decorrentes de falhas de um único enlace. Os outros 30% das falhas observadas (24% do total) são decorrentes de falhas de um único nó ou de SRLG.

Estudos semelhantes realizados durante o período de um ano no *backbone Michigan-Net* concluíram que 80% das falhas observadas afetam um único componente de rede e são transitórias com duração máxima de 5 minutos [WATSON et al, 2003].

3.3 Abordagens Reativas

Uma das primeiras abordagens reativas para tratar falhas transitórias de hardware e de congestionamento foi apresentada em [WANG and CROWCROFT, 1990]. Esta proposta propôs o algoritmo *SPF-Emergency Exit* (SPF-EE), que é uma extensão do algoritmo SPF padrão. Nessa abordagem, quando da ocorrência de congestionamento ou falhas de equipamentos, a rotina de roteamento do nó, ao invés de iniciar a atualização das tabelas de roteamento, encaminha temporariamente o tráfego por caminhos alternativos. Estes caminhos alternativos são escolhidos dinamicamente a partir da troca de informações de controle entre nós vizinhos. O principal requisito da escolha do caminho alternativo, além de localizar o próximo nó vizinho que tenha um caminho alternativo, é evitar *loops* de roteamento. Caso exista um caminho sem *loop* de roteamento, este nó é denominado "nó emergencial de saída". Caso não exista caminho sem *loop* de roteamento o algoritmo gera pacotes de controle para que seus vizinhos encontrem um "nó emergencial de saída"

adjacente a eles de forma a eliminar a existência de *loops*. Ao encontrá-lo, um *Reverse Alternate Path* (RAP) é estabelecido entre o "nó emergencial de saída" encontrado até o nó origem que iniciou a busca. Com o RAP definido, esse nó origem pode então desviar os pacotes pelo caminho definido. Caso um caminho sem *loop* ainda não seja encontrado o algoritmo tenta encontrar um caminho sem *loop* em nós mais distantes. Este procedimento é repetido até que um caminho sem *loop* seja encontrado ou as atualizações de roteamento global atualizem as tabelas e a manipulação de falhas/congestionamentos é desativada. Neste caso as entradas das RAPs são removidas das tabelas de roteamento. Na avaliação efetuada pelos autores os resultados foram considerados satisfatórios em termos de funcionalidade da abordagem (desvio do tráfego a partir de falhas transitórias/congestionamento). Entretanto, a abordagem gera um significativo tráfego adicional para encontrar e estabelecer um RAP. Além de ser lento em relação à reação global do OSPF, característica esta comum a toda abordagem reativa.

Uma outra abordagem reativa foi proposta em [NARVAEZ, 2000]. Nesta abordagem, o método proposto, denominado de "Algoritmo de Vetor-Métrica de re-convergência local", é executado a partir de uma detecção de falha de enlace. A proposta especifica que o algoritmo, ao calcular um caminho de recuperação capaz de contornar essa falha de enlace, notifica com mensagens as mudanças do estado de enlace referente à falha apenas aos nós pertencentes a esse caminho. Essa notificação é constituída também por um vetor cujos campos correspondem aos custos dos enlaces que devem ser utilizados pelos nós desse caminho. Esses nós por sua vez atualizam a RIB e FIB. Ao contrário da rotina de recuperação padrão do OSPF que propaga a informação da falha a todos os nós da topologia, esta abordagem propaga as informações das falhas somente aos nós do caminho de recuperação; reduzindo assim o período de convergência do algoritmo. No entanto, ela ainda é reativa e as falhas de nó ou SRLG não são suportadas nesta abordagem.

Outra abordagem para tratar com falhas de enlace foi apresentada em [FORTZ and THORUP, 2002]. A abordagem utiliza heurísticas de *busca local* em que os valores de custos dos enlaces são previamente distribuídos na topologia para diferenciar as variações na matriz de tráfego. Além disso, essa distribuição de custos evita modificar consideravelmente o estado do enlace durante a ocorrência de falha para que o período de convergência possa ser reduzido. Apesar de distribuir previamente os custos dos enlaces, essa abordagem depende de alterações no estado da rede o que resulta na solução reativa do OSPF com um tempo de convergência para contornar a falha. Além disso, a distribuição

de custos visa somente falhas de enlace.

A proposta apresentada em [LIU and REDDY, 2004] possui uma abordagem semelhante ao SPF-EE, e estende os protocolos de estado do enlace com o mecanismo *Fast Rerouting*. Esse mecanismo consiste na localização de um nó denominado *Feasible Next Hop* (FNH) e no encaminhamento de tráfego para esse nó quando ocorrer uma falha. Todo esse processo é feito antes da atualização da tabela de roteamento. O FNH possui como característica um menor caminho até o nó destino do tráfego capaz de contornar a falha considerada. Se o FNH não for o nó vizinho o mecanismo estabelece dinamicamente um caminho (*Rerouting Path*) até um FNH remoto para que consiga encaminhar os pacotes até ele. O fato do estabelecimento *Rerouting Path* também ser reativo limita a reação rápida à falha. Além de ser reativa, essa abordagem suporta apenas falha de enlace.

3.4 Abordagens Pró-Ativas

A principal característica das abordagens pró-ativas é disponibilizar um caminho antecipadamente que será utilizado no caso de falhas de um componente na topologia. Algumas abordagens se encaixam no padrão, denominado *IP Fast ReRouting* (IPFRR) [SHAND and BRYANT, 2008], definido pela IETF. Esse padrão é um *framework* teórico definido pelo grupo de trabalho *Routing Area Working Group* (RTGWG) cujos trabalhos incluem a área de roteamento e tratamento de ocorrências de falhas em redes IP. O mecanismo proposto pelo padrão IPFRR para tratar com falhas é bastante similar com a abordagem local de proteção antecipada de falhas voltadas para o domínio MPLS, que é nomeado *MPLS Fast-Rerouting* [PETERSSON, 2005] [SHARMA and HELLSTRAND, 2003] [PAN et al, 2005]. O padrão IPFRR recomenda que os mecanismos de recuperação de falhas utilizem caminhos pré-calculados, ou seja, antes da falha ocorrer, seguindo a ordem de prioridade: *Equal Cost Multi-Paths* (ECMP), *Loop-Free Alternates* (LFA) e *Multi-Hop* [SHAND and BRYANT, 2008]. O *framework* recomenda o uso do *Multi-Hop* apenas quando ECMP ou LFA, nessa ordem, não for aplicável.

Entre as abordagens de prioridade *Multi-Hop* citadas no *framework* está a *Failure Insensitive Routing* (FIR) [LEE et al, 2004], *Multiple Routing Configuration* (MRC) [KVALBEIN et al, 2006] e Not-Via [BRYANT et al, 2008]. A principal restrição das abordagens que seguem o padrão IPFRR é que elas são específicas para redes IP e são mais difíceis de serem implementadas devido às características do menor caminho usado no

roteamento IP.

3.4.1 Abordagem *Equal Cost Multi-Path* – ECMP

A abordagem *Equal Cost Multi-Paths* (ECMP) basicamente consiste em reutilizar a abordagem ECMP já disponível no padrão OSPF [MOY, 1998] descrita no capítulo anterior. Essa solução é por natureza pró-ativa e, na presença de uma falha adjacente (enlace, nó ou SRLG), um nó deve utilizar um ou mais caminhos já existentes de mesmo custo para um nó destino. Esses caminhos não devem possuir nas suas trajetórias a falha adjacente para que possam contorná-la.

Apesar de ser simples, essa abordagem não garante 100% de cobertura, pois é dependente da distribuição de custos aos enlaces, da topologia utilizada. Isto ocorre pois não há garantias de que sempre haverá caminhos de mesmo custo para cada nó destino a partir de um nó origem e que esses caminhos consigam sempre contornar uma falha adjacente.

3.4.2 Abordagem *Loop-Free Alternates* – LFA

A abordagem *Loop-Free Alternates* (LFA) [ATLAS and ZININ, 2008] é uma solução pró-ativa para um nó contornar um componente adjacente (enlace, nó ou SRLG) quando este estiver falho. A abordagem consiste na identificação de um nó vizinho, denominado nó vizinho de *backup*, que possua um caminho OSPF capaz de contornar esse componente adjacente. Esse nó vizinho de *backup* não pertence ao menor caminho OSPF para o endereço de destino dos pacotes. No entanto, se houver o desvio dos pacotes para esse nó vizinho de *backup*, eles devem ser encaminhados por um menor caminho OSPF, a partir desse nó, que seguramente contorne o componente adjacente.

Quando houver uma falha, o processo de encaminhamento deve utilizar o caminho alternativo pré-calculado pelo LFA que é representado por uma entrada na FIB, a qual permite desviar os pacotes para esse nó de vizinho *backup*.

Para utilizar essa abordagem, cada nó se considera nó origem e a partir da possibilidade de falha em cada um dos enlaces/nós adjacentes, realiza verificações para identificar quais nós vizinhos de *backup* podem ser utilizados. Para calcular os caminhos alternativos para contorno de falha, a abordagem LFA apenas é aplicável se as seguintes regras são obedecidas:

1. Distância(nó *backup*, nó destino) < (Distância(nó *backup*, nó origem) +

Distância(nó origem, nó destino))

2. $\text{Distância}(\text{nó } backup, \text{nó destino}) < (\text{Distância}(\text{nó } backup, \text{nó falho}) + \text{Distância}(\text{nó falho, nó destino}))$

Para que o nó origem consiga obter as distâncias do nó vizinho de *backup* e do nó falho adjacente nessas regras, o nó origem realiza cálculos SPF considerando cada um desses nós como a raiz da árvore de menor caminho.

A primeira regra elimina a existência de *loops* no caminho alternativo. Por outro lado, a segunda regra evita que o nó faltoso faça parte do caminho alternativo a partir do nó vizinho de *backup*. O uso das duas regras permite tratar falha do nó adjacente, enquanto que utilizar apenas a primeira regra restringe a solução para tratar falha de enlace. O uso apenas da primeira regra somente é adotado quando o nó destino for o nó adjacente, uma vez que não se pode contornar o nó adjacente por ser o nó destino (o que seria impedido pela segunda regra). Observe que a hipótese de falha de nó é mais abrangente e é usada para os demais nós destino, pois implicitamente é considerada a falha do enlace no cálculo do novo caminho. A abordagem também considera a presença de SRLG no cálculo das distâncias. Se durante o cálculo existir mais de um caminho alternativo para um nó destino, escolhe-se indistintamente um deles.

Considerando a configuração de *backbone* composta por 4 nós e 5 enlaces com custos assimétricos ilustrada na Figura 3.2 e a hipótese de falha no nó *d*, para o nó *b*, o nó *a* é o nó vizinho de *backup* quando da necessidade de encaminhar pacotes para o próprio nó *a* e o nó *c*.

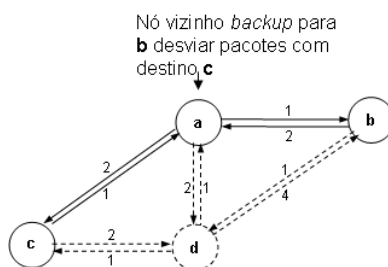


Figura 3.2 Exemplo LFA

Estudos realizados concluíram que as abordagens ECMP e LFA não garantem a cobertura de todas as falhas possíveis. Em alguns casos, a cobertura com essas duas abordagens pode chegar em até 80% das falhas únicas possíveis. Esta restrição ocorre devido à influência da topologia e da distribuição de custos aos enlaces no cálculo dos

caminhos alternativos. A cobertura completa somente é alcançada através de uma abordagem *Multi-Hop* [SHAND and BRYANT, 2008].

Apesar das abordagens ECMP e LFA não serem projetadas para tratarem múltiplas falhas independentes, a sua presença pode causar grande instabilidade com essas abordagens pelo fato de não ser identificada [ATLAS and ZININ, 2008] [SHAND and BRYANT 2008].

3.4.3 Abordagem *Failure Insensitive Routing* – FIR

Outra abordagem pró-ativa, denominada de *Failure Insensitive Routing* (FIR) foi apresentada em [LEE et al, 2004]. Nesta abordagem, cada nó cria uma tabela, denominada de *backwarding table*, com rotas alternativas para cada interface de rede. Estas rotas são previamente calculadas a partir da dedução, ou identificação, de falhas em enlaces no caminho original até o destino se existisse a chegada de pacotes por uma interface de rede não usual. Essa interface de rede normalmente seria usada para encaminhar e não para receber pacotes em condições normais. O nó previamente identifica por dedução quais enlaces de um caminho original poderiam estar com falha até o nó destino a partir do nó adjacente a sua interface de rede. Esse nó adjacente, nestas condições, redirecionaria o tráfego de volta para essa interface de rede. Esses enlaces deduzidos são então marcados como inutilizáveis. A partir deste momento calcula-se uma rota alternativa que não utilize esses enlaces para alcançar o nó destino e a disponibiliza na *backwarding table* da interface de rede em questão. Com as rotas disponíveis nessa interface, se existir a chegada real de pacotes que normalmente não chegaria nessa interface então essa tabela de rotas é usada para desviar esses pacotes.

Essa solução é apropriada para contornar falhas de um único enlace. Se ocorrer falha de nó pode gerar *loops* de roteamento. O exemplo da Figura 3.3 ilustra a ocorrência de um *loop* [ZHONG et al, 2005]. Suponha a existência de um fluxo de pacotes do nó 1 até o nó 6 pelo caminho 1-2-5-6. No caso de falha do nó 5, o tráfego do nó 2 com destino ao nó 6 é transferido para o nó 1. O nó 1 começa a receber pacotes do nó 2 destinados a 6 por uma interface não usual. Com as rotas alternativas já disponíveis nessa interface, o nó 1 usa a rota alternativa para os pacotes destinados a 6 que foi gerada com a dedução de falha no enlace 2-5. Essa rota encaminha os pacotes destinados ao nó 6 para o nó 3. Nesta situação, o tráfego direcionado para o nó 6 é encaminhado novamente para o nó 1, pois o nó 5 está falho o que gera o *loop* de roteamento 1-2-1-3-1-2-1-3... até terminar o TTL dos pacotes.

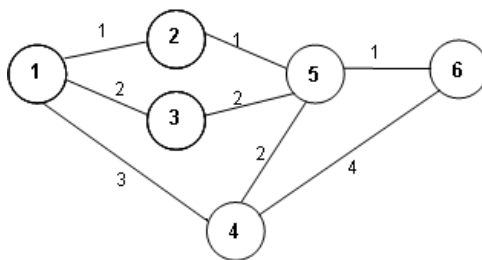


Figura 3.3 Exemplo de problema com a abordagem FIR

Este *loop* ocorre porque o nó 1 deduz os prováveis enlaces falhos de forma pró-ativa, e não dinamicamente, o que possibilitaria deduzir a falha também do enlace 3-5 após a chegada do pacote vindo do nó 3.

A abordagem *Failure Insensitive Routing Fast Rerouting* (FIFR) [ZHONG et al, 2005] aprimora a abordagem FIR, estendendo a solução também para casos de falhas de nó. Entretanto, existe a restrição de que a falha de enlace conectado ao último nó do caminho não pode ser solucionada [KVALBEIN et al, 2006]. Em consequência da criação da *backwarding table* por interface de rede as abordagens FIR e FIFR aumentam as informações de estado do enlace, além de existir a possibilidade do uso de caminhos mais longos a partir da dedução equivocada dos enlaces com falha. Outra restrição das abordagens FIR e FIFR é que as falhas de SRLG e o uso de enlaces com custos assimétricos não são suportadas.

3.4.4 Abordagem *Multiple Routing Configuration* – MRC

A abordagem MRC é uma adaptação da abordagem *Resilient Routing Layer* (RRL) [KVALBEIN et al, 2005]. A abordagem RRL consiste em gerar subconjuntos da topologia de rede original para redirecionar os tráfegos desviados em caso de falha. Esses subconjuntos são chamados de *layers* (camadas), e são gerados adotando estratégias de retirada dos enlaces da topologia de forma que cada camada contenha um caminho que interconecte todos os nós na camada gerada (após a exclusão dos enlaces). O fato de retirar um determinado enlace para compor uma camada indica que a mesma dispõe de um caminho que contorne a falha desse enlace, ou seja, isola esse enlace. As camadas são geradas por todos os nós da topologia de maneira que os tráfegos são desviados quando necessários.

A identificação de quais enlaces devem ser tirados da topologia para criar as camadas pode ser feito manualmente pelo administrador de rede ou por uma abordagem

automatizada, o que é interessante quando o tamanho da rede aumenta. [KVALBEIN et al, 2005] propuseram três estratégias para gerar as camadas de forma automatizada: (i) gerar poucas camadas; (ii) preservar a maior quantidade de enlaces em cada camada para obter um melhor caminho alternativo possível e; (iii) isolar a maior quantidade de enlaces em cada camada e ter mais chances de tratar múltiplas falhas de enlaces.

O ideal para se obter os melhores caminhos de recuperação é utilizar muitas camadas, pois se consegue isolar poucos enlaces por camada o que aumenta a possibilidade de obter melhores caminhos para contornar uma falha. Entretanto, o uso de muitas camadas demanda uma quantidade maior de recursos o que muitas vezes é inviável, pois para cada camada utilizada deve existir uma imagem da topologia na RIB e uma representação em separado na FIB. Esta é a justificativa para que poucas camadas fossem utilizadas [KVALBEIN et al, 2005].

Em caso de detecção de uma falha de enlace, o nó detector encaminha o tráfego para uma camada que consiga isolar o enlace comprometido e atingir o nó destino. O tráfego então é marcado para ser encaminhado pelo menor caminho seguindo a topologia dessa camada, de forma que essa marcação possa instruir os outros nós a encaminhar esses pacotes com a camada identificada.

A abordagem *Multiple Routing Configurations* (MRC) [KVALBEIN et al, 2006] adapta a abordagem RRL, fornecendo proteção de falha única de enlace ou de nó, para ser utilizada especificamente para protocolos de roteamento do estado do enlace. Essa adaptação da RRL não retira enlaces da topologia para isolá-los ao criar as camadas, ao invés disso, atribui custos altos em alguns enlaces para isolá-los e com isso estes enlaces são evitados no cálculo SPF. Conforme a abordagem RRL, a MRC procura também gerar poucas topologias com o objetivo de reduzir a quantidade de recursos usados para representá-las nos nós.

Um algoritmo pró-ativo foi desenvolvido para criar várias camadas a partir de uma topologia e atua em um laço percorrendo todos os nós até conseguir isolar cada um deles. O algoritmo segue algumas restrições estabelecidas [KVALBEIN et al, 2006]:

- Um nó não deve comportar qualquer tráfego passante na camada em que ele está isolado, e ainda esse nó deve ser capaz de gerar tráfego para outros nós e de receber tráfego destinado a esse nó.
- Um enlace não deve comportar qualquer tráfego em uma configuração de camada na qual ele esteja isolado.

- Em cada configuração, todos os pares de nós devem estar conectados por um caminho que não atravesse um nó isolado ou enlace isolado.
- Cada nó e cada enlace devem ser isolados em pelo menos uma das configurações de camada.

Na primeira restrição, para isolar um nó, o algoritmo atua sobre os enlaces que são adjacentes a ele especificando custos altos (não infinitos), o que restringe o uso desses enlaces apenas para tráfegos originados de ou destinados a esse nó. Para ter essa garantia, os custos altos atribuídos aos enlaces adjacentes desse nó devem cada um possuir um valor maior que a soma de todos os custos dos demais enlaces existentes na topologia.

Na segunda restrição, para realizar o isolamento total de um enlace, a camada deve ser configurada com uma atribuição de custo infinito para esse enlace impedindo a existência de uma rota que o utilize.

A terceira restrição é satisfeita atribuindo aos enlaces adjacentes ao nó isolado, quando possível, custo infinito, mantendo sempre a conectividade de todos os nós da camada.

A quarta restrição é satisfeita isolando cada nó e cada enlace em exatamente uma camada, satisfazendo também as três restrições anteriores.

O exemplo da Figura 3.4 apresentado em [KVALBEIN et al, 2006] é utilizado para explicitar o funcionamento do algoritmo. Nessa figura, os enlaces tracejados correspondem a custos altos e os pontilhados custos infinitos. Em *a*) o nó 5 é isolado pelo algoritmo atribuindo um peso alto aos enlaces 5-6, 5-4 e 5-3. Neste caso, estes enlaces são utilizados somente pelos tráfegos originados do nó 5 ou destinados ao mesmo. Em *b*) o algoritmo atribui um custo infinito ao enlace 3-5. Nesta situação, este enlace não é mais utilizado para acomodar nenhum tráfego nesta configuração de camada. Em *c*) ocorre a geração final de uma camada com o *backbone* 6-2-3, resultante do isolamento dos nós 1, 5 e 4 e dos enlaces 5-3, 5-4, 1-2. As demais camadas são geradas isolando os demais nós em pelo menos uma camada da topologia.

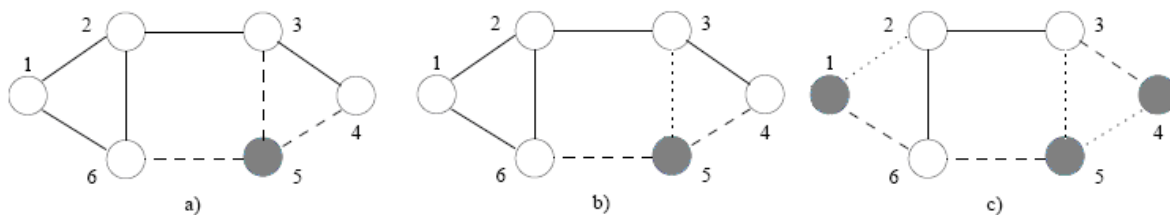


Figura 3.4 Exemplo de uma geração de camada utilizando MRC

O parâmetro de entrada do algoritmo é o número de camadas. Sendo assim, pode-se gerar tantas camadas quanto forem necessárias para isolar apenas um nó por camada. Isto permite a obtenção de melhores caminhos alternativos (isolando apenas poucos componentes por camada). Porém, adotando esse grande número de camadas não é uma abordagem escalável devido à grande quantidade de informações necessárias para representar todas essas camadas em cada nó. Em simulações feitas em [KVALBEIN et al, 2006] para algumas topologias reais, revela-se que entre 3 a 6 camadas são suficientes para isolar todos os elementos da rede, porém os menores caminhos alternativos ficam penalizados pelo aumento de sua extensão por considerar outros nós isolados a serem contornados em uma mesma camada.

Em caso de topologia livre de falhas, a topologia inteira é utilizada normalmente para o tráfego de pacotes (não havendo alteração no funcionamento normal das rotinas de roteamento). Em caso de falha detectada, o nó escolhe qual camada consegue isolar essa determinada falha adjacente para que os pacotes possam atingir o nó destino. Para cada camada gerada, existe uma RIB e FIB disponíveis e que são utilizadas para realizar o encaminhamento durante a presença de falha. Antes de encaminhar utilizando a FIB da camada escolhida, o nó marca os pacotes para sinalizar aos demais nós qual a camada foi utilizada para desviar os fluxos de pacotes. Quando os pacotes desviados chegam ao próximo nó, ele identifica pela marca qual camada está sendo utilizada pelos pacotes e realiza o encaminhamento utilizando a FIB respectiva. Esse procedimento ocorre até atingir o nó destino/saída da topologia, onde a marca é retirada.

Alguns problemas dessa abordagem são semelhantes à RRL como a quantidade de informações extras na RIB e FIB que cada nó deve armazenar para as camadas geradas e a qualidade dos caminhos pois, segundo os próprios autores, podem existir caminhos muito extensos para contorno de falha que influenciam consideravelmente os quesitos de desempenho de tráfego. Os caminhos utilizados são em geral mais longos que os caminhos obtidos em uma re-convergência das rotinas de roteamento IP (em torno de 5 a 16% dependendo do número de camadas utilizado), pois utiliza poucas camadas para diminuir a quantidade de informações extras armazenadas na RIB e FIB de cada nó.

Em um trabalho recente [KVALBEIN et al, 2007], há uma adaptação da abordagem para gerar sub-topologias não só reduzindo o número de sub-topologias necessárias, mas também considerando a distribuição de custos aos enlaces de cada sub-topologia gerada para ter uma boa performance balanceando a quantidade de tráfego desviado. Para realizar

essa distribuição de custos, utiliza-se uma adaptação da heurística de *busca local* [FORTZ and THORUP, 2000], porém para reduzir sua complexidade, apenas alguns enlaces identificados como críticos (enlaces mais internos do *backbone*) são considerados nesse esquema. Essa adaptação funciona apenas para falha de um único enlace dentre um grupo de enlaces críticos (os demais enlaces não pertencentes ao grupo não são considerados).

Segundo os autores de [KVALBEIN et al, 2006], quando existe a ocorrência de falha envolvendo SRLGs, a abordagem MRC pode ser alterada para isolar todos os enlaces que pertencem a um mesmo SRLG em uma mesma camada, porém não descreve como pode ser realizado e as possíveis dificuldades que podem existir. Mesmo no trabalho mais recente publicado pelos desenvolvedores do MRC [HANSEN et al, 2007], a falha de SRLG é ainda considerada como trabalho futuro [HANSEN et al, 2007] [KVALBEIN et al, 2006]. Entretanto, como observado em [MARKOPOULOU et al, 2004], falhas de SRLG ocorre em situações reais e, portanto deve ser considerado em uma abordagem de recuperação de falha [SHAND and BRYANT, 2008].

3.4.5 Abordagem Not-Via

A abordagem pró-ativa Not-Via possui sua última versão da *Internet-draft* em [BRYANT et al, 2008]. Nesta abordagem cada nó da topologia possui rotas para endereços IP específicos. Cada endereço IP específico é denominado endereço *not-via*, que representa um enlace/nó a ser contornado. Cada nó divulga os seus endereços *not-via* a todos os demais nós. A abordagem estabelece que todos os nós calculem uma rota para cada endereço *not-via* isolando individualmente o componente que esse endereço representa. No caso de uma falha, cria-se um túnel virtual a partir do nó adjacente a esse componente falho encapsulando os pacotes com o endereço *not-via* respectivo. Com os pacotes utilizando um endereço *not-via* específico, os demais nós conseguem encaminhar corretamente esses pacotes até o nó que contém esse endereço *not-via* contornando corretamente o componente faltoso. Ao chegar nesse nó, os pacotes são desencapsulados.

O cálculo da rota para cada endereço *not-via* é feito utilizando o algoritmo *Incremental-SPF* removendo o componente que ele representa. Segundo [BRYANT et al, 2008], mesmo utilizando o *Incremental-SPF*, o tempo computacional necessário para gerar as rotas *not-via* em uma rede de 40 a 400 nós seria o mesmo que 5 a 13 execuções inteiras do algoritmo SPF padrão.

A Figura 3.5 obtida de [BRYANT et al, 2008] ilustra o uso dos endereços *not-via*.

Nessa figura, tem-se os endereços *not-via* para os nós adjacentes a P Sp, Bp, Cp, Ap que são endereços capazes de contornar o nó P. Os endereços *not-via* localizados em P P_s, P_c, P_b, P_a são capazes de contornar os enlaces $S-P, C-P, B-P, A-P$ na visão do nó S, C, B e A respectivamente. Os demais nós consideram esses endereços *not-via* localizados em P para contornar os nós S, C, B, A respectivamente. Cada nó, realiza o cálculo do caminho até estes endereços utilizando o algoritmo *Incremental-SPF*, sendo que o cálculo do menor caminho até um nó destino é efetuado sem considerar o componente a ser contornado.

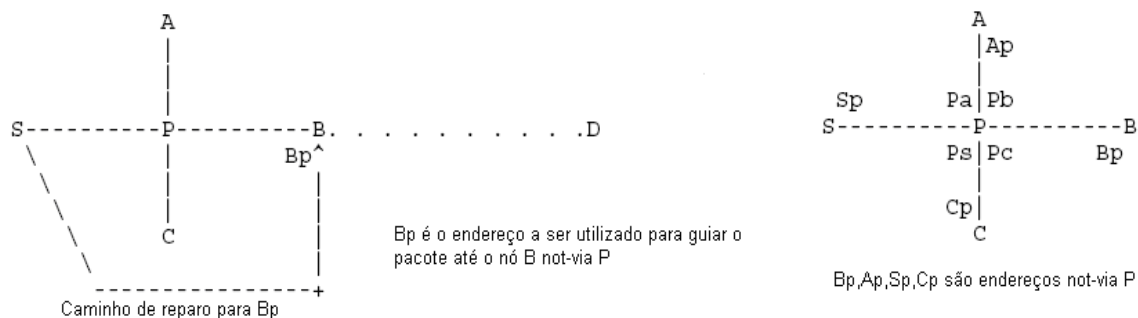


Figura 3.5 Endereços Not-Via

Por exemplo, o nó S na Figura 3.5 realiza o cálculo do menor caminho para os endereços *not-via* Bp, Cp, Ap e P_s . Para os endereços *not-via* Bp, Cp e Ap retira-se o nó P da topologia para esse cálculo, já no caso do menor caminho para o endereço *not-via* P_s retira-se o enlace $S-P$ da topologia apenas. Todos os nós da topologia não mostrados na figura efetuam o cálculo do menor caminho para Sp, Bp, Cp, Ap contornando o nó P, já para os endereços P_s, P_c, P_b, P_a o cálculo é feito para contornar o nó S, C, B, A respectivamente.

Quando esse nó S detectar alguma falha no enlace $S-P$, esse nó é o responsável por realizar o desvio dos pacotes. Os pacotes destinados a qualquer nó diferente de P, por exemplo um nó D, e que utilizam P na rota de menor caminho até D, devem ser encapsulados com algum endereço *not-via* P, ou seja, Bp, Cp, Ap . Para identificar qual desses endereços deve ser o escolhido, o nó S precisa manter informações sobre o *next-next-hop* além do *next-hop* existente para cada prefixo de rede de destino, o que implica em um campo extra na FIB para armazenar informações sobre o *next-next-hop*. No exemplo ilustrado, o *next-next-hop* adotado seria Bp , pois o fluxo destinado a um nó D passava por P e seguia em direção ao B (nó que detém o *next-next-hop* Bp) antes da falha de P. Nesse desvio, vários outros nós são utilizados para guiar os pacotes encapsulados, e como eles já possuem as rotas para qualquer endereço *not-via* P, eles sabem como rotear os pacotes desviados para Bp , possibilitando contornar o componente P atingindo B corretamente. Ao

chegar em B, os pacotes encapsulados com endereço *not-via Bp* são desencapsulados e os pacotes seguem o caminho original até chegar ao destino D.

Caso os pacotes estejam destinados ao nó P, S precisa contornar apenas o enlace S-P encapsulando os pacotes para o endereço *not-via Ps*, que representa um endereço “*not-via*” S-P na visão do nó S. A visão dos demais nós já possui rotas para o endereço *not-via Ps* como uma rota de contorno do nó S para chegar a Ps, o que evita utilizar S no menor caminho de recuperação (geraria *loop* de roteamento). Ao contornar o nó S, os demais nós encaminham os pacotes encapsulados para contornar também o enlace S-P.

A abordagem Not-Via permite alcançar 100% de cobertura das falhas de nós, enlaces e SRLG. Uma restrição do uso da abordagem ocorre quando os pacotes utilizam o limite estabelecido no *Maximum Transfer Unit* (MTU). Nesse caso, há a fragmentação dos pacotes além do *overhead* no processamento desse recurso nos roteadores em redes IPv4. Além dessa restrição, se o bit *Don't Fragment* estiver habilitado, o que ocorre em mais de 90% do tráfego de *backbone* Internet [WOLFGANG and TAFVELIN, 2007], há o descarte dos mesmos. Em redes IPv6 há o descarte desses pacotes [DEERING and HINDEN, 1998]. O fator de escala decorrente da quantidade de rotas adicionais necessárias para armazenar cada endereço *not-via*, além dos *next-next-hops* para cada entrada na FIB é outra restrição desta abordagem. Por fim, a abordagem pode permitir que pacotes percorram em algumas situações mais nós do que os necessários no desvio para atingir o nó destino.

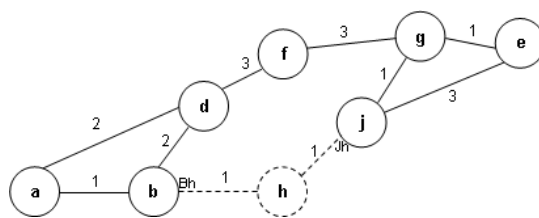


Figura 3.6 Exemplo de caminho mais extenso com Not-Via

A topologia simplificada com custos simétricos em cada enlace da Figura 3.6 ilustra a possibilidade de um caminho ter mais nós do que o necessário. Caso exista um fluxo de *a* para *e*, o caminho OSPF traçado para esse fluxo é *a,b,h,j,g,e* e as respectivas rotas em cada nó são preenchidas normalmente. Na ocorrência de uma falha no nó *h*, o nó *b* deve encapsular os pacotes destinados a *e* para o endereço *not-via h*, identificado como *Jh*. Isso é necessário já que a abordagem LFA não se aplica aqui: ao considerar o nó *d* como “nó vizinho *backup*” geraria *loop* de roteamento (*b,d,b,d,...*), pois o menor caminho

de d até e utiliza os nós b, h, j, g, e . Quando os pacotes encapsulados com o endereço *not-via* Jh chegam ao nó j , os pacotes são desencapsulados para seguirem o caminho OSPF original. Observa-se que os pacotes encapsulados seguem a partir de b em d, f, g, j , e apenas em j é feito o desencapsulamento para seguir o menor caminho até g novamente e por fim e , sendo uma trajetória ineficiente utilizando duas vezes o nó g para alcançar o destino.

Na ocorrência de falha envolvendo SRLGs, a abordagem *Not-Via* altera o conceito de endereços *not-via* estendendo-os para considerar enlaces pertencentes ao mesmo SRLG da falha. Dessa forma, ao considerar um endereço *not-via* que contorna algum enlace pertencente a um SRLG, os demais enlaces desse SRLG também serão retirados da topologia para realizar o cálculo *Incremental-SPF*. A mesma situação ocorre para contornar um nó, devendo retirar da topologia todos os enlaces que pertencem aos mesmos SRLGs dos enlaces adjacentes ao nó sendo contornado.

As abordagens FIR/FIFR, MRC ou *Not-Via* são recomendadas pela IETF [SHAND and BRYANT, 2008] apenas para quando nenhuma abordagem com ECMP [MOY, 1998] ou LFA [ATLAS and ZININ, 2008] for possível. Isso é justificável, pois utilizar ECMP ou LFA, quando possível, torna-se bastante simples de ser implementado e não requer grandes modificações no funcionamento dos nós. Dentre as abordagens vistas até agora, as abordagens FIR/FIFR não cobrem falha de SRLG e não suporta topologias com enlaces de custos assimétricos. Já a abordagem MRC não suporta falha de SRLG. A abordagem *Not-Via* foi a mais adequada por fornecer 100% de cobertura de nó, enlace e SRLG.

3.4.6 Outras Abordagens Pró-Ativas Relacionadas

Além das abordagens apresentadas existem outras abordagens pró-ativas que não atingem 100% de cobertura de falhas de enlace, nó ou SRLG. Algumas possuem certa consonância com o *framework* IPFRR outras apresentam algumas variações.

A abordagem *IP Local Node Protection* (IPLNP) [SU et al, 2007] foi proposta recentemente com o intuito de aperfeiçoar alguns itens na escolha do nó vizinho de *backup* quando a abordagem LFA não for aplicável para contorno de nós. A abordagem verifica se o “nó vizinho *backup*” é adjacente ao nó considerado falho. Nesse caso, o nó origem consegue deduzir que o “nó vizinho *backup*” também encontra, a partir dele, um próximo nó vizinho a esse nó considerado falho para alcançar o nó destino. Ao antecipar o caminho encontrado pelo “nó vizinho *backup*”, pois é adjacente ao mesmo, o nó origem pode encontrar um “nó vizinho *backup*” quando a abordagem LFA não consegue. Da mesma

forma que a abordagem LFA, essa abordagem não atinge 100% de cobertura por ser dependente da estrutura da topologia e é limitado apenas para contorno de nó. Outra restrição é que a IETF não divulgou ainda a confiabilidade dessa proposta.

A abordagem pró-ativa [NUCCI et al, 2003] é semelhante à abordagem reativa [FORTZ and THORUP, 2002] diferenciando por encontrar os valores de custos dos enlaces através de heurísticas de *busca tabu* para conseguir um possível isolamento de falha de qualquer enlace da topologia. A abordagem encontra custos de enlace para restringir a carga dos enlaces a 90% de sua capacidade com o intuito de acomodar o tráfego em caso de falha. Esta abordagem, no entanto, é restrita a falhas de enlace.

A abordagem pró-ativa proposta em [SCHOLLMEIER et al, 2003] especifica o algoritmo O2 que é responsável por identificar enlaces denominados de **curinga**. O enlace curinga é escolhido a partir de um esquema, denominado de **triângulo**, em que cada nó é relacionado com os seus vizinhos. Neste esquema, um dos enlaces do triângulo é escolhido como curinga. O tráfego é desviado para o enlace curinga quando da ocorrência de falha no enlace utilizado pelo algoritmo de roteamento. Não existe restrição de um nó utilizado pelo algoritmo de roteamento ser o nó curinga de outro enlace. A principal desvantagem desta abordagem é que ela é restrita a casos particulares de topologias de rede. Além disto, o algoritmo de roteamento não utiliza a estratégia de menor caminho original do OSPF.

A abordagem *U-Turn Alternate* [ATLAS, 2006] é basicamente uma extensão da abordagem LFA com a possibilidade dos nós vizinhos encontrarem os próximos nós *backup*. Se o nó vizinho *backup* não atender às restrições para ser um LFA, o seu nó vizinho pode ser o *U-Turn Alternate* se esse encontrar um próximo nó que seja um nó *backup*. Para eliminar a possibilidade de *loop* de roteamento os pacotes são marcados para que o nó vizinho *backup* possa encaminhá-los ao nó *U-Turn Alternate*. Apesar de ser uma solução simples, essa abordagem não atinge 100% de cobertura por ser dependente da estrutura da topologia.

A abordagem de *Tunnels* apresentada em [BRYANT et al, 2005] especifica um mecanismo de encapsulamento que é utilizado, quando da ocorrência de falhas, para desviar os pacotes por um caminho, definido pelo túnel, até um nó que possa encaminhar os pacotes pelo menor caminho até o nó destino contornando a falha. Apesar de ser uma abordagem simples, segundo os próprios autores, não suporta custos de enlaces assimétricos, dependente da técnica de encapsulamento e não é interativo com as demais abordagens IPFRR, como por exemplo, o LFA.

A abordagem *SS_LINK/SS_NODE* [KANG and CHAO, 2007] utiliza os algoritmos *Depth-First Search* e *Breadth-First Search* em grafos para estabelecer os caminhos de recuperação (ao invés do SPF padrão adotado pelo OSPF). Esses caminhos são identificados por uma seqüência de *bkp_next_hop/bkp_port* que é definida em cooperação com as informações geradas pelos demais nós. Cada nó possui campos adicionais em cada entrada na FIB para armazenar esses *bkp_next_hop/bkp_port*. A abordagem traz uma solução simples, porém, como o próprio trabalho [KANG and CHAO, 2007] revela, gera maior custo computacional que as buscas SPF, gera caminhos de recuperação mais extensos que o necessário em alguns casos e não suporta falha de SRLG.

3.5 Sumário das Abordagens em Redes IP

Entre as abordagens apresentadas nos itens anteriores algumas são propostas específicas para redes IP de forma a auxiliar as rotinas de roteamento OSPF e de encaminhamento IP até o fim do período de convergência. Encaixam nesta classificação as abordagens: [WANG and CROWCROFT, 1990], [NARVAEZ, 2000], [FORTZ and THORUP, 2002], [LIU and REDDY, 2004], [NUCCI et al, 2003], [SCHOLLMEIER et al, 2003], [LEE et al, 2004], [ZHONG et al, 2005], [ATLAS and ZININ, 2008], [BRYANT et al, 2008], [KVALBEIN et al, 2006], [ATLAS, 2006] e [BRYAN et al, 2005]. As quatro primeiras abordagens são reativas ([WANG and CROWCROFT, 1990], [NARVAEZ, 2000], [FORTZ and THORUP, 2002] e [LIU and REDDY, 2004]), sendo que as abordagens [NUCCI et al, 2003], [SCHOLLMEIER et al, 2003], [LEE et al, 2004], [ZHONG et al, 2005], [ATLAS and ZININ, 2008], [BRYANT et al, 2008], [KVALBEIN et al, 2006], [ATLAS, 2006] e [BRYAN et al, 2005] são pró-ativas calculando antecipadamente, isto é, antes da ocorrência de falhas existem rotas que possibilitem contornar as falhas.

Os resultados indicam que as abordagens pró-ativas são mais úteis devido a abordagem já reativa do OSPF. Observa-se ainda que o período em que as rotas auxiliares são utilizadas é relativamente curto por atuarem dentro do escopo local. As abordagens reativas, apesar de oferecerem um solução para a manipulação de falhas/congestionamento, apresentam tempo de reação superior as pró-ativas (como era de se esperar). Além disso, é difícil justificar o uso de abordagens reativas como alternativa à utilizada no OSPF padrão em que o período de convergência pode ser inferior a 1 segundo.

As análises das abordagens permitem ainda concluir que:

- As abordagens ECMP e ou LFA basicamente definem que o nó vizinho consegue identificar rotas alternativas para o destino dos pacotes. Estas abordagens são relativamente simples e recomendadas pela IETF, porém não atingem 100% de cobertura. Esta restrição pode ser eliminada através de abordagens *Multi-Hop*;
- Dentre as abordagens pró-ativas analisadas que são *Multi-Hop*, apenas a abordagem Not-Via [BRYANT et al, 2008] consegue 100% de cobertura de falhas de enlace, nó, ou SRLG em uma topologia com custos de enlaces assimétricos ou não;

Dentre todas as abordagens vistas, a opção por contornar falha de um componente de rede (enlace, nó ou SRLG) torna-se uma escolha razoável por ser o tipo de ocorrência mais comum [VASSEUR et al, 2004] [IYER et al, 2003]. Isso reduz a complexidade das abordagens, sendo um fator importante para se obter um caminho de recuperação, pois somente deve ser usado em um curto espaço de tempo (caráter emergencial) até que a falha transitória seja reparada ou até que um novo caminho seja definido globalmente com novas rotas (OSPF).

Além disso, [PETERSSON, 2005] revela que a maioria dos provedores de serviços de transporte de dados não planejam uma recuperação para múltiplas falhas simultâneas de enlaces ou nós, pois seria muito caro definir formas de proteção contra eventos que geralmente possuem uma probabilidade muito baixa de ocorrência, além do que a complexidade das abordagens aumenta em relação ao período de convergência do OSPF (podendo ser < 1 segundo).

Dentre todas as abordagens analisadas, apenas a abordagem [KVALBEIN et al, 2006] com a adaptação [KVALBEIN et al, 2007] realiza algum controle no desvio dos pacotes para evitar congestionamentos na topologia, porém ficou restrito à falha de um único enlace (dentre os enlaces ditos críticos). As demais ocorrências de falhas (enlaces não críticos) com a abordagem [KVALBEIN et al, 2007] e as demais outras abordagens não fornecem meios para evitar congestionamentos, o que pode implicar em mais instabilidade na rede influenciando tráfegos originalmente não afetados pela falha. Segundo a IPFRR [SHAND and BRYANT, 2008], recomenda-se tratar esse tipo de problema.

A Tabela 3.1 apresenta a síntese da análise das abordagens apresentadas neste capítulo.

Tabela 3.1 Sumário das Abordagens de Reroteamento em Redes IP

Abordagem	(Reativo Pró-ativo) (Suporta Falha de: Enlace Nó SRLG)	Propósito	Restrição
SPF-EE [WANG and CROWCROFT, 1990]	Reativo, Enlace, Nó	Contornar falhas e congestionamentos encontrando caminhos a partir da troca informações com os nós vizinhos.	Puramente reativo.
<i>VectorMetric</i> [NARVAEZ, 2000]	Reativo, Enlace	Encontrar um caminho alternativo limitando a atualização do estado do enlace apenas para os nós desse caminho alternativo calculado.	Puramente reativo e funciona para falhas de enlace apenas.
OSPF com Busca Local para custos de enlaces [FORTZ and THORUP, 2002]	Reativo, Enlace	Busca Local para distribuir custos aos enlaces para que necessitem de poucas mudanças nos seus valores em caso de falhas.	Recálculo dos menores caminhos devido às poucas mudanças nos custos dos enlaces, além de suportar apenas falhas de enlace e ser reativo.
Feasible Next Hop [LIU and REDDY, 2004]	Reativo, Enlace	Identificação de <i>Feasible Next Hop</i> para contorno de um enlace falho, criando um <i>Rerouting Path</i> dinamicamente.	Contorna apenas enlace e o processo é reativo.
OSPF utilizando Busca Tabu para custos de enlaces [NUCCI et al, 2003]	Pró-ativo, Enlace	Busca Tabu para encontrar a atribuição de custos aos enlaces para contornar falhas de enlace através da distribuição do tráfego em múltiplos caminhos com ECMP.	Funciona para falhas de enlace, além de permitir a divulgação da mudança de estado do enlace para todos os nós da rede, não suporta SRLG.
O2 routing [SCHOLLMEIER et al, 2003]	Pró-ativo, Enlace	Roteamento O2 que utiliza enlaces “curinga” para contornar um enlace principal. Cada enlace pode ser considerado principal ou “curinga” dependendo da falha.	Alta limitação da topologia de rede (formação de “triângulos”) e substituição do protocolo de roteamento existente.
FIR [LEE et al, 2004]	Pró-ativo, Enlace	Reroteamento local com encaminhamento baseado em FIBs por interface obtida na dedução de enlaces com falha.	Funciona para falhas de enlace, pode gerar <i>loops</i> de roteamento, não suporta enlaces assimétricos, não suporta SRLG.

FIFR [ZHONG et al, 2005]	Pró-ativo, Enlace, Nó	Melhora a abordagem de [LEE et al, 2004] obtendo FIBs por interface com rotas que contornam falhas deduzidas de nó.	Não consegue contornar falha no último enlace do menor caminho, não suporta custos de enlace assimétricos, não suporta SRLG.
LFA [ATLAS and ZININ, 2008]	Pró-ativo, Enlace, Nó, SRLG	Estipula regras para identificar um nó vizinho <i>backup</i> para obter um caminhos alternativo sem <i>loops</i> .	Não tem 100% de cobertura das falhas e depende da topologia.
IPLNP [SU et al, 2007]	Pró-Ativo, Nó	Permite utilizar um nó vizinho, que também é adjacente ao nó falho, que não gere <i>loop</i> quando a abordagem LFA não pode ser aplicada.	Funciona somente para nó, não tem 100% de cobertura de falhas e não considera SRLG.
Not-Via [BRYANT et al, 2008]	Pró-ativo, Enlace, Nó, SRLG	Rotas para endereços not-via, utilizados para encapsular os fluxos comprometidos pela falha de um determinado componente.	Uso de encapsulamento, grande aumento de informações extras na FIB e pode gerar caminhos mais longos que o necessário.
MRC [KVALBEIN et al, 2006], [KVALBEIN et al, 2007], [HANSEN et al, 2007]	Pró-ativo, Enlace, Nó	Criar várias sub-topologias onde cada uma contorna determinadas falhas, mantendo uma RIB e FIB para cada uma dessas sub-topologias.	Uso de várias RIBs e FIBs, pode gerar caminhos mais longos que o necessário e não suporta SRLG.
U-Turn [ATLAS, 2006]	Pró-ativo, Enlace, Nó, SRLG	Localiza um nó a frente do nó vizinho que consegue atender à abordagem LFA.	Não tem 100% de cobertura de falhas e depende da topologia.
Tunnels [BRYAN et al, 2005]	Pró-ativo, Enlace, Nó, SRLG	Cria túneis a partir do nó origem até um nó capaz de contornar a falha com as rotas existentes.	Não suporta enlances assimétricos, não tem 100% de cobertura de falhas, não interage com outras abordagens IPFRR (LFA) e usa encapsulamento.
SS_LINK/NODE [KANG and CHAO, 2007]	Pró-ativo, Enlace, Nó	Usa <i>Depth-First Search</i> e <i>Breadth-First Search</i> em grafos para encontrar uma seqüência de <i>bkp_next_hop</i> até um nó de saída para atingir o nó destino.	Possui complexidade extra com algoritmos de busca, gera caminhos mais extensos que o necessário e não suporta falha de SRLG.

OBS: Nenhuma das abordagens realiza um gerenciamento dos pacotes desviados para evitar a ocorrência de congestionamentos (afeta outros fluxos de tráfego) em toda a topologia.

3.6 Conclusão

Esse capítulo descreve as características das falhas e as implicações no período de convergência. São descritas as principais abordagens de recuperação de falhas em redes IP, e meios para auxiliar as rotinas de roteamento durante a ocorrência de falhas. Dentre essas abordagens, a que alcança 100% de cobertura para falhas de enlace, nó e SRLG foi a *Not-Via* em conjunto com ECMP e LFA. Foram descritas algumas ineficiências dessa abordagem em relação ao tamanho do caminho de recuperação, da quantidade de recursos necessários para desviar os fluxos comprometidos e depende da técnica de encapsulamento.

De posse desse levantamento, o próximo capítulo descreve a abordagem proposta nesse trabalho, que apresenta um novo esquema de reroteamento rápido do tipo IPFRR para aumentar a confiabilidade e robustez de uma topologia de rede.

Capítulo 4

ESQUEMA DE CAMINHOS EMERGENCIAIS RÁPIDOS – E-CER

Quando existe uma mudança na topologia de rede, devido a uma falha ou restauração de enlace/nó ou como um resultado de ação de manutenção, os nós passam por um período de convergência para que todos atinjam uma visão única da topologia atualizada. Durante esse período, vários fluxos de pacotes podem ter o seu processo de encaminhamento afetado. Isso ocorre devido ao tempo necessário para propagação da mudança do novo estado da topologia a todos os nós e também devido ao tempo que cada nó necessita para atualizar a RIB, calcular os menores caminhos e atualizar a FIB. No caso de uma falha, os pacotes destinados às redes afetadas por essa mudança podem ser descartados devido às contínuas tentativas de encaminhamento pelo componente falho (ocasionadas pela não atualização da FIB no nó adjacente à falha), e também devido aos *loops* de roteamento. Os *loops* de roteamento são consequência da atualização não sincronizada das FIBs nos nós da topologia.

Mesmo o período de convergência podendo ser reduzido para décimos de segundo em algumas topologias, ainda existe uma alta taxa de pacotes perdidos principalmente em redes mais rápidas.

Observa-se que as falhas são passíveis de ocorrer no dia a dia de operação de uma topologia de *backbone* IP. Estudos realizados em topologias reais apontam que a grande maioria das falhas atinge um único componente de rede (enlace ou nó) ou de componentes de rede dependentes (SRLG). Além disso, a grande maioria dessas falhas (80%) é transitória (de alguns segundos até no máximo alguns minutos) [IANNACCONE et al, 2004] [IYER et al, 2003] [MARKOPOULOU et al, 2004] [WATSON et al, 2003].

Existem algumas abordagens para auxiliar os protocolos de roteamento durante a

existência de falhas. Dentre essas abordagens, as indicadas no *framework* IPFRR [SHAND and BRYANT, 2008] são ECMP, LFA e uma abordagem *Multi-Hop* (U-Turn, Tunnels, FIR/FIFR, MRC ou Not-Via), somente quando ECMP e LFA não forem aplicáveis. Porém, foi verificado que as abordagens U-Turn, Tunnels e FIR/FIFR não alcançam 100% cobertura de falhas de enlace ou nó. Além disso, Tunnels e FIR/FIFR não suportam peso de enlace assimétrico. Tanto a abordagem MRC quanto a FIR/FIFR não suportam falha do tipo SRLG. É importante ressaltar que a ocorrência de falha SRLG existe em ambientes reais [MARKOPOULOU et al, 2004], sendo necessário ter esse suporte. A abordagem mais promissora observada é a Not-Via para prover 100% de cobertura de falhas (enlace, nó e SRLG) e independente dos custos atribuídos aos enlaces. No entanto, ela pode gerar caminhos mais extensos que o necessário, utilizar recursos excessivos para representar seus caminhos de recuperação na FIB além de depender da técnica de encapsulamento. Em qualquer abordagem, se houver congestionamento causado pelos fluxos de tráfego desviados, não existe um mecanismo de detecção para que evite comprometer os demais fluxos não afetados pela falha.

Dentro desse contexto, este trabalho visa descrever um novo mecanismo de recuperação de falha seguindo o *framework* IPFRR como um esquema distribuído para auxiliar os protocolos de roteamento, no caso o OSPF, a reduzir as instabilidades na rede e as perdas de pacotes durante o período de convergência. Esse esquema, nomeado “Esquema de Caminhos Emergenciais Rápidos” (E-CER), consiste de caminhos de recuperação previamente calculados (Caminhos Emergenciais Rápidos, i.e. CERs) para cada componente adjacente ao nó, que o está executando, e os disponibiliza diretamente na FIB. O E-CER atinge 100% de cobertura de falhas (enlace, nó e SRLG), independe dos custos atribuídos aos enlaces (assimétricos ou não) e realiza o processo de desvio dos pacotes pelo percurso do caminho de recuperação somente se houver possibilidade (impede a ocorrência de congestionamentos durante o desvio descartando os pacotes para não comprometer os demais fluxos até que o OSPF termine o período de convergência).

O fato de antecipar os cálculos de caminhos de recuperação (ser pró-ativo) tem por meta disponibilizar os CERs antes de um sinal de falha ocorrer, e esta é uma prática justificável e comumente encontrada nas principais abordagens de reroteamento [SHAND and BRYANT, 2008], pois elimina todo o tempo necessário para calcular os caminhos após a detecção de falha.

Na ocorrência de um sinal de falha, o E-CER antecipa a reação do protocolo de

roteamento OSPF para que, antes de iniciar um período de convergência, o nó utilize os CERs já disponíveis na FIB. Essa medida procura desviar os fluxos de pacotes comprometidos que já seriam descartados no nó detector de falha devido à inconsistência de dados na FIB, além dos descartes ocasionados por *loops* de roteamento decorrentes da inconsistência entre as FIBs dos demais nós durante o período de convergência.

O E-CER atua em escopo local e distribuído em cada nó da rede. Essa abordagem é separada em dois módulos: módulo *CER_pró-ativa* no plano de controle e módulo *CER_EDif* no plano de dados. O módulo *CER_pró-ativa* tem por objetivo calcular e gerar os CERs logo após o término da execução do OSPF, para então adicionar uma representação dos mesmos na FIB. A geração dos CERs utiliza os dados existentes na RIB e os resultados da árvore de menores caminhos mantida pelo OSPF. A geração de um CER é obtida seguindo a recomendação da IETF IPFRR [SHAND and BRYANT, 2008], que foi personalizada aqui em termos de níveis de classificação, de acordo com a característica do caminho: ECMP, LFA e para *Multi-Hop* o nível *Signaling*. O módulo *CER_EDif* é uma rotina de encaminhamento auxiliar para as rotinas padrão de encaminhamento IP, cujo objetivo é desviar os fluxos de pacotes afetados por uma falha utilizando os CERs já disponibilizados na FIB.

A reunião desses dois módulos tem por objetivo auxiliar o OSPF a aumentar a robustez e confiabilidade da rede durante a presença de uma falha. Cada um desses módulos é descrito mais detalhadamente no decorrer deste capítulo.

4.1 Módulo: CER_pró-ativa

A FIB existente em cada nó da rede IP situa-se no plano de dados, e é uma tabela utilizada pelas rotinas de encaminhamento IP, que consultam o endereço IP destino de cada pacote para realizar a decisão de encaminhamento. A consulta na FIB baseia-se na localização do prefixo de rede mais específico que o endereço IP de destino se encaixa [FULLER et al, 1993], retornando o próximo nó, i.e. *next-hop*, que o pacote deve ser enviado e a interface de saída [BAKER, 1995]. Além desse padrão encontrado no plano de dados, há a necessidade da adição de dados extras na FIB para representar os CERs, que são gerados pelo módulo *CER_pró-ativa*. Esses dados são necessários para que o módulo *CER_EDif* consiga auxiliar as rotinas de encaminhamento IP a contornar uma falha por meio dos CERs.

A *CER_pró-ativa* é um módulo do E-CER que atua em escopo local e no plano de controle de cada nó da topologia de *backbone* sob domínio do protocolo OSPF. Esse módulo tem por objetivo encontrar previamente os CERs para uma falha e disponibilizá-los na FIB para uso imediato quando for necessário. Para um nó encontrar os CERs, a *CER_pró-ativa* é executada após as rotinas do OSPF para reutilizar a RIB e os menores caminhos já calculados. A rotina calcula menores *caminhos alternativos* para alcançar os demais nós sem utilizar um dos componentes de rede adjacentes ao nó origem por vez (simulando uma falha). Desses menores *caminhos alternativos* para cada nó destino remoto, identifica-se um sub-caminho que é chamado caminho emergencial rápido, i.e. CER. Um CER é composto por uma seqüência de nós, que identifica esse sub-caminho, armazenados em forma de vetor (Vetor_CER). Uma representação do Vetor_CER é sintetizada pela *CER_pró-ativa* e adicionada na FIB.

Cada componente de rede adjacente a ser contornado é uma antecipação de um sinal de falha do mesmo. Com essa antecipação, na presença de falha real, o nó pode redirecionar os fluxos comprometidos rapidamente utilizando a informação CER na FIB que possibilita o contorno desse componente.

Quando acontece alguma mudança no estado do enlace, como uma adição de nó ou enlace, o OSPF atualiza normalmente as RIBs em todos os nós, calcula o menor caminho SPF utilizando a RIB e atualiza a FIB. Em seguida, a *CER_pró-ativa* é executada novamente após o período de convergência do OSPF para manter os CERs sempre atualizados com as informações do estado do enlace existentes.

4.1.1 Condições para o Correto Funcionamento da CER_pró-ativa

A primeira condição necessária para o correto funcionamento da *CER_pró-ativa* é a existência de uma configuração de estrutura física de topologia de AS que permaneça interligada mesmo na ocorrência de uma falha em qualquer ponto da topologia, podendo ser ou de enlace ou de nó ou de SRLG. Essa condição também se mostra válida para qualquer abordagem de roteamento, uma vez que deve existir pelo menos um caminho alternativo disponível para contornar alguma falha. Conforme observado em [NUCCI et al, 2003] e também em análises práticas ([IANNACCONE et al, 2004] [IYER et al, 2003] [MARKOPOULOU et al, 2004] [WATSON et al, 2003]) considerar falha de apenas um componente de rede mostra-se razoável, uma vez que múltiplas falhas simultâneas

independentes são raras de acontecer em topologias reais. Suportar múltiplas falhas acarreta em uma complexidade adicional para computar previamente os caminhos, o que é indesejável em relação aos resultados obtidos com a convergência do OSPF. Utilizando a Teoria de Grafos, podem-se representar os nós de uma topologia de rede como sendo os vértices, e os enlaces com seus custos como sendo as arestas. Para que um grafo qualquer consiga sustentar uma falha de vértice mantendo ainda uma conectividade de todos os vértices, ou seja, ainda existir um caminho alternativo que conecte qualquer dois vértices do grafo, é necessário que o grafo obedeça ao *Teorema de Menger* [DIESTEL, 2005], sendo *k-conexo* com $k=2$. A variável k indica que mesmo removendo $k-1$ vértices do grafo, o grafo ainda mantém uma conectividade entre todos os vértices. Portanto um grafo mínimo deve ser pelo menos *2-conexo*, ou seja, um grafo com $k=2$ suporta total conectividade entre seus vértices se removido no máximo $(2-1)$ vértices do grafo.

A segunda condição ocorre na presença de SRLGs, onde uma falha dos enlaces pertencentes a um SRLG não pode ocasionar falta de conectividade entre os nós da topologia.

A terceira condição necessária está relacionada com o planejamento antecipado da rede, de modo que a utilização de recursos e a distribuição de tráfego na topologia façam com que utilizem no máximo 50% da capacidade dos enlaces. Essa condição, observada em [IYER et al, 2003], tem por objetivo permitir uma acomodação de tráfegos desviados por motivos de falha. Para tanto, pode-se utilizar uma abordagem de ET *offline* para calcular exaustivamente em um espaço de soluções e localizar qual a melhor distribuição de custos aos enlaces para que restrinja o uso da capacidade dos enlaces em no máximo 50%, como os trabalhos de [FORTZ and THORUP, 2000] e [NUCCI et al, 2003].

Tendo uma topologia de AS com características de ao menos *2-conexo*, manter a conectividade em casos de SRLG, e com a distribuição de tráfego obedecendo à restrição de utilização da capacidade dos enlaces ser menor que 50%, a atuação da *CER_pró-ativa* para obtenção de *caminhos emergenciais* possui as condições necessárias para ser aplicável.

4.1.2 Geração de Caminhos Alternativos

A Tabela 4.1 apresenta uma descrição dos principais símbolos utilizados no desenvolvimento das formulações para a elaboração da *CER_pró-ativa* e *CER_EDif*.

Após a execução das rotinas do OSPF em um nó, a RIB e FIB já estão atualizadas e preenchidas e as rotinas de encaminhamento de pacotes estão funcionando corretamente. Após essa execução, a *CER_pró-ativa* deve iniciar o cálculo dos *CAs*.

Tabela 4.1 Notação Utilizada na Descrição dos Níveis e na Formulação Geral

Notação	Descrição
NO	Nó que está executando a $CER_{pró-ativa}$ para obter os CERs.
NA	Nó adjacente conectado ao (NO,NA) simulado com falha.
NV	Nó vizinho: primeiro nó do CA ou $ALT_{NO,ND,NA}$
ND	Nó que tem o menor caminho original OSPF afetado pela falha simulada do componente de rede adjacente.
(NO,NA)	Enlace adjacente ao NO simulado com falha.
CA	Caminho Alternativo: Menor caminho obtido com $ISPF$ a partir de NO até um ND que contorna um componente adjacente ao NO .
$nós_intermediários$	Seqüência de nós que não obedeceram ao nível SIG.
CER	Sub-caminho do CA que é suficiente para contornar a falha simulada com sucesso.
$nó_CER$	Nó integrante do CA que possui, a partir dele, uma rota OSPF que contorna a falha simulada com sucesso.
$DistCA(x,y)$	Distância do sub-caminho de x até y integrante do menor CA para um ND .
$NumNosCA(x,y)$	Número de nós do CA de x até y .
$Z_{NO,J,NA}$	Função custo dos menores caminhos de NO até qualquer nó destino J , sendo que existem alguns menores caminhos que são CA para alguns nós J
$\varphi_{NO,J,NA}$	Conjunto dos menores caminhos para qualquer nó destino J que obedecem a $Z_{NO,J,NA}$
$\varphi_{NO,ND,NA}$	Conjunto dos menores CA para ND que pertencem a $\varphi_{NO,J,NA}$
$ALT_{NO,ND,NA}$	Um CA de $\varphi_{NO,ND,NA}$
$nó_CER$	Nó pertencente a $ALT_{NO,ND,NA}$ que identifica o final do $CER_{NO,ND,NA}$
$CER_{NO,ND,NA}$	Sub-caminho (CER) do NO até $nó_CER$ de um $ALT_{NO,ND,NA}$, candidato a ser $S_CER_{NO,ND,NA}$
$S_CER_{NO,ND,NA}$	CER escolhido para ND que obedece a $Z'_{NO,ND,NA}$
$Z'_{NO,ND,NA}$	Função custo que minimiza o custo e o número de nós para cada $CER_{NO,ND,NA}$
$num_CER_{NO,ND,NA}$	Especifica o número de nós dependendo do nível utilizado (ECMP, LFA ou SIG)
$custo_CER_{NO,ND,NA}$	Especifica a soma dos custos dos enlaces dependendo do nível utilizado (ECMP, LFA ou SIG)
J	Todos os nós destino da topologia a partir de NO
n	Número de nós na topologia
$c_{i,j}$	Custo ou peso do enlace (i,j)
$x_{i,j}$	Fluxo do enlace (i,j)
k	Um NV ou um $nó_TMP$ sendo analisado de $ALT_{NO,ND,NA}$
$DistOSPF(x,y)$	Distância do menor caminho OSPF do nó x até y
$NumNosOSPF(x,y)$	Número de nós do menor caminho OSPF do nó x até y
$DistCER_{NO,ND,NA}$	Soma dos custos dos enlaces de $CER_{NO,ND,NA}$
$NumNosCER_{NO,ND,NA}$	Número de nós do $CER_{NO,ND,NA}$
$OSPF_Enlaces(x,y)$	Conjunto dos enlaces integrantes do menor caminho OSPF de x até y
$OSPF_Nos(x,y)$	Conjunto dos nós integrantes do menor caminho OSPF de x até y
$enlaces_SRLG$	Enlaces pertencentes a um mesmo SRLG
$SRLG(x,y)$	Conjunto de enlaces pertencentes ao mesmo SRLG do enlace (x,y)
$Vizinhos_NO$	Conjunto de NV a NO

Um nó que está executando a $CER_{pró-ativa}$ é dito NO , pois os CA s terão como raiz esse nó. Cada nó destino que se torna inalcançável na árvore original OSPF de

menores caminhos na presença de uma possível falha no componente de rede adjacente ao NO , recebe o nome de ND e é utilizado como referência para obter os CAs . Os CAs são gerados para contornar um componente adjacente e atingir cada ND . O componente de rede adjacente a ser contornado pode ser o enlace adjacente (NO,NA) ao NO , os enlaces pertencentes ao mesmo SRLG ($enlaces_SRLG$), quando existirem, e o nó adjacente ao NO conectado pelo (NO,NA) denominado NA .

A geração dos CAs é realizada com o *Incremental-SPF* (ISPF) [MCQUILLAN et al, 1978] [NARVAEZ, 2000], identificando as interfaces ativas do NO e simulando uma falha no componente adjacente (retirando da base de informações do estado do enlace para poder contornar esse componente) em cada uma dessas interfaces. Ao retirar um componente de rede, o ISPF apenas realiza os cálculos na sub-árvore comprometida, obtendo apenas os CAs para cada um dos NDs afetados pela retirada do componente. A complexidade do algoritmo ISPF é, no pior caso, igual ao de uma execução completa de um SPF clássico [DIJKSTRA, 1959], ou seja, $O(n\log(n))$ com n sendo o número de nós na rede. Entretanto, em situações reais esse tempo é reduzido, pois apenas uma parte da sub-árvore fica comprometida e a execução pode terminar a partir do momento que encontrar os caminhos para todos os NDs afetados.

Primeiramente retira-se da topologia o componente (NO,NA) , realizando um cálculo do ISPF para obter apenas os menores caminhos de mesmo menor custo do NO até um único ND somente, que é o nó adjacente ao NO interligado pelo (NO,NA) . A equação (4.1) representa a chamada ISPF para encontrar os CAs do NO para alcançar o ND retirando da topologia o (NO,NA) .

$$CAs(ND) = ISPF(NO,ND,(NO,NA)) \quad (4.1)$$

Caso o (NO,NA) pertença a um SLRG, então há uma modificação na equação (4.1), devendo considerar não apenas o (NO,NA) , mas também os $enlaces_SRLG$. A equação (4.2) apresenta como é representada a chamada ISPF considerando o (NO,NA) e também os $enlaces_SRLG$ quando existir.

$$CAs(ND) = ISPF(NO,ND,(NO,NA) \cup enlaces_SRLG) \quad (4.2)$$

Nesse ponto, consegue-se obter os CAs que contornam o componente (NO,NA) , podendo considerar os $enlaces_SRLG$, e são destinados ao ND em questão, i.e. o nó adjacente ao componente (NO,NA) .

Para cada um dos demais NDs , diferentes do nó adjacente, que possui menores caminhos OSPF utilizando o (NO,NA) , deve-se obter os menores CAs com ISPF do NO até

cada um dos demais *NDs* (retirando agora o componente *NA*) de forma a contornar o *NA*. A equação 4.3 representa a chamada do ISPF para contornar o *NA*.

$$CA_s(ND) = ISPF(NO, ND, NA) \quad (4.3)$$

Caso algum dos enlaces do *NA* pertença a um SRLG, então há uma modificação na equação (4.3) para considerar não apenas o *NA* com seus enlaces, mas também os enlaces pertencentes ao SRLG. A equação (4.4) representa a chamada do ISPF considerando o *NA* e os enlaces_SRLG quando existir.

$$CA_s(ND) = ISPF(NO, ND, NA \cup enlaces_SRLG) \quad (4.4)$$

Como o *NO*, ao detectar uma falha real, não possui informações suficientes para identificar se a ocorrência de uma falha real é do enlace ou do nó adjacente, deve-se dar preferência para contornar sempre o *NA*. Ao contornar um *NA*, automaticamente está-se contornando o (NO, NA) . Portanto, apenas calculam-se *CA*s para contornar o (NO, NA) quando o *ND* é igual ao nó adjacente. Para os demais *NDs* afetados, deve-se obter *CA*s para contornar o *NA*. Caso não seja possível encontrar um *CA* viável, as condições especificadas na seção 4.1.1 não foram observadas.

O resultado total dessa primeira etapa é armazenado em um conjunto de vetores de nós, cada um representando uma seqüência de nós dos menores *CA*s do *NO* até um *ND* comprometido pelo componente retirado: Caminhos_Alternativos[*ND*]. Cada um desses caminhos indica qual a trajetória ideal por onde os pacotes deveriam ser desviados para alcançar um *ND* contornando um determinado componente. Se o *ND* for o nó adjacente, então os *CA*s contornam apenas o (NO, NA) . Se o *ND* for diferente do nó adjacente, então os *CA*s contornam o *NA*. Quando existir um SRLG, os *CA*s consideram todos os componentes integrantes do SRLG em ambas as gerações de caminhos.

Dentre os Caminhos_Alternativos[*ND*] para cada *ND* gerados e armazenados nesses vetores, deve-se identificar qual o melhor *CA* a ser utilizado. Para tanto, realiza-se uma busca em cada um deles pelo primeiro nó que, a partir dele, consiga utilizar o menor caminho original do OSPF até o *ND* e que consiga contornar o componente retirado sem ocasionar *loops* de roteamento. Essa busca ocorre a partir do *NO* seguindo cada um dos nós na seqüência do vetor até encontrar esse determinado nó que satisfaça a restrição. Esse nó especial é denominado *nó_CER*, e tem a função de indicar até onde um sub-caminho deve se estender, pois a partir do *nó_CER* os pacotes já podem ser encaminhados pelo menor caminho do OSPF com sucesso. O sub-caminho que se estende a partir do *NO* até esse *nó_CER* recebe a denominação de *caminho emergencial rápido*, i.e. CER. Cada *CA* para

um *ND* sempre possui um CER respectivo, sendo que no pior caso o *nó_CER* será o próprio *ND* e o CER será o próprio *CA*. Dentre os *CAs* possíveis com seus respectivos CERs para um *ND*, há a necessidade de escolher um CER (ou mais dependendo do caso) para ser o responsável por realizar o contorno do componente falho. Para tanto, uma classificação em níveis é realizada adaptando a idéia proposta no *framework* IPFRR descrita em [SHAND and BRYANT, 2008]. Porém, a classificação definida pela *CER_pró-ativa* é baseada em prioridades de acordo com a característica do *CA* e com a localização do *nó_CER*. Os três níveis de prioridade definidos são: *Equal Cost MultiPath (ECMP)*, *Loop Free Alternates (LFA)* e *Signaling (SIG)*. Após ter escolhido o CER mais prioritário, ele é armazenado em um *Vetor_CER[ND]*.

O nível ECMP é o nível de prioridade preferencial, sendo o único que consegue escolher mais de um CER, porém caso não seja possível utilizar esse nível, verifica-se o próximo nível nomeado nível LFA. Caso esse nível também não seja obedecido, utiliza-se o último nível SIG. Esses dois últimos níveis somente escolhem um único CER. Uma descrição do funcionamento de cada um dos níveis é abordada a seguir e a formulação englobando todos os níveis é apresentada na seqüência.

Os cálculos de distância dos caminhos utilizados nas fórmulas de cada um dos níveis fazem uso da base de informações do estado do enlace no *NO*, para que identifique as distâncias relativas de outros nós (considerando-os como a raiz da árvore).

4.1.3 Nível ECMP

O primeiro nível, ECMP, é o mais restritivo sendo apenas possível quando existirem um ou mais menores caminhos originais OSPF de mesmo custo até um *ND*. Quando se retira um componente da topologia para ser tratado, tanto (*NO,NA*) quanto *NA*, apenas uma parte desses menores caminhos originais é afetado e cada um dos menores caminhos restantes já é um *CA*. O *nó_CER* de cada um desses caminhos restantes localiza-se imediatamente no primeiro nó do *CA* após o *NO* (Nó Vizinho - *NV*), pois esses caminhos são caminhos originais OSPF que devem contornar naturalmente o componente retirado. Para realizar o desvio, basta o *NO* encaminhar os pacotes desviados para o *nó_CER*, i.e. *NV*, que os pacotes são encaminhados para o *ND* com sucesso. A formulação (4.5) encontra *CAs* de mesmo custo OSPF que o caminho original do *NO* para *ND*.

$$DistOSPF(NO,NA) + DistOSPF(NA,ND) = DistOSPF(NO,NV) + DistOSPF(NV,ND) \quad (4.5)$$

Sujeito a:

$$NA \notin OSPF_Nós(NV,ND), \text{ se } NA \neq ND \quad (4.5.a)$$

$$OSPF_Enlaces(NV,ND) \cap \begin{cases} SRLG(NO,NA) = \emptyset, \text{ se } NA = ND \\ SRLG(NA,?) = \emptyset, \text{ se } NA \neq ND \end{cases} \quad (4.5.b)$$

A restrição (4.5.a) impede o uso de *CA*s se existe um caminho OSPF original a partir do *NV* até o *ND* que contenha o *NA* quando o $NA \neq ND$. A restrição (4.5.b) impede o uso de *CA*s os quais existe um caminho OSPF original a partir do *NV* para o *ND* que contenha algum enlace pertencente ao $SRLG(NO,NA)$ se $NA = ND$, ou, caso contrário, pertencente ao $SRLG(NA,?)$, onde "?" indica qualquer nó adjacente ao *NA* (incluindo o *NO*). Essas duas restrições são necessárias, pois a partir do *NV*, pode existir um caminho OSPF que pode utilizar o componente simulado com falha. É importante citar que apenas o *NO* realiza o cálculo do *CA* com o componente retirado, devendo considerar apenas os caminhos originais OSPF nos demais nós.

O ECMP possui alta prioridade, pois o CER consegue contornar o componente retirado naturalmente, sem a possibilidade de *loops* de roteamento. Caso exista mais de um CER que utilize diferentes *nós_CER* (obtidos a partir de *CA*s que utilizam interfaces diferentes de saída a partir do *NO*), eles são utilizados para permitir um balanceamento de carga com distribuição igual de fluxos entre rotas de mesmo custo através do ECMP padrão [MOY, 1998].

É importante citar que poderia ser possível uma distribuição desigual de fluxos se fosse utilizada uma extensão proposta para o OSPF nomeada *Optimized Multipath* [VILLAMIZAR, 1999]. Essa abordagem, em particular, consiste em coletar informações de carga utilizada nos enlaces da topologia por cada nó e divulgá-las através de LSAs específicos. Dessa forma, os fluxos poderiam ser distribuídos desigualmente entre os caminhos ECMP, porém essa abordagem não revela a quantidade de carga que isso pode gerar e também foi abandonada pela IETF por falta de atualização dos seus autores.

4.1.4 Nível LFA

O segundo nível de prioridade, LFA, apenas pode ser utilizado quando o nível ECMP não puder ser obedecido. O nível LFA é possível quando, para um determinado *ND*, existe um ou mais *CA* que permitem naturalmente (utilizando as rotas originais do OSPF) o contorno do componente retirado apenas a partir do *NV*. Esses *CA*s não têm o mesmo custo do caminho original OSPF a partir do *NO* (nível ECMP), mas são os menores

caminhos originais OSPF a partir do NV ao NO . Caso exista algum CA com essa característica, o CER encontrado nesse caso é de nível LFA e se estende até o $nó_CER$, identificado como sendo o NV .

Uma adaptação da abordagem LFA original, que ainda está em desenvolvimento [ATLAS and ZININ, 2008], é realizada na formulação (4.6), que minimiza a distância do menor caminho original OSPF do NV até o ND . O processo de minimização também considera o número de nós $NumNósOSPF$ apenas se existir mais de um caminho LFA de mesmo custo devido ao fator de multiplicação 1000. O valor de 1000 é suficiente, pois não existe número de nós em um caminho OSPF maior que esse valor.

$$Min(1000DistOSPF(NV,ND) + NumNósOSPF(NV,ND)) \quad (4.6)$$

Sujeito a:

$$DistOSPF(NV,ND) < DistOSPF(NV,NO) + DistOSPF(NO,ND) \quad (4.6.a)$$

$$NA \notin OSPF_Nós(NV,ND), \text{ se } NA \neq ND \quad (4.6.b)$$

$$OSPF_Enlaces(NV,ND) \cap \begin{cases} SRLG(NO,NA) = \emptyset, \text{ se } NA = ND \\ SRLG(NA,?) = \emptyset, \text{ se } NA \neq ND \end{cases} \quad (4.6.c)$$

A restrição (4.6.a) evita o uso do NO como integrante do menor caminho OSPF original a partir do NV para o ND (*loop* de roteamento). As restrições (4.6.b) e (4.6.c) seguem a mesma definição das restrições (4.5.a) e (4.5.b) na formulação de nível ECMP, as quais evitam CAs que contenham um NV que, em seu menor caminho original OSPF para o ND , utiliza o NA (se $NA \neq ND$) e enlaces pertencentes ao SRLG.

Caso exista mais de um CER para um mesmo ND e que utilizem diferentes $nós_CER$, escolhe-se apenas um deles. Nesse caso, não se pode utilizar vários CERs, pois o OSPF não permite dividir os fluxos de pacotes com ECMP em vários caminhos que não tenham a mesma soma de custos de enlaces a partir do NO que o caminho original.

A adaptação do LFA original feita nesse nível também permite a escolha de um caminho mais adequado considerando o número de nós desse caminho, o que diminui a extensão do caminho percorrido. Já a abordagem LFA original indica a escolha de qualquer um dos LFAs possíveis sem considerar o número de nós [ATLAS and ZININ, 2008].

4.1.5 Nível SIG

O terceiro e último nível de prioridade, SIG, é necessário quando o CA não possui características que se encaixam em qualquer um dos primeiros níveis (ECMP ou LFA). Isso pode ocorrer naturalmente em uma topologia, uma vez que nem sempre o uso de

ECMP ou LFA é factível por motivos de estrutura da topologia e da distribuição dos custos aos enlaces. No nível SIG a seguinte regra deve ser obedecida: deve-se identificar qual dos nós na seqüência do *CA* analisado, após o *NV*, é distante o suficiente para conseguir, a partir dele, alcançar o *ND* contornando o componente retirado sem gerar *loop* de roteamento através do menor caminho original OSPF. A localização do *nó_CER* é feita percorrendo e verificando cada um desses nós a partir do *NV*, e cada nó a ser verificado é nomeado *nó_TMP*.

A verificação de cada *nó_TMP* é ditado por uma extensão da abordagem adaptada LFA na formulação (4.6). A formulação (4.7) minimiza a distância do sub-caminho (*NO* – *nó_TMP*), observando cada *nó_TMP* (na seqüência a partir do *NV*) do *CA* para o *ND*. A minimização também considera o número de nós do *CA* analisado (*NO* – *nó_TMP*), apenas se existir mais de um caminho com mesmo custo devido ao fator de multiplicação 1000.

$$\text{Min}(1000\text{DistCA}(\text{NO}, \text{nó_TMP}) + \text{NumNósCA}(\text{NO}, \text{nó_TMP})) \quad (4.7)$$

Sujeito a:

$$\text{DistOSPF}(\text{nó_TMP}, \text{ND}) < \text{DistOSPF}(\text{nó_TMP}, \text{NO}) + \text{DistOSPF}(\text{NO}, \text{ND}) \quad (4.7.a)$$

$$\text{NA} \notin \text{OSPF_Nós}(\text{nó_TMP}, \text{ND}), \text{ se } \text{NA} \neq \text{ND} \quad (4.7.b)$$

$$\text{OSPF_Enlaces}(\text{nó_TMP}, \text{ND}) \cap \begin{cases} \text{SRLG}(\text{NO}, \text{NA}) = \emptyset, \text{ se } \text{NA} = \text{ND} \\ \text{SRLG}(\text{NA}, ?) = \emptyset, \text{ se } \text{NA} \neq \text{ND} \end{cases} \quad (4.7.c)$$

A restrição (4.7.a) evita o uso do *NO* como integrante do menor caminho OSPF original a partir do *nó_TMP* para o *ND* (*loop* de roteamento). As restrições (4.7.b) e (4.7.c) seguem as mesmas definições das respectivas restrições do nível ECMP e LFA, as quais evitam *CAs* que contenham um *nó_TMP* que, em seu menor caminho original OSPF para o *ND*, faz uso do *NA* (se *NA* \neq *ND*) e de enlaces pertencentes ao SRLG.

O primeiro *nó_TMP* que satisfazer a formulação (4.7) é identificado como o *nó_CER*, e o CER se estende do *NV* até o *nó_CER* (*NV*, *nós_TMP* que não satisfizeram a formulação (4.7), nomeados *nós_intermediários*, e por fim o *nó_CER*).

Se existir mais de um CER para um mesmo *ND*, escolhe-se apenas um CER, pois da mesma forma que o nível LFA, o OSPF não permite dividir os fluxos de pacotes com ECMP em vários caminhos com custos maiores a partir do *NO* que o caminho original.

4.1.6 Exemplificação dos Níveis

A topologia ilustrada na Figura 4.1 é utilizada para exemplificar a verificação dos níveis de prioridade (ECMP, LFA e SIG). Considerando o nó *b* como sendo o *NO*, o (*NO, NA*) como sendo (*b, a*) e o *NA* *a*, a *CER_pró-ativa* calcula os menores *CAs* com *ISPF*

para os NDs a, c, f, i . A Tabela 4.2 ilustra os CAs obtidos para esses NDs.

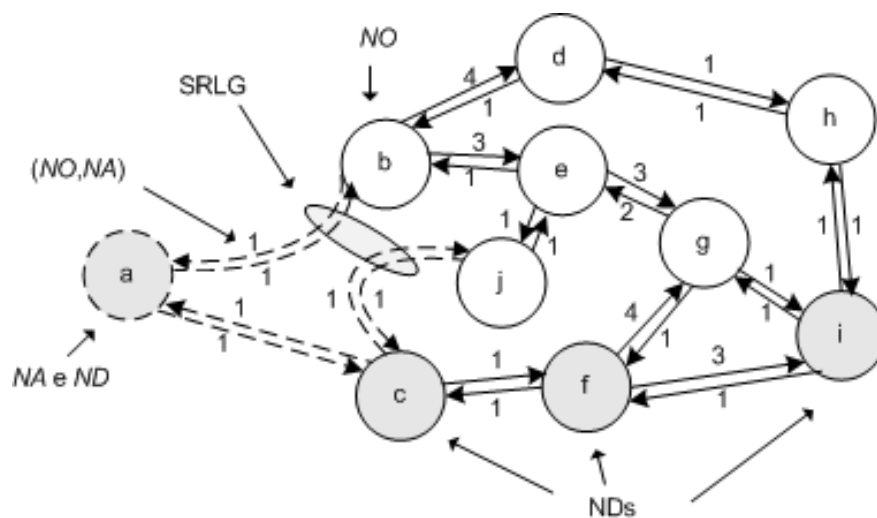


Figura 4.1 Topologia Exemplo para Identificação dos Níveis da CER_{pró-ativa}

Com os CAs gerados, o próximo passo é classificá-los em um dos níveis (ECMP, LFA e SIG), de acordo com as seções 4.1.3, 4.1.4 e 4.1.5, para identificar o *nó_CER* e obter o respectivo CER para o ND. Para o ND a , existem 2 CAs (e, g, f, c, a e d, h, i, f, c, a) que contornam apenas o (NO, NA) (b, a) e também os enlaces pertencentes ao seu SRLG, i.e. (j, c), pelo fato do ND a ser o NA. O nível obtido é SIG com o caminho e, g, f, c, a . Isso ocorre, pois com o *nó_TMP* em g existe um menor caminho OSPF que consegue atingir o nó a (nó g é o *nó_CER*), portanto, seguindo a descrição da seção 4.1.5, *DistCA* (de b até g) é 6, que multiplicado por 1000 e adicionado com *NumNosCA* (de b até g) de valor 2, gera um total de 6002. Já o caminho d, h, i, f, c, a encontra o seu *nó_CER* em i , e portanto *DistCA* é 6 e *NumNosCA* é 3 dando um total de 6003. O processo de minimização SIG escolhe e, g, f, c, a por ter um menor valor ($6002 < 6003$) e portanto o CER = $\{e, g\}$.

Tabela 4.2 CAs e CERs para a Topologia Exemplo

ND	Falha Simulada	CAs(ND)	CERs
a	$(NO, NA) + SRLG$	e, g, f, c, a (soma dos custos: 9) d, h, i, f, c, a (soma dos custos: 9)	SIG: $\{e, g\}$
c	$NA + SRLG$	e, g, f, c (soma dos custos: 8) d, h, i, f, c (soma dos custos: 8)	SIG: $\{d, h\}$
f	$NA + SRLG$	e, g, f (soma dos custos: 7) d, h, i, f (soma dos custos: 7)	LFA: $\{d\}$
i	$NA + SRLG$	d, h, i (soma dos custos: 6)	ECMP: $\{d\}$

Para o ND c , existem 2 CAs (e, g, f, c e d, h, i, f, c) que contornam o NA a por ser um

$ND \neq NA$ (nó $c \neq$ nó a). O nível obtido é SIG com o caminho d,h,i,f,c . Isso ocorre, pois com o nó TMP em h existe um menor caminho OSPF que seguramente consegue atingir o nó c (nó h é o nó CER), portanto, seguindo a descrição da seção 4.1.5, $DistCA$ (de b até h) é 5 e $NumNosCA$ (de b até h) é 2 dando um total de 5002. Já o caminho e,g,f,c encontra o nó CER em g , e portanto $DistCA$ (de b até g) é 6 e $NumNosCA$ (de b até g) é 2 dando um total de 6002. O processo de minimização SIG escolhe d,h,i,f,c com o $CER=\{d,h\}$.

Para o $ND f$, gera-se também CAs para contornar o $NA a$. Existem 2 CAs (e,g,f e d,h,i,f) e o nível obtido é LFA com o caminho d,h,i,f . Isso ocorre, pois com o NV em d existe um menor caminho OSPF que seguramente consegue atingir o $ND f$ (nó d é o nó CER), portanto, seguindo a descrição da seção 4.1.4, $DistOSPF$ (de d até f) é 3 e $NumNosOSPF$ (de d até f) é 3 dando um total de 3003. Já o caminho e,g,f somente consegue encontrar o nó CER com nível SIG (nó CER em g), o que perde em nível de prioridade para o LFA. O processo de minimização LFA, de mais alta prioridade que SIG, escolhe o caminho d,h,i,f com o $CER=\{d\}$.

Para o $ND i$, existe 1 CA (d,h,i,f,c) para contornar o $NA a$. O nível obtido é ECMP nesse caminho. Isso ocorre, pois satisfaz a descrição da seção 4.1.3 com sucesso, sendo o CA um menor caminho OSPF original que consegue contornar o $NA a$ (nó d é o nó CER). Portanto, escolhe-se d,h,i,f,c com o $CER=\{d\}$.

4.1.7 Formulação Geral da CER_ pró-ativa

Considere uma representação de topologia de rede utilizando um grafo $G(V,A)$, onde V é o conjunto de nós e A o conjunto de enlaces. Os nós representam os roteadores e as arestas representam enlaces físicos que conectam dois roteadores. A topologia deve obedecer às condições definidas para a existência da $CER_pró-ativa$ na seção 4.1.1.

Considere $(i, j) \in A$ como a representação de um enlace ligando os roteadores i e j . Para cada enlace (i, j) existe um custo atribuído $c_{i,j}$ por um administrador de rede na configuração do OSPF.

Cada nó da topologia realiza os cálculos de menores caminhos considerando o seu próprio nó como NO na formulação. Para o cálculo dos menores caminhos SPF busca-se minimizar a soma dos custos dos enlaces $c_{i,j}$ do NO até qualquer nó destino. A primeira formulação matemática da $CER_pró-ativa$ foi criada tendo como base a formulação da árvore de caminhos mínimos [AHUJA et al, 1993]. Para o cálculo dos menores caminhos

deseja-se encontrar o custo mínimo $Z_{NO,J,NA}$ obtendo os menores caminhos de NO até qualquer nó destino J , procurando evitar o componente NA , ou o (NO,NA) .

$$Z_{NO,J,NA} = \min \sum_{i=1}^n \sum_{j=1}^n c_{i,j} x_{i,j}$$

Sujeito a:

$$\sum_{j=1}^n x_{i,j} - \sum_{k=1}^n x_{k,i} = \begin{cases} n-1, & \text{se } i=1 \\ -1, & \text{se } i \neq 1 \end{cases} \quad (a)$$

$$c_{NO,j} = \infty \text{ se } j = NA, \text{ e se } J = NA \quad (b)$$

$$c_{i,j} = \infty \text{ se } j = NA, \text{ e se } J \neq NA \quad (c)$$

$$c_{i,j} = \infty \text{ se } (i,j) \in \begin{cases} SRLG(NO,NA), & \text{se } J = NA \\ SRLG(NA,?), & \text{se } J \neq NA \end{cases} \quad (d)$$

$$i \geq 0, j \geq 0, i \neq j ; x_{i,j} \geq 0 ; c_{i,j} \geq 0 \quad (e)$$

A restrição (a) seleciona os menores caminhos transmitindo $n-1$ unidades de fluxo a partir do NO e que são consumidas de 1 unidade em cada nó atingido pelo menor caminho. As restrições (b) e (c) impedem o uso do (NO,NA) ou o do NA (todos os enlaces adjacentes a NA) como integrantes de um menor caminho, pois define um custo ∞ aos enlaces. Os enlaces pertencentes ao mesmo SRLG de (NO,NA) , se o contorno é do (NO,NA) , ou os enlaces pertencentes ao mesmo SRLG de $(NA,?)$, se o contorno é do NA , são considerados em (d) recebendo um custo ∞ e conseqüentemente evitados no cálculo de menor caminho. Os resultados obtidos com a formulação acima, sem as restrições (b), (c) e (d) obtêm os menores caminhos, sendo resolvido com o algoritmo de Dijkstra, o qual é utilizado para geração dos caminhos OSPF. As restrições (b), (c) e (d) apenas são utilizadas se o menor caminho para um nó destino J utilizasse um desses componentes, portanto, os demais menores caminhos não são afetados. A solução para este problema é obtida com uma simples modificação do algoritmo de Dijkstra, ou *Incremental-SPF*, para considerar as restrições (b), (c) e (d).

Os menores caminhos que obedecem $Z_{NO,J,NA}$ são representados no conjunto $\varphi_{NO,J,NA}$. Considere os menores caminhos para um ND pertencentes a $\varphi_{NO,J,NA}$, onde $ND \in J$. Se NO tem os seus menores caminhos originais OSPF para ND (considerando todos os componentes da topologia) utilizando no mínimo o (NO,NA) , então estes menores caminhos $\varphi_{NO,J,NA}$ para ND são *CAs* para ND e representados no conjunto $\varphi_{NO,ND,NA}$. Cada *CA* de $\varphi_{NO,ND,NA}$ é representado como $ALT_{NO,ND,NA}$, e possui uma seqüência de nós de NO até ND , que deveriam ser seguidos pelos pacotes para contornar (NO,NA) ou NA .

A segunda formulação matemática da $CER_{pró-ativa}$ foi criada para possibilitar uma identificação de qual nó de $ALT_{NO,ND,NA}$ é capaz de contornar o componente simulado com falha, i.e. (NO,NA) ou NA , para atingir ND utilizando um menor caminho OSPF original. Para essa identificação, são utilizados os níveis de classificação (ECMP, LFA ou SIG baseados nas definições anteriores) para descobrir esse nó, que é nomeado $nó_CER$. Cada nó da seqüência de nós de $ALT_{NO,ND,NA}$ analisado como candidato a $nó_CER$ é representado por k na formulação. Portanto, independente do nível utilizado, gera-se um sub-caminho de cada $ALT_{NO,ND,NA}$, do NO para $nó_CER$, que é definido como $CER_{NO,ND,NA}$. Todos os $CER_{NO,ND,NA}$ obtidos para $\varphi_{NO,ND,NA}$ são candidatos a ser o CER escolhido para esse ND , sendo identificado como $S_CER_{NO,ND,NA}$.

$$Z'_{NO,ND,NA} = \min(1000custo_CER_{NO,ND,NA} + num_CER_{NO,ND,NA})$$

Sujeito a:

$$\varphi_{NO,ND,NA} \subset \varphi_{NO,J,NA} \quad (a)$$

$$ALT_{NO,ND,NA} \in \varphi_{NO,ND,NA} \quad (b)$$

$$CER_{NO,ND,NA} \subset ALT_{NO,ND,NA} \quad (c)$$

$$nó_CER = k, \text{ se } DistOSPF(NO,k) + DistOSPF(k,ND) = DistOSPF(NO,NA) + DistOSPF(NA,DR), \text{ \& } k \in Vizinhos_NO \quad (d)$$

$$nó_CER = k, \text{ se } DistOSPF(k,ND) < DistOSPF(k,NO) + DistOSPF(NO,ND), \text{ \& } k \in Vizinhos_NO \quad (e)$$

$$nó_CER = k, \text{ se } DistOSPF(k,ND) < DistOSPF(k,NO) + DistOSPF(NO,ND), \text{ \& } k \notin Vizinhos_NO \quad (f)$$

$$NA \notin OSPF_Nós(k,ND), \text{ se } NA \neq ND \quad (g)$$

$$OSPF_Enlaces(k,ND) \cap \begin{cases} SRLG(NO,NA) = \emptyset, \text{ se } NA = ND \\ SRLG(NA,?) = \emptyset, \text{ se } NA \neq ND \end{cases} \quad (h)$$

$$k \in CER_{NO,ND,NA}, \text{ se nenhuma restrição na sequencia (d), (e) ou (f) for possível} \quad (i)$$

$$k \in CER_{NO,ND,NA} \text{ e é o último nó, se } (nó_CER == k) \text{ \& } (\text{qualquer restrição na ordem (d), (e) ou (f) for possível}) \quad (j)$$

$$num_CER_{NO,ND,NA} = \begin{cases} 1, \text{ se a restrição (d) for possível} \\ NumNosOSPF(k,ND), \text{ se a restrição (e) for possível} \\ NumNosCER_{NO,ND,NA}, \text{ se a restrição (f) for possível} \end{cases}, \text{ se restrição (j) for possível} \quad (k)$$

$$custo_CER_{NO,ND,NA} = \begin{cases} DistOSPF(k,ND), \text{ se a restrição (d) ou (e) for possível} \\ DistCER_{NO,ND,NA}, \text{ se a restrição (f) for possível} \end{cases}, \text{ se restrição (j) for possível} \quad (l)$$

$$k \in ALT_{NO,ND,NA} ; NA \notin Vizinhos_NO \quad (m)$$

Para identificar $S_CER_{NO,ND,NA}$, a função custo $Z'_{NO,ND,NA}$ minimiza o valor de $1000custo_CER_{NO,ND,NA}$ considerando o número de nós $num_CER_{NO,ND,NA}$. Devido ao fator de multiplicação 1000, $num_CER_{NO,ND,NA}$ apenas influencia o processo de minimização se existir mais de um $CER_{NO,ND,NA}$ com mesmo valor de $custo_CER_{NO,ND,NA}$. O resultado da minimização é a identificação do $S_CER_{NO,ND,NA}$.

A localização do $nó_CER$ é obtida analisando cada nó na seqüência de cada $ALT_{NO,ND,NA}$, sendo esse nó representado pela variável k . A variável k apenas se torna o $nó_CER$ caso uma das restrições seja satisfeita nessa ordem: (d) para nível ECMP obedecendo ao item 4.1.3 com k sendo o NV , (e) para nível LFA obedecendo ao item 4.1.4 com k sendo o NV , ou (f) para nível SIG obedecendo ao item 4.1.5 e k sendo diferente de NV . Caso alguma dessas restrições seja obedecida, então k se torna o $nó_CER$, e é o último nó de $CER_{NO,ND,NA}$ segundo a restrição (j). Caso contrário, a variável k pertence ao $CER_{NO,ND,NA}$, segundo a restrição (i), e deve-se buscar outro k na seqüência de $ALT_{NO,ND,NA}$. Quando k não pertence ao conjunto de nós vizinhos a NO , que é verificado na restrição (f), então k se comporta como o $nó_TMP$ do item 4.1.5, e todos os $nó_TMP$ que não satisfizerem a restrição (f) são armazenados como $nós_intermediários$. A restrição (g) desconsidera um $ALT_{NO,ND,NA}$ quando existir um menor caminho OSPF a partir de k para ND que utilize NA , se $NA \neq ND$. A restrição (h) desconsidera um $ALT_{NO,ND,NA}$ quando existir um menor caminho OSPF a partir de k até ND que utilize algum enlace pertencente a um SRLG. Um $CER_{NO,ND,NA}$ obtido é um candidato a $S_CER_{NO,ND,NA}$ para o ND e está contido em $ALT_{NO,ND,NA}$ por ser um sub-caminho, conforme a restrição (b) e (c). A restrição (k) define um valor para $num_CER_{NO,ND,NA}$ de acordo com o nível obtido ECMP (d), LFA (e) ou SIG (f), sendo que define um valor 1 quando o nível ECMP for obtido para permitir vários caminhos de mesmo custo que o OSPF original (permitir balanceamento de carga com ECMP). A restrição (l) define um valor para $custo_CER_{NO,ND,NA}$ de acordo com o nível obtido ECMP (d), LFA (e) ou SIG (f).

Todos os $CER_{NO,ND,NA}$ para os $\varphi_{NO,ND,NA}$ são localizados por meio dessa classificação e encontra-se o melhor $S_CER_{NO,ND,NA}$ que satisfazer $Z'_{NO,ND,NA}$. Cada $S_CER_{NO,ND,NA}$ é armazenado em um vetor: Vetor_CER[ND].

Cada nó da topologia deve calcular $Z_{NO,J,NA}$ e em seguida $Z'_{NO,ND,NA}$ de forma distribuída, utilizando a base de dados do estado do enlace, previamente organizada pelo OSPF. Cada nó realiza apenas o cálculo referente ao seu nó (NO) para contornar cada um dos componentes adjacentes (NO, NA) ou nó NA , dependendo da localização do ND . Com os $S_CER_{NO,ND,NA}$ armazenados em Vetores_CER, é necessário adotar uma representação dos mesmos na FIB para que possam ser utilizados na presença de uma falha adjacente.

Essa representação é definida em forma de um par *CER_Marca/Interface de Rede(IR)*, cuja geração é detalhada na próxima seção.

4.1.8 Extensão Adicionada na FIB: CER_Marca/IR

Para representar os Vetores_CER na FIB, uma extensão é necessária na FIB e ela consiste de um par *CER_Marca/IR*.

Um *CER_Marca* é uma marca gerada pelo *NO* para representar um Vetor_CER selecionado na FIB. A *IR* é a interface de rede a ser seguida para a *CER_Marca* correspondente. A geração de todas as marcas é realizada após ter obtido todos os Vetores_CER. Cada *CER_Marca*, que for definida, indicará aos pacotes (através de marcação) o desvio dos pacotes para o respectivo Vetor_CER escolhido (mais detalhes sobre o processo de desvio será abordado mais adiante).

Uma marca *CER_Marca* consiste de 16 bits divididos em 10 bits para identificar o *NO* (*NO_id*), e 6 bits para identificar um Vetor_CER (*CER_id*). O *NO_id* consegue representar até 1024 nós_origens. O *CER_id* consegue representar até 64 Vetores_CER por cada *NO*. A restrição de 1024 *NOs* é mais que suficiente para representar uma única área com OSPF, uma vez que uma topologia com essa quantidade de nós ou mais gera um elevado tráfego de controle (LSAs) e não é escalável. Em situações práticas, a documentação da Cisco recomenda no máximo 200 nós [DOYLE, 1998] por área, e alguns testes verificados pela IETF apontam no máximo 350 nós [MOY2, 1998]. Durante as avaliações da E-CER, as 64 possíveis representações de Vetores_CER são também mais que suficientes, pois nas várias topologias artificiais e reais (ver Apêndice B) utilizadas para avaliação foram necessárias no máximo 30 representações de CERs em um nó isolado e em média foram necessárias de 12 a 15 por nó. A Figura 4.2 ilustra a separação dos 16 bits para a *CER_marca*.

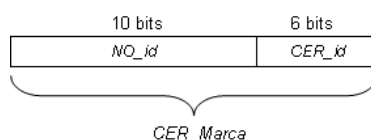


Figura 4.2 CER_Marca

Como esses 16 bits de uma *CER_Marca* são utilizados para marcar os pacotes durante o desvio, há a necessidade de representar esses bits nos cabeçalhos IPv4 e IPv6 para evitar o uso de encapsulamento. Para tanto alguns campos são reutilizados de cada

versão do cabeçalho IP:

- IPv4: O campo *Fragment Offset* pode ser reusado caso a abordagem [MATHIS and HEFFNER, 2007] seja usada. Esse campo somente é utilizado em caso de fragmentação do datagrama IP pelos nós da topologia, porém essa ocorrência é rara uma vez que o tamanho padrão de um segmento TCP e de um datagrama IP (limita também o UDP) são 536 e 576 bytes [POSTEL, 1983] respectivamente. Caso seja necessário um tamanho maior de segmento/datagrama, o mecanismo de *Path MTU Discovery* [MATHIS and HEFFNER, 2007] [MOGUL and DEERING, 1990] deve ser utilizado atualmente para identificar o tamanho máximo a ser utilizado entre as máquinas origem e destino sem haver fragmentação. Além do mecanismo *Path MTU Discovery* ser uma norma, ela também é adotada pelo IPv6 [DEERING and HINDEN, 1998]. Além disso, medições atuais de tráfego em topologias de *backbone* [WOLFGANG e TAFVELIN, 2007] revelam que apenas 0,06% de todo o tráfego teve fragmentação. Houve estudos para utilizar o campo *Options* para marcação dos pacotes, no entanto, tornou-se inviável pela necessidade de remontar o pacote IP a cada adição/retirada de marca o que ocasiona carga de processamento nos nós.
- IPv6: O campo *Flow Label* pode ser usado se especificar os primeiros 3 bits com valor "111", que identifica os 17 bits seguintes para uso futuro segundo a definição de uso desse campo para aplicações com requisitos de QoS [TANG et al, 2002], porém utiliza-se apenas 16 bits. O uso do campo *Flow Label* não é definido explicitamente em sua especificação [RAJAHALME et al, 2004], no entanto um dos seus objetivos é permitir o desenvolvimento de aplicações com QoS que influenciam as decisões dos nós do *backbone*. O fato de permitir a definição de parâmetros de QoS a partir da máquina origem pode afetar o desempenho do *backbone* com ataques de negação de serviço, segundo a própria especificação [RAJAHALME et al, 2004]. Esse problema pode ocorrer, pois cada usuário poderá alocar os maiores privilégios de tráfego para ele, o que dificultaria o gerenciamento por parte do administrador de rede. A falta de uma definição clara do uso do *Flow Label* para QoS deixa esse campo ainda em desuso e reforça o uso do típico *DiffServ* para redes IPv6 [NICHOLS et al, 1998], pois consegue-se prover QoS através do campo *Class of Service* ao invés do *Flow Label*, e ao mesmo tempo ter garantia de estabilidade no *backbone*.

Para representar o *IR*, utilizam-se 8 bits para atingir 256 identificadores de interfaces de rede por nó. Este número é mais que suficiente para os *hardwares* de roteadores atuais.

A *CER_pró-ativa* poderia gerar um par *CER_Marca/IR* para cada Vetor_CER obtido, mas para reduzir o número de *CER_Marca/IR* gerados, essa geração deve inicialmente obedecer a duas regras:

- Todas as entradas na FIB anunciadas por um mesmo nó destino (informação essa obtida da base de dados do estado do enlace), sendo esse nó um *ND* de acordo com a formulação *CER_pró-ativa*, devem usar o mesmo Vetor_CER[*ND*]. Neste caso, essas entradas da FIB devem referenciar um mesmo par *CER_Marca/IR* representante desse Vetor_CER, pois os pacotes com prefixos de rede destino anunciados por *ND* devem ser desviados para seguir o mesmo caminho do CER e alcançar *ND*.
- Se existir dois ou mais Vetor_CER para diferentes *ND*, porém com a mesma seqüência de nós no CER, então todas as entradas da FIB com prefixos de rede anunciados por estes *NDs* devem referenciar um mesmo par *CER_Marca/IR* representante desse CER. Isto é possível pois o CER será o mesmo para os diferentes *NDs*.

Com essas duas regras definidas, os pares *CER_Marca/IR* são então gerados e adicionados na FIB seguindo um dos dois casos: *NO_case* ou *Not_NO_case*.

O *NO_case* ocorre quando um nó gera seus próprios Vetores_CER. Este nó é portanto o *NO* na formulação *CER_pró-ativa*. Neste caso, para cada Vetor_CER[*ND*], uma *CER_Marca* é gerada com o *NO_id* correspondendo aos últimos 10 bits dos 32 bits de endereço *loopback* do *NO*. Essa abordagem é possível, pois os endereços de *loopback* são identificados como *Router ID* [KATZ et al, 2003] [MOY, 1998] [COLTUN et al, 2006] e são geralmente organizados pelos administradores de rede para estarem dentro de um mesmo grupo/faixa de endereços IP projetados para os nós IGP [DOYLE, 1998]. Outra possibilidade para gerar o *NO_id* é utilizar um valor de *hash* de 10 bits do endereço IP do *Router ID*. Os demais bits da *CER_Marca* correspondem ao *CER_id*, que é gerado de acordo com o nível de classificação usado na formulação (ECMP, LFA ou SIG) para obter o Vetor_CER[*ND*]. Se for obtido através do nível ECMP ou LFA então o *CER_id* terá 6 bits com valor "0". Se for obtido através do nível SIG, então o *CER_id* é um *número identificador* > 0 , que é incrementado a cada geração de *CER_Marca* nível SIG. Como o

Vetor_CER de nível SIG se estende a um *nó_CER* a frente do *NV*, a *CER_Marca* de nível SIG deve ser divulgada até o *nó_CER* através de uma das abordagens: *CER_Signal* ou *CER_pró-ativa_ext*, que são detalhadas mais adiante. Portanto, o *CER_id* somente será diferente de 0 se for utilizado o nível SIG, pois o desvio a ser utilizado pelos níveis ECMP e LFA termina no *NV* e não necessita de uma identificação diferenciada de *CER_id*. No caso do nível SIG é necessário ter o *CER_id* diferenciado, pois o Vetor_CER se estende até um *nó_CER* a frente do *NV* e os nós integrantes devem ter o par *CER_Marca/IR* adicionados na FIB (abordado em *Not_NO_case* a seguir) para realizar o correto desvio dos pacotes. O campo *IR* armazena a identificação da interface de rede do *NO* conectada ao primeiro nó do Vetor_CER[*ND*], i.e. *NV*.

O *Not_NO_case* ocorre quando um nó pertence a um Vetor_CER de nível SIG gerado em outro nó da topologia. Os nós pertencentes ao *Not_NO_case* são o *NV*, *nós_intermediários* e o *nó_CER* relacionados a um Vetor_CER[*ND*] gerado por um outro nó da topologia. Neste caso, o par *CER_Marca/IR* é obtido usando as informações de Vetor_CER[*ND*], *ND*, *NO*, *CER_Marca* adquiridos por uma das abordagens: *CER_Signal* ou *CER_pró-ativa_ext*, que são detalhadas na próxima seção. Com estas informações disponíveis, cada nó de *Not_NO_case* (*NV*, *nós_intermediários* e *nó_CER*) necessita somente identificar o *IR* para completar o par *CER_Marca/IR*. Para realizar essa identificação, considere um nó Y, como sendo um dos nós possíveis de *Not_NO_case*. Esse nó Y define *IR* como sendo a identificação da interface de rede conectada ao próximo nó, após Y, na seqüência de nós do Vetor_CER[*ND*].

Com todos os pares *CER_Marca/IR* gerados, a Figura 4.3 ilustra como estes pares são representados na FIB. Existem 3 pares *CER_Marca/IR* referenciados pelas entradas da FIB, mas para simplificar o exemplo, somente são mostradas duas entradas com os endereços destino anunciados por um mesmo nó (D). Estas duas entradas apontam um mesmo par *CER_Marca/IR* representante do Vetor_CER[*D*], de acordo com as duas regras iniciais de geração da *CER_Marca* definidas para representar esses dados na FIB, o que reduz o uso de recursos. A referência da entrada na FIB para um par *CER_Marca/IR* é alcançada através de um campo adicional nomeado *Ref* de 8 bits, que mostrou-se suficiente para referenciar todos os *CER_Marca/IR* obtidos nas avaliações feitas.

A quantidade de pares utilizados em um nó é altamente dependente da infraestrutura da topologia, pois além de gerar os pares relacionados aos Vetores_CER gerados em *NO_case*, um nó necessita adicionar os pares *CER_Marca/IR* gerados em

Not_NO_case. Em experimentos realizados com várias representações de topologia, foram utilizados no máximo 6 bits de *Ref*, e em média apenas 4 bits são necessários. Espera-se que os 8 bits forneçam um fator de escala suficiente para a quantidade de nós existentes em uma área do OSPF em topologias reais (entre 20 e 200 nós) e no máximo 1024 nós.

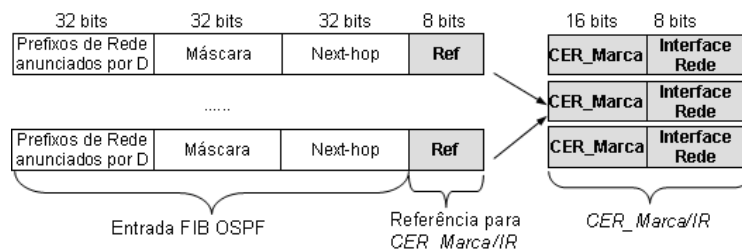


Figura 4.3 Adição da *CER_Marca/IR* na FIB

É importante enfatizar que apenas os prefixos de rede OSPF na FIB são atualizados com o campo adicional *Ref*, pois os prefixos de rede anunciados por BGP [REKHTER et al, 2006] existentes na FIB (se existirem) usam as rotas IGP para alcançar a referência *next_hop* BGP.

Em situações de várias áreas OSPF, as informações dos prefixos de rede de outras áreas OSPF ou BGP são sumarizadas pelos *Area Border Routers (ABR)* [MOY, 1998], i.e. nós de borda de uma área OSPF, que anunciam esse sumário de prefixos de rede para dentro da sua área OSPF. Da mesma forma que as regras são utilizadas para os prefixos de rede anunciados por um *ND*, esses prefixos de redes externos também seguem essa regra, sendo um *ABR* também um *ND* anunciador de prefixos de rede.

Com os pares *CER_Marca/IR* adicionados na FIB, os CERs podem ser utilizados na ocorrência de falhas e este procedimento é realizado pelo módulo do E-CER nomeado *CER_EDif*, que é descrito adiante.

4.1.9 CER_Signal e CER_pró-ativa-ext

Todos os Vetores_CER de nível SIG gerados por um *NO* estendem-se até um *nó_CER* a frente do *NV*. O *CER_EDif*, na presença de uma falha, deve ser capaz de desviar os pacotes do *NO* até o *nó_CER* impedindo *loops* de roteamento causado pelo encaminhamento IP padrão (rotas OSPF de nós ainda não atualizadas com o novo estado do enlace). Para tanto, algumas informações são previamente obtidas para que esses nós do Vetor_CER consigam realizar esse desvio de nível SIG. Essas informações são distribuídas a partir do *NO_case* (visto na seção anterior) com uma das abordagens possíveis:

CER_Signal e *CER_pró-ativa-ext*. Com essas informações distribuídas, o *Not_NO_case* realiza a adição da *CER_Marca/IR* na FIB corretamente.

O *CER_Signal* é uma extensão simplificada da troca de mensagens de controle OSPF (LSA) para prover previamente uma troca de informações relativas às marcas a serem utilizadas em Vetores_CER de nível SIG. Cada *NO* utiliza cada um dos Vetor_CERs obtidos e selecionados do nível SIG para identificar a seqüência de nós do CER (*NV*, *nós_intermediários* e *nó_CER*) que devem encaminhar diferencialmente os pacotes para desviá-los até o *nó_CER*. Uma mensagem então é gerada em *NO* para sinalizar esses nós utilizando uma definição genérica disponível de mensagens LSA no OSPF, nomeada *Opaque LSA* [COLTUN, 1998], que é especificada para prover futuras extensões para o OSPF. A mensagem gerada é do tipo *Opaque LSA Link-State Type 9*, que indica uma mensagem em escopo de enlace apenas (entre um nó e o seu nó adjacente, não sendo divulgado para os demais nós da topologia). O conteúdo dessa mensagem já foi gerado pelo *NO* e é composto por: Vetor_CER[*ND*], *NO*, *ND*, e a *CER_Marca*. Essa mensagem é enviada em escopo local para a interface de rede conectada ao primeiro nó da seqüência de nós do Vetor_CER, i.e. *NV*. Quando o pacote é recebido no *NV*, o processo da *CER_pró-ativa* recebe esse pacote e configura o par *CER_Marca/IR* (através do *Not_NO_case* descrito anteriormente) para o Vetor_CER obtido da mensagem *Opaque LSA*. Em seguida, a *CER_pró-ativa* desse nó atualiza a mensagem *Opaque LSA* como sendo um novo *Link-State Type 9* e envia em escopo local para a interface de rede conectada ao próximo nó na seqüência de nós do Vetor_CER (isto minimiza a quantidade de mensagens *Opaque*, uma vez que restringe a mensagem a um par de nós adjacentes [COLTUN, 1998]). Esse processo de encaminhamento e configuração dos nós com a mensagem *Opaque LSA* ocorre até alcançar o *nó_CER* (último nó do Vetor_CER). Ao atingir o *nó_CER*, essa mensagem é enviada de volta até chegar em *NO* através da seqüência reversa de nós do Vetor_CER. O objetivo é confirmar a definição do par *CER_Marca/IR* configurado em cada nó desse Vetor_CER. É importante salientar, que essas mensagens obedecem ao padrão de confiabilidade já existente nas mensagens do tipo LSA [MOY, 1998]. Ao atingir *NO* a mensagem é descartada confirmando com sucesso que o encaminhamento *CER_EDif* para esse CER pode ser utilizado quando necessário na presença de falha.

Este processo realizado pela abordagem *CER_Signal* é repetido para todos os Vetores_CER com marcas geradas de nível SIG no *NO_case*. Portanto, esse processo é importante para publicar a *CER_Marca* gerada pelo *NO* para o *NV*, *nós_intermediários* e

nó_CER, que devem configurar a respectiva *IR* a ser usada para essa *CER_Marca*.

A outra abordagem possível para obter a *CER_Marca* foi definida para evitar a troca adicional de mensagens entre os nós. Essa abordagem, nomeada *CER_pró-ativa-ext*, é uma extensão do cálculo *CER_pró-ativa*. Para entender seu funcionamento, considere um nó X que, após realizar o cálculo da *CER_pró-ativa* padrão, calcule quais Vetores_CER gerados de nível SIG pelos outros nós utilizariam X como *NV*, *nós_intermediários* ou *nó_CER*. O nó X realiza esse cálculo simulando cada nó da base de informações do estado do enlace, diferente de X, como sendo o *NO* e aplicando o cálculo padrão da *CER_pró-ativa* para gerar os Vetores_CER (a base de informações do estado do enlace contém todas as informações necessárias de enlaces e nós adjacente relacionados a todos os nós da topologia). A partir de então, utilizando o procedimento *Not_NO_case*, o nó X consegue identificar qual a *CER_Marca* para um Vetor_CER (utiliza o nó X em seu CER) e, também consegue identificar qual a *IR* está conectada ao próximo nó, após o nó X, na seqüência do Vetor_CER. Portanto, o nó X consegue identificar os Vetor_CERs e encontrar as respectivas *CER_Marcas/IR* sem trocar mensagens de controle entre os nós. Entretanto, este processo aumenta consideravelmente a complexidade (um cálculo *CER_pró-ativa* realizado para cada nó da topologia). Mesmo com esse aumento de complexidade, essa abordagem não afeta o encaminhamento dos pacotes OSPF, uma vez que pode ser executado em *background* em cada nó logo após o término de execução do processo OSPF.

4.2 Módulo: CER_EDif

O processo de encaminhamento padrão IP, ilustrado no fluxograma de execução na Figura 4.4, é baseado no IP de destino apenas. Esse processo realiza o descarte de pacotes quando a informação de *next-hop* for invalidada por uma falha, sendo que esse estado se mantém até que o OSPF termine seu processamento reativo.



Figura 4.4 Fluxograma do Encaminhamento IP Padrão

Para auxiliar esse processo durante a presença de falha, um encaminhamento diferenciado utiliza os pares *CER_Marca/IR* já existentes na FIB para realizar o desvio dos pacotes corretamente do *NO* até *ND* até o término da reação do OSPF. Esse encaminhamento diferenciado recebe o nome de *CER_EDif* e modifica o fluxograma de execução do encaminhamento IP padrão, conforme a ilustração da Figura 4.5. Se não existir uma falha na topologia, o encaminhamento padrão IP não sofre alterações e o encaminhamento pelo IP de destino é realizado normalmente (indicação (1) na Figura 4.5 nos quadros em cinza).

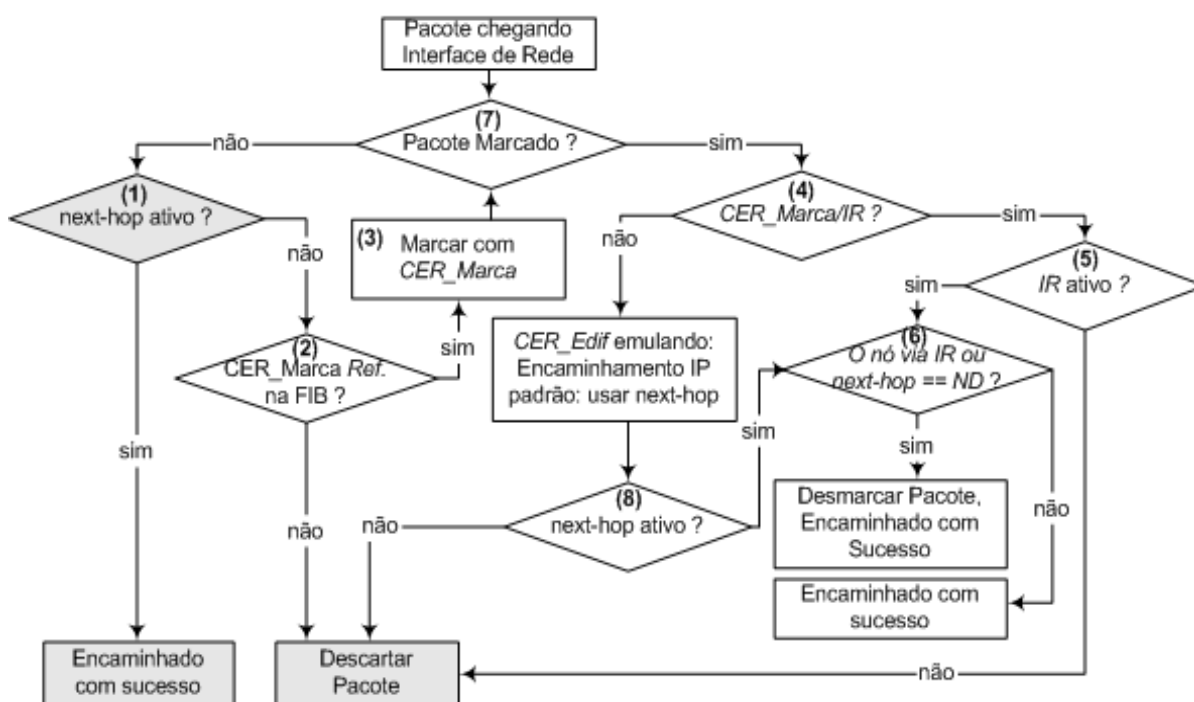


Figura 4.5 Fluxograma CER_EDif

Na presença de uma falha (indicação (1)), o *CER_EDif* é acionado pois o *next-hop* contém uma informação inválida (inativo). O nó que detecta a falha é o *NO* do caminho emergencial. O *CER_EDif* verifica se existe uma marca referenciada na FIB (indicação (2)) através do campo *Ref.* Se não existir essa marca, os pacotes são descartados por não ter um CER, sendo resultado de uma topologia que não permite 100% de cobertura de falha (seção 4.1.1). Se existir uma marca, os pacotes são marcados (indicação (3)) com *CER_Marca*. O objetivo é indicar qual Vetor_CER em específico deve ser usado no contorno da falha adjacente (sem ter o conhecimento se a falha é do *(NO,NA)* ou do *NA*). Com essa marca adicionada, o desvio dos pacotes é alcançado com um encaminhamento não apenas baseado no IP de destino, mas também na marca *CER_Marca* adicionada nos

pacotes, que indica uma *IR* a ser usada de acordo com o par *CER_Marca/IR* existente na FIB do nó (indicações (7) e (4)). Se a *IR* estiver ativa (indicação (5)) e o "nó via *IR*" não for *ND*, o encaminhamento é feito com sucesso (indicação (6)). Se a *IR* estiver inativa, então houve a ocorrência de múltiplas falhas independentes e descartam-se os pacotes.

Apenas o *NO* realiza a marcação dos pacotes para iniciar o desvio de forma diferenciada (indicações (2), (3), (7), (4), (5) e (6)). Os demais nós do Vetor_CER (*NV*, *nós_intermediários* até o *nó_CER*), ao receberem o pacote que foi marcado e desviado por *NO*, apenas verificam a marcação dos pacotes (indicação (7)) e identificam a respectiva *IR* a ser utilizada para encaminhamento (indicação (4)). Se a *IR* estiver ativa (indicação (5)) e o "nó via *IR*" não for *ND*, o encaminhamento é feito com sucesso (indicação (6)). Se a *IR* estiver inativa, então houve a ocorrência de múltiplas falhas independentes e descartam-se os pacotes desviados.

Ao atingir o *nó_CER*, a marca contida no pacote não é encontrada no par *CER_Marca/IR* existente na FIB desse nó (atingiu o final do Vetor_CER), e nesse caso, o *CER_EDif* emula o encaminhamento IP padrão (uso do *next-hop* das rotas OSPF) (indicações (4) e (8)). A marca é mantida nos pacotes para que os demais nós possam emular o encaminhamento IP (indicações (7), (4), (8) e (6)). A emulação do encaminhamento IP ocorre até que o "nó via *next-hop*" seja *ND* (indicação (6)). Nesse caso, a marca é então retirada para encaminhar os pacotes ao *ND*. A partir do *ND*, os pacotes podem ser encaminhados normalmente com base no endereço IP. A *CER_Marca* mantida nos pacotes também evita múltiplas recuperações (aumentaria a instabilidade na rede) causadas por múltiplas falhas independentes (SRLG não pertence a esse tipo) tanto no desvio do Vetor_CER (do *NO* até *nó_CER* os pacotes marcados são descartados na indicação (5)), quanto na emulação do encaminhamento IP até *ND* (do *nó_CER* até *ND* os pacotes marcados são descartados na indicação (8)). Isto é necessário, pois se existir outra falha que afete o desvio dos pacotes (Vetor_CER ou encaminhamento IP emulado até *ND*), esses pacotes são descartados para evitar uma nova ação do *CER_EDif* na tentativa de contornar essa nova falha.

Se o Vetor_CER identifica o *nó_CER = ND*, então o nó anterior ao *nó_CER* identifica que "nó via *IR*" é o *ND* (indicação (6)). Nesse caso, o nó encaminha para o *nó_CER* retirando a marca dos pacotes (nesse caso não há emulação do encaminhamento IP padrão por já alcançar o *ND*).

Esse procedimento de desvio dos pacotes é mantido até que o OSPF termine o

período de convergência, onde os *next-hops* são atualizados e o *NO* não mais realiza a marcação dos pacotes para desviá-los.

É importante ressaltar que o processo de encaminhamento realizado para o caso de múltiplas falhas independentes adotado pelo *CER_EDif* não existe em outra abordagem relacionada. De forma contrária, o Not-Via é incapaz de identificar esta ocorrência o que possibilitaria múltiplas recuperações gerando instabilidades na rede. Já a abordagem original LFA, nesse caso, revela também a possibilidade de *loop* de roteamento.

Se durante o processo de desvio, os pacotes desviados afetarem os demais fluxos não afetados pela falha, então o *CER_EDif* adota a seguinte regra ainda na indicação (6) antes de enfileirar o pacote para transmissão: somente continuar a encaminhar os pacotes desviados se o tamanho da fila de transmissão estiver menor que 80% da capacidade. Acima desse valor, o *CER_EDif* descarta os pacotes desviados. O valor de 80% de capacidade da fila é um valor estipulado que se mostrou suficiente em todas as topologias testadas para não prejudicar os demais tráfegos quando o CER estiver sendo usado. Ao contrário das demais abordagens de IPFRR, o *CER_EDif* evita a ocorrência de instabilidades na rede com congestionamentos, descartando os pacotes desviados (fila > 80%) o que afetaria outros fluxos de pacotes não prejudicados pela falha.

4.2.1 CER_EDif: Particularidades em Redes IPv4 e IPv6

No caso de redes IPv4, uma marcação extra é necessária: *CER_EDif* define o primeiro bit do campo *Flags* com "1" além da marcação com *CER_Marca*. Isso é necessário para indicar que o campo *Fragment Offset* foi modificado dentro da topologia de *backbone* devido ao desvio dos pacotes e não por um processo de fragmentação IPv4. Esse primeiro bit do campo *Flags* é indicado como "reservado" com valor "0" [DARPA, 1981]. Portanto, uma mudança no padrão desse bit é feita e pode ser realizada, pois o *CER_EDif* atua no desvio dos pacotes auxiliando o encaminhamento padrão IP. Quando os pacotes chegarem em *ND* esse bit é definido com "0".

No caso de redes IPv6, uma marcação extra não é necessária, pois o campo *Flow Label* somente poderia ser marcado pelas máquinas fim a fim do fluxo e não pelos nós da topologia de *backbone*.

Porém, se os bits a serem usados para *CER_Marca* no cabeçalho IP, tanto no IPv4, através do processo de fragmentação [DARPA, 1981], quanto no IPv6, com a definição de QoS por uma aplicação fim a fim [TANG et al, 2002], já estão utilizados, então é

necessário um processo alternativo de desvio: o *CER_EDif* utiliza um processo de encapsulamento. O *CER_EDif* encapsula os pacotes para o endereço do *ND* para então adicionar a marcação. Os pacotes seguirão o mesmo CER que o endereço IP original seguiria, pois o endereço de rede do IP de destino original dos pacotes é anunciado pelo *ND*. Quando os pacotes atingirem *ND*, os pacotes são desencapsulados.

É importante esclarecer que o processo de encapsulamento somente é usado quando os bits dos campos utilizados para marcação estão usados. Para tanto, antes do processo *CER_EDif* adicionar uma marca aos pacotes na presença de uma falha adjacente, ele verifica se o campo (*Fragment Offset* ou *Flow Label*) do cabeçalho já está utilizado. No caso do cabeçalho IPv4, verifica-se se o campo *Fragment Offset* está diferente de todos os bits com "0" e se o primeiro bit do campo *Flags* está desabilitado. Nesse caso, indica que houve um processo de fragmentação sendo necessário utilizar o processo de encapsulamento. No entanto, se o primeiro bit do campo *Flags* está ativo e o campo *Fragment Offset* possui todos os bits diferentes de "0", então indica a existência de um ambiente com múltiplas falhas independentes, pois um processo de desvio com *CER_EDif* já foi realizado por outro nó na topologia de *backbone*.

No caso do IPv6, verifica-se se o campo *Flow Label* contém os 3 primeiros bits diferentes de "111", o que indica uma marcação realizada pelas máquinas fim a fim com definição de QoS, e nesse caso utiliza-se o processo de encapsulamento. Caso os 3 primeiros bits estejam com "111", então indica um ambiente com múltiplas falhas independentes, pois um processo de desvio com *CER_EDif* já foi realizado.

De qualquer forma, os bits dos cabeçalhos usados para marcação são pouco utilizados atualmente pelos tráfegos de rede o que evita o uso do processo alternativo com encapsulamento. O campo *Fragment Offset* do IPv4 é pouco usado devido ao processo de *Path MTU Discovery*, o qual é amplamente disseminado pelas implementações TCP/IP hoje [WOLFGANG and TAFVELIN, 2007] além de ser uma padrão [MATHIS and HEFFNER, 2007] [MOGUL and DEERING, 1990]. Além disso, se esse processo não é usado na implementação das pilhas TCP/IP fim a fim, o tamanho padrão de segmento TCP e de datagrama IP devem ser utilizados: 536 e 576 bytes respectivamente [POSTEL, 1983]. O campo *Flow Label* do IPv6 ainda não está totalmente definido e, além disso, pode causar problemas de segurança e instabilidades por permitir aos usuários fim a fim priorizar seus fluxos [RAJAHALME et al, 2004]. Portanto, os administradores de rede IPv6 utilizam apenas o campo *Class of Service* para definição de QoS (apenas os nós de borda da rede

(ex. domínio *DiffServ*) conseguem determinar a prioridade dos fluxos de acordo com o administrador de rede).

4.2.2 Observações sobre a Execução do CER_EDif

Durante a ocorrência de uma falha real, o *CER_EDif* em *NO* identifica na FIB qual a marca a ser utilizada para desviar um determinado fluxo de pacotes com rotas para o IP de destino comprometido pela falha. Ao realizar o processo de desvio pela respectiva *IR* até que atinja o *nó_CER*, apenas os nós pertencentes ao Vetor_CER terão suas rotas OSPF para esse IP de destino afetadas pela falha durante o período de convergência. Isso ocorre pois estes nós são os únicos, no *CA* de *NO* até *ND*, que têm os seus menores caminhos afetados na sua árvore SPF. A prova para essa afirmação está na localização do *nó_CER*, que é o primeiro nó do *CA* que tem seus menores caminhos SPF para *ND* sem gerar *loops* e sem utilizar o componente simulado com falha ((*NO,NA*) ou *NA*, dependendo da localização de *ND*).

O *CER_EDif* tem por objetivo manter o processo de desvio durante o período de convergência para que o encaminhamento dos pacotes desviados por um CER possa atingir o *nó_CER*. A partir desse nó, os pacotes são encaminhados até *ND* utilizando o encaminhamento emulado do padrão IP utilizando as rotas OSPF já disponíveis. O caminho seguido pelos pacotes desviados é muitas vezes o mesmo caminho obtido pelo OSPF, a partir de *NO*, após os nós integrantes do Vetor_CER terminarem a atualização das rotas para o novo estado do enlace. O fato de não ser sempre o mesmo caminho OSPF deve-se à antecipação da falha, ou seja, se a falha for do (*NO,NA*), então o caminho percorrido pelo desvio pode ser maior que o necessário devido à preferência pelo contorno do *NA*, uma vez que o cálculo é pró-ativo. Em caso de falha do *NA*, então o caminho percorrido pelo desvio é o mesmo que o OSPF teria após a reação a partir do *NO*. Essa característica permite uma adaptação gradual da maioria ou até de todos os nós integrantes do Vetor_CER para o novo caminho OSPF.

Durante o período de convergência com a atualização dos nós com OSPF, o desvio dos pacotes feito pelo *CER_EDif* é mantido no *NO* por um espaço de tempo suficiente para que todos os nós do Vetor_CER tenham as rotas atualizadas com OSPF. Este intervalo pode ser especificado em 10 segundos de acordo com as configurações de tempo padrão das rotinas OSPF para recuperação (entre 8 e 9 segundos [IANNACCONE et al, 2004]). Porém, uma alternativa que atinge um espaço de tempo mais preciso é o uso da abordagem

oFIB [FRANCOIS and BONAVENTURE, 2005]. Essa abordagem estipula um intervalo de tempo para então atualizar a FIB no nó mais perto da falha (*NO*). Esse tempo é suficiente para que os demais nós, que tiverem as suas árvores SPF afetadas pela falha, tenham atualizado a FIB com as novas rotas OSPF. Essa abordagem impede a ocorrência de *loops* de roteamento durante o período de convergência, porém o *NO* descartaria os pacotes até o término desse intervalo de tempo. Utilizando o *oFIB* em conjunto com *CER_EDif*, não haveria descartes de pacotes e, ao invés disso, seriam encaminhados para o Vetor_CER. Ao fim do intervalo estipulado pelo *oFIB* em *NO*, o OSPF atualizaria a FIB e então o *CER_EDif* automaticamente interromperia a marcação e desvio dos pacotes, uma vez que as novas rotas do OSPF não mais são encaminhadas para um *next-hop* inacessível e conseqüentemente não acionam o *CER_EDif* (ver Figura 4.5).

4.3 Conclusão

Neste capítulo foi descrito os módulos integrantes da E-CER: *CER_pró-ativa* e *CER_EDif*. A *CER_pró-ativa* atua no plano de controle de cada nó da topologia para calcular os CERs através de níveis de prioridades (ECMP, LFA e SIG) e armazená-los em Vetores_CER. Os Vetores_CER são selecionados para ter uma representação na FIB em forma de *CER_Marca/IR* para que o módulo *CER_EDif* possa utilizá-los em caso de falha real. Define-se um processo de mensagens extras OSPF (Opaque LSA) com *CER_Signal* ou uma extensão *CER_pró-ativa-ext* para encontrar os Vetores_CER que utilizam nós distantes do nó origem (CERs de nível SIG). O módulo *CER_EDif* descreve a processo do E-CER pertencente ao plano de dados, que auxilia as rotinas de encaminhamento padrão IP em caso de falhas. Foi descrito como é realizado o desvio e o processo de marcação dos pacotes durante a presença de uma falha. No próximo capítulo uma implementação da abordagem E-CER é descrita e uma avaliação da mesma é feita em um simulador.

Capítulo 5

IMPLEMENTAÇÃO E AVALIAÇÃO DO E-CER

5.1 Introdução

Com os dois módulos integrantes do E-CER especificados, uma implementação foi realizada para a avaliação do E-CER. Primeiramente, será apresentado um algoritmo para tratar com o problema da formulação geral da *CER_pró-ativa*. Em seguida, esse algoritmo será adaptado em conjunto com o código OSPF (módulo *CER_pró-ativa*) e com o código de encaminhamento (módulo *CER_EDif*) do simulador de rede J-SIM [TYAN, 2008] para analisar o comportamento da abordagem em uma rede com atividade. A escolha desse simulador teve como fator determinante o código fonte do OSPF, que foi portado do projeto de implementação de protocolos de roteamento de livre distribuição nomeado Zebra [ZEBRA, 2008]. Esse projeto possui um código OSPF com um comportamento semelhante ao definido pela IETF [MOY, 1998]. Além disso, o simulador J-SIM fornece um ambiente de desenvolvimento modular para redes TCP/IP (ver Apêndice A) com interface bem definida, onde a separação dos componentes de roteamento e encaminhamento favoreceu o desenvolvimento dos módulos *CER_pró-ativa* e *CER_EDif*.

5.2 Implementação do módulo *CER_pró-ativa*

O propósito do módulo *CER_pró-ativa* é ser executado no plano de controle dos nós atuando em conjunto com as rotinas de roteamento OSPF. O objetivo é reutilizar as informações geradas pelo OSPF para gerar os CERs de acordo com a formulação descrita no capítulo anterior.

5.2.1 Algoritmo CER_ pró-ativa

O algoritmo *CER_ pró-ativa* foi desenvolvido com o propósito de obter os Vetores_CER e as respectivas marcas *CER_Marca/IR*. Os passos do algoritmo estão descritos na Tabela 5.1. Um nó que está sendo considerado para análise é denominado *NO* no algoritmo. Dada uma configuração de topologia, o algoritmo reutiliza os caminhos SPF (gerados pelo OSPF) a partir do *NO* (passo 1). Na sequência, o algoritmo obtém os elementos de $\varphi_{NO,ND,NA}$ para cada *ND* afetado por uma falha simulada na interface de rede podendo ser o enlace adjacente (*NO,NA*) ou nó adjacente *NA* (passos 2, 3, 4, 5 e 6) conforme visto no capítulo anterior. A seguir, busca-se o *nó_CER* utilizando os 3 níveis de classificação para então selecionar $S_{CER_{NO,ND,NA}}$ (passo 7). Por fim, obtém-se os Vetores_CER e as respectivas marcas *CER_Marca/IR* a partir dos $S_{CER_{NO,ND,NA}}$ escolhidos (passo 8).

Tabela 5.1 Algoritmo *CER_ pró-ativa*

Passos	Descrição dos Passos
1.	Obter os caminhos SPF a partir do <i>NO</i> utilizando os resultados da execução normal do OSPF.
2.	Para cada interface de rede ativa de <i>NO</i> , faça:
3.	Identificar quais <i>NDs</i> ficariam comprometidos na árvore SPF obtida no passo 1 se houvesse o desligamento dessa interface ativa. Para cada <i>ND</i> identificado faça:
4.	<i>ND</i> é igual ao <i>NA</i> ?
5.	Se SIM: Obtém-se o <i>ISPF</i> retirando o (<i>NO,NA</i>) a essa interface para encontrar os <i>CAs</i> para um $ND = NA$ a esse enlace. Caso o (<i>NO,NA</i>) pertença a um SRLG, os enlaces pertencentes ao mesmo SRLG são retirados da topologia para esse cálculo.
6.	Se NÃO: Já realizou o cálculo <i>ISPF</i> retirando o <i>NA</i> ? Se SIM: reutilizar o resultado do cálculo <i>ISPF</i> para <i>ND</i> , sem o <i>NA</i> , já realizado. Se NÃO: Obtém-se o <i>ISPF</i> retirando o <i>NA</i> a essa interface para encontrar os <i>CAs</i> para todos os <i>NDs</i> (passo 3) $\neq NA$ a esse enlace. Caso o (<i>NA,?</i>) pertença a um SRLG, os enlaces pertencentes ao mesmo SRLG são retirados da topologia para esse cálculo. O valor de "?" significa qualquer nó adjacente a <i>NA</i> .
7.	Utilizar o resultado do passo 5 ou 6 para encontrar em cada <i>CA</i> o <i>nó_CER</i> de acordo com os níveis de prioridade (ECMP, LFA, SIG). Ao identificar o <i>nó_CER</i> temos o CER também identificado e escolhe-se o CER de maior prioridade para cada <i>ND</i> .
8.	Com os CERs escolhidos, armazene-os em Vetores_CER[<i>ND</i>]. Verifica-se a possibilidade de reuso dos Vetores_CER conforme seção 4.1.8 e 4.1.9 para então gerar as marcas <i>CER_Marca/IR</i> .

A complexidade do algoritmo é $O(n^3 \log(n))$, onde n é o número de nós na topologia. Essa complexidade foi obtida a partir da quantidade de *loops* de repetições analisados como necessários para gerar todo o cálculo da CER_ pró-ativa. No entanto, o limite de 1024 nós da abordagem E-CER limita a grandeza dessa complexidade. O uso do *ISPF* também reduz a grandeza dessa complexidade restringindo os cálculos apenas nos caminhos que tenham os nós destinos afetados pelo componente retirado. Por fim, em um ambiente real, esse algoritmo é executado em *background* no plano de controle de cada nó após o término da convergência do OSPF, ou seja, com as rotas OSPF já calculadas e funcionais (ver capítulo 4). O objetivo é executar esse cálculo de forma a não influenciar a execução do plano de encaminhamento do nó com rotas OSPF.

5.2.2 Implementação da CER_ pró-ativa em Java

Ao invés de implementar o algoritmo direto em um simulador de rede, resolveu-se averiguar a sua implementação e sua execução primeiramente como um aplicativo desktop. Esse aplicativo gera os Vetores_CER e as respectivas CER_Marca/IR a partir de uma topologia de rede representada a partir de uma matriz de adjacência. O aplicativo foi desenvolvido em linguagem Java devido a sua portabilidade, facilidade e pelo código fonte em Java do simulador J-SIM, o qual foi utilizado para avaliação deste algoritmo em um ambiente simulado (descrito mais adiante). Para o cálculo dos menores caminhos, foi utilizado o pacote Java *Graphs and Dijkstra's algorithm* [GARNIER, 2002] de origem acadêmica. Porém, foram necessárias algumas modificações no código fonte desse pacote, pois havia algumas funcionalidades não implementadas, como por exemplo, o suporte a custos assimétricos de enlace.

Para representar SRLGs, a matriz de adjacência contém um atributo que indica quando um enlace pertence ou não a um SRLG. Essa abordagem simula a divulgação dos identificadores de SRLG através do OSPF: um atributo adicional obtido das mensagens LSA [KOMPELLA and REKHETER, 2005] e adicionados na RIB.

Uma classe *CER_ pró-ativa* foi definida com o método construtor contendo como parâmetro de entrada um arquivo em disco representando uma matriz de adjacência da topologia de rede. Esse método calcula o SPF padrão para obter os menores caminhos entre os nós. A geração desses caminhos SPF padrão emula a geração dos caminhos que o OSPF realizaria no passo 1 do algoritmo. Também foi definido um método público *calcula_Vetores_CER*, que recebe como parâmetro a identificação de um nó dessa

topologia. Essa identificação corresponde ao *NO* do algoritmo e é utilizada para calcular/obter todos os CERs para esse nó correspondendo aos passos 2, 3, 4, 5, 6, e 7 do algoritmo. Os VetoresCER e as respectivas *CER_Marca/IR* também são obtidas no método *calcula_Vetores_CER* correspondendo ao passo 8 do algoritmo.

Para realizar o passo 8, tanto o *CER_Signal* quanto a *CER_pró-ativa-ext* descritas na seção 4.1.9 são capazes de distribuir os Vetores_CER e as *CER_Marcas* entre os nós do Vetor_CER de nível SIG. Como essa implementação é um aplicativo desktop, a troca de mensagens entre os nós de rede fica inviável e optou-se por utilizar a *CER_pró-ativa-ext*. Seguindo a descrição da seção 4.1.9, os Vetores_CER e as *CER_Marcas* gerados pelos nós remotos são obtidos corretamente.

Para averiguar a implementação do algoritmo em Java, vários testes foram realizados com sucesso em uma variedade de configurações de topologias de teste, variando o número de nós entre 10 e 40. Utilizando marcadores de tempo em Java para início e fim da execução do método *calcula_Vetores_CER* por nó, procurou-se estimar aproximadamente o tempo de execução desse algoritmo. Sem aplicar otimizações elaboradas no desenvolvimento do código fonte, os testes (realizados em um computador Pentium IV barramento de 333MHz com 512MBytes) reportaram um tempo máximo aproximado de 590 milissegundos para calcular todos os Vetores_CER e identificar os *CER_Marca/IR* que cada nó deve conter.

5.2.3 Implementação da CER_pró-ativa no J-SIM

O próximo passo da implementação foi adaptar o código fonte da *CER_pró-ativa* no código fonte do J-SIM. Após analisar o código fonte desse simulador, o código da *CER_pró-ativa* foi adicionado no plano de controle do nó como um adendo ao código OSPF padrão do J-SIM para executar o algoritmo. A *CER_pró-ativa* foi programada para ser executada após o término do período de convergência do OSPF. Isso é necessário para obter a RIB atualizada do OSPF com o estado do enlace de toda a topologia e também o resultado do cálculo SPF feito pelo OSPF. Ao invés de usar o pacote *Graphs and Dijkstra's algorithm* para cálculo SPF, o algoritmo reusou o código SPF já instalado no OSPF do simulador, que basicamente é uma implementação do algoritmo de *Dijkstra*. Modificou-se o código fonte da *CER_pró-ativa* para usar *CER_Signal* ao invés do *CER_pró-ativa-ext* (seção 4.1.9), pois pode-se utilizar a opção de troca de mensagens entre os nós da rede com mensagens *Opaque LSA Link-State Type 9*. Essa troca de mensagens seguiu a semântica

descrita em [COLTUN, 1998] e foi implementada como uma classe especializada da classe LSA (responsável pelas mensagens LSA do OSPF) original do código fonte OSPF do J-SIM.

Para que a abordagem *CER_pró-ativa* possa utilizar a RIB atualizada pelo OSPF, um temporizador foi estipulado com um tempo de 10 segundos [FRANCOIS et al, 2005] suficiente para emular o término do período de convergência do OSPF em cada nó. Após esse tempo expirar, a execução da *CER_pró-ativa* pode garantidamente utilizar a RIB atualizada. Conforme visto no capítulo anterior, uma implementação do *oFIB* [FRANCOIS e BONAVENTURE, 2005] no J-SIM poderia estipular um valor mais preciso para o início da execução da *CER_pró-ativa*. Porém, como a descrição do *oFIB* ainda está em fase de elaboração pela IETF e a sua implementação no J-SIM está fora do escopo desse trabalho, ela deve ser realizada em trabalhos futuros.

Com a *CER_pró-ativa* programada para executar após o OSPF, todos os Vetores_CER locais de um nó da rede são gerados corretamente e, em conjunto com o *CER_Signal*, todos os Vetores_CER remotos são identificados para então gerar todas as *CER_Marca/IR* necessárias a um nó de rede. O próximo passo é preencher as *CER_Marca/IR* na FIB utilizando como referência um campo adicional *Ref*. As entradas na FIB a serem modificadas são obtidas da base de informações do estado do enlace mantidas pelo OSPF, que revela quais são os prefixos de rede anunciados por cada *ND* do domínio OSPF. O campo *Ref* é adicionado na FIB modificando as propriedades da classe de encaminhamento de pacotes do J-SIM. Uma classe Lista foi criada para armazenar as *CER_Marca/IR* obtidas. Os integrantes dessa lista são referenciados na FIB pelo campo adicional *Ref*.

Com a representação *CER_Marca/IR* adicionada nas entradas da FIB, o módulo *CER_EDif* pode realizar os desvios dos pacotes na presença de uma falha.

5.3 Implementação do módulo CER_EDif

O módulo *CER_EDif* é projetado para ser executado no plano de encaminhamento dos nós atuando em conjunto com o encaminhamento padrão IP. O objetivo é usar as informações de *CER_Marca/IR* na FIB para que sejam imediatamente usadas na presença de uma falha. O uso dessas informações realiza um encaminhamento diferenciado dos pacotes para que sejam desviados da falha corretamente.

5.3.1 Algoritmo e Implementação da CER_EDif no J-SIM

Um algoritmo foi desenvolvido para executar o módulo *CER_EDif* e a sua seqüência de passos é apresentada na Tabela 5.2. A descrição do funcionamento do *CER_EDif* na seção 4.2 e o fluxograma da Figura 4.5 foram utilizados como referência para o desenvolvimento deste algoritmo.

Tabela 5.2 Algoritmo *CER_EDif*

Passos	Descrição dos Passos
1.	- Pacote está marcado com <i>CER_Marca</i> (IPv4: <i>Fragmentation offset</i> está \neq "0" e <i>Reserved Bit</i> = "1", ou IPv6: <i>Flow Label</i> está com os primeiros 3 bits = "111") ?
2.	- Se SIM: Existe uma entrada na FIB para o endereço IP de destino do pacote com uma referência para <i>CER_Marca/IR</i> , ou uma <i>CER_Marca/IR</i> na FIB que é a mesma <i>CER_Marca</i> contida no pacote ?
3.	- Se SIM: o respectivo <i>IR</i> a essa marca está ativo e a capacidade da fila < 80% ?
4.	- Se SIM: o próximo nó acessado via <i>IR</i> é o <i>ND</i> ?
5.	- Se SIM: desmarcar pacote e encaminhá-lo pela <i>IR</i> . (FIM algoritmo).
6.	- Se NÃO: encaminhar pacote pela <i>IR</i> . (FIM algoritmo).
7.	- Se NÃO: descartar o pacote (FIM algoritmo).
8.	- Se NÃO: o <i>next-hop</i> contido na entrada da FIB (rota OSPF) está ativo e a capacidade da fila < 80% ?
9.	- Se SIM: o próximo nó acessado via <i>next-hop</i> é o <i>ND</i> ?
10.	- Se SIM: desmarcar pacote e encaminhá-lo para o <i>next-hop</i> . (FIM algoritmo).
11.	- Se NÃO: encaminhar pacote p/ <i>next-hop</i> . (FIM algoritmo)
12.	- Se NÃO: descartar o pacote. (FIM algoritmo).
13.	- Se NÃO: a entrada na FIB para o endereço IP de destino do pacote possui o <i>next-hop</i> ativo ?
14.	- Se SIM: usar o encaminhamento padrão IP com rota OSPF. (FIM algoritmo).
15.	- Se NÃO: o pacote IP possui o campo <i>Fragmentation offset</i> vazio com <i>Reserved Bit</i> = "0" (IPv4) ou <i>Flow Label</i> vazio (IPv6) ?
16.	- Se SIM: existe uma entrada na FIB para o IP de destino do pacote que tem referência para uma <i>CER_Marca/IR</i> ?
17.	- Se SIM: marcar o pacote com essa <i>CER_Marca</i> e vá para o passo 3.
18.	- Se NÃO: descartar o pacote. (FIM algoritmo).
19.	- Se NÃO: Encapsular o pacote com endereço IP de destino = <i>ND</i> e vá para o passo 16.

Esse algoritmo é capaz de marcar os pacotes na presença de uma falha, quando uma falha de um componente adjacente (nó ou enlace) for detectada, e encaminhar diferenciadamente os pacotes marcados de acordo com o *CER_Marca/IR*. Além disso, esse algoritmo realiza o encaminhamento IP padrão se os pacotes não estiverem marcados ou se não houver uma falha. As informações de *CER_Marca/IR* são geradas previamente pelo módulo *CER_pró-ativa*.

Esse algoritmo foi implementado em Java diretamente na classe responsável pelo encaminhamento padrão IP no J-SIM. O encaminhamento padrão IP somente é contornado quando existe um pacote marcado ou uma falha no *next-hop* (passo 1, ou 13 e 16). Caso não haja falha ou marca, o encaminhamento padrão IP é usado (passos 1, 13 e 14). Diante de uma falha, o algoritmo marca os pacotes (passo 17) para que sejam encaminhados de acordo com o par *CER_Marca/IR* e consigam atingir o *nó_CER* ou o *ND* (passos 2, 3, 4, 5, e 6). Se não existir um *CER_Marca/IR* para iniciar o desvio os pacotes, eles são descartados indicando que não existe um caminho emergencial para o destino dos pacotes (passo 18). A partir do *nó_CER* os pacotes são encaminhados seguindo as rotas OSPF até que atinjam *ND*, onde os pacotes são desmarcados (passos 8, 9, 10, e 11). Caso uma fragmentação IPv4 ou a QoS de aplicação fim a fim no IPv6 tenha sido usada nos pacotes, usa-se o recurso de encapsulamento (passo 15 e 19). Durante o caminho de desvio percorrido até *ND*, se existir outra falha que afete o desvio desses pacotes, então esses pacotes são descartados para evitar uma nova ação do *CER_EDif* na tentativa de contornar essa nova falha (passos 7 e 12). Em caso de mais de uma falha, o E-CER evita um ambiente mais instável na rede ao descartar os pacotes desviados. Uma reação do OSPF é desejável nesse caso uma vez que aumentaria a complexidade de uma solução emergencial se fosse tratado [NUCCI et al, 2003]. Além disso, o E-CER é planejado apenas para ser usado durante o período de convergência. Ao término da reação do OSPF no nó que está marcando e dando início ao desvio dos pacotes, o *CER_EDif* automaticamente encerra o processo de desvio (passo 17) pois as rotas OSPF nesse nó não mais utiliza um *next-hop* falho (passo 13 e 14). Os demais pacotes que ainda estão marcados trafegando na rede continuam o desvio (passo 1) até atingir o *ND*.

5.4 Análise Experimental

Além da abordagem E-CER implementada no J-SIM, foi necessário implementar a

abordagem Not-Via para realizar uma comparação. A escolha da abordagem Not-Via deve-se ao fato de ser a única abordagem que alcança 100% de cobertura da rede para uma única falha (falha de enlace, nó ou SRLG). A implementação do Not-Via seguiu a semântica descrita em [BRYANT et al, 2008] e foi adaptada para somente gerar os dados necessários para uma análise comparativa com o E-CER. De forma similar à abordagem E-CER, o Not-Via necessitou de várias modificações no código fonte das classes do OSPF (plano de controle) e do encaminhamento padrão IP (plano de encaminhamento) do J-SIM.

Para avaliar a abordagem proposta, 3 análises foram efetuadas:

- Extensão Total dos Caminhos de Recuperação (E-CER x Not-Via)
- Total de Informações Extras Adicionadas na FIB (E-CER x Not-Via)
- Avaliação do E-CER em Execução no Simulador

A primeira análise avalia a extensão dos caminhos de recuperação que os pacotes devem percorrer até atingir o nó destino da topologia em cada abordagem. Esse quesito permite identificar a quantidade de nós necessários para realizar o desvio dos pacotes, quanto maior o desvio maior a quantidade de recursos de rede necessários para desviar os pacotes. A segunda análise considera a quantidade de informações extras adicionadas na FIB que cada abordagem necessita para realizar os desvios. Quanto maior essa quantidade, mais recursos da FIB são necessários em cada nó. A terceira análise é a operação do E-CER em um ambiente simulado com tráfego, tendo por objetivo analisar qual a redução da quantidade de pacotes perdidos utilizando ou não o E-CER em conjunto com o OSPF durante o período de convergência.

5.4.1 Cenário de Testes e Simulação

Utilizou-se nesse estudo várias topologias reais e artificiais como cenário de testes e simulação. Essas topologias foram escolhidas por apresentarem *backbones* que atendiam às restrições apresentadas na seção 4.1.1. Foram escolhidas 11 topologias de *backbone* reais: 10 topologias (Abilene, AGIS, ATHome, AT&T, CAIS, Level 3, NFSNET, QWest, Servint e Sprint) obtidas com MapNet [CAIDA, 2008] e a GEANT2 obtida de <http://www.geant2.net>. A representação das 11 topologias está disponível no Apêndice B. Além dessas 11 topologias reais, 50 topologias artificiais com 32 nós e 64 enlaces foram geradas com *Boston University Representative Internet Topology Generator* (BRITE) [MEDINA et al, 2002] utilizando o modelo Waxman [WAXMAN, 2001]. A definição dos

parâmetros de Waxman utilizados foram os mesmos utilizados na avaliação feita em [HANSEN et al, 2006].

5.4.2 Parâmetros dos Testes e Simulação

Em todas as topologias reais os custos dos enlaces foram definidos com um valor inversamente proporcional à taxa de transmissão dos enlaces encontrados. Os custos dos enlaces definidos para as topologias artificiais foram de valor 1.

Por falta de informações de SRLGs nas topologias reais, alguns cenários aleatórios de SRLGs foram considerados (obedecendo às restrições de 4.1.1) e ambas as abordagens conseguiram obter caminhos de recuperação com sucesso.

Para realizar a primeira e segunda análise, executou-se o protocolo OSPF em conjunto com a abordagem E-CER para cada uma dessas topologias. Os CERs e as marcas *CER_Marca/IR* foram coletadas de todos os nós para fazer análise. Em seguida o mesmo cenário foi executado, porém com o OSPF em conjunto com a abordagem Not-Via. As informações referentes aos endereços *not-via*, *next-next-hop* e os caminhos Not-Via foram coletados de cada nó da topologia para fazer análise.

Para realizar a terceira análise, foi necessário configurar o código OSPF do J-SIM para que os nós possam alcançar um período de convergência < 1 segundo. Várias modificações nos temporizadores e nos procedimentos internos do código OSPF possibilitaram a detecção e reação mais rápida a uma falha. A maioria dessas modificações seguiu as descrições apontadas em [ALAETTINOGLU et al, 2000] [FRANCOIS et al, 2005] [IANNACCONE et al, 2004]. Com as alterações feitas no código fonte, foi possível reduzir o período de convergência para aproximadamente 200 milissegundos, o que fica de acordo com o mínimo aproximado alcançado em [FRANCOIS et al, 2005]. Durante a ocorrência de uma falha, o OSPF ou o E-CER somente é sinalizado após 20 milissegundos [FRANCOIS et al, 2005] (entre 0 e 20 milissegundos após uma falha, os pacotes são descartados no nó que está adjacente à falha). Com essa redução do período de convergência, foi possível avaliar a redução da perda de pacotes alcançada pelo E-CER durante o período de convergência do OSPF < 1 segundo. Para que as restrições do E-CER descritas na seção 4.1.1 possam ser 100% obedecidas, foi definido nessa análise um ambiente onde os enlaces ficariam ocupados em no máximo 50% de sua capacidade, o que é um ambiente planejado por muitos *backbones* reais [IANNACCONE et al, 2004] [IYER et al, 2003].

5.4.3 Análise da Extensão Total dos Caminhos de Recuperação (E-CER x Not-Via)

Com os dados obtidos de todos os nós de cada topologia testada, as Figura 5.1, 5.2 e 5.3 revelam os gráficos por topologia. Essa figura ilustra a porcentagem de todos os caminhos de recuperação por abordagem (E-CER com níveis ECMP, LFA e SIG, e Not-Via com abordagens originais ECMP e LFA) gerados em termos do número de nós a serem percorridos pelos pacotes para atingirem os *NDs*. As topologias artificiais obtidas com BRITE geraram resultados similares e foram sintetizadas em um único gráfico para simplificação.

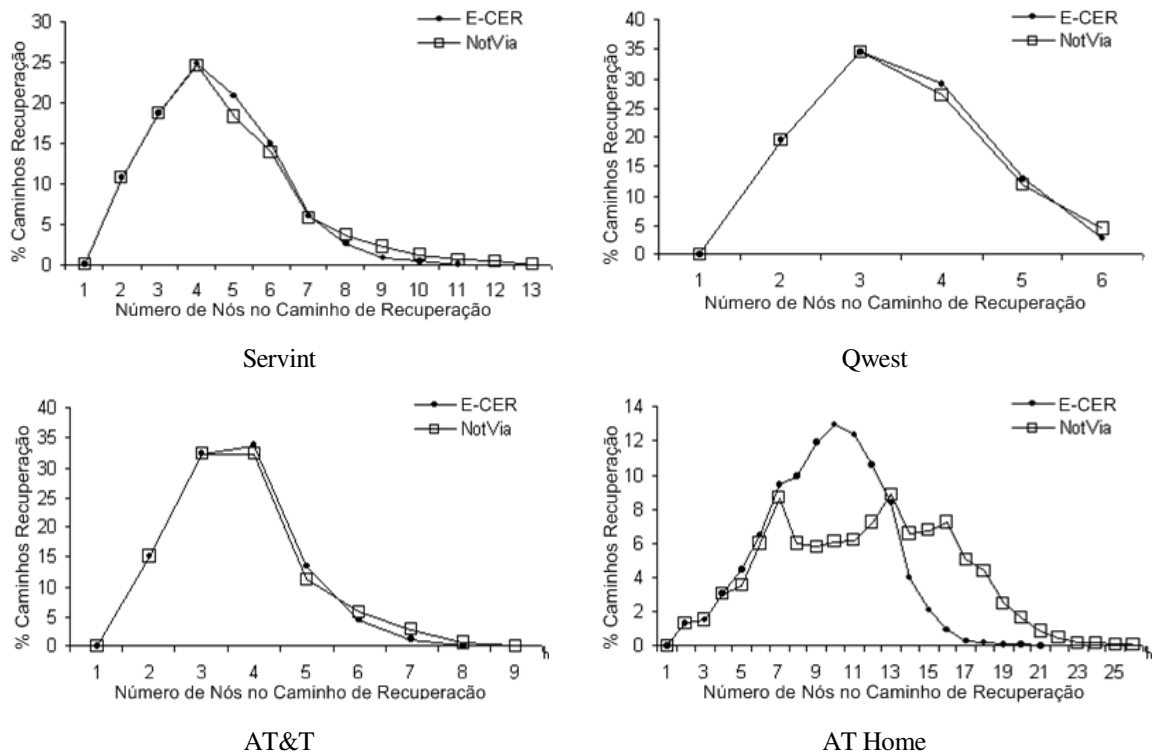


Figura 5.1 N° de Saltos Necessários para Alcançar os *NDs* (Servint, Qwest, AT&T, AT Home)

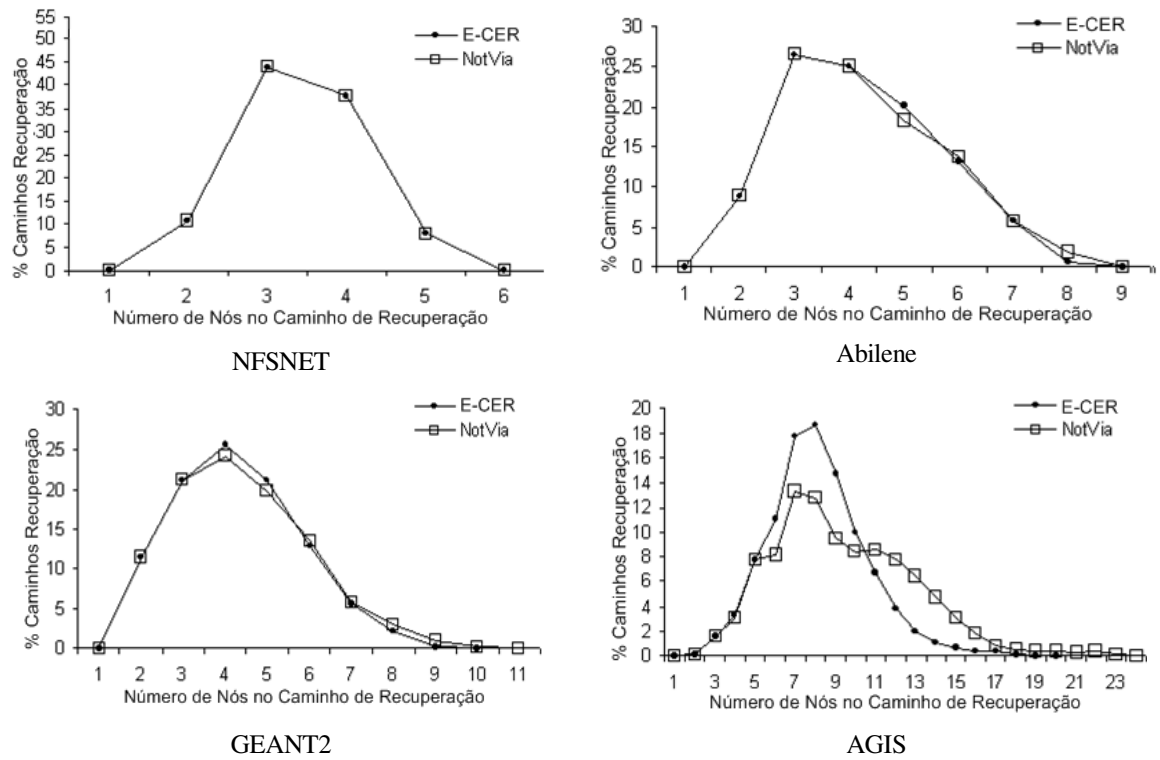


Figura 5.2 N° de Saltos Necessários para Alcançar os NDs (NFSNET, Abilene, GEANT2, AGIS)

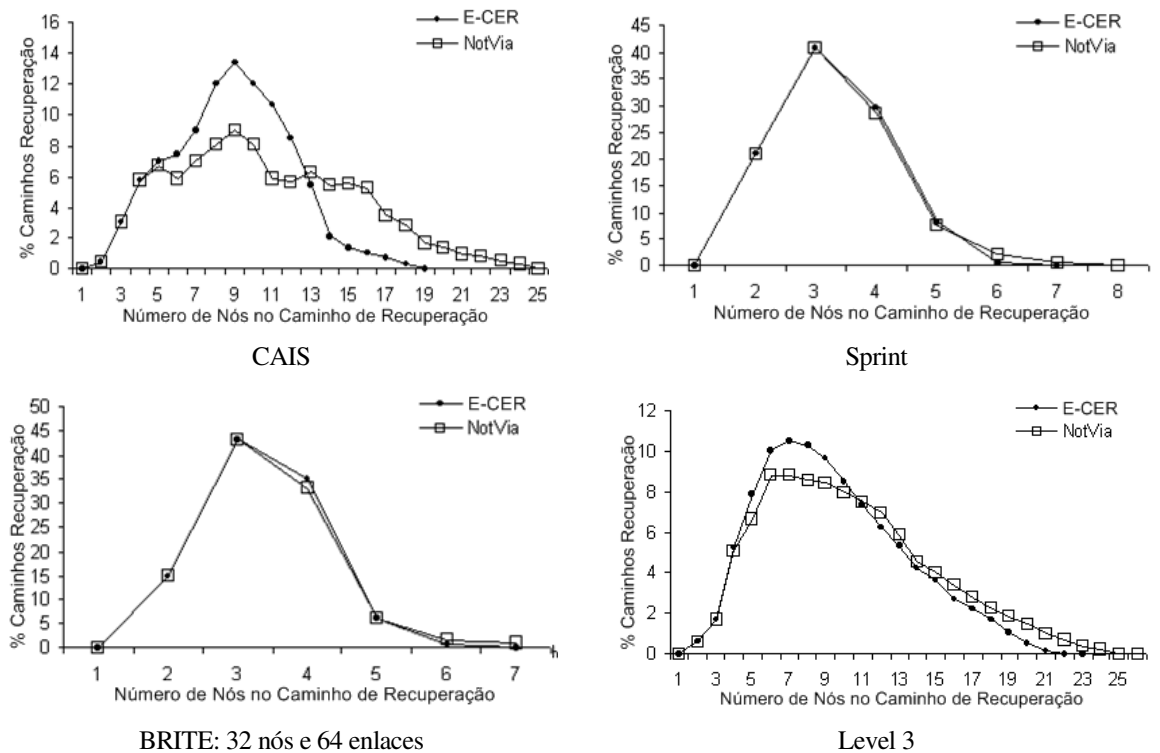


Figura 5.3 N° de Saltos Necessários para Alcançar os NDs (CAIS, Sprint, BRITE, Level3)

Os caminhos obtidos com ECMP, tanto com E-CER (nível ECMP) quanto com

Not-Via (indica a abordagem ECMP padrão para ser utilizada nesse caso), tiveram os mesmos resultados, pois ambos reusam quando possível o recurso do ECMP do OSPF. Já os caminhos obtidos com E-CER nível LFA conseguiram obter em alguns casos menores caminhos que o Not-Via. Isso ocorre pelo fato da abordagem Not-Via apenas indicar o uso da abordagem LFA original para ser usada quando possível [BRYANT et al, 2008]. Na abordagem original LFA [ATLAS and ZININ, 2008] se existir mais de um caminho LFA possível, escolhe-se qualquer um desses caminhos sem considerar o número de nós ou a distância dos caminhos. Já o processo de minimização do LFA adaptado ao nível E-CER considera a distância OSPF em conjunto com o número de nós no caso de vários caminhos LFA disponíveis, o que possibilita a escolha de um menor caminho de recuperação. No entanto, a diferença mais significativa obtida foi resultado das abordagens *Multi-Hop*.

Quando o nível ECMP ou LFA não for possível, a abordagem E-CER utiliza o nível SIG como *Multi-Hop* (obedecendo ao *framework* IPFRR visto na seção 3.4) e o Not-Via usa a sua própria abordagem como *Multi-Hop*. Na maioria das topologias analisadas, o nível SIG gera caminhos mais curtos que o Not-Via sendo estes os que mais influenciaram os resultados dos gráficos, uma vez que desvia os pacotes até o nó necessário (*nó_CER*) para então serem encaminhados com as rotas OSPF. Outra característica que justifica esse ganho é que os caminhos E-CER são quase sempre os mesmos caminhos obtidos com as rotas OSPF, a partir do nó adjacente à falha, após o término do período de convergência (ver seção 4.2.2). Já o Not-Via gera caminhos na maioria dos casos um pouco mais extensos por guiar os pacotes encapsulados até atingir o nó com o endereço *not-via*. Isso acarreta em um caminho mais extenso que o necessário comparado com E-CER, pois é apenas a partir do nó com endereço *not-via* que os pacotes são desencapsulados para seguirem o menor caminho OSPF até o nó destino. Esta característica é melhor visualizada nas topologias de AGIS, AT Home, CAIS e Level 3, pois possuem topologias com seqüências de nós em série. Isto faz com que os pacotes devam percorrer um caminho maior para atingir o nó com endereço *not-via* para então encaminhar com OSPF até o nó destino (o que justifica a grande diferença nos gráficos dessas topologias).

A única topologia na qual o Not-Via obteve os mesmos caminhos de recuperação em relação ao E-CER é a NFSNET. Isso ocorre devido a sua distribuição de nós na topologia. Essa distribuição permite à abordagem Not-Via gerar caminhos encapsulados para atingir os *next-next-hops* que são os mesmos caminhos que a abordagem E-CER utiliza para gerar os CERs e atingir o *nó_CER*, ou seja, *next-next-hop* = *nó_CER*. Quanto mais nós estiverem

interconectados em uma topologia de forma a ter *next-next-hop* = *nó_CER*, então mais próximos os caminhos de recuperação do Not-Via se igualam ao E-CER.

O número médio de nós utilizado nos caminhos de recuperação das abordagens por topologia é apresentado na Tabela 5.3. O E-CER revela um melhor ganho médio em termos do número de nós utilizados nos caminhos de recuperação comparado ao Not-Via.

Tabela 5.3 Número Médio de Nós nos Caminhos de Recuperação por Topologia

	E-CER	Not-Via
Servint	4,46 nós	4,65 nós
Qwest	3,53 nós	3,62 nós
AT&T	3,66 nós	3,76 nós
AT Home	9,46 nós	11,57 nós
NFSNET	3,42 nós	3,42 nós
Abilene	4,22 nós	4,29 nós
GEANT2	4,47 nós	5,0 nós
AGIS	8,1 nós	9,4 nós
CAIS	8,85 nós	10,69 nós
Sprint	3,26 nós	3,35 nós
BRITE	3,34 nós	3,43 nós
Level 3	9,45 nós	10,9 nós

Além de gerar caminhos de recuperação em geral mais curtos que o Not-Via, o E-CER possui a vantagem de não depender obrigatoriamente do processo de encapsulamento. Isso evita problemas de carga extra nos nós (processo de encapsulamento), além de evitar uma possível fragmentação (IPv4) ou até mesmo descarte dos pacotes por ultrapassar a MTU tanto no IPv4, com o bit de "*don't fragment*" (*DF*) ativado, quanto no IPv6 (não possui suporte a fragmentação) conforme visto na seção 4.2.1.

5.4.4 Análise do Total de Informações Extras Adicionadas na FIB (E-CER x Not-Via)

Os caminhos de recuperação gerados pelo E-CER são aqueles que se estendem do *NO* até o *nó_CER*: *NV* (se ECMP ou LFA) ou um nó distante o suficiente para possibilitar o contorno da falha com rotas OSPF (se SIG). Nesse último caso, o desvio dos pacotes pela marca segue um CER que é limitado pelo *nó_CER* distante. Após esse nó, os pacotes seguem o encaminhamento com rotas OSPF até *ND*. Como o número de nós utilizados no

CER é minimizado nas formulações, isso influencia diretamente na quantidade necessária de *CER_Marca/IR* adicionada na FIB dos nós. Além disso, existe um processo de reaproveitamento de *CER_Marca/IR* definida pelos *NO_case* e *Not_NO_case* na seção 4.1.8, o que ajuda a reduzir a quantidade de informações adicionadas na FIB.

Para comparar a quantidade de informações extras adicionadas na FIB pela abordagem E-CER e Not-Via, a Tabela 5.4 apresenta os procedimentos adotados por cada abordagem para representar suas informações para prover os caminhos de recuperação na FIB.

Tabela 5.4 Representação Usada pelas Abordagens E-CER e Not-Via na FIB

	E-CER	Not-Via
Informações Extras na FIB	Pares de <i>CER_Marca/IR</i> gerados de acordo com a seção 4.1.8 e referenciados (<i>Ref</i>) nas entradas existentes na FIB (prefixos de rede anunciados pelos nós do domínio OSPF).	Novas entradas na FIB para todos os endereços <i>not-via</i> divulgados na topologia, além de adicionar um <i>next-next-hop</i> por cada entrada existente na FIB (prefixos de rede anunciados pelos nós do domínio OSPF).
Quantidade em bytes para identificação	- <i>Ref</i> : 1 byte - <i>CER_Marca/IR</i> : 2 + 1 = 3 bytes	- Entrada <i>not-via</i> na FIB (<i>end. rede/IP + mask + next-hop</i>) [MOY, 1998], representado por <i>nova_ent_FIB</i> : 12 bytes - <i>next-next-hop</i> , ou <i>n-n-h</i> : 4 bytes

Uma média estimada das informações adicionadas na FIB foi realizada para cada uma das 11 topologias reais e para os resultados das 50 topologias geradas com BRITTE.

A Tabela 5.5 compara a quantidade aproximada de recursos (memória em bytes) necessárias em cada FIB dos nós para garantir a operação das 2 abordagens utilizando o padrão IPv4. A indicação de “Entradas_OSPF” representa o número de entradas na FIB referentes aos prefixos de redes IGP com rotas geradas pelo OSPF.

O E-CER demonstra um ganho significativo em termos de economia de recursos na FIB em todas as topologias. Isso é explicado principalmente pela extensão de cada Vetor_CER de nível SIG (até o *nó_CER*) minimizado nas formulações, o que resulta em menos nós com informações extras necessárias para ter um encaminhamento diferenciado. Além disso, há a definição da marca *CER_Marca/IR* (menor que o *next-next-hop*) e o reuso das marcas quando os Vetores_CER puderem ser reutilizados. De forma contrária, o Not-Via necessita que todos os nós tenham as mesmas informações extras sobre endereços *not-*

via para conseguir encaminhar além dos *next-next-hop*.

Tabela 5.5 Quantidade de Informações Adicionadas na FIB por Topologia

	E-CER	Not-Via
Servint	$12 \times (CER_Marca/IR) + Entradas_OSPF \times (Ref)$ Total: 36 + Entradas_OSPF bytes	$58 \text{ novas_ent_FIB} + Entradas_OSPF \times (n-n-h)$ Total: 696 + 4×(Entradas_OSPF) bytes
Qwest	$10 \times (CER_Marca/IR) + Entradas_OSPF \times (Ref)$ Total: 30 + Entradas_OSPF bytes	$93 \text{ novas_ent_FIB} + Entradas_OSPF \times (n-n-h)$ Total: 1116 + 4×(Entradas_OSPF) bytes
AT&T	$11 \times (CER_Marca/IR) + Entradas_OSPF \times (Ref)$ Total: 33 + Entradas_OSPF bytes	$72 \text{ novas_ent_FIB} + Entradas_OSPF \times (n-n-h)$ Total: 864 + 4×(Entradas_OSPF) bytes
AT Home	$36 \times (CER_Marca/IR) + Entradas_OSPF \times (Ref)$ Total: 108 + Entradas_OSPF bytes	$106 \text{ novas_ent_FIB} + Entradas_OSPF \times (n-n-h)$ Total: 1272 + 4×(Entradas_OSPF) bytes
BRITE	$10 \times (CER_Marca/IR) + Entradas_OSPF \times (Ref)$ Total: 30 + Entradas_OSPF bytes	$128 \text{ novas_ent_FIB} + Entradas_OSPF \times (n-n-h)$ Total: 1536 + 4×(Entradas_OSPF) bytes
Abilene	$9 \times (CER_Marca/IR) + Entradas_OSPF \times (Ref)$ Total: 27 + Entradas_OSPF bytes	$30 \text{ novas_ent_FIB} + Entradas_OSPF \times (n-n-h)$ Total: 360 + 4×(Entradas_OSPF) bytes
GEANT2	$12 \times (CER_Marca/IR) + Entradas_OSPF \times (Ref)$ Total: 36 + Entradas_OSPF bytes	$76 \text{ novas_ent_FIB} + Entradas_OSPF \times (n-n-h)$ Total: 912 + 4×(Entradas_OSPF) bytes
AGIS	$31 \times (CER_Marca/IR) + Entradas_OSPF \times (Ref)$ Total: 93 + Entradas_OSPF bytes	$149 \text{ novas_ent_FIB} + Entradas_OSPF \times (n-n-h)$ Total: 1788 + 4×(Entradas_OSPF) bytes
CAIS	$30 \times (CER_Marca/IR) + Entradas_OSPF \times (Ref)$ Total: 90 + Entradas_OSPF bytes	$88 \text{ novas_ent_FIB} + Entradas_OSPF \times (n-n-h)$ Total: 1056 + 4×(Entradas_OSPF) bytes
Sprint	$9 \times (CER_Marca/IR) + Entradas_OSPF \times (Ref)$ Total: 27 + Entradas_OSPF bytes	$60 \text{ novas_ent_FIB} + Entradas_OSPF \times (n-n-h)$ Total: 720 + 4×(Entradas_OSPF) bytes
NFSNET	$8 \times (CER_Marca/IR) + Entradas_OSPF \times (Ref)$ Total: 24 + Entradas_OSPF bytes	$41 \text{ novas_ent_FIB} + Entradas_OSPF \times (n-n-h)$ Total: 492 + 4×(Entradas_OSPF) bytes
Level 3	$22 \times (CER_Marca/IR) + Entradas_OSPF \times (Ref)$ Total: 66 + Entradas_OSPF bytes	$184 \text{ novas_ent_FIB} + Entradas_OSPF \times (n-n-h)$ Total: 2208 + 4×(Entradas_OSPF) bytes

Para ilustrar essa diferença de forma comparativa em um exemplo, supondo um valor de 1000 *Entradas_OSPF* em cada nó, o seguinte gráfico apresentado na Figura 5.4 revela os valores aproximados de recursos na FIB usado tanto pelo E-CER quanto pelo Not-Via em cada topologia analisada.

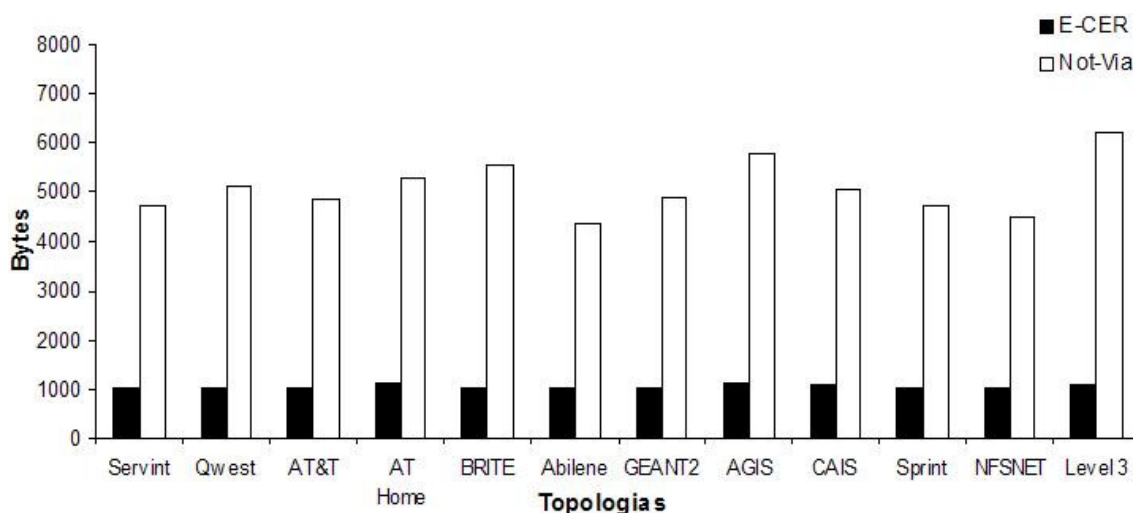


Figura 5.4 Exemplo do Uso de Recursos na FIB por Topologia (E-CER x Not-Via)

5.4.5 Avaliação do E-CER em Execução no Simulador

Com o E-CER completo adicionado ao código original OSPF do J-SIM, uma avaliação do seu comportamento foi realizada para verificar a redução da taxa de pacotes perdidos em relação ao OSPF configurado com período de convergência < 1 segundo.

Para realizar essa avaliação utilizou-se a topologia GEANT2 por ter sido analisada com sucesso em [FRANCOIS et al, 2005] com um período de convergência < 1 segundo. A representação da topologia GEANT2 é ilustrada na Figura 5.5 e possui enlaces de 10Gbps, todos com tráfego para ocupar 50% de sua capacidade (ver seção 5.4.2). Os testes são focados em 6 fluxos de tráfego (origem, destino): (2,18), (18,2), (1,17), (17,1), (9,11) e (11,9). Os menores caminhos destes 6 fluxos de tráfego estão planejados para utilizar o nó 6, enlace 6-8 e nó 8.

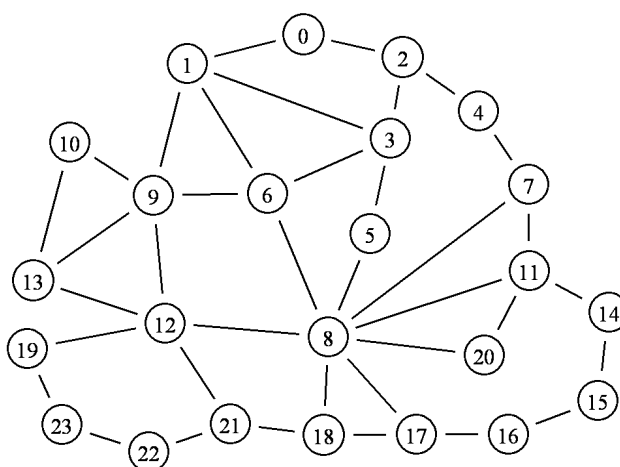


Figura 5.5 Topologia GEANT2

Todos estes 6 fluxos são gerados com taxa de bits constante em 1600Mbps com pacotes de 256 bytes para ajustar o enlace 6-8 a usar no máximo 4800Mbps (menor que 50%) em ambas as direções do enlace. Para realizar a avaliação do comportamento da E-CER *online* no simulador, foram projetados três cenários de falha independentes: enlace 6-8, nó 6 e nó 8. Estes cenários têm por objetivo analisar o pior caso de falha: nós centrais de uma topologia.

Primeiramente os testes foram realizados com o OSPF modificado (período de convergência < 1 segundo) e depois, o OSPF modificado com o E-CER para auxiliar o OSPF durante o período de convergência. Todos os pacotes dos fluxos de tráfego analisados, quando desviados para outros enlaces, podem ser ajustados na largura de banda extra (50% da capacidade dos enlaces de 10 Gbps). Este cenário isolado permite analisar a taxa de perdas de pacotes causadas pelo período de convergência, o que não pode ser realizado em um ambiente com alta taxa de tráfego não planejado (perdas por congestionamentos não ocasionados pelo período de convergência). Entretanto, o E-CER, num ambiente com alta taxa de tráfego na topologia, consegue impedir a existência de congestionamento causado pelo desvio desses pacotes através do descarte dos mesmos para não afetar os demais fluxos de tráfego em outros enlaces. Isso é possível, pois o E-CER apenas continua a desviar os pacotes se a capacidade da fila de transmissão dos nós está menor que 80% da capacidade da fila de transmissão (seção 4.2.2). Esse tipo de avaliação será realizado mais adiante.

A Tabela 5.6 apresenta a percentagem de pacotes perdidos durante o período de convergência (~ 200 milissegundos) para cada um dos cenários de falha e para cada fluxo de pacotes analisado. O OSPF, mesmo tendo um período de convergência < 1 segundo tem uma alta taxa de pacotes perdidos em todos os cenários e em todos os 6 fluxos analisados devido a informações inconsistentes (FIB não atualizada ainda e por *loops* de roteamento momentâneos) de rotas no nó que está adjacente à falha. O E-CER consegue reduzir a taxa de pacotes perdidos ao atuar em conjunto com o OSPF durante o período de convergência, pois durante esse tempo, o módulo *CER_EDif* atua desviando corretamente os pacotes. A pequena taxa de pacotes perdidos com a abordagem E-CER deve-se ao retardo de sinalização da falha de 20 milissegundos (ver seção 5.4.2).

Durante o período de convergência do OSPF, obteve-se um ganho expressivo na redução de pacotes perdidos utilizando o E-CER, o que cumpre o objetivo da abordagem. A abordagem Not-Via não foi analisada por já se saber que consegue obter esse mesmo

ganho na redução dos pacotes perdidos durante o período de convergência.

Tabela 5.6 % Pacotes Perdidos durante o Período de Convergência (200ms)

	OSPF < 1 segundo			OSPF + E-CER		
	Enlace 6-8	Nó 6	Nó 8	Enlace 6-8	Nó 6	Nó 8
(2,18)	60,9 %	62,5 %	65,7 %	3,5 %	6,6 %	6,9 %
(18,2)	59,7 %	61,4 %	65,4 %	3,2 %	6,4 %	6,9 %
(1,17)	60,0 %	63,1 %	68,0 %	3,6 %	6,5 %	7,0 %
(17,1)	59,1 %	62,0 %	67,1 %	3,3 %	6,3 %	6,7 %
(9,11)	59,3 %	61,7 %	66,2 %	3,2 %	6,5 %	6,9 %
(11,9)	59,8 %	61,5 %	68,1 %	3,4 %	6,3 %	7,1 %

Depois de 200 milissegundos, todos os nós atualizaram as rotas OSPF na FIB, e o *CER_EDif* não mais é utilizado pois os pacotes já seguem as novas rotas atualizadas. Porém, o *CER_EDif*, nessa avaliação, é programado para manter o desvio dos pacotes por um tempo suficiente de 10 segundos para garantir que todos os nós do domínio OSPF tenham convergido (caso não terminem o período de convergência).

Em ambas as abordagens (OSPF e OSPF + E-CER), o tráfego de fundo definido para ocupar 50% da capacidade dos enlaces não foi afetado (taxa de perdas = 0 %) pelos pacotes desviados devido ao cenário planejado de tráfego. Porém, em casos de tráfego excedente ao planejado (acima de 50% da capacidade dos enlaces) a abordagem E-CER consegue evitar a ocorrência de congestionamentos causados pelos tráfegos desviados durante o período de convergência.

Para simular esse ambiente, o cenário anterior foi modificado para avaliar a influência do tráfego desviado em relação aos demais tráfegos não afetados pela falha. Nesse novo cenário, o tráfego de fundo, que antes era planejado para ocupar 50% da capacidade do enlace, agora ocupa 80% (8000 Mbps). Quando o nó que está adjacente à falha desviar o tráfego, esse tráfego irá influenciar os demais fluxos de pacotes (tráfego de fundo). Alguns custos dos enlaces foram modificados para que os 6 fluxos utilizem obrigatoriamente o enlace 9-12/12-9 após a ocorrência das falhas. Para avaliar a influência desses 6 fluxos desviados, definiu-se, por simplificação, apenas dois fluxos de dados (9,12) e (12,9) de fundo a serem analisados para usar esse enlace com taxa aproximada de 8000 Mbps. Alterou-se também o tempo de manutenção do desvio realizado pelo E-CER de 10 segundos para 500ms. Esse valor mostrou-se suficiente para demonstrar o desvio do tráfego utilizando E-CER ainda após o período de convergência do OSPF (~200ms), uma

vez que valores acima desse tempo (o valor padrão antes era de 10 segundos) causaram grande sobrecarga no simulador pela quantidade de objetos gerados no simulador em um ambiente de redes Gigabit. Durante esse tempo de 500 milissegundos, o E-CER mantém o desvio do tráfego, após esse tempo as rotas OSPF são utilizadas. A Tabela 5.7 apresenta uma média dos resultados obtidos durante 500 milissegundos de análise após o início do período de convergência.

Tabela 5.7 % Pacotes Perdidos do Tráfego de Fundo (500ms)

	OSPF < 1 segundo			OSPF + E-CER		
	Enlace 6-8	Nó 6	Nó 8	Enlace 6-8	Nó 6	Nó 8
(9,12)	15,1 %	16,9 %	15,2 %	0,0 %	0,0 %	0,0 %
(12,9)	25,2 %	22,3 %	10,8 %	0,0 %	0,0 %	0,0 %

Observa-se que a abordagem OSPF faz com que o desvio dos pacotes prejudique o tráfego de fundo em ambos os cenários de falha (enlace 6-8, nó 6 e nó 8). Essa ocorrência é resultado do OSPF não possuir uma política de descarte de pacotes dos pacotes desviados. Já com a abordagem E-CER, não houve perda de pacotes durante os 500 milissegundos relacionados ao tráfego de fundo, pois apenas continuou a desviar os pacotes pelo enlace 9-12 e 12-9 se a capacidade da fila de transmissão estava menor que 80 %. Após 500 milissegundos, o E-CER interrompe o desvio e o encaminhamento padrão IP com as rotas OSPF é utilizado. Nesse momento, os tráfegos de fundo possuem o seu encaminhamento afetado pelos demais fluxos agora encaminhados normalmente com as rotas OSPF.

5.5 Análise Geral do E-CER

O E-CER atinge seus objetivos fornecendo uma abordagem que garante 100% de cobertura de uma única falha (enlace, nó ou SRLG). Essa abordagem encontra em geral caminhos de recuperação menores que a abordagem Not-Via, o que reduz a quantidade necessária de recursos para possibilitar o desvio os pacotes. Além disso, necessita de menos recursos em cada nó para utilizar os caminhos de recuperação quando comparado com Not-Via. A operação do E-CER em um ambiente simulado com tráfego de rede satisfaz os objetivos reduzindo significativamente a taxa de pacotes perdidos durante o período de convergência.

5.6 Conclusão

Nesse capítulo foi apresentado a implementação do E-CER em um simulador de rede (J-SIM). Foram definidos os cenários de testes e as avaliações realizadas. Utilizou-se a abordagem Not-Via para confrontar resultados com E-CER por ser a única abordagem que consegue 100% de cobertura de falha. Foram realizados 3 análises: a extensão dos caminhos de recuperação, a quantidade de informações extras adicionadas na FIB e a simulação do E-CER em execução junto com OSPF. O E-CER apresentou um significativo ganho em relação à abordagem Not-Via quando considerado os caminhos de recuperação, e apresentou resultados bem expressivos em relação à quantidade de informações extras adicionadas na FIB sem depender do processo de encapsulamento. Os testes em um ambiente com tráfego simulado verificaram a usabilidade do E-CER em conjunto com o OSPF para reduzir a taxa de pacotes perdidos durante o período de convergência. Os resultados no simulador também revelam que E-CER consegue evitar instabilidades causadas pelo desvio dos pacotes em ambientes com alta taxa de tráfego.

Capítulo 6

CONCLUSÃO E TRABALHOS FUTUROS

6.1 Conclusão

Nesta tese foram abordados os conceitos dos planos de controle e de encaminhamento que atuam em um nó. Os protocolos de roteamento também foram abordados, sendo os responsáveis por organizar a distribuição de tráfego na topologia em que atuam. Esses protocolos se dividem em protocolos intra-domínio e inter-domínio. Os protocolos de roteamento intra-domínio atuam no interior de sistemas autônomos enquanto os protocolos inter-domínio atuam entre sistemas autônomos. Tendo o foco em protocolos intra-domínio, o OSPF é o principal exemplo desse tipo de protocolo e também o indicado pela IETF para tal função. Foi apresentado um estudo do OSPF apontando as suas características de funcionamento. Pelo fato de ser um protocolo de estado do enlace, todos os nós da topologia necessariamente trocam mensagens para que cada um tenha uma visão completa da topologia (cada nó possui a mesma base de estado do enlace). Esse protocolo gera rotas para cada nó da topologia em que atua com base na busca pelo menor caminho considerando a soma dos custos dos enlaces. Qualquer mudança na configuração da topologia (manutenção de equipamentos, expansão da topologia ou falhas) faz com que todos os nós devam atualizar as suas bases de estado do enlace para que tenham a configuração correta da topologia, o que torna esse protocolo reativo. Essa característica faz com que esse protocolo passe por uma série de etapas para que todos os nós possam recalculer suas rotas. Essa série de etapas ocasiona um período de convergência, o qual gera um período instável com rotas erradas ou inexistentes ocasionando grande quantidade de pacotes descartados. Esse tipo de problema tem sido agravado atualmente com o aumento da velocidade dos *backbones* e pelo aumento dos usuários e aplicativos que utilizam as infra-estruturas de rede para os mais diversos fins.

Foram apresentadas as características das falhas em topologias de redes. Vários estudos revelam a ocorrência comum de falhas e que a maioria delas possui característica transitória e única (no máximo alguns minutos e atingem apenas um enlace, nó ou SRLG). Como o protocolo OSPF é reativo, um período de convergência sempre existirá na presença de uma falha. Vários trabalhos procuraram reduzir o tempo necessário nas etapas do período de convergência, o que possibilitou ao OSPF reduzir todo esse período para décimos de segundo. No entanto, essa redução mostrou-se dependente de equipamentos e estrutura de topologia o que pode provocar em vários casos a criação de um ambiente com rotas instáveis. Outras abordagens que atuam em escopo local foram propostas para auxiliar o protocolo OSPF e o encaminhamento IP a reduzir a taxa de pacotes perdidos durante a ocorrência de falha. Essas abordagens se dividem em reativas e pró-ativas. As abordagens reativas não alcançam um nível satisfatório de uso pelo fato do OSPF já ser reativo e trabalhar em escopo global, além de poder ter o tempo de reação reduzido em décimos de segundo. Além disso, as abordagens reativas avaliadas não atingem 100% de cobertura da topologia (falha de enlace, nó ou SRLG). As abordagens pró-ativas apresentam melhores resultados por já tornar disponíveis rotas alternativas no momento em que uma falha ocorrer. Dentre as abordagens estudadas, existem aquelas que seguem o framework *IP FastRerouting (IPFRR)* desenvolvido pela IETF: ECMP, LFA, FIR, MRC e Not-Via. Essas abordagens mostraram-se como as mais promissoras e foram detalhadas nesse trabalho. Identificou-se que apenas a abordagem Not-Via foi capaz de atingir 100% de cobertura de falha (enlace, nó ou SRLG) em uma topologia mesmo com custos de enlaces assimétricos. No entanto, algumas restrições foram apontadas: a dependência da técnica de encapsulamento (pode ocasionar o descarte de pacotes em redes IPv4 e IPv6), a adição de uma quantidade considerável de informações extras na FIB além de gerar caminhos de recuperação mais extensos que o necessário.

Essa tese definiu a abordagem Esquema de Caminhos Emergenciais Rápidos (E-CER) que adota o *framework IPFRR* e é capaz de atingir 100% de cobertura de falha. Um CER é um caminho que se estende de um nó (adjacente à falha) até um nó especial denominado *nó_CER*. Esse nó especial possui rotas OSPF que não são afetadas pela falha considerada. Quando ocorrer uma falha, o nó que está adjacente a ela deve desviar os pacotes para que utilizem o CER e atinjam o *nó_CER* e assim contornar a falha. Foi definido o projeto conceitual do E-CER que identifica dois módulos: *CER_pró-ativa* e *CER_EDif*. O módulo *CER_pró-ativa* é responsável por calcular os CERs no plano de

controle de um nó baseado nas informações já processadas do OSPF. Definiu-se uma formulação para gerar os CERs o qual criou 3 níveis de classificação: ECMP, LFA e SIG. O nível ECMP reutiliza o recurso de múltiplos caminhos de mesmo custo já existente no OSPF, porém para ser utilizado quando for possível contornar uma falha. O nível LFA adaptou a abordagem LFA original para realizar uma melhor escolha de um LFA dentre vários LFA possíveis quando isso ocorrer, o que reduz a extensão do caminho de recuperação. Tanto o nível ECMP quanto o LFA identificam o *nó_CER* no *NV*. O último nível, denominado SIG, é utilizado quando o nível ECMP ou o nível LFA não puder ser utilizado. Esse nível encontra o *nó_CER* além do *NV*, o que implica em um encaminhamento diferenciado para que os pacotes, quando desviados de uma falha, atinjam o *nó_CER*. No entanto, esses nós necessitam ter informações extras para poder realizar esse encaminhamento. O módulo *CER_pró_ativa* desses nós obtém essas informações através da *CER_pró_ativa-ext* ou do *CER_Sig*. Com todos os CERs necessários a um nó, eles são representados na FIB com um campo adicional *CER_Marca/IR*, que representa uma marca (*CER_Marca*) e a respectiva interface de rede (*IR*) que deve ser utilizada para encaminhamento.

Com os *CER_Marca/IR* adicionados na FIB, o módulo *CER_EDif* é o responsável por aplicar essas informações no encaminhamento dos pacotes para desviá-los de uma falha quando necessário. Esse módulo é responsável por marcar os pacotes quando uma falha adjacente for detectada ou por encaminhar pacotes já marcados para seguirem um CER e atingir o *nó_CER* corretamente. Definiu-se um procedimento para uso da *CER_Marca* nos pacotes em redes com tecnologia IPv4 ou IPv6, além de um procedimento de encapsulamento quando a simples marcação não for possível. Esse módulo ainda impede múltiplas ações de desvio em caso de múltiplas falhas independentes ou quando o desvio dos pacotes ameaçar um congestionamento de outros fluxos de tráfego não afetados pela falha. Isto se torna possível ao manter o encaminhando dos pacotes com *CER_EDif* mesmo após atingir o *nó_CER*, sendo que nesse caso o *CER_EDif* emula o encaminhamento IP padrão com as rotas OSPF até atingir o nó destino de saída da topologia. Se ocorrer uma nova falha nesse trajeto, esses pacotes não são desviados e sim descartados.

Um algoritmo para o módulo *CER_pró_ativa* e outro para o *CER_EDif* foram desenvolvidos, implementados em Java e adaptados no simulador J-SIM. Além desses módulos, a abordagem Not-Via foi parcialmente implementada para apenas obter

resultados com a abordagem proposta. Realizaram-se três avaliações: a 1ª avaliação confronta a extensão dos caminhos de recuperação do E-CER versus Not-Via, a 2ª avaliação confronta a quantidade de informações extras necessárias na FIB pelo E-CER versus Not-Via e a última avaliação verifica a execução da abordagem E-CER em um ambiente com tráfego simulado. Foram utilizadas diversas topologias reais e artificiais para realizar essas avaliações. A 1ª avaliação revelou que a abordagem E-CER obteve em geral menores caminhos de recuperação que a abordagem Not-Via. Observou-se que isto se deve à localização dos endereços *not-via* em relação ao *nó_CER*. O *nó_CER* identifica um nó em que os pacotes desviados podem seguramente seguir as rotas OSPF até o nó destino da topologia. Já os endereços *not-via* forçam os pacotes encapsulados a seguirem um caminho até um nó após a falha para então seguir as rotas com OSPF. A 2ª avaliação revelou que a abordagem E-CER necessita de menos informações extras que a abordagem Not-Via na FIB para utilizar os caminhos de recuperação. Isto ocorre devido ao número reduzido de bits necessários para representar *CER_Marca/IR*, ao uso do campo adicional *Ref* nas entradas da FIB para referenciar uma *CER_Marca/IR* e também pelo reuso das marcas quando for possível. Já o Not-Via necessita da informação de *next-next-hop* adicionado a cada entrada na FIB além de novas entradas para comportar os endereços *not-via*. A última avaliação verifica a execução da abordagem E-CER em um ambiente simulado, cujo objetivo é identificar qual a redução de pacotes perdidos durante o período de convergência do OSPF com e sem o E-CER. Ao utilizar a abordagem E-CER, verificou-se uma redução considerável na porcentagem de pacotes perdidos durante o período de convergência quando comparado à execução do OSPF padrão. Verificou-se também que em um ambiente com altas taxas de tráfego, os pacotes desviados com E-CER não influenciam os demais tráfegos não afetados diretamente pela falha.

Da mesma forma que outras abordagens relacionadas, o E-CER está limitado à configuração da topologia para prover 100% de cobertura de falha. São necessárias também modificações no plano de controle e plano de encaminhamento dos nós para comportar os módulos *CER_pró-ativa* e *CER_EDif* respectivamente. No entanto, o E-CER apenas atua em pacotes que estão marcados ou quando houver uma falha adjacente a um nó (realiza a marcação e desvio dos pacotes apenas se existir uma entrada na FIB com *Ref* identificando um *CER_Marca/IR*).

Essa tese cumpriu os objetivos definidos, criando a abordagem E-CER para auxiliar o protocolo OSPF durante o período de convergência. A partir da abordagem definiram-se

2 módulos para atuarem no plano de controle e de encaminhamento dos nós. Esses módulos podem trabalhar tanto em redes IPv4 quanto redes IPv6. Foi realizada uma avaliação da abordagem E-CER comparado com a abordagem Not-Via, demonstrando resultados bastante satisfatórios. Por fim, foi analisada a usabilidade dessa abordagem num ambiente simulado, demonstrando uma redução na porcentagem de pacotes perdidos durante o período de convergência.

6.2 Trabalhos Futuros

Como trabalhos futuros, deve-se desenvolver a abordagem E-CER em um ambiente de testes real. Em uma primeira etapa, pode-se adaptar o E-CER no software Zebra (base do código OSPF do J-SIM) para realizar os mesmos testes com roteadores Linux.

O uso de um ambiente misto com nós com e sem E-CER também deve ser alvo de novas pesquisas para que o E-CER consiga suportar esse tipo de ambiente.

Um aprimoramento da abordagem E-CER para considerar outras métricas para encontrar melhores CERs e também para trabalhar em conjunto com abordagens de engenharia de tráfego poderiam ser investigados em trabalhos futuros.

Aperfeiçoar a abordagem E-CER para múltiplas falhas independentes para prover 100% de cobertura de falhas únicas ou múltiplas. No entanto, esse tipo de abordagem tende a ser bastante complexa em escopo local para justificar o desenvolvimento.

Além da medição da quantidade de pacotes perdidos, pode-se verificar qual o impacto que o E-CER propiciaria na redução de pacotes perdidos em um ambiente com aplicativos de tráfego de tempo real (voz, vídeo, etc...) e também relacionado a outras abordagens pró-ativas.

Adaptar o E-CER para protocolos IGP de vetor de distância, ou para protocolos de roteamento BGP também são considerados trabalhos futuros. Além desses protocolos, poderia também adaptar o E-CER para o ambiente de redes sem fio *mesh*. Em uma primeira etapa, poderia utilizar o protocolo *Optimized Link State Routing Protocol* (OLSR) por ser um protocolo do estado do enlace, o qual o E-CER está inicialmente planejado.

Finalmente, sugere-se adaptar a abordagem E-CER para atuar em topologias com tecnologia MPLS. Poder-se-ia utilizar o conceito de CERs em caminhos virtuais do MPLS para definir os caminhos de recuperação. No entanto, uma análise mais profunda é necessária para verificar quais as implicações desse conceito.

REFERÊNCIAS BIBLIOGRÁFICAS

- AHUJA, R. K. MAGNANTI, T. L. ORLIN, B. O. Network Flows: Theory, Algorithms, and Applications. *Prentice-Hall*, ISBN: 013617549x. 1993.
- ALAEETTINOGLU, C. JACOBSON, V. YU, H. Toward Mili-second IGP Convergence. *Expired Internet Draft alaettinoglu-isis-convergence-00.txt*. 2000.
- ATLAS, A. ZININ, A. Basic Specification for IP Fast-Reroute: Loop-Free Alternate. *Internet Draft draft-ietf-rtgwg-ipfrr-spec-base-12.txt*, Março 2008.
- ATLAS, A., U-Turn Alternate for IP/LDP Fast-Reroute. *Internet Draft draft-atlas-ip-local-protect-uturn-03.txt*, Fevereiro 2006.
- BAKER, F. Requiriments for IP Version 4 Routers, *Request for Comments RFC 1812*, Junho, 1995.
- BASAK, D. KAUR H. T. KALYANAMARAN S. Traffic Engineering Techniques and Algorithms for the Internet. *Technical Report*, 2002
- BASU, A. RIECKE, J. Stability Issues in OSPF routing. *ACM SIGCOMM Conference on Applications, technologies, architectures, and protocols for computer communications*, pp. 225-236, San Diego. 2001.
- BERTSEKAS, D., GALLAGER, R. Data Networks, Second Edition, Prentice-Hall International Editions, 1992.
- BOUTREMANS, C. IANNACCONE, G. DIOT, C. Impact of link failures on VoIP performance. *NOSSDAV International Workshop on Network and Operating Systems Support for Digital Áudio Vídeo*, Maio, 2002.
- BRADEN, R. D. CLARK, J. CROWCROFT, B. DAVIE, S. DEERING, D. ESTRIN, S. FLOYD, V. JACOBSON, G. MINSHALL, C. PARTRIDGE, L. PETERSON, K. RAMAKRISHNAN, S. SHENKER, J. WROCLAWSKI, and L. ZHANG. Recommendations on Queue Management and Congestion Avoidance in the Internet. *Request for Comments RFC 2309*, Abril 1998.
- BRYAN, S., Filsfils, C., PREVIDI, S., and SHAND, M., IP Fast Reroute using Tunnels, *Internet Draft draft-bryant-ipfrr-tunnels-02.txt*, Outubro, 2005.

- BRYANT, S. SHAND, M. PREVIDI, S. IP Fast Reroute Using Not-via Addresses. *Internet Draft draft-ietf-rtgwg-ipfrr-notvia-addresses-02.txt*, Fevereiro, 2008.
- CAIDA. Cooperative Association for Internet Data Analysis – Project MapNet, *Disponível em 2008 no site <http://www.caida.org/tools/visualization/mapnet/Backbones/>*, Junho, 2008.
- CISCO. *OSPF Design Guide*. Disponível em <http://www.cisco.com/warp/public/104/1.html>. 2005.
- CLARK, D. D. The Design Philosophy of the DARPA Internet Protocols. *ACM SIGCOMM Computer Communications Review*. pp. 106-104. Agosto 1988.
- COLTUN, R. ACOUSTRA, T. FERGUSON, D. MOY, J. LINDEM, A. OSPF for IPV6. *Internet Draft draft-ietf-ospf-ospfv3-update-08.txt*. Março 2006.
- COLTUN, R., The OSPF Opaque LSA Option. *Request for Comments RFC 2370*, 1998.
- DARPA Project, Internet Protocol, *Request for Comments RFC 791*, 1981.
- DEERING, S. HINDEN, R. Internet Protocol, Version 6 (IPv6) Specification, *Request for Comments RFC 2460*. 1998
- DIESTEL, R. *Graph Theory*, Springer-Verlag 3ª edição, 2005.
- DIJKSTRA, E. W. A Note on Two Problems in Connection with Graphs. *Numerische Mathematik*, 1:269-271, 1959.
- DOYLE, J., CCIE Professional Development Routing TCP/IP – Volume1. *Cisco Press*, Setembro, 1998.
- FLOYD, S., JACOBSON, V. Random Early Detection gateways for Congestion Avoidance *IEEE/ACM Transactions on Networking*, V.1 N.4, pp. 397-413, Agosto 1993.
- FORTZ, B. and THORUP, M. Internet Traffic Engineering by optimizing OSPF Weights, *Proceedings of IEEE INFOCOM'00*, Tel Aviv. Israel. 2000.
- FORTZ, B. and THORUP, M. Optimizing OSPF/IS-IS Weights in a Changing World. *Journal on Selected Areas in Communications*, vol 20, no 4, 2002.
- FRANCOIS, P. FILSFILS, C. EVANS, J. BONAVENTURE, O. Achieving sub-second IGP convergence in large IP networks. *ACM SIGCOMM Computer Communication Review*. vol. 35, num. 2, Julho 2005.
- FRANCOIS, P., BONAVENTURE, O., Avoiding Transient Loops During IGP Convergence in IP Networks. *IEEE INFOCOM Computer and*

- Communications Societies*, pp. 237-247, 2005.
- FULLER, V. LI, T. YU, J. VARADHAN, K. Classless Inter-Domain Routing (CIDR): an Address Assignment and Aggregation Strategy. *Request for Comments RFC 1519*. 1993.
- GARNIER, J. Graphs and Dijkstra's algorithm. *Disponível em 2006* <http://rollerjm.free.fr/pro/graphs.html>, 2002.
- GLAZER, D. W., TROPPER, C. A New Metric for Dynamic Routing Algorithms. *IEEE Transactions on Communications*. Vol. 38, No. 3, Março 1990.
- GOYAL, M., RAMAKRISHNAN, K. K., FENG, W. Achieving Faster Failure Detection in OSPF Networks. *IEEE International Conference on Communications*, pp. 296-300, 2003.
- GROSS, P. Choosing a "Common IGP" for the IP Internet (The IESG's Recommendation to the IAB). *Request for Comments RFC 1371*, 1992.
- HANSEN, A. F., CICIC, T., and GJESSING, S. Alternative Schemes for Proactive IP Recovery. *Next Generation Internet Design and Engineering*. Abril, 2006.
- HANSEN, A. F., LYSNE, O., CICIC, T., and GJESSING, S., Fast Proactive Recovery from Concurrent Failures. *IEEE International Conference on Communications*, Junho, 2007.
- IANNACCONE, G. CHUAH, C. DAVIS, BHATTACHARYYA, S. DIOT, C. Feasibility of IP Restoration in a Tier-1 Backbone. *IEEE Network* vol. 18, pp. 13-19 Agosto 2004.
- IANNACCONE, G. CHUAH, C. MORTIER, R. BHATTACHARYYA, S. DIOT, C. Analysis of link failures in na IP backbone. *ACM SIGCOMM Workshop on Internet measurement*. sessão 8, pp. 237-242. 2002.
- IYER, S. BHATTACHARYYA, S. TAFT, N. DIOT, C. "An Approach to alleviate link overlad as observed on an IP backbone", in Proc. *IEEE Communications Society INFOCOM* 2003.
- KANG, X., and CHAO, H. J., IP Fast Rerouting for Single-Link/Node Failure Recovery. *International Conference on Broadband Communications, Networks and Systems*, Setembro, 2007.
- KATZ, D. KOMPELLA, K. YEUNG, D. Traffic Engineering (TE) Extensions to OSPF Version 2. *Request for Comments RC 3630*. Setembro, 2003.
- KATZ, D. WARD, D. Bidirectional Forwarding Detection. *Internet Draft draft-*

- ietf-bfd-base-04.txt*. Outubro, 2005
- KESHAV, S. An Engineering Approach to Computer Networking: ATM Networks, the Internet, and the Telephone Network. Addison-Wesley Publishing Company, 1997.
- KHANNA, A. ZINKY, J. The Revised ARPANET Routing Metric, *Proceedings of ACM SIGCOMM*, pp 46-55, Setembro 1989.
- KOMPELLA, K., REKHTER, Y. OSPF Extensions in Support of Generalized Multi-Protocol Label Switching (GMPLS). *Request for Comments RFC 4203*, Outubro 2005.
- KVALBEIN, A. HANSEN, A. F. CICIC, T. GJESSING, S. LYSNE, O. Fast IP Network Recovery using Multiple Routing Configurations. *IEEE Conference on Computer Communications INFOCOM*. 2006.
- KVALBEIN, A. HANSEN, A. F. CICIC, T. GSESSING, S. LYSNE, O. Fast Recovery from Link Failures using Resilient Routing Layers. *IEEE Symposium on Computers and Communications*, Junho 2005.
- KVALBEIN, A., CICIC, T., and GJESSING, S., Post-Failure Routing Performance with Multiple Routing Configurations, *IEEE Conference on Computer Communications*. pp. 98-106. Maio, 2007.
- LEE, S. YU, Y. NELAKUDITI, S. ZHANG, Z. and CHUAH, C., "Proactive vs. Reactive Approaches to Failure Resilient Routing", *IEEE INFOCOM*, 2004.
- LIU, Y. and REDDY, A. L. N. A Fast Rerouting Scheme for OSPF/IS-IS Networks. *IEEE Computer Communications Networks*, pp. 47-52, 2004.
- MARKOPOULOU, A., IANNA CONNE, G., BHATTACHARYYA, S., CHUAH, C., and DIOT, C., Characterization of Failures in IP Backbone. *IEEE INFOCOM*, 2004.
- MATHIS, M. and HEFFNER, J., Packetization Layer Path MTU Discovery. *Request For Comments RFC 4821*, 2007.
- MCQUILLAN, J. M. RICHER, I. ROSEN, E. C. The New Routing Algorithm for the ARPANET. *IEEE Transactions on Communications*, 711-719, Maio 1980.
- MCQUILLAN, J. RICHER, I. and ROSEN, E. "ARPANET Routing Algorithm Improvements", *BBN Technical Report 3803*, Abril, 1978.
- MEDINA, A. LAKHINA, A. MATTA, I. BYERS, J. Boston University Representative Internet Topology Generator. *Disponível em 2006 no site*

- <http://www.cs.bu.edu/brite>, 2002.
- MOGUL, J., and DEERING, S., Path MTU Discovery. *Request for Comments RFC 1191*, 1990.
- MOY, J. OSPF Version 2. *Request for Comments RFC 2328*, 1998.
- MOY, J., OSPF Standardization Report. *Request for Comments RFC 2329*. Abril, 1998.
- NARVAEZ, P. Routing Reconfiguration in IP Networks. *Instituto de Tecnologia de Massachusetts. Ph.D Dissertação*. Junho 2000.
- NICHOLS, K. BLAKE, S. BAKER, F. BLACK, D. Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers. *Request for Comments RFC 2474*, Dezembro, 1998.
- NUCCI, A., SCHOROEDER, B., BHATTACHARYYA, S., TAFT N., and DIOT C. "IGP Link Weight Assignment for Transient Link Failures", *International Teletraffic Congress*, 2003.
- PAN, P. SWALLOW, G. ATLAS, A. Fast Reroute Extensions to RSVP-TE for LSP Tunnels. *Request for Comments RFC 4090*, 2005.
- PETERSSON, J. M. O. MPLS Based Recovery Mechanisms. *Universidade de Oslo. Master Thesis*. Maio 2005.
- POSTEL, J., The TCP Maximum Segment Size. *Request for Comments RFC 893*, 1983.
- RAJAHALME, J., CONTA, A., CARPENTER, B., and DEERING, C., IPv6 Flow Label Specification, *Request for Comments RFC 3697*, Março, 2004.
- REKHTER, Y. LI, T. HARES, S. A Border Gateway Protocol 4 (BGP-4). *Request for Comments RFC 4271*. Janeiro 2006.
- ROSEN, E. VISWANATHAN, A. CALLON, R. Multiprotocol Label Switching Architecture. *Request for Comments RFC 3031*. Janeiro 2001.
- SCHOLLMEIER, G. CHARZINSKI, J. KIRSTADTER, A. REICHERT, C. SCHRODI, K. GLICKMAN, Y. WINKLER, C. Improving the Resilience in IP Networks. *High Performance Switching and Routing (HPSR)*, 2003.
- SHAND, M. BRYANT, S. IP Fast Reroute Framework. *Internet Draft draft-ietf-rtgwg-ipfrr-framework-05.txt*, Fevereiro 2008.
- SHARMA, V. and HELLSTRAND, E. F., Framework for MPLS-based Recovery. *Request for Comments RFC 3469*, Fevereiro 2003.

- SU, H., WU, C., and CHU, Y., IP Local Node Protection. *IEEE Conference on Systems and Networks Communications*. Setembro, 2007.
- TANEMBAUM, A. *Redes de Computadores*. Prentice-Hall. 4ª Edição. 2003.
- TANG, X., TANG, J., HUANG, G., and SIEW, C. QoS Provisioning Using IPv6 Flow Label In the Internet”, *IEEE Conference on Information, Communications and Signal Processing*, 2002.
- TYAN, H., J-SIM, *Disponível em 2008 no site <http://www.j-sim.org>*, Junho, 2008.
- VASSEUR, J. P., PICKAVET, M., DEMEESTER, P. *Network Recovery: Protection and Restoration of Optical, SONET-SDH, IP, and MPLS*. Morgan Kaufmann, Elsevier. 2004.
- VILLAMIZAR, C. OSPF Optimized Multipath (OSPF-OMP). *Expired Internet Draft draft-ietf-ospf-omp-02.txt*, Fevereiro, 1999.
- WANG, Z. CROWCROFT, J. Analysis of shortest-path routing algorithms in a dynamic network environment. *Computer Communication Review*, vol 22, no. 2, pp. 63-71, 1992.
- WANG, Z. CROWCROFT, J. Shortest Path First with Emergency Exits, *ACM SIGCOMM Symposium on Communications Architectures and Protocols*, 1990.
- WATSON, D., JAHANIAN, F. LABOVITZ, C. Experiences with Monitoring OSPF on a Regional Service Provider Network. *IEEE ISDCS International Conference on Distributing Computing Systems*. pp. 204-213, Maio 2003.
- WAXMAN, B. Routing of Multipoint Connections, *IEEE Journal of Selected Areas in Communications*, Janeiro, 2001.
- WOLFGANG, J. and TAFVELIN, S. Analysis of Internet Backbone Traffic and Header Anomalies observed. *ACM Internet Measurement Conference*, 2007.
- ZEBRA, GNU, *Disponível em 2008 no site <http://www.zebra.org>*, Junho, 2008.
- ZHONG, Z. NELAKUDITI, S. YU, Y. LEE, S. WANG, J. CHUAH, C. Failure Inferencing based Fast Rerouting for Handling Transient Link and Node Failures. *Proceedings of IEEE Global Internet*, Março, 2005.

Apêndice A

SIMULADOR JAVASIM (J-SIM)

Neste apêndice é apresentado em linhas gerais a ferramenta JavaSim (J-SIM) [TYAN, 2008] capaz de trabalhar com simulação de eventos discretos e simulação de processos baseados em tempo real. O J-SIM é um ambiente de simulação escrito em linguagem Java e teve a sua estrutura construída com base na *Arquitetura de Componentes Autônomos* (ACA). Essa arquitetura tem como base entidades denominadas *componentes*. Cada *componente* possui uma função definida e uma ou mais *portas* através das quais se podem conectar a outros *componentes* para enviar/receber dados. A execução de um *componente* é realizada em threads controladas para processar os dados de entrada/saída que são recebidos/enviados por suas *portas*. As *portas* possuem um padrão de interface e funcionalidade definidas que devem ser seguidos para o correto funcionamento no intercâmbio de dados. Um *componente* pode ser composto por outros *componentes* encapsulados que trocam dados internamente entre suas *portas* conectadas. O uso de conceito de *componentes* e *portas* permite a criação ou modificação de código fonte sem interferir nas demais implementações de outros *componentes*. O reuso de *componentes* mais simples para que em conjunto realizem funções mais complexas quando encapsuladas no desenvolvimento de um *componente* maior permite ter uma grande granularidade no seu desenvolvimento.

O J-SIM utiliza uma thread principal denominada *runtime* que possui o controle global da execução das threads dos *componentes* ativos no sistema e do tempo de simulação. Esse tempo é observado por todas as threads. A thread *runtime* é também um intermediário para controlar a troca de dados entre as *portas* conectadas dos *componentes*.

Utilizando a base ACA, o J-SIM fornece um *modelo generalizado de rede IP de pacotes*. Este modelo define uma estrutura genérica de *componentes* de nó (um *host* ou um roteador) denominada *Core Service Layer (CSL)*. Além do *componente* nó, há

componentes para representar enlaces físicos e para representar processos de rede padrão TCP/IP (por exemplo: protocolo IP, encaminhamento de pacotes IP, protocolos roteamento, transporte e aplicativos). Um *componente* nó basicamente encapsula *componentes* de datagrama IP, de encaminhamento de pacotes, de filas de transmissão, e de interfaces de rede. Esses *componentes* fornecem as rotinas necessárias para o funcionamento básico das camadas de rede IP, enlace e física do modelo TCP/IP na troca de mensagens entre os nós de rede. O componente nó possibilita o envio de dados das camadas superiores do TCP/IP representado por componentes de transporte, roteamento e de aplicativos para outros componentes nó e vice-versa. O encaminhamento de pacotes armazena a tabela de encaminhamento (FIB), a qual é atualizada pelo componente de roteamento sempre que necessário. O componente de roteamento sempre utiliza o componente nó para receber e enviar mensagens referentes ao protocolo (ex: OSPFv2) que foi implementado através de *portas* definidas. A Figura A.1 adaptada de [TYAN, 2008] ilustra o componente nó e a separação das camadas TCP/IP em componentes do *modelo generalizado de rede IP de pacotes* do J-SIM. Observa-se que o componente CSL realiza as funções das camadas de rede IP, enlace e física. Um CSL se comunica com outro CSL por meio do componente interface de rede, que se conecta a um *componente* de enlace físico, e que por sua vez se conecta a um componente interface de rede de outro CSL. As funções de transporte, roteamento e aplicativos se comunicam com o CSL para utilizar as funções IP básicas e de transmissão/recepção de mensagens.

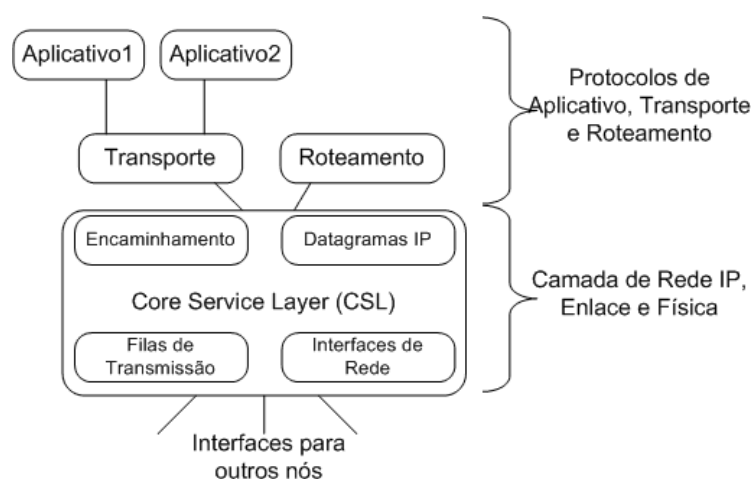


Figura A.1 Visão geral da componente Nó no J-SIM

Com o *modelo generalizado de rede IP de pacotes* do J-SIM, pode-se criar novos componentes que sigam esse padrão ou até mesmo alterar o comportamento desses

componentes. Esta tese teve como maior foco a alteração dos componentes de roteamento e encaminhamento de pacotes para incorporar a abordagem E-CER.

Além dos *componentes* escritos em Java, o simulador J-SIM fornece um conjunto de classes para gerar uma interface de scripts TCL. Essa interface permite a interação do usuário de forma a iniciar, parar, configurar e programar um ambiente de simulação utilizando os *componentes* J-SIM. A partir dessa interface é possível gerar resultados e até gráficos utilizando classes Java do pacote PtPlot5.1 (<http://ptolemy.berkeley.edu/>) que já estão disponíveis como pacotes adicionais ao J-SIM.

Como o simulador J-SIM é escrito em Java, utilizamos o ambiente de desenvolvimento integrado NetBeans (<http://www.netbeans.org>) ao invés da sintaxe via linha de comando conforme o manual do J-SIM. Esse ambiente facilita a execução, a criação ou modificação de componentes do J-SIM. A sua execução é simples, basta executar a classe *drcl.ruv.System* que o J-SIM é carregado e a interface de scripts TCL é carregada. A Figura A.2 ilustra a interface TCL esperando por comandos do usuário após o carregamento final do J-SIM.

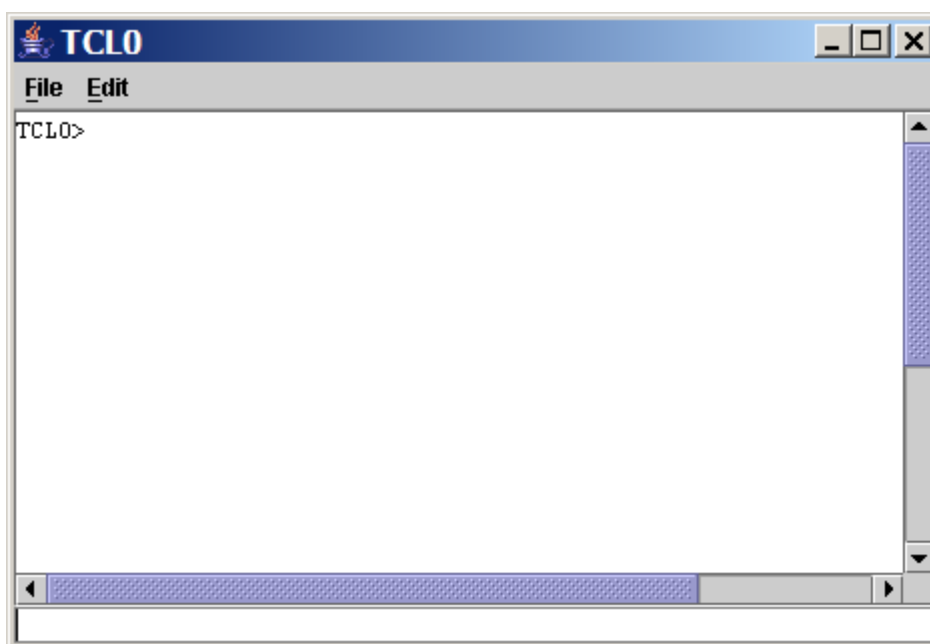


Figura A.2 Interface TCL para o simulador J-SIM

Por meio dessa interface, basta digitar comandos TCL para criação e execução dos componentes ou colar um texto de script já criado para acionar o simulador. A sintaxe específica que essa interface suporta pode ser encontrada em [TYAN, 2008].

Apêndice B

TOPOLOGIAS

Uma pequena descrição das representações de topologias utilizadas na avaliação realizada com as abordagens E-CER e Not-Via são descritas a seguir.

A topologia Abilene de 12 nós demonstrada na Figura B.1 é um *backbone* de alta velocidade projetado para a comunidade da Internet2. Seu uso está voltado para o meio científico e educacional.

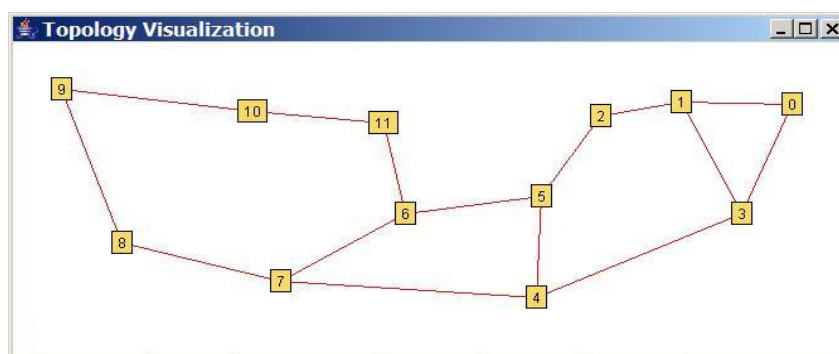


Figura B.1 Abilene

A topologia AGIS de 61 nós demonstrada na Figura B.2 é um *backbone* para prover serviços de roteamento de rede a grandes empresas e provedores de serviço.

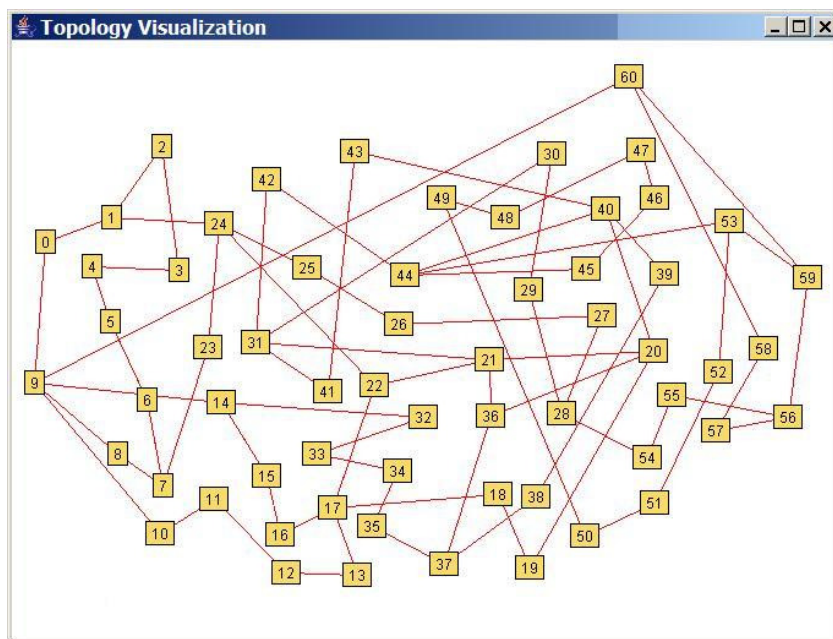


Figura B.2 AGIS

A topologia AT Home Network de 43 nós demonstrada na Figura B.3 é um *backbone* voltado para disponibilizar acesso à Internet para clientes domésticos. No entanto, esse *backbone* também provê atualmente serviços de Internet para empresas.

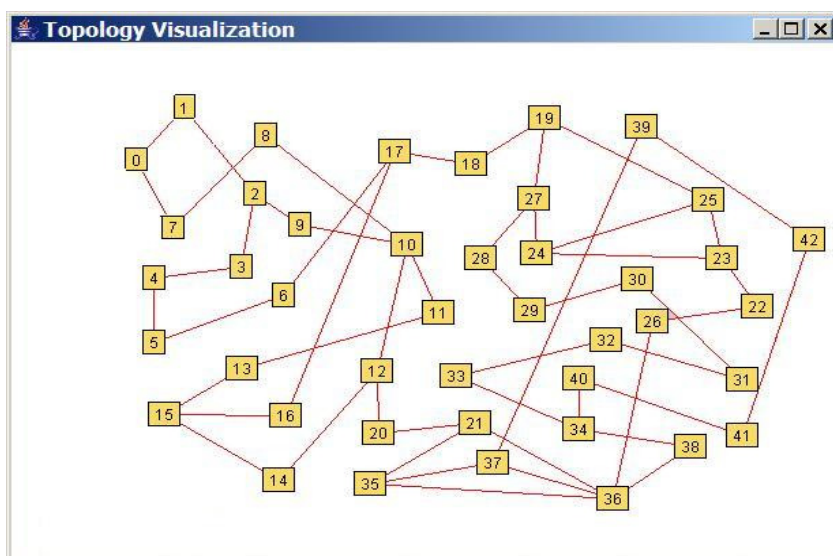


Figura B.3 AT Home

A topologia AT&T WorldNet de 23 nós demonstrada na Figura B.4 é um *backbone* para prover acesso a Internet, portal web e telefonia a clientes residenciais e empresariais.

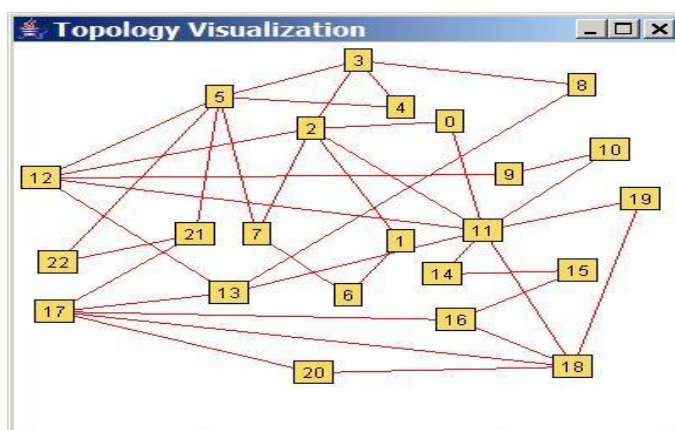


Figura B.4 AT&T

A topologia BRITE de 32 nós demonstrada na Figura B.5 é um *backbone* gerado com a ferramenta BRITE [MEDINA et al, 2002].

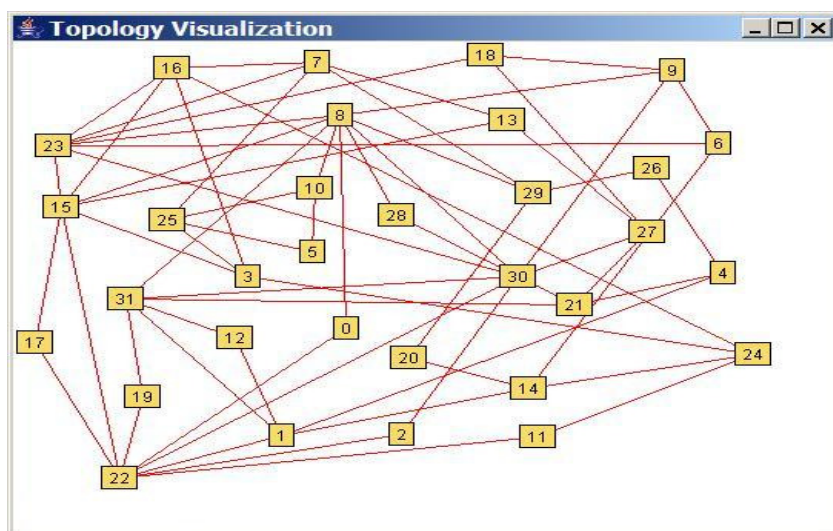


Figura B.5 BRITE 32 Nós e 64 Enlaces

A topologia CAIS de 36 nós demonstrada na Figura B.6 é um *backbone* voltado para interligar redes de hotéis e centros de conferências a uma alta velocidade.

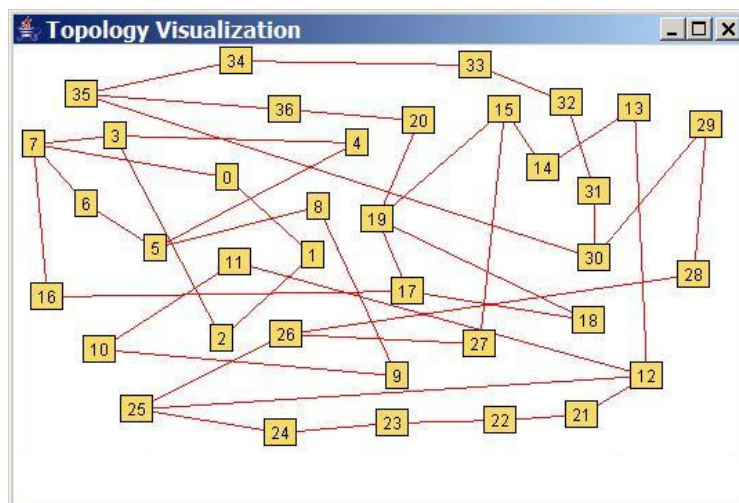


Figura B.6 CAIS

A topologia GEANT2 de 24 nós demonstrada na Figura B.7 é um *backbone* de pesquisa e ensino acadêmico europeu de alta velocidade conectando vários países.

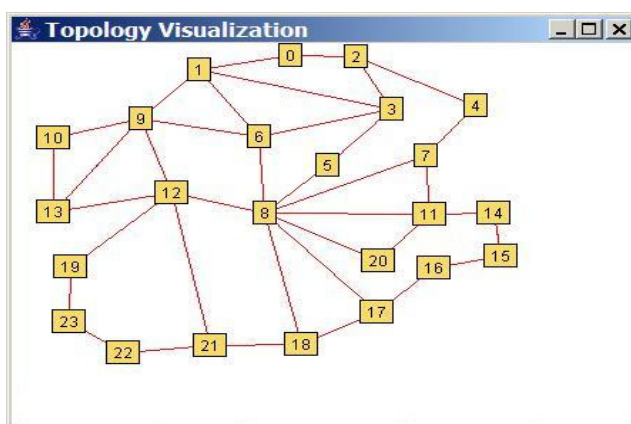


Figura B.7 GEANT2

A topologia Level 3 de 73 nós demonstrada na Figura B.8 é um *backbone* que fornece acesso à Internet e a serviços de telefonia para residências e empresas.

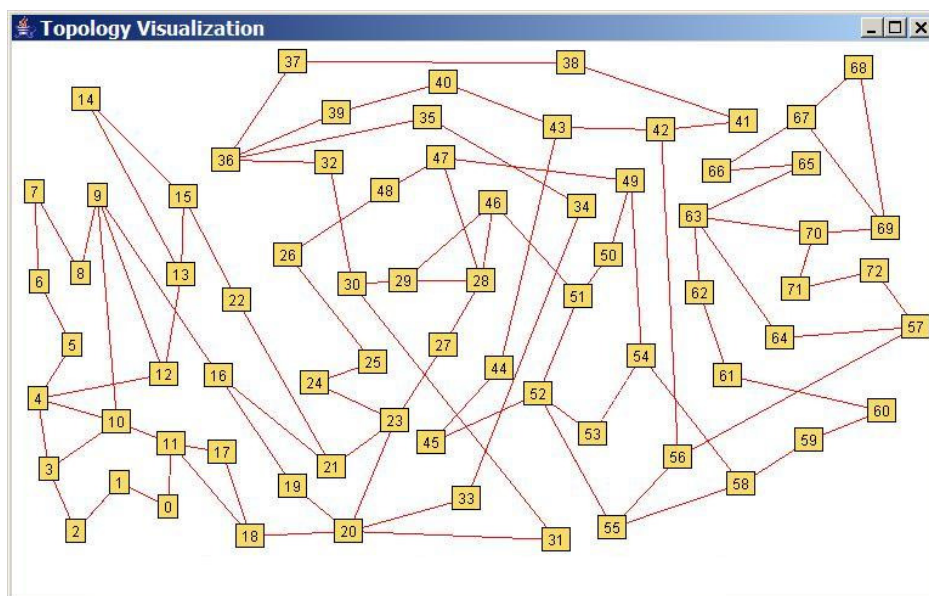


Figura B.8 Level 3

A topologia NFSNET de 14 nós demonstrada na Figura B.9 é um *backbone* acadêmico que interliga principalmente departamentos de ciências de computação.

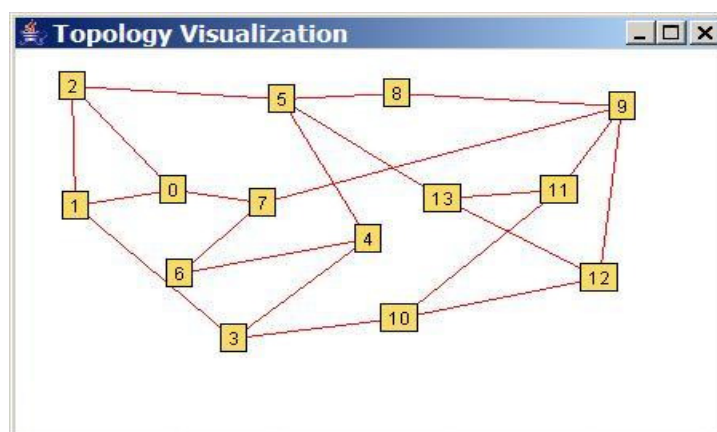


Figura B.9 NFSNET

A topologia Qwest de 25 nós demonstrada na Figura B.10 é um *backbone* de comunicação para rede de voz e dados para pequenas e grandes empresas além de acesso residencial à Internet.

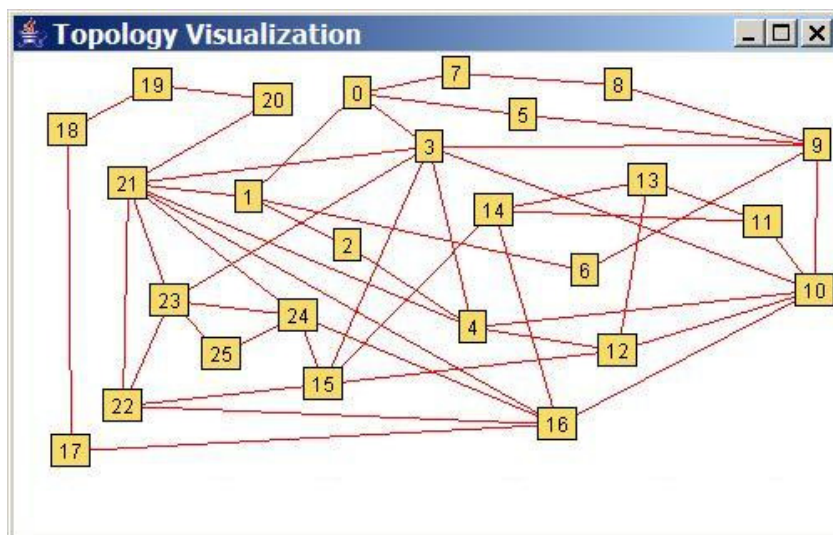


Figura B.10 Qwest

A topologia Servint de 20 nós demonstrada na Figura B.11 é um *backbone* de alta velocidade para empresas de todos os tamanhos terem acesso a Internet além de serviço de hospedagem de páginas web.

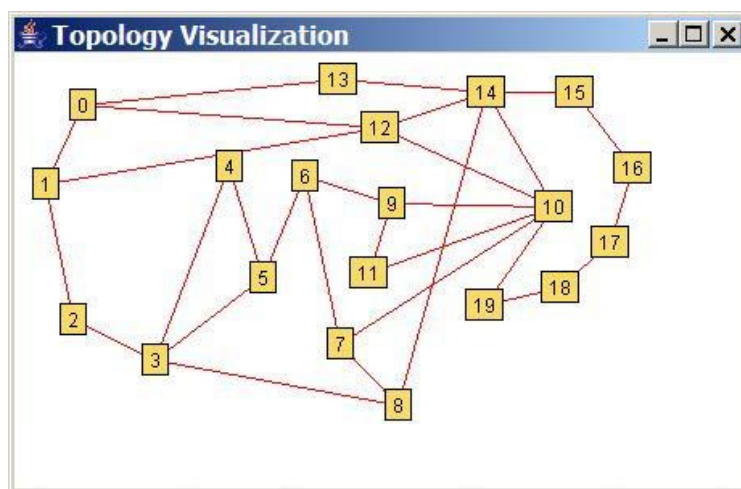


Figura B.11 Servint

A topologia Sprint de 18 nós demonstrada na Figura B.12 é um *backbone* de alta velocidade para prover acesso a Internet com transmissão de voz, dados e vídeo a clientes empresariais.

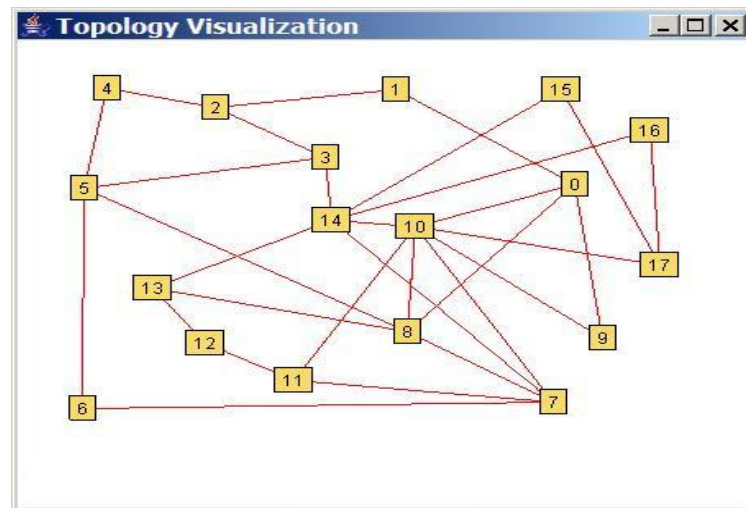


Figura B.12 Sprint