COPPE/UFRJ

NOVAS TÉCNICAS PARA O PROBLEMA DE ESTIMAÇÃO DA MATRIZ DE TRÁFEGO

Zhou Xiang

Tese de Doutorado apresentada ao Programa de Pós-graduação em Engenharia de Sistemas e Computação, COPPE, da Universidade Federal do Rio de Janeiro, como parte dos requisitos necessários à obtenção do título de Doutor em Engenharia de Sistemas e Computação.

Orientadores: Edmundo de Souza e Silva
Susana Scheimberg de Makler

Rio de Janeiro
Fevereiro de 2009

NOVAS TÉCNICAS PARA O PROBLEMA DE ESTIMAÇÃO DA MATRIZ DE TRÁFEGO

Zhou Xiang

TESE SUBMETIDA AO CORPO DOCENTE DO INSTITUTO ALBERTO LUIZ COIMBRA DE PÓS-GRADUAÇÃO E PESQUISA DE ENGENHARIA (COPPE) DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE DOUTOR EM CIÊNCIAS EM ENGENHARIA DE SISTEMAS E COMPUTAÇÃO.

Aprovada por:

Prof. Edmundo Albuquerque de Souza e Silva, Ph.D.

Profa. Susana Scheimberg de Makler, D.Sc.

Prof. Nelson Maculan Filho, D.Habil.

Prof. Henrique Pacca Loureiro Luna, Dr.

Prof. Harry Richard Gail, Ph.D.

Prof. Aloysio de Castro Pinto Pedroza, Dr.

RIO DE JANEIRO, RJ - BRASIL

FEVEREIRO DE 2009

XIANG, ZHOU

Novas Técnicas para O Problema de Estimação da Matriz de Tráfego/ Zhou Xiang. - Rio de Janeiro: UFRJ/COPPE, 2009.

XV, 128 p.:il; 29,7 cm.

Orientadores: Edmundo de Souza e Silva

Susana Scheimberg de Makler

Tese (doutorado) - UFRJ/COPPE/Programa de Engenharia de Sistemas e Computação, 2009.

Referência Bibliográficas: p.122-128.

1. Matriz de Tráfego. 2. Otimização. 3. Roteamento. I. Silva, Edmundo De Souza et al.. II. Universidade Federal do Rio de Janeiro, COPPE, Programa de Engenharia de Sistemas e Computação. III. Título.

Aos meus pais.

Agradecimentos

Agradecer aos meus pais por terem suportado os encargos dos meus estudos, e pela confiança que me incutiram ao longo dos meus anos de vida, sei que é a você que devo o facto de ser aquilo que sou hoje.

Agradeço aos meus orientadores Prof. Edmundo A. de Souza e Silva e Profa. Susana Scheimberg de Makler por me aceitarem, pela confiança que depositaram em mim, pelo apoio científico que sempre me concederam e pela amizade,

Agradeço ao Prof. Nelson Maculan Filho que me aconselhou nesta pesquisa de tese.

Agradeço profundamente a Carolina Maria de Vilar Moreira Vieira que me ajudou a rever o português e me ajudou muito ao longo dos tempos.

Por fim, agradeço a todos os professores e alunos do LAND - Laboratório de Modelagem, Análise e Desenvolvimento de Sistemas de Computação e Redes de Computadores, que direta ou indiretamente, me ajudaram na confecção desta tese e me ajudaram a tirar dúvidas com a língua portuguesa. Em especial, ao Daniel Sadoc Menasché, Guilherme Dutra Gonzaga Jaime e Fernando Jorge Silveira Filho que discutiram comigo tirando minhas dúvidas e me ofereceram suas ajudas durante estes anos, à Profa. Rosa Maria Meri Leão, ao Prof. Daniel Ratton ,ao Bernardo, Ed, Allyson, Guto, Hugo, Kleber, Luiz, Bruno, Fabrício e outros.

Resumo da Tese apresentada à COPPE/UFRJ como parte dos requisitos necessários para a obtenção do grau de Doutor em Ciências (D.Sc.)

NOVAS TÉCNICAS PARA O PROBLEMA DE ESTIMAÇÃO DA MATRIZ DE TRÁFEGO

Zhou Xiang

Fevereiro/2009

Orientadores: Edmundo de Souza e Silva
Susana Scheimberg de Makler
Programa: Engenharia de Sistema e Computação

A estimação da matriz de tráfego em redes de comunicação, tem recentemente recebido muita atenção da comunidade de pesquisa. Neste trabalho, realizamos estudos visando a estimar a matriz de tráfego em uma rede de telecomunicação.

O mecanismo de roteamento sendo adotado na rede é um fator crucial para a estimação da matriz de tráfego. Consideramos dois diferentes mecanismos de roteamentos: 1) roteamento pelo caminho mais curto, 2) roteamento ótimo por múltiplas rotas. Propomos novos métodos para estimar a matriz de tráfego quando cada um mecanismo de roteamento é empregado na rede.

Os método tradicionais para a estimação da matriz de tráfego são baseados em protocolos de roteamento pelo caminho mais curto. O método de tomogravidade é um dos métodos mais conhecidos e utilizados. Mas este método não produz resultados precisos quando o erro resultante da medição do tráfego nos canais é grande. Nesta tese, propomos uma abordagem que formula o problema de estimação da matriz de tráfego como um problema de otimização com restrição de não-negatividade e, em seguida, um método de projeção é utilizado para resolvê-lo. Foram realizados experimentos com dados sintéticos e dados reais obtidos da Rede Abilene. Mostramos que o nosso método estima a matriz de tráfego mais precisamente do que o método tomogravity.

Desenvolvemos também uma nova abordagem para estimar a matriz de tráfego quando o roteamento ótimo por múltiplas rotas está sendo empregado. Formulamos o problema como um problema de programação em dois níveis. Um algoritmo genético foi proposto para resolver o problema.

Abstract of Thesis presented to COPPE/UFRJ as a partial fulfillment of the requirements for the degree of Doctor of Science (D.Sc.)

NEW SOLUTION TECHNIQUES FOR THE TRAFFIC MATRIX ESTIMATION PROBLEM

Zhou Xiang

February / 2009

Advisors: Edmundo de Souza e Silva
Susana Scheimberg de Makler
Department: Computer and System Engineering

Estimating traffic matrices(TM) in communication networks, has recently been received much attention in the research community. In this thesis, we address the traffic matrix estimation problem and propose new efficient techniques.

The routing mechanism is a crucial factor for estimating the traffic matrices. We consider two different routing mechanisms: 1) shortest path routing, 2) optimal multipath routing. We propose new methods for estimating the traffic matrix when each routing mechanism is employed.

The traditional methods for estimating the traffic matrix are based on the shortest path routing. The tomogravity method is one of the most extensively used methods. Therefore it may produce inaccurate results when the link traffic measurement errors are large. In this thesis, we propose an approach which formulates the traffic matrix estimation problem as a non-negativity constrained optimization problem, and then a projection method is used to solve it. We conduct experiments both on synthetic and real measurement data obtained from the Abilene network. The results indicate that the traffic matrices are more accurately estimated when our approach is used than when the tomogravity method is employed. We also develop a novel approach for estimating traffic matrices when optimal multi-path routing is employed. We formally formulate the problem as a bilevel programming problem. Then a genetic algorithm is used to solve it.

Sumário

Resumo **vi**

Abstract **vii**

1 Introdução **1**

1.1 Objetivo da Tese 6

1.2 Contribuições 8

1.2.1 Problema A 8

1.2.2 Problema B 8

1.3 Roteiro 10

2 Trabalhos Relacionados **11**

2.1 Métodos baseados em Inferência Estatística 12

2.1.1 Inferência Bayesiana 13

2.1.2 Estimação de Máxima Verossimilhança e EM 14

2.2 Métodos baseados em Otimização 16

2.2.1 Programação Linear 16

2.2.2 Programação Não Linear 17

2.3 Outros Métodos de Estimação da Matriz de Tráfego 22
2.3.1 Método fanout 23
2.3.2 Um Método que Leva em Conta Possíveis Mudanças de Rotas 23
2.3.3 Um Método baseado no Filtro de Kalman 25
2.3.4 Um Método baseado na Análise dos Componentes Principais . 26
2.3.5 Modelo de Conexão Independente 28
2.3.6 Método de Zhao 29
2.4 Resumo 30

3 Fundamentos Teóricos **32**

3.1 Roteamento Ótimo por Múltiplas Rotas 32
3.1.1 Roteamento: Múltiplas Rotas versus Caminho mais curto . . 37
3.2 Programação em Dois Níveis 38
3.2.1 Introdução 38
3.2.2 Métodos Numéricos 42
3.2.3 Uma Aplicação ao Problema de Roteamento Ótimo por Múltiplas Rotas 44
3.3 O Método dos Mínimos Quadrados 44
3.4 Ajuste Proporcional Iterativo 45
3.5 O Algoritmo Genético 46
3.5.1 Introdução 46
3.5.2 Elementos básicos 47
3.5.3 Um Exemplo Simples 51

4 Matriz de Tráfego para Roteamento pelo Caminho mais curto 55

4.1 Incerteza nos Dados SNMP 55

4.2 Não-negatividade das Variáveis 57

4.3 O Método de Projeção do Gradiente 58

4.4 Resultados para o Problema A 62

4.4.1 Uso de Dados Sintéticos 62

4.4.2 Utilização de Dados Reais 64

4.4.3 Complexidade do algoritmo de projeção do gradiente 68

5 Matriz de Tráfego para Roteamento Ótimo por Múltiplas Rotas 72

5.1 O Sistema 72

5.2 Modelo de Dois Níveis 74

5.2.1 Nível Superior 76

5.2.2 Nível Inferior 77

5.2.3 Modelo Proposto 77

5.3 Um Algoritmo Genético para a Estimação da Matriz de Tráfego . . . 82

5.4 Resultados Computacionais 83

5.5 Complexidade 86

5.5.1 Complexidade do algoritmo genético proposto 86

5.5.2 Vantagens e desvantagens do algoritmo proposto 88

5.6 Aplicação do método proposto às redes que suportam o roteamento pelo caminho mais curto 88

6 Conclusão e Trabalhos Futuros 91

6.1 Conclusão 91

6.1.1 Algoritmos para resolver programação em dois níveis 93

A Código Fonte **95**

A.1 Algoritmo de Projeção do Gradiente 95

A.1.1 Código Principal 95

A.1.2 Código Fonte do Algoritmo de Projeção do Gradiente 99

A.1.3 Código Fonte do Método Tomogravity 100

A.1.4 Código do Ajuste Proporcional Iterativo 101

A.2 Algoritmo Iterativo 102

A.2.1 Código Principal 102

A.2.2 Código Fonte do Nível Superior 105

A.2.3 Código Fonte Referente ao Nível Inferior 105

A.2.4 Código Fonte da Função Objetivo do Nível Inferior 106

A.3 Algoritmo Genético 107

A.3.1 Código Principal 107

Referências Bibliográficas **122**

Lista de Figuras

1.1 Exemplo de estimação de matriz de tráfego 3

1.2 As curvas de desempenho dos mecanismos de roteamento. 6

1.3 Mecanismos de roteamento. 6

1.4 Comparação entre os métodos clássicos de estimação da matriz de tráfego e o método proposto. 9

2.1 O método iterativo do Vaton-Gravey 14

2.2 Worst-case bound na rede Europa e Americana 18

2.3 A Priori obtido pelo worst-case bound 18

2.4 Ilustração do método de tomogravidade 21

2.5 Fanout nos três níveis de granularidade 24

3.1 Exemplo de roteamento ótimo por múltiplas rotas 33

3.2 Jogo Microsoft-Netscape 39

3.3 A estrutura básica do algoritmo genético 47

3.4 Cruzamento com um ponto de corte 50

3.5 Mutação 50

3.6 Roleta 52

4.1 Topologia de uma rede de 4 nós 63

4.2 Rede Abilene 66

4.3 A função de distribuição cumulativa empírica das estimativas das matrizes de tráfego 67

4.4 Comparação com o método de tomogravidade com ruído de diferentes níveis para uma matriz de tráfego determinada 67

4.5 Comparação com o método de tomogravidade com ruído de diferentes níveis 68

4.6 Complexidade de tempo e número de iterações do algoritmo proposto 69

4.7 Função de distribuição percentual cumulativa empírica do tempo de execução e número de iteração 70

4.8 Comparação com o algoritmo usado na segunda etapa do método de tomogravidade 71

5.1 O método de estimação da matriz de tráfego proposto. 73

5.2 Exemplo ilustrativo do sistema considerado 75

5.3 Versão detalhada do método de estimação da matriz de tráfego proposto. 75

5.4 Topologia de 4 nós 84

5.5 Algoritmo Genético para Estimação da Matriz de Tráfego 85

5.6 Comparação da matriz de tráfego real com a matriz de tráfego estimada 86

5.7 Tempo de execução do roteamento ótimo 87

5.8 Comparação entre o método de tomogravidade e o nosso método nos ambientes que suportam o roteamento pelo caminho mais curto . . . 89

Lista de Tabelas

2.1 Taxonomia de métodos para estimação da matriz de tráfego 12

3.1 Diferenças entre o roteamento pelo caminho mais curto e o roteamento ótimo por múltiplas rotas 38

3.2 Perdas da Microsoft ($F(\mathbf{x}, \mathbf{y})$) 39

3.3 Perdas da Netscape ($f(\mathbf{x}, \mathbf{y})$) 40

3.4 Perdas da Netscape (nível inferior) 40

3.5 Perdas da Microsoft (nível superior) 41

3.6 Os resultados obtidos pelo método da roleta 52

3.7 Os resultados no processo de seleção 53

3.8 Após o cruzamento 53

3.9 Geração nova 54

4.1 Comparação com o método de tomogravidade 65

4.2 Diferenças de MRE para diferentes níveis de ruído 68

4.3 Ambiente de experimentos 68

5.1 Comparação dos algoritmos 88

Notação

$\mathbf{R}$	=	conjunto de nós
$\mathbf{N}$	=	conjunto de pares origem-destino
$\mathbf{M}$	=	conjunto de canais
R	=	número de nós
N	=	número de pares origem-destino
M	=	número de canais
x_i	=	tráfego do par origem-destino i, $i = 1, \ldots, N$
x_{ij}	=	tráfego na rota j do par OD i
y_a	=	tráfego no canal a, $a = 1, \ldots, M$
$\mathbf{U}, \mathbf{V}$	=	matrizes de peso
l_a	=	função de custo associada ao canal a
C_a	=	capacidade do canal a
$\mathbf{x}$	=	$(x_1, x_2, \ldots, x_N)^T$ vetor de tráfego entre os pares OD (matriz de tráfego)
$\mathbf{y}$	=	$(y_1, y_2, \ldots, y_M)^T$ tráfego nos canais
$\mathbf{x}^{(p)}$	=	matriz de tráfego *a priori*
$\mathbf{x}_g$	=	estimativa da matriz de tráfego obtida por modelo de gravidade
$\hat{\mathbf{x}}$	=	$(\hat{x_1}, \hat{x_2}, \ldots, \hat{x_N})^T$ matriz de tráfego *a priori*
$\hat{\mathbf{y}}$	=	$(\hat{y_1}, \hat{y_2}, \ldots, \hat{y_M})^T$ tráfego medido nos canais ($\hat{\mathbf{y}} = \mathbf{y} + \eta$, onde η é o erro de medição)
δ_a^{ij}	=	fração que tráfego da rota j do par OD i passa o canal a.
$\mathbf{A}_{MxN}$	=	matriz de roteamento
Γ_n^i	=	rota i do par origem destino n

Capítulo 1

Introdução

O problema chave enfrentado pelos engenheiros de tráfego consiste em determinar qual a melhor estratégia para transmissão de dados de forma a atender os requisitos de desempenho das aplicações. Quando uma nova rede é concebida, seus administradores precisam estabelecer como o tráfego será distribuído (balanceado) evitando os congestionamentos nos roteadores e conseqüentemente mantendo os retardos de transmissão de pacote dentro de níveis aceitáveis. Uma vez implementada a rede, é necessário adaptá-la às necessidades de seus usuários. Neste contexto, a matriz de tráfego é uma peça fundamental. Ela descreve o volume de tráfego que flui entre os nós origem e destino em uma rede de comunicação, refletindo as demandas dos usuários. No contexto da Internet, os nós podem ser links, roteadores, Pontos de Presença (PoPs), etc. Por exemplo, em uma matriz de tráfego de roteador-roteador, o tráfego de um determinado roteador inclui todos os clientes conectados a este roteador.

Obter a matriz de tráfego é muito importante para os operadores de redes porque muitas tarefas de engenharia de tráfego e de gerenciamento de redes necessitam do conhecimento fornecido pela matriz de tráfego para melhorar o desempenho e a eficiência de redes. Estas tarefas incluem o projeto de topologia lógica, o planejamento de capacidade, a configuração de protocolo de roteamento, a provisão para acordo de nível de serviço (SLAs), o balanceamento de carga e análise de confiabilidade de redes.

Embora seja importante conhecer a matriz de tráfego, a sua exata determinação não é factível nas redes de computadores modernas. Isto ocorre porque o suporte nos roteadores para a medição da matriz de tráfego é deficiente devido ao grande número de pares origem-destino, ao enorme volume de tráfego em cada link e a falta de infra-estrutura de medição. Assim, os operadores de rede precisam estimar a matriz de tráfego a partir dos dados disponíveis, descritos a seguir [42].

Considere uma rede envolvendo $R = |\mathbf{R}|$ nós e $M = |\mathbf{M}|$ canais (*links*), onde C_m é a capacidade de um canal m, e cada canal é uma aresta direcionada. Um par origem-destino n conectando uma fonte s_n a um destino d_n ($s_n, d_n \in \mathbf{R}$) é um par ordenado (s_n, d_n), e $\mathbf{N}$ é o conjunto composto por todos os N pares origem-destino(OD). Um par origem-destino n está relacionado a uma rota Γ_n, que é um subconjunto de $\mathbf{M}$.[1] Cada usuário u_n transmite dados a uma taxa x_n. Seja $\mathbf{A}$ a matriz de roteamento, onde $A_{mn} = 1$ se $m \in \Gamma_n$, e $A_{mn} = 0$ caso contrário. Finalmente, $\boldsymbol{C} = (C_1, \ldots, C_M)^T$, e $\mathbf{x} = (x_1, \ldots, x_N)^T$.

Tipicamente, o administrador da rede conta com as medições de tráfego feitas em cada um dos canais da rede, ou seja, com um vetor $\mathbf{y} \leq \boldsymbol{C}$, bem como com a estratégia de roteamento sendo empregada, caracterizada por $\mathbf{A}$, e o seu objetivo é estimar $\mathbf{x}$.

A relação entre a matriz de tráfego, a matriz de roteamento e o tráfego nos canais pode ser descrita pelo seguinte sistema de equações lineares:[2]

$$\mathbf{A}_{(M\times N)}\mathbf{x}_{(N\times 1)} = \mathbf{y}_{(M\times 1)} \tag{1.1}$$

$\mathbf{A}_{(M\times N)}$ matriz de roteamento

$\mathbf{x}_{(N\times 1)}$ matriz de tráfego

$\mathbf{y}_{(M\times 1)}$ tráfego nos canais

[1] Adiante iremos assumir que entre cada par origem destino podem existir múltiplas rotas.

[2] Por motivos históricos, o *vetor* $\mathbf{x}$ é chamado de *matriz* de tráfego.

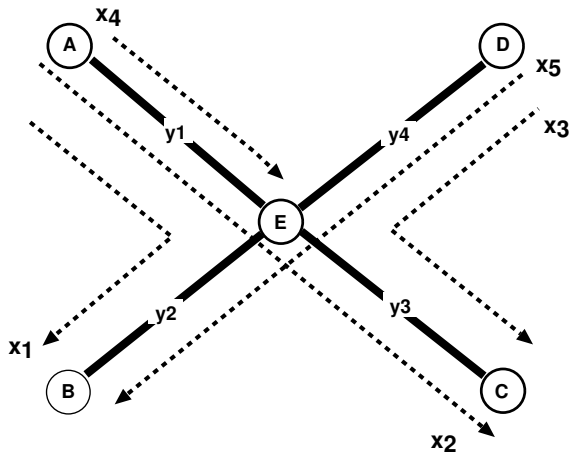


Figura 1.1: Exemplo de estimação de matriz de tráfego

Considere o exemplo apresentado na Figura 1.1. Temos quatro nós A, B, C e D interconectados por um quinto nó, o roteador E. Além disto, dispomos de oito canais direcionados, mas vamos nos deter naqueles quatro por onde passa o tráfego: L_1 (AE), L_2 (EB), L_3 (EC) e L_4 (DE) – os outros quatro canais são EA, ED, CE e BE. Assumimos que as informações sobre o tráfego individual nestes canais estão disponíveis, e compõem o vetor $\mathbf{y} = (y_1, y_2, y_3, y_4)^T$. [3] Os pares OD estão representados pelas linhas tracejadas. O problema chave consiste em estimar a matriz de tráfego $\mathbf{x} = (x_1, x_2, x_3, x_4, x_5)^T$. A relação $\mathbf{y} = \mathbf{A}\mathbf{x}$ é dada por:

$$\begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{bmatrix} \tag{1.2}$$

Em particular, x_1 representa o tráfego transmitido do *host* A para o *host* B e $y_1 = x_1 + x_2 + x_4$.

Conforme mencionado anteriormente, nos ambientes de rede modernos $\mathbf{y}$ e $\mathbf{A}$ estão disponíveis para o administrador. De fato, o tráfego nos canais pode ser obtido

[3] $\mathbf{x}^T$ é a transposta da matriz $\mathbf{x}$.

a partir de medições feitas com o protocolo SNMP. Já a matriz de roteamento pode ser computada, por exemplo, se assumirmos que o administrador possui informações sobre a topologia da rede – ou seja, sobre o 'custo'[4] de transmissão de dados em cada um dos canais. Afinal, tendo em mãos a informação sobre o custo de cada canal pode-se aplicar um algoritmo de roteamento (e.g. algoritmo de Dijkstra) para determinar o caminho de menor custo entre cada par de *hosts* na rede. Cada caminho corresponde a uma das colunas da matriz $\mathbf{A}$. O caminho ótimo depende de fatores como custo, capacidade e política administrativa. Mais detalhes sobre este tópico, estão incluidos no Capítulo 3.

Em resumo: o administrador do sistema conhece $\mathbf{A}$ e $\mathbf{y}$ e deseja computar $\mathbf{x}$ tal que $\mathbf{Ax} = \mathbf{y}$. Aparentemente, *basta* resolver este sistema de equações lineares. Entretanto, o problema é que o número de canais (linhas na matriz $\mathbf{A}$) é sempre muito menor que o número de pares origem-destino (colunas na matriz $\mathbf{A}$), o que significa que o sistema $\mathbf{Ax} = \mathbf{y}$ apresenta infinitas soluções! Portanto, para estimar o valor de $\mathbf{x}$ precisaremos conhecer propriedades extras sobre o sistema que estamos analisando, a fim de poder estabelecer uma classificação das soluções do sistema, de acordo com suas qualidades, e escolher aquelas que melhor atendam às necessidades do operador da rede. É justamente este problema que iremos abordar nos próximos capítulos.

Mecanismos de Roteamento

Um ingrediente fundamental dos métodos de estimação da matriz de tráfego é o mecanismo de roteamento sendo adotado na rede. Em redes de automóveis, por exemplo, a estimação da matriz de tráfego é feita baseando-se na hipótese de que os usuários são egoístas (*selfish routing*). Não existem protocolos estabelecendo como cada usuário deve selecionar suas rotas, o roteamento é não cooperativo.

Já nas redes de computadores, os mecanismos de roteamento podem ser descritos da seguinte forma seguindo os conceitos dados em Kurose e Ross[31].

[4] o custo de um canal pode, por exemplo,ser uma medida de desempenho como retardo ou utilização ou mesmo custo em dinheiro por unidade de transmissão daquele canal

> Para transferir pacotes de um *host* origem a um *host* destino, a camada de rede deve determinar o caminho ou a rota que os pacotes devem seguir. No núcleo de qualquer protocolo de roteamento está o algoritmo (o algoritmo de roteamento) que determina o trajeto para um pacote. A finalidade do algoritmo de roteamento é simples: dada uma série de roteadores conectados por meio de canais, o algoritmo de roteamento descobre um 'bom' caminho entre a fonte e o destino. Um 'bom' caminho é aquele que otimiza a métrica de "custo"escolhida. Para o caminho de menor custo, a soma dos custos dos canais no caminho é a mínima dentre todos os possíveis caminhos entre a fonte e o destino. Note que se todos os canais tiverem o mesmo custo (por exemplo, igual a 1) o caminho de menor custo será o caminho mais curto (isto é, o caminho que passa pelo menor número de canais entre a origem e o destino).

Na Internet de hoje os protocolos de roteamento intra-domínio mais populares são aqueles que aqui denotamos como *protocolos tradicionais de roteamento por caminho mais curto*, como o protocolo OSPF (Open Shortest Path First). Os usuários da rede seguem estritamente o protocolo tradicional de roteamento por caminho mais curto, almejando convergir para um estado ótimo do ponto de vista global. Nesta linha, uma outra classe de protocolos de roteamento consiste nos chamados *protocolos de roteamento ótimo por múltiplas rotas*. As diferenças entre os protocolos de roteamento de caminho mais curto e os protocolos de roteamento ótimo por múltiplas rotas serão descritas brevemente na próxima seção, e em mais detalhes no Capítulo 3. A seguir, descrevemos as semelhanças.

Tanto o protocolo de roteamento por caminho mais curto quanto o protocolo de roteamento ótimo por múltiplas rotas têm objetivos comuns [5] (Figura 1.2). Em cenários envolvendo uma alta taxa de dados injetada na rede (lado direito da Figura 1.2) o efeito de um bom roteamento é aumentar a vazão, para um certo valor de atraso médio. Em cenários envolvendo uma baixa ou média taxa de dados injetada na rede (lado esquerdo da Figura 1.2) o efeito de um bom roteamento é diminuir o atraso [5, Seção 5.1].

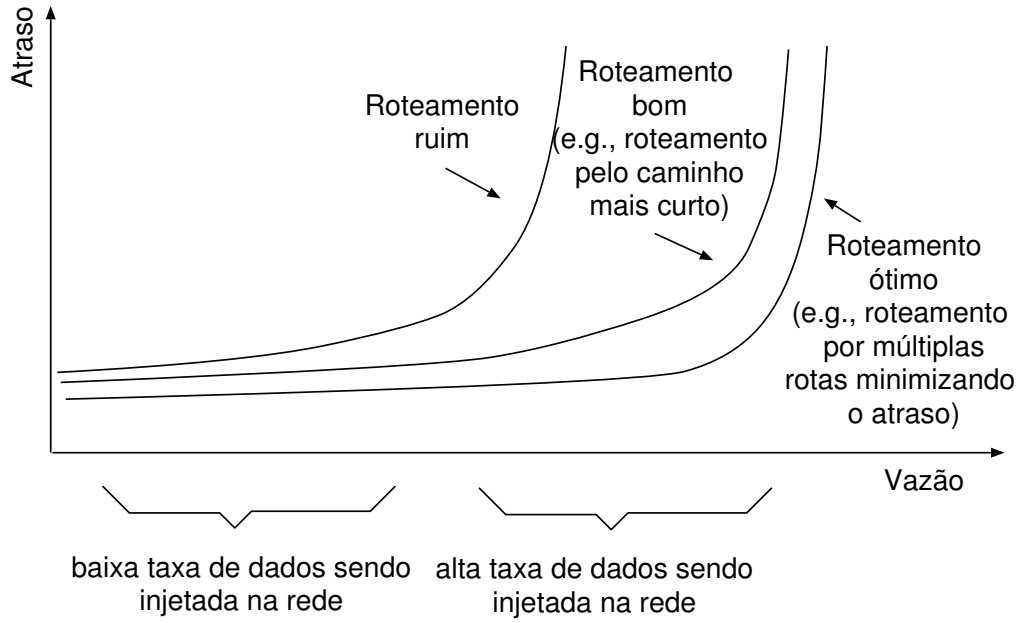


Figura 1.2: As curvas de desempenho dos mecanismos de roteamento.

Na Figura 1.3 temos uma classificação dos métodos de roteamento considerados. No presente trabalho, iremos focar nos métodos cooperativos de roteamento.

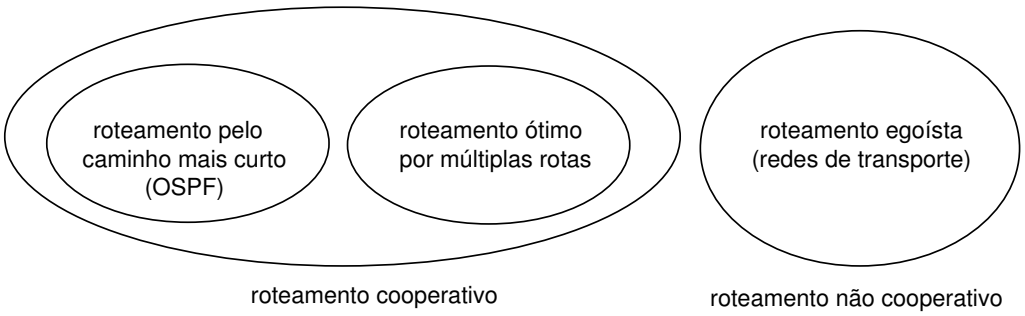


Figura 1.3: Mecanismos de roteamento.

1.1 Objetivo da Tese

As abordagens tradicionais [37] para a estimação da matriz de tráfego partem do princípio de que a matriz de roteamento é estática, e possui elementos iguais a zeros e uns. Estas suposições são adequadas quando assumimos que o protocolo é

um protocolo de roteamento pelo caminho mais curto tradicional, como o OSPF [5]. Novas abordagens [48] assumem uma matriz de tráfego dinâmica, mas ainda assim consideram que a cada instante só existirá uma rota sendo utilizada entre cada par origem-destino.

A estimação da matriz de tráfego em redes envolvendo o *selfish routing* tem sido estudada há décadas pelos engenheiros de transportes. A comunidade dedicada às redes de computadores tem recentemente voltado atenção para o *selfish routing* [44], mas tradicionalmente vem lidando com cenários envolvendo o roteamento pelo caminho mais curto tradicional (como o OSPF). Neste trabalho estudamos a estimação da matriz de tráfego usando uma abordagem baseada no roteamento ótimo por múltiplas rotas (Figura 1.3), além de abordagem tradicional onde um protocolo de roteamento pelo caminho mais curto é usado.

Os novos protocolos de roteamento, dentre os quais destaca-se o *roteamento ótimo por múltiplas rotas* (Capítulo 3), diferem dos protocolos de roteamento pelo caminho mais curto tradicionais em questões relativas à forma como o *custo de transmissão dos dados* em cada rota é calculado e questões relativas ao *número de rotas* concomitantemente usadas entre cada par origem destino. Mais precisamente, neste trabalho assumimos que:

- o custo de transmissão em um canal é função de seu tráfego (ou seja, se o tráfego em um canal for x, o custo de transmissão de dados no mesmo é caracterizado por $l(x)$);
- concomitantemente múltiplas rotas são usadas entre cada par origem-destino, e estas rotas podem variar substancialmente ao longo do tempo.

[5] No OSPF, podem existir múltiplos caminhos de custo igual para um mesmo par origem-destino. Neste caso, o tráfego é distribuído igualmente nestes caminhos de custo igual e a matriz de tráfego possui elementos fracionários. Nesta tese, iremos considerar apenas o caso em que só existe um caminho ótimo par cada par origem-destino

O objetivo deste trabalho é responder à seguinte questão:

Problema A

- Como estimar a matriz de tráfego, assumindo-se que o roteamento na rede é executado por um protocolo de roteamento pelo caminho mais curto padrão como o OSPF?

Problema B

- Como estimar a matriz de tráfego, assumindo-se que o roteamento na rede não é executado por um protocolo de roteamento pelo caminho mais curto padrão como o OSPF, nem por um mecanismo de roteamento egoísta (*selfish routing*), mas sim por um mecanismo de *roteamento ótimo por múltiplas rotas*?

1.2 Contribuições

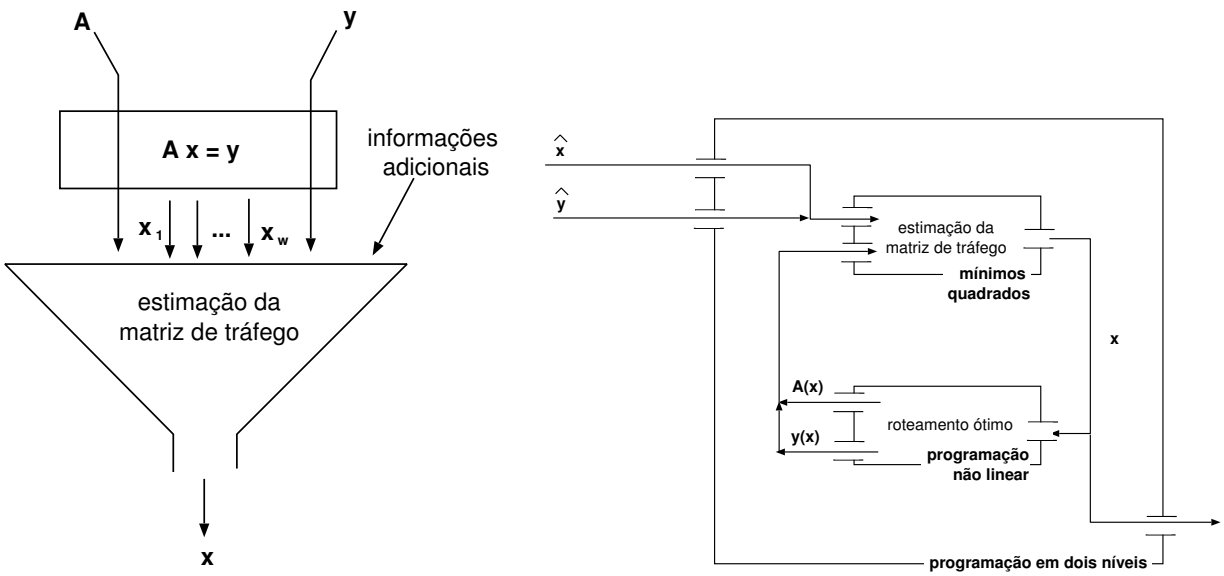
1.2.1 Problema A

Propomos uma abordagem que formula o problema da estimação da matriz de tráfego como um problema de otimização com restrições de não-negatividade. Um método de projeção é usado para resolver o problema. Fizemos experimentos com dados sintéticos e dados reais obtidos da Rede Abilene. Finalmente, mostramos que a nossa abordagem estima a matriz de tráfego mais precisamente do que o método tomogravity[58], que é um método mais conhecido e usado para estimar a matriz de tráfego, além de ser uma abordagem precisa e rápida.

1.2.2 Problema B

Introduzimos um modelo para estimar a matriz de tráfego quando se considera *roteamento ótimo por múltiplas rotas.*

Na Figura 1.4(a), ilustramos como funcionam as estratégias clássicas para estimar a matriz de tráfego: dados $\mathbf{A}$ e $\mathbf{y}$, e informações adicionais sobre o sistema



(a) (b)

Figura 1.4: Comparação entre os métodos clássicos de estimação da matriz de tráfego e o método proposto.

sendo analisado, como por exemplo, uma estimativa *a priori* $\mathbf{x}^{(p)}$ da matriz de tráfego, obtém-se a estimativa final da matriz de tráfego $\mathbf{x}$.

Na Figura 1.4(b), ilustramos o esquema do modelo proposto. Consideramos um sistema composto por dois módulos: um responsável pela estimação da matriz de tráfego, e o outro pelo *roteamento ótimo por múltiplas rotas*, que será introduzido no Capítulo 3. Para analisar este sistema formado por dois componentes, utilizamos a técnica da *programação em dois níveis*, que emerge como uma metodologia natural para modelar esse sistema, e será também apresentada no Capítulo 3.

Formulamos o problema sendo de programação em dois níveis. Um algoritmo iterativo e um algoritmo genético foram propostos para resolver o problema.

É importante destacar que assumimos que a rede sendo analisada necessariamente suporta o mecanismo de roteamento ótimo por múltiplas rotas. Em alguns casos, é possível que a rede execute um algoritmo de roteamento pelo caminho mais curto, cujo desempenho aproxima-se daquele associado ao roteamento ótimo por múltiplas rotas (Figura 1.2). Nestes casos, a estimação da matriz de tráfego tradicional(caminho mais curto) pode ser empregada usando a abordagem proposta da

Figura 1.4(b).

1.3 Roteiro

O restante deste trabalho está organizado da seguinte forma: no capítulo 2 será apresentado um breve histórico dos métodos para estimar a matriz de tráfego; no capítulo 3 consta uma introdução aos fundamentos teóricos, incluindo uma breve introdução ao roteamento ótimo por múltiplas rotas e à programação em dois níveis; o capítulo 4 apresenta nossa primeira contribuição. É apresentado o modelo e o algoritmo propostos para estimar a matriz de tráfego quando o roteamento empregado é o roteamento pelo caminho mais curto. O algoritmo foi implementado e foram feitas comparações numéricas com o método tomogravity. O capítulo 5 responde à questão: como estimar a matriz de tráfego, assumindo-se que o roteamento na rede é roteamento ótimo por múltiplas rotas? O capítulo 6 traz as conclusões e os trabalhos futuros.

Capítulo 2

Trabalhos Relacionados

Nos últimos anos, muitos métodos foram propostos para estimar a matriz de tráfego e a Tabela 2.1 mostra as abordagens principais. Para resolver um sistema com infinitas soluções(indeterminado), $\mathbf{Ax}=\mathbf{y}$, onde $\mathbf{A}$ tem N linhas e M colunas, com $N >> M$, o fator crucial para a estimação é a informação extra sobre o problema a ser resolvido. Até hoje, os modelos para obter as informações extras são: modelo de gravidade; modelo que assume a relação média-variância de tráfego e modelo de conexão independente. Baseado nas técnicas e nas informações extras, podemos classificar os métodos presentes na literatura para estimar a matriz de tráfego em três classes. Na primeira classe constam aqueles que procuram enquadrar a matriz de tráfego dentro de uma estrutura pré-estabelecida, diminuindo, assim, o número de soluções para o sistema indeterminado apresentado no primeiro capítulo. Na segunda classe constam os métodos que, ao invés de fixarem uma estrutura pré-estabelecida para a matriz de tráfego, buscam as soluções mais adequadas dentro do conjunto de todas as soluções. Neste caso, estabelece-se uma função que caracterize a qualidade das soluções encontradas, e aplica-se um algoritmo de otimização para encontrar a melhor delas. Na terceira e última classe constam os métodos recentes que não se enquadram em nenhuma das duas anteriores. Estes métodos são categorizados por Soule et al. [47]. como métodos de terceira geração. O caráter comum deste métodos é que os monitores de fluxos são usados para calibrar o seu próprio modelo em todos os métodos. Finalmente, introduzimos o modelo de conexão inde-

Métodos	Informação extra
Simple gravity model[58]	modelo de gravidade
Generalized gravity model[58]	modelo de gravidade
Choice model[37]	modelo de gravidade Mlogit
Fanout model [42, 22]	fanout
Tomogravidade (Tomogravity) [58]	modelo de gravidade
Information theoretic approach[59]	modelo de gravidade
Bayesian inference [51]	distribuição de Poisson
Iterative Bayesian inference[54]	distribuição de Poisson
Network tomography [53]	Poisson, relação média-variância(Poisson)
Time varying network tomography[10]	Poisson, relação média-variância(geral)
Scalable likelihood approach [11]	Normal, relação média-variância(c=1)
Pseudo likelihood approach [34]	Poisson, relação média-variância(geral)
Quick method [25]	normal, relação média-variância(geral)
Linear programming [21, 16]	nenhum
Worst Case bounds by LP [22]	nenhum
Principal components analysis [32]	nenhum
Kalman filter [47, 49]	nenhum
Independent connection model[15]	modelo de conexão independente
Route change method [48]	habilidade para mudar os pesos de canais

Tabela 2.1: Taxonomia de métodos para estimação da matriz de tráfego

pendente e o método que usa os dados do NetFlow[12] nos elementos da matriz de tráfego a priori, propostos recentemente.

2.1 Métodos baseados em Inferência Estatística

Uma das maneiras de diminuir o número de soluções do sistema indeterminado $\mathbf{Ax} = \mathbf{y}$, apresentado no primeiro capítulo, consiste em introduzir informações adicionais sobre a matriz de tráfego no modelo. Tipicamente, escolhe-se uma certa distribuição para os elementos da matriz de tráfego e, posteriormente, infere-se os

parâmetros desta distribuição que melhor se adaptam aos dados colhidos [53, 10, 51]. Daí segue o nome desta abordagem: inferência estatística.

A distribuição escolhida para os elementos da matriz de tráfego pode ser, por exemplo, Poisson ou Gaussiana. Para inferir os parâmetros de uma distribuição, diferentes técnicas foram propostas na literatura: Vardi et al. [53] usaram o método da geração de momentos, Tebaldi et al. [51] adotaram o método da inferência bayesiana, e Cao et al. [10] empregaram o método de estimação da máxima verossimilhança. Um estudo comparativo destes métodos foi realizado por Medina et al. [37]. Os autores deste último trabalho concluíram que os métodos de inferência estatística são muito sensíveis a erros, e pequenas distorções nos dados de entrada podem comprometer fortemente a qualidade dos resultados.

2.1.1 Inferência Bayesiana

Em [51], um método de inferência bayesiana foi proposto. O objetivo é computar a distribuição condicional da probabilidade, $P(X|Y)$, sendo X o tráfego dos pares OD e Y o tráfego nos canais. Para alcançar este objetivo, é necessário ter uma distribuição a priori para X, ou seja $p(X)$. Em [51], é assumido que $P(X_i)$ segue uma distribuição de Poisson com media λ_i, ou seja, $X_j \sim \text{Poisson}(\lambda_i)$, independentemente para todos os pares OD. Então o objetivo é obter a distribuição condicional $P(X, \lambda|Y)$, ou seja, computar a distribuição conjunta de X e λ dado o tráfego nos canais Y. Como é difícil calcular a distribuição posterior, o método de simulação iterativa como MCMC(Markov Chain Monte Carlo) foi usado para obter a distribuição posterior. O amostrador foi usado nas iterações seguintes

$$\begin{cases} \lambda^i & = & P(\lambda|x^i, y) \\ x^{i+1} & = & P(x^i|\lambda^i, y) \end{cases} \tag{2.1}$$

Vaton et al. [54] propõem um método iterativo para melhorar a distribuição a priori dos elementos da matriz de tráfego. A matriz de tráfego estimada de uma medição do tráfego nos canais é usada na estimação seguinte usando medições novas

do tráfego nos canais. A Figura 2.1 ilustra as iterações e as informações trocadas entre duas "caixas", que melhoram significativamente a estimação da matriz de tráfego. Na segunda "caixa", o algoritmo EM (Expectation Maximization) é usado para reestimar os parâmetros dos fluxos OD. Estes parâmetros são escolhido como melhor a priori para a primeira "caixa". O processo é repetido até que nenhuma mudança significativa seja feita na matriz de tráfego estimada.

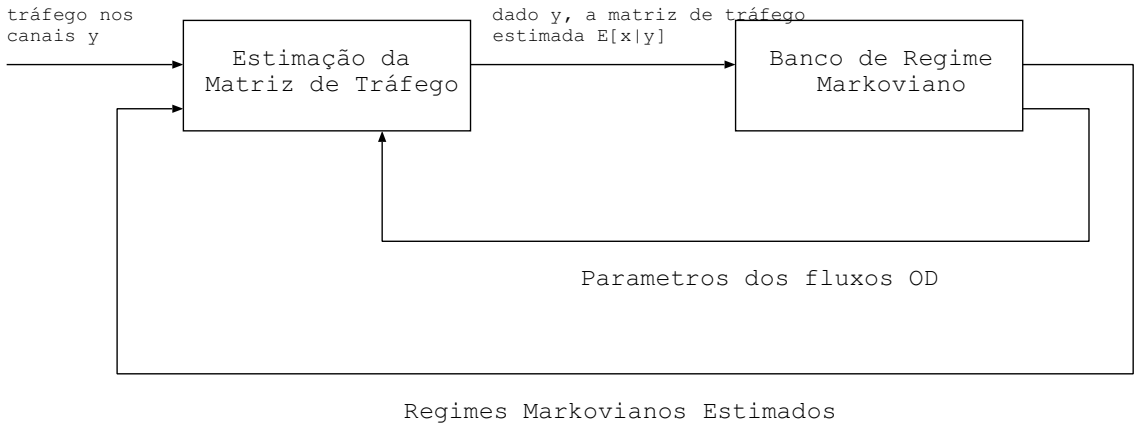


Figura 2.1: O método iterativo do Vaton-Gravey

2.1.2 Estimação de Máxima Verossimilhança e EM

Os dois métodos de inferência bayesiana, introduzidos acima, assumem que os fluxos OD seguem a distribuição de Poisson. Nesta abordagem [53, 10] o tráfego dos pares OD é modelado de acordo com uma distribuição Gaussiana, $\mathbf{x} \sim Normal(\lambda, \Sigma)$, onde os componentes de $\mathbf{x}$, $\mathbf{x}_i$ são modelados como variáveis aleatórias normais e independentemente distribuídas. Em função da relação $\mathbf{Ax} = \mathbf{y}$, a suposição Gaussiana em $\mathbf{x}$ implica que $\mathbf{y} \sim Normal(A\Lambda, A\Sigma A')$, onde $\Lambda = (\lambda_1, \ldots, \lambda_n)$ é vetor de tráfego médio dos pares OD e a matriz de covariância é dada por $\Sigma = \phi diag(\lambda_1^b, \ldots, \lambda_c^b)$, onde ϕ é um parâmetro e c é um número constante.

Uma das diferenças entre este método e os dois métodos precedentes é que este último incorpora conjuntos múltiplos de medições nos canais. Seja $My_1, \ldots, My_K$ o conjunto do K medições SNMP (Simple Network Management Protocol) e $\theta = (\Sigma, \phi)$ o conjunto dos parâmetros a estimar. O estimação da máxima verossimilhança

é computada encontrando o máximo da seguinte função de log-verossimilhança,

$$l(\theta|My_1,\ldots,My_K) = -\frac{k}{2}\log|A\Sigma A'| - \frac{1}{2}\Sigma_{k=1}^{K}(y_k - A\lambda)'(A\sigma A')^{-1}(y_k - A\Lambda) \quad (2.2)$$

O algoritmo EM é o algoritmo amplamente aplicável que fornece um procedimento iterativo para computar estimação de máxima verossimilhança nas situações onde a máxima verossimilhança não seja trivial devido à ausência de alguns dos dados.

Vardi[53] foi o primeiro a introduzir o termo "Network tomography"para estimar a matriz de tráfego. Com a suposição da distribuição de Poisson para o tráfego dos pares OD em uma topologia geral de rede, ele especifica a identificação do modelo de Poisson e desenvolve um algoritmo EM (Expectation Maximization) para estimar os parâmetros do modelo de Poisson. Para diminuir a dificuldade em executar o algoritmo EM, é proposto um método de estimação de momento e discute brevemente o modelo Normal como uma aproximação ao modelo de Poisson.

Cao et al. [10] modificam o método de Vardi para que possa ser usado em uma rede real. O método de Cao tem os seguintes passos: primeiramente, um algoritmo EM é usado para computar a estimativa de θ, isso é, $\hat{\theta} = (\hat{\Sigma}, \hat{\phi})$. Em seguida, cada par OD j é estimado no tempo t por $\hat{x}_{j,t} = E[x_{j,t}|\hat{\theta}, Y]$. No último passo, um algoritmo IPF (Iterative Propotional Fitting) é usado. Um novo método foi posposto em [11]. Este método é um método de tipo "Dividir e Conquistar". A sua idéia é dividir um problema em vários subproblemas e resolvê-los separadamente.

Liang e Yu [34] propõem um método de pseudo-verossimilhança para acelerar a estimação de parâmetros.

Juva et al. [25] propõem um método rápido (*quick method*) baseado na matriz de covariância do tráfego nos canais. O método obtém a distribuição a priori baseado na matriz de covariância do tráfego nos canais e na relação funcional entre média e variância.

2.2 Métodos baseados em Otimização

Uma maneira de filtrar as melhores soluções de um sistema do tipo $\mathbf{Ax} = \mathbf{y}$ consiste em estabelecer uma função objetivo que de alguma forma possa ser usada para ordenar as soluções do problema de acordo com suas qualidades, e, em seguida, aplicar um algoritmo de otimização para encontrar as soluções ótimas dessa função. Se a função objetivo for linear, aplica-se o método da programação linear (PL), e é exatamente com esse que começaremos.

2.2.1 Programação Linear

O método de estimação da matriz de tráfego usando Programação Linear (PL) foi proposto por Goldschmidt et al. [21]. A formulação do problema é dada por:

$$\max \sum_{j=1}^{N} w_j x_j \tag{2.3}$$

$$\text{s.t.} \sum_{j=1}^{N} A_{ij} x_j \leq y_l \qquad l = 1, ...M \tag{2.4}$$

$$\sum_{l=(i,j)} y_l a_{lk} - \sum_{l=(j,i)} y_l a_{lk} = \begin{cases} x_k & \text{se } j = \text{origem de } k \\ -x_k & \text{se } i = \text{destino de } k \\ 0 & \text{caso contrário} \end{cases} \tag{2.5}$$

$$x_j \geq 0 \qquad \forall j \tag{2.6}$$

onde w_j é o peso associado ao par OD j e a_{lk} é fração do tráfego x_k passa no canais l. O objetivo (2.3) consiste em maximizar o tráfego ponderado transportado na rede. As desigualdades (2.4) são restrições de canais, as igualdades (2.5) são restrições de fluxo e (2.6) são restrições de não negatividade. Em [21], o autor considera que $w_i = 1$ para todo $i = 1, 2, ...N$ vai levar às soluções em que o tráfego dos pares OD curtos (nós vizinhos) serão atribuídos valor muito grande, enquanto os pares distantes, ou seja, aqueles com muitos nós entre eles) serão atribuídos freqüentemente o valor zero. Nós sabemos que estas não são as soluções que estamos procurando na prática. Conseqüentemente, em [21] é sugerido que uma função objetiva boa é uma

na qual os w_i refletem os comprimentos de rotas. Em [37], é mostrado que o método Programação Linear não funciona bem porque os erros dos tráfegos estimados são realmente altos. Então, o problema da abordagem apresentada por Goldschmidt et al., segundo Medina et al. [37], consiste no fato de que ela não é escalável, não podendo, portanto, ser empregada em redes de larga escala. Porém, Eum et al. [16] resolveram o problema de Programação Linear e concluem que os resultados obtidos são muitos bons e tão exatos como o método de tomogravidade e a obtenção é rápida. Eles resolvem o problema de Programação Linear usando o método de pontos interiores, ao invés do método tradicional *Simplex*.

Gunnar et al. [22] aplicam outro método baseado em programação linear para achar o limite inferior e superior do tráfego dos pares OD no pior caso(*worst-case bound*). A formulação é: para cada $i = 1, .., N$ fixo, deve-se resolver

$$\begin{aligned} \min(ou\,\max) \quad & \mathbf{x}_i \\ \text{sujeito às restrições} \quad & \mathbf{Ax} = \mathbf{y} \\ & \mathbf{x} \geq 0 \end{aligned} \tag{2.7}$$

Obviamente, este método é somente interessante quando é achado um limite superior menor do que $\max_j y_j$ (canal j na rota do par OD i) e um limite inferior maior do que zero. Nota-se também que o método é computacionalmente caro, pois requer resolver duas programações lineares para cada tráfego de pares OD. A Figura 2.2, retirada de Gunnar et al. [22] mostra que o valor médio do limite inferior e superior de cada fluxo OD dá uma estimativa de tráfego relativamente boa. A Figura 2.3, retirada de Gunnar et al. [22], mostra que este valor médio pode ser usado como a distribuição a priori para o problema de estimar a matriz de tráfego. Para o conjunto de dados considerados em [22], o método de *worst case bounds* oferece uma estimativa a priori melhor que o modelo de gravidade.

2.2.2 Programação Não Linear

Nesta seção abordamos os métodos de estimação da matriz de tráfego que possuem uma função objetivo não linear. Em geral, a não linearidade desta função

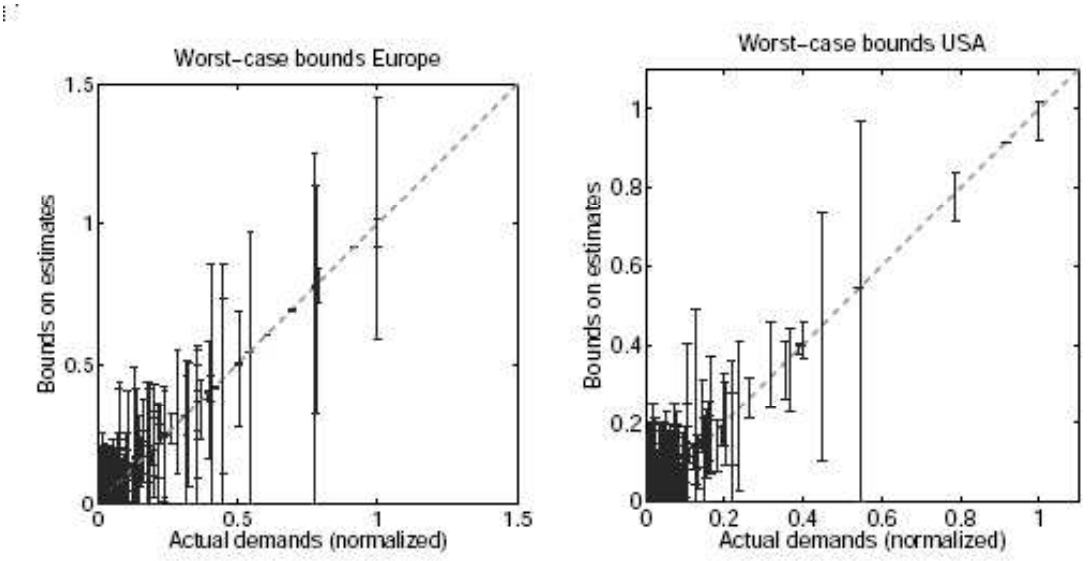


Figura 2.2: Worst-case bound na rede Europa e Americana

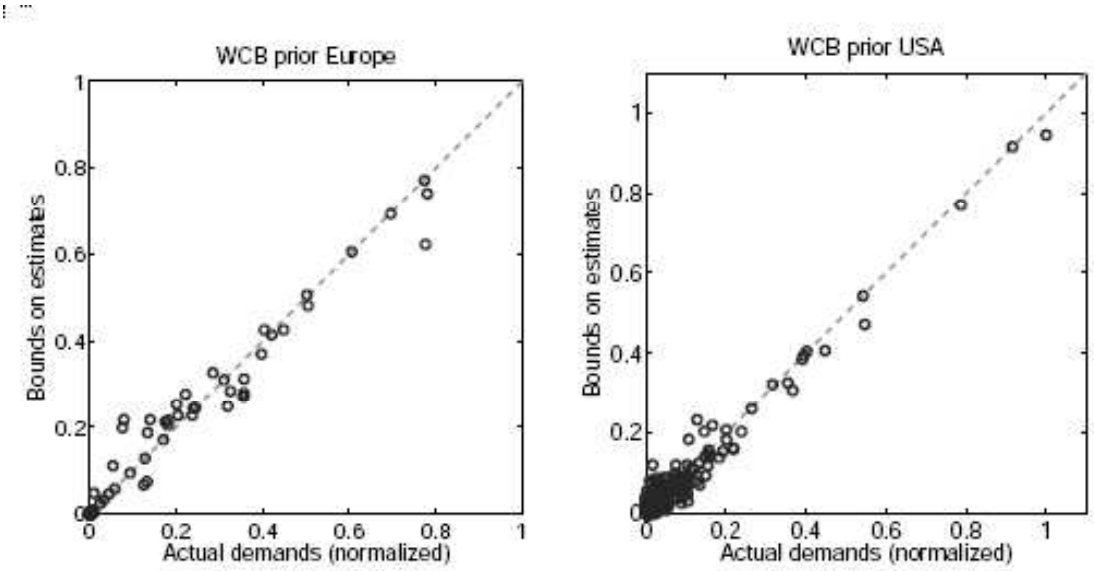


Figura 2.3: A Priori obtido pelo worst-case bound

advém do fato de estes métodos assumirem que o objetivo do problema de programação é minimizar a distância entre duas quantidades, e tipicamente a função distância é não linear.

Método de Kruithof

A idéia de que a programação não linear pode ser aplicada ao problema de estimação da matriz de tráfego foi primeiramente notada por Kruithof [29]. Kruithof considerou o problema de estimar o tráfego ponto a ponto em uma rede telefônica, baseando-se em uma matriz de tráfego conhecida (a matriz de tráfego *a priori*, ou $\mathbf{x}^{(p)}$) e nas medidas do tráfego total que entra e sai de cada nó na rede [1] . O modelo de Kruithof foi analisado por Krupp [30], que encontrou uma nova interpretação, usando a teoria de informação: o método de Kruithof minimiza a distância de Kullback-Leibler entre a matriz de tráfego estimada e a matriz de tráfego *a priori*.

Seja $D(\mathbf{a}, \mathbf{b})$ a função de distância entre $\mathbf{a}$ e $\mathbf{b}$. Krupp reescreveu o método de Kruithof da seguinte forma:

$$\begin{aligned} \min \quad & D(\mathbf{x}, \mathbf{x}^{(p)}) \\ \text{sujeito às restrições} \quad & \mathbf{A}\mathbf{x} = \mathbf{y} \\ & \mathbf{x} > 0 \end{aligned} \tag{2.8}$$

Claramente, o objetivo do problema de Kruithof consiste em minimizar a distância entre o tráfego estimado e o tráfego conhecido *a priori*.

Método de Tomogravidade (Tomogravity)

Zhang et al. [58] desenvolveram um método para estimar a matriz de tráfego baseado na seguinte suposição:

$$x_{ij} \propto x_{i*} \cdot x_{*j} \tag{2.9}$$

onde $\propto$ significa propocional, x_{ij} é o tráfego de i para j, x_{i*} é o tráfego total gerado pelo i e x_{*j} é o tráfego total direcionado para j. Note que

$$x_{ij}/x_{i*} \propto x_{*j} \tag{2.10}$$

ou seja, independente de qual seja o origem i, a fração do tráfego gerado de i para j (lado esquerdo da equação 2.10) é proporcional à quantidade total de tráfego

[1] Aqui, na expressão $\mathbf{x}^{(p)}$ a letra p remete-nos à palavra *priori*.

direcionado de todos os nós para j (lado direito da equação 2.10). A fração de pacotes gerados por i direcionados para j é independente de i.

Segundo a lei da gravitação de Newton,

$$F_{\text{grav}} \propto m_1 \cdot m_2 \tag{2.11}$$

onde F_{grav} é a força gravitacional entre os corpos de massas m_1 e m_2. Devido à semelhança entre as equações 2.9 e 2.11, os modelos que pressupõem uma relação do tipo 2.9 são chamados de *modelo de gravidade simples*.

O modelo de gravidade simples essencialmente assume a independência total entre fontes e destinos. No entanto, como indicado em [58], o tráfego transmitido entre as redes par-a-par se comporta muito diferente. Neste caso, é necessário considerar *o modelo de gravidade generalizado*, onde o tráfego entre *peering* é igual a zero.

Seja $\mathcal{Q}$ o conjunto de canais de acesso e $\mathcal{P}$ ou conjunto de canais de *peering*. O modelo de gravidade generalizado é definido a seguir,

$$x(i,j) = \begin{cases} 0 & para \quad i \in \mathcal{P}, j \in \mathcal{P} \\ \frac{x(i,*)}{\sum_{i \in Q} x(i,*)} x(*,j) & para \quad i \in \mathcal{Q}, j \in \mathcal{P} \\ x(i,*) \frac{x(*,j)}{\sum_{j \in Q} x(*,j)} & para \quad i \in \mathcal{P}, j \in \mathcal{Q} \\ \rho \frac{x(i,*)}{\sum_{i \in Q} x(i,*)} x(*,j) & para \quad i \in \mathcal{Q}, j \in \mathcal{Q} \end{cases} \tag{2.12}$$

onde ρ é um constante de normalização.[58]

O modelo de gravidade é usado por Zhang et al. para obter uma estimativa *a priori*, $\mathbf{x}_g$, da matriz de tráfego. Mas, normalmente, esta estimativa *a priori*, $\mathbf{x}_g$ não satisfaz a equação linear, $\mathbf{y} = \mathbf{A}\mathbf{x}$. Então, o modelo de gravidade foi usado em um problema de otimização de mínimos quadrados. O problema de otimização de mínimos quadrados é definido como,

$$\begin{aligned} \min \quad & (\mathbf{x} - \mathbf{x}_g)^T (\mathbf{x} - \mathbf{x}_g) \\ \text{sujeito a:} \quad & (\mathbf{A}\mathbf{x} - \mathbf{y})^T (\mathbf{A}\mathbf{x} - \mathbf{y}) = 0 \end{aligned} \tag{2.13}$$

A idéia central deste problema de otimização é encontrar, dentre todas as matrizes de tráfego que satisfazem à restrição $\mathbf{y} = \mathbf{A}\mathbf{x}$, aquelas que mais se aproximem

do modelo de gravidade.

A solução dos mínimos quadrados(2.13) pode ter valores negativos que obviamente não têm qualquer significado físico. A fim de obter uma solução rápida e eficaz, ao invés de formular o problema como sendo de otimização com restrições, o ajuste proporcional iterativo(Iterative Proportional Fitting, IPF) é também usado para garantir não-negatividade.

Conforme descrito em [58], o primeiro ponto de partida não é tão complexo como aquele em [10]. No IPF, é usado simplesmente a solução dos mínimos quadrados como o primeiro ponto com zero substituindo os elementos negativos da matriz. Também é relatado em [58] que leva somente algumas iterações para reduzir erros nas equações de restrições ao ponto em que eles são insignificantes na prática.

O método funciona em três etapas: uma estimativa inicial $\mathbf{x}_g$ é calculada, e, em seguida, o problema de otimização (2.13) é resolvido. Finalmente, o IPF é usado para eliminar os elementos negativos da matriz tráfego.

Na Figura 2.4, temos uma ilustração de como funciona o método proposto por Zhang et al.: primeiramente, determina-se a solução $\mathbf{x}_g$ do problema adotando-se a restrição imposta pelo modelo da gravidade (equação 2.9) e, em seguida, procura-se a solução do sistema $\mathbf{y} = \mathbf{A}\mathbf{x}$ que mais se aproxima de $\mathbf{x}_g$. Este método para estimar a matriz de tráfego é chamado método de tomogravidade (*tomogravity*).

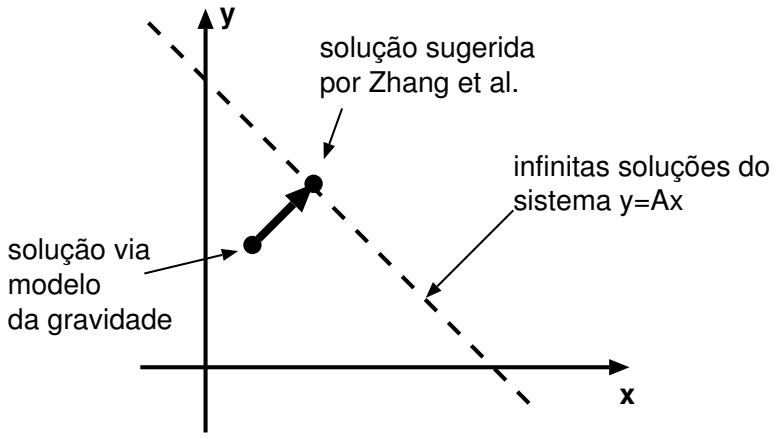


Figura 2.4: Ilustração do método de tomogravidade

Resumo

Todos os modelos de otimização não linear introduzidos têm uma mesma forma geral:

$$\begin{aligned} \min \quad & F(\mathbf{x}, \mathbf{y}) \\ \text{sujeita a} \quad & \mathbf{A}\mathbf{x} = \mathbf{y} \\ & \mathbf{x} \geq 0 \end{aligned} \tag{2.14}$$

Eles assumem uma matriz de roteamento estática. O objetivo é sempre selecionar a melhor solução dentro de todas as disponíveis, baseando-se no conjunto de equações lineares $\mathbf{A}\mathbf{x} = \mathbf{y}$. Supõe-se que os usuários da rede estão executando um protocolo de roteamento pelo caminho mais curto, e que este define completamente a matriz de roteamento $\mathbf{A}$ em questão.

Considere, por exemplo, o protocolo OSPF (Open Shortest Path First). O administrador do sistema estabelece os custos dos canais, que são mantidos fixos ao longo do tempo. Dentre todos os caminhos existentes entre cada par origem-destino, apenas aquele com menor custo será automaticamente selecionado e usado a longo prazo. Por isto, a matriz de roteamento é formada por constantes (zeros e uns) e invariável no tempo[2].

2.3 Outros Métodos de Estimação da Matriz de Tráfego

Nos últimos anos, alguns métodos novos foram propostos. Estes métodos são categorizados por Soule et al. [47] como métodos de terceira geração. O caráter comum dos métodos é que os monitores de fluxos são usados para calibrar o seu próprio modelo. O método fanout [42] [22] é introduzido na seção 2.3.1. Nas seções

[2] A matriz de roteamento pode sofrer modificações caso o administrador do sistema mude o custo de algum dos canais, ou caso um canal falhe. No presente trabalho, assumimos que estes eventos são raros e desprezíveis.

2.3.2 e 2.3.3 apresentamos dois trabalhos de Soule et al. [48, 49] para estimação da matriz de tráfego. O primeiro procura modelar possíveis correlações temporais na matriz de roteamento **A**. O método baseado no filtro de Kalman é introduzido na seção 2.3.3. Na seção 2.3.4, o método baseado na análise dos componentes principais é introduzido. Finalmente, o modelo de conexão independente é introduzido na seção 2.3.5.

2.3.1 Método fanout

O método fanout é um método puramente dirigido por dados que usam medições somente para obter a matriz de tráfego. Nenhuma matriz de roteamento é usada e nenhuma inferência é executada. O fanout de um nó i é definido como o vetor que captura a fração do tráfego entrando neste nó i fluindo a nó j da rede. Seja $f(i,j,t) = \frac{x(i,j,t)}{\Sigma_j x(i,j,t)}$ a fração do tráfego entrando no i e saindo da rede no j no tempo t. O fanout de um nó é dado pelo vetor $\mathbf{f}(i,*,t) = \{f(i,j,t) \quad \forall j\}$

Em [42] e [22], os autores encontraram, independentemente, que os fanouts dos nós exibem fortes padrões diários, e são notavelmente previsíveis no ciclo de um dia (Figura 2.5,[42]). Esta estabilidade de fanout significa que o fanout do nó i, em um horário de um dia, pode ser usado para predizer a linha i da matriz de tráfego no mesmo horário do dia seguinte. Esta técnica de medição direta explora o encontro de fanouts previsíveis.

2.3.2 Um Método que Leva em Conta Possíveis Mudanças de Rotas

Soule et al. [48] propõem uma abordagem dinâmica para estimar a matriz de tráfego. Os autores assumem que ao longo do tempo as rotas entre os pares origem-destino irão mudar, e sugerem que as correlações temporais da matriz de roteamento **A**, bem como as correlações temporais no tráfego **y** passando nos canais podem ser usadas para resolver o problema. Monitora-se a evolução da matriz **A** e do tráfego

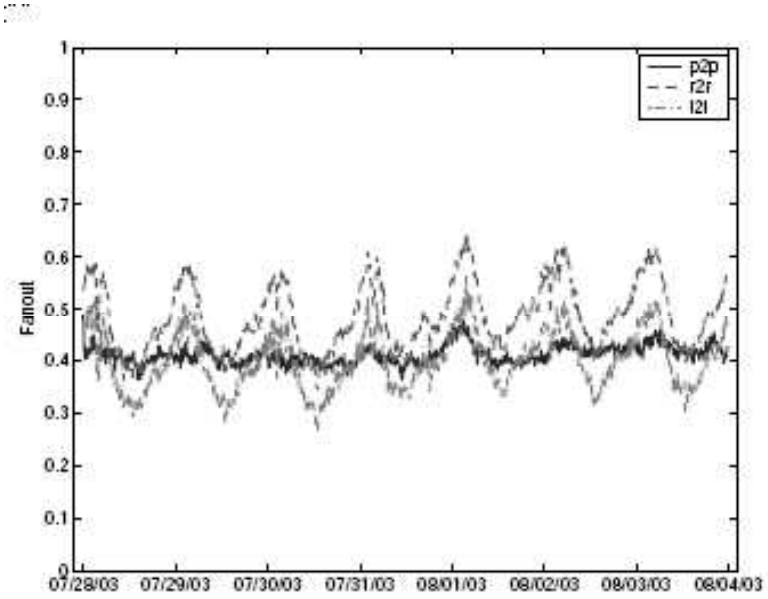


Figura 2.5: Fanout nos três níveis de granularidade

$\mathbf{y}$ nos canais, que passam a ser caracterizados respectivamente por $\mathbf{A}(t)$ e $\mathbf{y}(t)$. Discretizando o tempo em *slots* de tamanho 1, ao invés de resolver um sistema mal condicionado do tipo $\mathbf{Ax} = \mathbf{y}$ os autores solucionam um problema que possui a seguinte estrutura:

$$\begin{pmatrix} \mathbf{A}(0) \\ \mathbf{A}(1) \\ \cdots \\ \mathbf{A}(K-1) \end{pmatrix} \mathbf{x} + \mathbf{QW} = \begin{pmatrix} \mathbf{y}(0) \\ \mathbf{y}(1) \\ \cdots \\ \mathbf{y}(K-1) \end{pmatrix} \tag{2.15}$$

onde $\mathbf{QW}$ é um vetor associado ao ruído das medições. O sistema (2.15) é, na sua forma de matriz,

$$\dot{\mathbf{Y}} = \dot{\mathbf{A}}\mathbf{x} + \mathbf{QW} \tag{2.16}$$

O objetivo é estimar a matriz de tráfego $\mathbf{x}$, notado como $\hat{\mathbf{x}}$. Uma maneira possível para obter $\hat{\mathbf{x}}$ é minimizar a norma Euclideana de $\dot{\mathbf{Y}} - \dot{\mathbf{A}}\mathbf{x}$,

$$\hat{\mathbf{x}} = \arg\min_{z}\{(\dot{\mathbf{Y}} - \dot{\mathbf{A}}\mathbf{x})^T(\dot{\mathbf{Y}} - \dot{\mathbf{A}}\mathbf{x})\} \tag{2.17}$$

$$= (\dot{\mathbf{A}}^T\dot{\mathbf{A}})^{-1}\dot{\mathbf{A}}^T\dot{\mathbf{Y}} \tag{2.18}$$

Supondo que $\dot{\mathbf{A}}^T\dot{\mathbf{A}}$ é uma matriz não singular.

Este é o estimador Pseudo-Inverso, e não requer conhecimento das estatísticas dos ruídos dos tráfegos medidos $\mathbf{W}$. Os autores mostram que este novo sistema (2.16) é determinado (possui solução única), caso as rotas variem significativamente ao longo do tempo (significa que $\dot{\mathbf{A}}^T\dot{\mathbf{A}}$ é não singular) .

2.3.3 Um Método baseado no Filtro de Kalman

Soule et al. [47, 49], inspirados em modelos advindos da literatura de engenharia de transporte [13], assumem que a matriz de tráfego $\mathbf{x}_t$ no tempo t respeita a seguinte relação:

$$\mathbf{x}_{t+1} = \mathbf{B}_t\mathbf{x}_t + \mathbf{w}_t \tag{2.19}$$

onde $\mathbf{w}_t$ captura fatores variáveis ou aleatórios.

O sistema a ser resolvido é:

$$\mathbf{y}_t = \mathbf{A}_t\mathbf{x}_t + \mathbf{m}_t \tag{2.20}$$

onde $\mathbf{m}_t$ corresponde a erros de medição.

Seja $\hat{\mathbf{x}}_{t|t-1}$ a predição de $\mathbf{x}_t$ no tempo t baseada em toda a informação disponível até o tempo t-1. A tarefa consiste em determinar $\hat{\mathbf{x}}_{t+1|t+1}$ dado um conjunto de observações $\mathbf{y}_1, \ldots, \mathbf{y}_{t+1}$. O erro da estimação no tempo t é dado por $\hat{\mathbf{x}}_{t|t} - \mathbf{x}_t$. Os autores almejam um estimador da matriz de tráfego que seja ótimo no sentido de que ele minimiza a variância do erro. A matriz de variância de erro no tempo t é dada por:

$$E[||\mathbf{x}_{t+1} - \hat{\mathbf{x}}_{t+1}||^2] = E[(\mathbf{x}_{t+1} - \hat{\mathbf{x}}_{t+1})^T(\mathbf{x}_{t+1} - \hat{\mathbf{x}}_{t+1})] \tag{2.21}$$

Para resolver este problema os autores sugerem o uso de um algoritmo tempo-discreto, sequencial e recursivo, conhecido como *filtro de Kalman.*

O filtro de Kalman estima um processo através de uma forma do controle do gabarito usando dois tipos de equações.

Passo de predição: Este passo prediz o estado e a variânça no tempo $t + 1$

dependente na informação no tempo t.

$$\hat{\mathbf{x}}_{t+1|t} = \mathbf{B}_t \hat{\mathbf{x}}_{t|t} \tag{2.22}$$

$$\mathbf{P}_{t+1|t} = \mathbf{B}\mathbf{P}_{t|t}\mathbf{B}^T + \mathbf{Q}_t \tag{2.23}$$

Onde $\mathbf{P}_{t|t} = E[(\mathbf{x}_t - \hat{\mathbf{x}}_{t|t})(\mathbf{x}_t - \hat{\mathbf{x}}_{t|t})^T]$ é a matriz de covariância de erros no tempo t e $\mathbf{Q}_t$ é a matriz de covariância de ruido no tempo t.

Passo de estimação: Este passo renova o estado e a variância usando a combinação do estado predito e da observação Y_{k+1}.

$$\hat{\mathbf{x}}_{t+1|t+1} = \hat{\mathbf{x}}_{t+1|t} + \mathbf{G}_{t+1}[\mathbf{y}_{t+1} - \mathbf{A}\hat{\mathbf{x}}_{t+1|t}] \tag{2.24}$$

$$\mathbf{P}_{t+1|t+1} = (\mathbf{ID} - \mathbf{G}_{t+1}\mathbf{A})\mathbf{P}_{t+1|t}(\mathbf{ID} - \mathbf{G}_{t+1}\mathbf{A})^T + \mathbf{G}_{t+1}\mathbf{R}_{t+1}\mathbf{G}_{t+1}^T \tag{2.25}$$

Onde **ID** é a matriz de identidade, $\mathbf{G}_{t+1}$ é chamado matriz de ganho de Kalman(*Kalman gain matrix*) e $\mathbf{R}_{t+1}$ é a matriz de covariância de ruido no tempo $t+1$.

As equações acima (2.22)-(2.24), junto com as condições iniciais, define o algoritmo tempo-discreto, sequencial e recursivo para determinar a estimativa de variância mínima e linear conhecido como o filtro de Kalman.

2.3.4 Um Método baseado na Análise dos Componentes Principais

O método baseado na análise dos componentes principais (PCA) que corresponde à decomposição de uma matriz em valores singulares ataca o problema de estimação da matriz de tráfego estudando a dimensionalidade intrínseca do conjunto de fluxos OD. O PCA reduz o número de variáveis espectrais a poucas variáveis ortogonais denominadas pelas componentes principais, que melhor se correlacionam com os valores medidos das propriedades.

Os autores de [32] encontram que o conjunto inteiro de N fluxos OD, quando examinado em escalas de tempo longas, podem exatamente ser capturadas por re-

presentações de dimensão baixa. Em conseqüência, os fluxos OD podem ser representados tipicamente pela a combinação de 5 até 10 esquemas temporais, que são chamadas como *eigenflows*.

Então, pode-se aproveitar esta vantagem para desenvolver um método para a estimação da matriz de tráfego. A idéia chave é que ao invés de estimar todos os fluxos de N pares OD, somente se necessita estimar K *eigenflows* mais importantes. Por causa de $K \ll N$, o problema de estimar os *eigenflows* se torna bem-condicionado (*well-posed*).

Formalmente, seja $\mathbf{X}$ as séries temporais T por N de fluxos OD. Então, pode-se usar PCA para escrever $\mathbf{X}$ como:

$$\mathbf{X} = \mathbf{USV}^T \tag{2.26}$$

onde $\mathbf{U}$ é uma matriz de séries temporais de *eigenflows* T por N e $\mathbf{V}$ é uma matriz T por N com componentes principais nas suas colunas. Finalmente, $\mathbf{S}$ é uma matriz diagonal N por N de valores singulares. $S(i,i)$ é uma medida de energia capturada pelo componente principal i. Pode-se reduzir a dimensionalidade selecionando somente K componentes principais. Ou seja, no tempo t,

$$\mathbf{x}_t \approx \mathbf{V}'\mathbf{S}'\mathbf{u}'_t \quad t = 1, \ldots, T \tag{2.27}$$

onde $\mathbf{V}'$ é uma matriz N por K com somente K componentes principais, $\mathbf{S}'$ é uma matriz diagonal correspondente K por K. O vetor $\mathbf{u}'_t$ tem os valores de K *eigenflows* mais significativos no tempo t.

Em [32], é assumido que as matrizes $\mathbf{V}'$ e $\mathbf{S}'$ já são conhecidas (e.g. de medições a priori). Então, o problema se torna estimar os K *eigenflows*, $\mathbf{u}'_t$, do conjunto de medições de canais,

$$\mathbf{y}_t = \mathbf{A}\mathbf{V}'\mathbf{S}'\mathbf{u}'_t \quad t = 1, \ldots, T \tag{2.28}$$

Como a dimensão de $\mathbf{u}'_t$ é muito menor do que $\mathbf{x}$, a equação (2.28) é bem condicionada. Para resolver $\mathbf{u}'_t$, o método pseudo-inverso é usado e uma estimativa $\hat{\mathbf{u}}_t{}'$ é obtida. Então, voltando na equação (2.27), uma estimativa da matriz de tráfego $\hat{\mathbf{x}}_t$ é computada.

2.3.5 Modelo de Conexão Independente

Ao invés de usar o modelo de gravidade para obter a matriz de tráfego a priori, Erramilli et al. [15] propõem um modelo de conexão independente.

A suposição chave do modelo de gravidade é que o tráfego entrando na rede em um nó i e saindo da rede por um nó j é proporcional à quantidade total do tráfego saindo da rede por este nó j. Isso pode ser considerado que, para qualquer pacote os seus pontos de egresso e ingresso são independentes. Porém na Internet, esta suposição de independência é realista. A razão é que a maioria do tráfego da rede consiste em conexões nos dois sentidos. Por exemplo, no protocolo TCP (Transmission Control Protocol), é assumido que as conexões são nos dois sentidos.

Cada conexão tem um iniciador (um *host* que peça conexão) e um respondedor (um *host* que aceita a conexão pedida). Ao invés de assumir que os pontos de egresso e ingresso são independentes, Erramilli et al. supõem que o iniciador e o respondedor de conexões são independentes.

Seja A_i o tráfego total gerado pelas conexões iniciadas no nó i, f_{ij} a porção do tráfego direto no tráfego total, ou seja, o tráfego direto dividido pelo tráfego direto mais o tráfego reverso, P_i a possibilidade que o respondedor escolha o no i e X_{ij} a quantidade do tráfego entre os nós i e j. Então o modelo de conexão independente é,

$$X_{ij} = \frac{f_{ij} A_i P_j}{\sum_{i=1}^{N} P_i} + \frac{(1 - f_{ji}) A_i P_j}{\sum_{i=1}^{N} P_i} \quad i, j = 1, \ldots, N \tag{2.29}$$

Para comparar o modelo de conexão independente com o modelo de gravidade, os autores usaram os dados da rede GÉANT[3] e verificam que o modelo pode melhor aproximar a matriz de tráfego real, do que o modelo de gravidade. Eles também mostram que a estimação da matriz de tráfego com a matriz de tráfego a priori

[3] A Géant (Rede Gigabit de pesquisa pan-européia) é uma rede de alta capacidade que engloba mais de três mil instituições de ensino e pesquisa em 32 países, através de 28 redes nacionais e regionais de ensino e pesquisa. Atuando desde dezembro de 2001, seu objetivo principal é dar continuidade e melhorar a versão anterior da rede de pesquisa pan-européia.

obtida do modelo de conexão independente tem resultados melhores do que com a matriz de tráfego a priori obtida do modelo de gravidade.

2.3.6 Método de Zhao

Em [60], os autores usam múltiplas fontes de dados para estimar a matriz de tráfego. Estes dados são o tráfego medido nos canais pelo protocolo SNMP (Simple Network Management Protocol) e o tráfego medido de pares OD pelo protocolo NetFlow [12]. Eles mostram que com ambas as fontes de dados (SNMP e NetFlow), a estimação da matriz de tráfego é muito mais precisa do que aqueles que somente usam uma fonte de dados (SNMP ou NetFlow), mesmo que os dados de NetFlow estejam disponíveis em apenas um subconjunto de pares OD.

O problema formulado por eles é

$$\hat{X} = X + \epsilon^X \tag{2.30}$$

$$\hat{Y} = AX + \epsilon^Y \tag{2.31}$$

onde ϵ^X e ϵ^Y são as variáveis aleatórias gaussianas com média zero, referindo-se aos erros na medição NetFlow e SNMP, respectivamente.

Então, o sistema pode ser escrito como

$$Y = HX + N \tag{2.32}$$

onde $H = (I : A)$ é uma matriz $m + n$ por n que concatena verticalmente uma matriz identidade I e a matriz de roteamento A; Y é o vetor concatenado por $\hat{X}$ e $\hat{Y}$, e N é o vetor concatenado de ϵ^X e ϵ^Y. Então, o objetivo atual é obter uma boa estimativa de X do Y observado.

O sistema se encaixa bem no esquema do teorema Gauss-Markov [26]. O teorema diz que, em um modelo linear onde os erros são não-correlatos e têm média zero, o melhor estimador linear sem vício (Best linear unbiased estimator, BLUE) do coeficiente são os estimadores dos mínimos quadrados.

Então, a estimativa de X em (2.32), é achada pela solução de pseudo-inversa

$$X = (H^T K^{-1} H)^{-1} H^T K^{-1} Y \tag{2.33}$$

onde K é a matriz de covariância de N.

Finalmente, o ajuste proporcional iterativo (Iterative Proportional Fitting, IPF) também é usado em [60] para garantir a não-negatividade de X.

O método mostra a vantagem de usar os dados de NetFlow como a matriz de tráfego a priori na estimação da matriz de tráfego. Claramente, os dados de NetFlow são muito mais precisos do que o modelo de gravidade. Porém a desvantagem é que NetFlow nem sempre está disponível na rede total, ou seja, o custo de usar NetFlow na rede total é muito caro. Um plano efetivo é que a matriz de tráfego a priori consiste nos dados de NetFlow parciais e no modelo de gravidade. Isso significa que, para alguns elementos da matriz de tráfego a priori, usa-se os dados de NetFlow que estão disponíveis na rede e para os elementos restantes da matriz de tráfego a priori usa-se o modelo de gravidade.

2.4 Resumo

Medina et al. [37] avaliam os métodos de primeiro classe, ou seja, métodos de inferência estatística. Eles mostram que as suposições dos modelos estatísticos (Poisson ou Gaussiano) não são justificadas. Os métodos são limitados a tipos específicos de tráfego. Então, os métodos de inferência estatística podem ser mal executados quando as suas suposições são violadas. Os autores também concluíram que os métodos de inferência estatística são muito sensíveis a erros da matriz de tráfego a priori e nenhum método de inferência estatística foi satisfatório em uma grande rede.

Os métodos de terceira classe usam medição de fluxo de 24 horas para calibrar o modelo. Em particular, o método de mudanças de rotas foi consequência da observação de forte padrão diário de tráfego. O método baseado na análise dos componentes principais é construído sobre a observação de que os fluxos têm baixa

dimensionalidade intrínseca. O método fanout é baseado na estabilidade da fanouts.

Sabendo que entre os fluxos OD existem as duas correlações: correlação espacial e temporal, e que o sistema é uma sistema linear ($\mathbf{y} = \mathbf{Ax}$), o método baseado no filtro de Kalman constroi um sistema linear e dinâmico que pode capturar os dois tipos de correlações em uma única equação.

Comparado com os método de terceira classe, o método de tomogravidade tem as seguinte vantagens,

- adaptatividade rápida. É claro que cada método define a sua taxa de adaptatividade. Mas normalmente, o método de tomogravidade pode renovar o seu modelo cada 10 minutos porque os dados que ele usa estão disponíveis em um nível de granularidade. A obtenção dos dados utilizados pelos outros métodos é mais custosa. Normalmente eles esperam 24 horas antes de adaptar os dados.

- pode estimar a matriz de tráfego de tipos diferentes. Se o tráfego medido nos canais é uma fotografia de tráfego nos canais, pode estimar a matriz de tráfego de fotografia. Se o tráfego medido nos canais é uma série temporal, pode estimar a matriz de tráfego de tipo série temporal também nesses casos.

- computação rápida. Segundo Zhang et al. [58], o método pode ser usado para uma rede grande (mais de 1000 roteadores) e ele é muito rápido. Por exemplo, para uma rede grande, leva somente 5 segundos para obter o resultado .

Além disso, o método de tomogravidade é um método muito preciso, especialmente para os elementos maiores na matriz de tráfego[58]. Então, neste trabalho, para estimar a matriz de tráfego no roteamento pelo caminho mais curto, propomos o nosso método baseado no método de tomogravidade.

Capítulo 3

Fundamentos Teóricos

Neste capítulo estabelecemos os fundamentos teóricos que dão sustentação aos modelos propostos nesta tese. Primeiramente, na seção 3.1, apresentamos o roteamento ótimo por múltiplas rotas , que se diferencia em vários aspectos do roteamento pelo caminho mais curto tradicional (e.g. OSPF), abordado no capítulo 1. Na seção 3.2, introduzimos a programação em dois níveis, elemento unificador dos modelos apresentados no Capítulo 5. Também apresentamos uma rápida discussão dos métodos numéricos existentes para resolver este problema. Uma breve recordação do método dos mínimos quadrados e do método de ajuste iterativo proporcional encontra-se na Seção 3.3 e 3.4.

3.1 Roteamento Ótimo por Múltiplas Rotas

O principal objetivo do roteamento ótimo por múltiplas rotas consiste em distribuir o tráfego na rede de tal modo que uma métrica de custo total seja minimizada. Parte-se do princípio de que existe uma autoridade central, o administrador da rede, ou um protocolo pré-acordado que detém o controle sobre os recursos disponíveis, e que este pode balancear o tráfego gerado pelos usuários da forma que lhe convenha [28].

No exemplo a seguir (Figura 3.1), ilustramos o emprego do *roteamento ótimo*

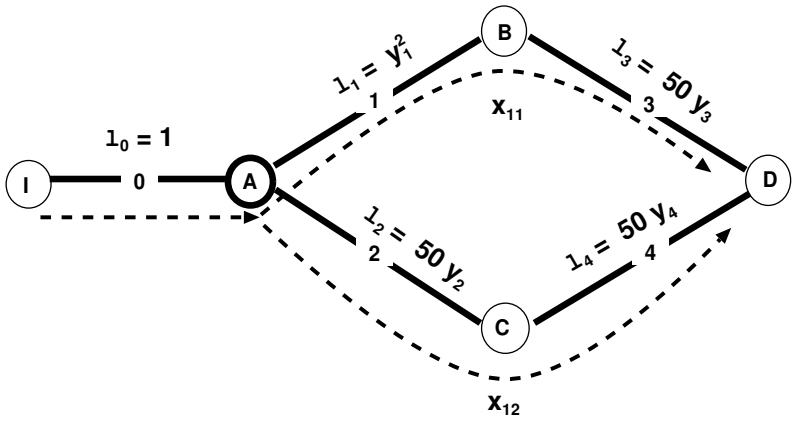


Figura 3.1: Exemplo de roteamento ótimo por múltiplas rotas

por múltiplas rotas em uma rede envolvendo 5 nós $\{I, A, B, C, D\}$, 5 canais $\{L_0, L_1, L_2, L_3, L_4\}$, e um fluxo de dados de I para D. Observe que cada canal está etiquetado com uma função de custo distinta, que depende da quantidade de tráfego que passa pelo mesmo. Por exemplo, quanto maior o fluxo de dados no canal, maior o atraso (*delay*) experimentado pelos pacotes que lá trafegam. O custo total experimentado pelos usuários é, por exemplo, o somatório dos custos de todos os canais.

O nó A é um *ponto estratégico* – quando os pacotes lá chegam, eles são direcionados para B ou para C. É responsabilidade do algoritmo de *roteamento ótimo por múltiplas rotas* determinar a quantidade de tráfego que será direcionada para cada ponto, de tal forma que o custo total seja minimizado. No exemplo apresentado, seja x_{11} o tráfego que passa pela rota 1 (IABD), x_{12} o tráfego que passa pela rota 2 (IACD), e $x_1 = x_{11} + x_{12}$ o tráfego total entre o par origem destino I-D. Além disso, y_i é o tráfego no canal i e l_i é o custo associado no canal L_i, para cada i=0,1,2,3,4. Note que $y_1 = y_3 = x_{11}$ e $y_2 = y_4 = x_{12}$. Então temos o seguinte problema:

$$\begin{aligned} \min_{y,x_{11},x_{12}} \quad & 1 + {y_1}^2 + 50y_3 + 50y_2 + 50y_4 \\ \text{sujeito a} \quad & y_0 = x_{11} + x_{12} \\ & y_1 = x_{11} \\ & y_2 = x_{12} \\ & y_3 = x_{11} \\ & y_4 = x_{12} \\ & x_{11} + x_{12} = x_1 \\ & x_{11} \geq 0 \\ & x_{12} \geq 0 \end{aligned} \tag{3.1}$$

Supondo que o tráfego total a ser transmitido de I para D é de $x_1 = 100$ kbps, e resolvendo o problema de otimização acima, obtemos a seguinte estratégia ótima de roteamento: $x_{11} = 25$ kbps, e $x_{12} = 75$ kbps.

O roteamento ótimo por múltiplas rotas é geralmente empregado em cenários envolvendo:

- *múltiplas rotas* entre cada par origem destino;
- *pontos estratégicos*, onde um fluxo de dados principal é dividido em múltiplos fluxos;
- funções de custo que em geral *dependem da quantidade de fluxo* passando pelo canal.

No caso de roteamento ótimo por múltiplas rotas, um par origem-destino usa múltiplas rotas. Então a matriz de roteamento é uma matriz cujo elemento é fração de tráfego de uma rota que flui por cada canal. Neste caso, a matriz de roteamento não é uma matriz que possui apenas elementos 0 e/ou 1. Ela possui elementos que variam entre 0 e 1.

Retornando ao exemplo, aplicando a estratégia de roteamento ótima, temos a

seguinte matriz de roteamento:

$$\mathbf{A} = 1/(x_{11}+x_{12})\,(\quad y_0 \quad y_1 \quad y_2 \quad y_3 \quad y_4 \quad)^T = (\quad 1 \quad 1/4 \quad 3/4 \quad 1/4 \quad 3/4 \quad)^T$$

Como só temos um par origem-destino, a matriz de roteamento possui apenas 1 linha e 5 colunas. Note que, neste caso, a matriz de roteamento possui elementos que variam entre 0 e 1. Isto reflete o fato de que 1/4 dos pacotes transmitidos de I para D seguem a rota IABD e o restante segue a rota IACD. A matriz de tráfego neste caso é composta por um único elemento, igual a 100kbps, que corresponde ao tráfego entre a origem I e o destino D do único par OD (Figura 3.1).

Formulação do problema

Formalmente, no caso geral o problema do roteamento ótimo por múltiplas rotas é definido da seguinte forma [5, 28]:

$$\begin{aligned} \min_{\mathbf{y},x_{ij}} \quad & \textstyle\sum_{a\in\mathbf{M}} l_a(y_a) \\ \text{sujeito a} \quad & \textstyle\sum_j x_{ij} = x_i, i \in \mathbf{N} \\ & \textstyle\sum_{ij} x_{ij}\delta_a^{ij} = y_a \\ & x_{ij} \geq 0 \end{aligned} \tag{3.2}$$

Recordando a notação, $\mathbf{R}$ é o conjunto de R nós e $\mathbf{M}$ o conjunto de M canais. O grafo direcionado $G = (\mathbf{R}, \mathbf{M})$ caracteriza a rede sendo estudada. Seja y_a o tráfego que passa pelo canal a. Uma função de custo l_a associa a cada canal $a \in \mathbf{M}$ um custo, que pode depender de y_a. Denotamos por x_i a demanda entre os nós $(o, d) \in \mathbf{N}$, ou seja, o tráfego que é gerado pela origem o e destinado a d.

Finalmente, sabemos que podem existir múltiplas rotas associadas a um dado par OD. Denotamos por x_{ij} o tráfego associado ao par OD i que passa pela rota j e Γ_i o conjunto de todas as rotas associado ao par OD i.

A função indicadora δ_a^{ij} é definida da seguinte forma:

$$\delta_a^{ij} = \begin{cases} 1, \text{se a rota } j \text{ do par OD } i \text{ passa pelo canal } a \\ 0, \text{caso contrário} \end{cases}$$

Retornando à formulação geral do problema de roteamento ótimo por múltiplas rotas (3.2), identificamos que a função objetivo corresponde à minimização da métrica de custo total sendo utilizada. A primeira restrição caracteriza o tráfego do par origem-destino OD $i \in \mathbf{N}$ como sendo o somatório dos tráfegos que passam em cada uma das rotas associadas a i. A última restrição relaciona o tráfego total que passa pelo canal $a \in \mathbf{M}$ com o tráfego que passa em cada uma das rotas envolvendo a.

A função de custo

Seja C_a a capacidade do canal $a \in \mathbf{M}$. Uma função de custo muito popular é derivada do modelo M/M/1, e relaciona o custo de um canal com o atraso (*delay*) experimentado pelos pacotes que nele trafegam [27, 28]:

$$l_a(y_a) = \frac{y_a}{C_a - y_a} \tag{3.3}$$

Retornando ao exemplo apresentado na Figura 3.1, vamos manter todos os parâmetros fixos, com exceção da função de custo, que passará a ser definida conforme 3.3. Então, a função objetivo do problema de otimização 3.1 pode ser reescrita como:

$$\min_y \frac{y_0}{C_0 - y_0} + \frac{y_1}{C_1 - y_1} + \frac{y_2}{C_2 - y_2} + \frac{y_3}{C_3 - y_3} + \frac{y_4}{C_4 - y_4}$$

Algoritmos

Existem inúmeros algoritmos para resolver o problema de otimização 3.2. Os mais populares são o método de Frank-Wolfe,[5] [18] (também chamado de método de desvios de fluxos – *flow deviation method*, desenvolvido para o problema por Kleinrock e Gerla [28]) e o método da projeção [4], ambos baseados na estratégia de *direção viável* (*feasible direction*) [5, seção 5.7].

3.1.1 Roteamento: Múltiplas Rotas versus Caminho mais curto

Vamos a seguir resumir as principais diferenças entre o cenário envolvendo um algoritmo de roteamento pelo caminho mais curto, como o OSPF, e aquele envolvendo um algoritmo de roteamento ótimo por múltiplas rotas :

- ao adotarmos um mecanismo de roteamento pelo caminho mais curto tradicional, implicitamente caracterizamos uma matriz de tráfego cujos elementos são zeros e uns. Quando adotamos um mecanismo de roteamento ótimo por múltiplas rotas, pelo fato de serem usadas concomitantemente múltiplas rotas entre cada par origem-destino, a matriz de roteamento pode conter elementos fracionários, conforme exemplificamos no início deste capítulo;

- muitos dos mecanismos de roteamento pelo caminho mais curto, associam um custo fixo a cada canal. Este custo pode ser ajustado (automaticamente ou manualmente) pelo operador de rede. A qualidade do OSPF depende altamente da escolha dos pesos dos canais. Mesmo assim, a Cisco recomenda que o custo de um canal seja configurado simplesmente como o inverso de sua capacidade, sem levar em consideração a demanda pelo mesmo [17]. Ao considerarmos os mecanismos de roteamento ótimo por múltiplas rotas, por outro lado, assumimos que o custo associado a cada canal é uma função do tráfego que nele passa. O custo no canal a é caracterizado pela função $l_a(y_a)$;

- a matriz de roteamento clássica possui entradas fixas. Já os elementos da matriz de roteamento associada a um algoritmo de roteamento ótimo por múltiplas rotas são funções, que podem depender do tráfego que passa nos canais. Como o tráfego nos canais $\mathbf{y}$ depende da matriz de tráfego $\mathbf{x}$, $\mathbf{A}$ trata-se de uma matriz do tipo $\mathbf{A}(\mathbf{x})$.

Os métodos convencionais de estimação da matriz de tráfego partem do pressuposto de que a matriz de roteamento é independente da matriz de tráfego. Entretanto, ao considerarmos o roteamento ótimo por múltiplas rotas esta suposição

Tabela 3.1: Diferenças entre o roteamento clássico e o roteamento ótimo

roteamento	elementos da matriz de roteamento	custos dos canais	matriz de roteamento	tráfego nos canais
pelo caminho mais curto	0, 1	números constantes (ajustáveis)	$\mathbf{A}$	$\mathbf{y}$
ótimo por múltiplas rotas	[0, 1]	$l(\mathbf{y})$	$\mathbf{A}(\mathbf{x})$	$\mathbf{y}(\mathbf{x})$

torna-se inadequada.

Conforme mencionado acima (Tabela 3.1), ao adotarmos o roteamento ótimo por múltiplas rotas, a matriz de roteamento $\mathbf{A}$ passa a depender da matriz de tráfego $\mathbf{x}$. Ao invés de uma relação linear $\mathbf{y} = \mathbf{A}\mathbf{x}$, temos uma relação não linear entre $\mathbf{x}$ e $\mathbf{y}$. Para estimar o valor de $\mathbf{x}$ precisamos combinar a estimação da matriz de tráfego com o roteamento ótimo, de tal forma que ambos constituam partes de um todo. A *programação em dois níveis* presta-se a este fim unificador, e emerge como uma metodologia natural para modelar este sistema.

3.2 Programação em Dois Níveis

3.2.1 Introdução

Considere o seguinte jogo, envolvendo a Microsoft e a Netscape [19, exemplo 5.2], [38]. A Microsoft e a Netscape planejam introduzir um novo tipo de navegador (*browser*) para a *web*. Elas precisam escolher uma dentre duas plataformas, Java ou ActiveX. Se elas introduzirem plataformas distintas (incompatíveis), seus prejuízos serão de 2 (ou, equivalentemente, seus ganhos serão negativos, e iguais a -2). Se introduzirem a mesma plataforma, seus prejuízos serão iguais a 0, com um prejuízo extra de 1 para a Microsoft se a plataforma for Java, e 1 extra para a Netscape se a plataforma for ActiveX. É importante observar que neste jogo a Microsoft tem o

poder de *decidir primeiro* a plataforma que irá adotar, e a Netscape poderá tomar sua decisão baseada na observação que fizer sobre a ação adotada pela Microsoft. Neste caso, é conveniente descrever os prejuízos obtidos pelos jogadores por meio de uma árvore (Figura 3.2). Cada folha da árvore está etiquetada com um par ordenado (X, Y), onde X e Y representam os prejuízos da Microsoft e da Netscape, respectivamente.

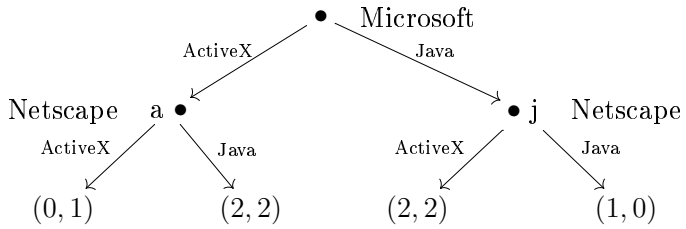


Figura 3.2: Jogo Microsoft-Netscape

A Microsoft pode tomar sua decisão final resolvendo um problema de otimização em dois níveis: primeiramente, ela estima qual será a *reação* da Netscape, ao Java e ao ActiveX. Em seguida, a Microsoft, sempre visando a maximizar seus ganhos, toma sua decisão levando em conta a previsível resposta da Netscape, e escolhe o ActiveX.

Formulação do problema

Antes de colocar a formulação geral, retornamos ao exemplo da Microsoft *versus* Netscape, temos duas estratégias disponíveis para cada empresa, logo $\mathbf{x} \in \{$ ActiveX, Java $\}$ e $\mathbf{y} \in \{$ ActiveX, Java $\}$. A função F($\mathbf{x}$, $\mathbf{y}$) caracteriza as perdas da Microsoft, que podem ser descrita pela seguinte tabela:

$\mathbf{x}$ \ $\mathbf{y}$	ActiveX	Java
ActiveX	0	2
Java	2	1

Tabela 3.2: Perdas da Microsoft (F($\mathbf{x}, \mathbf{y}$))

Denominando $x_1 = x = ActiveX$, $x_2 = x = Java$, $y_1 = y = Java$, tem-se que $F(x_1, y_1) = 0$, $F(x_1, y_2) = F(x_2, y_1) = 2$, $F(x_2, y_2) = 1$.

A função f($\mathbf{x}$, $\mathbf{y}$) caracteriza as perdas da Netscape, que podem ser descritas pela seguinte tabela:

x \ **y**	ActiveX	Java
ActiveX	1	2
Java	2	0

Tabela 3.3: Perdas da Netscape (f($\mathbf{x}, \mathbf{y}$))

Isto é, $f(x_1, y_1) = 1$, $f(x_1, y_2) = f(x_2, y_1) = 2$, $f(x_2, y_2) = 0$.

Ao resolvermos o problema de otimização da Netscape (nível inferior) obtemos a seguinte função de $\mathbf{x}$, que será utilizada pela Microsoft para prever a ação da Netscape:

$\mathbf{x}$	Perda para Netscape ($\min_{\mathbf{y}} f(\mathbf{x}, \mathbf{y})$)
ActiveX	1
Java	0

Tabela 3.4: Perdas da Netscape (nível inferior)

Tendo em mãos os resultados descritos na Tabela 3.4, a Microsoft pode resolver o seu problema de minimização (nível superior):

$\mathbf{x}$	Perda da Microsoft ($\min_{\mathbf{x}} F(\mathbf{x}, \mathbf{y})$)
ActiveX	0
Java	1

Tabela 3.5: Perdas da Microsoft (nível superior)

De acordo com os resultados apresentados na Tabela 3.5, concluímos que a Microsoft selecionará a estratégia ActiveX, pois desse maneira obriga a Netscape selecionar também a ActiveX, por minimizar o seu prejuízo.

Matematicamente, a formulação do problema é dado por

$$\begin{aligned} \min_{x} \quad & F(x,y) \\ \text{sujeito a} \quad & y \text{ é solução de} \\ & \min_y f(x,y) \end{aligned} \tag{3.4}$$

Mais formalmente, o problema da programação em dois níveis é definido da seguinte forma:

Sejam $\mathbf{x} \in \Re^{n_1}$, $\mathbf{y} \in \Re^{n_2}$. O líder resolve o problema

$$\begin{aligned} \min_{\mathbf{x},\mathbf{y}} \quad & F(\mathbf{x},\mathbf{y}) \\ \text{sujeita a} \quad & G(\mathbf{x},\mathbf{y}) \leq 0 \end{aligned} \tag{3.5}$$

onde $\mathbf{y}$, por sua vez, é solução de

$$\begin{aligned} \min_{\mathbf{y}} \quad & f(\mathbf{x},\mathbf{y}) \\ \text{sujeita a} \quad & g(\mathbf{x},\mathbf{y}) \leq 0 \end{aligned}$$

As variáveis do problema 3.5 são divididas em duas classes: as variáveis do nível superior, $\mathbf{x}$, e as variáveis do nível inferior, $\mathbf{y}$. As funções $F : \Re^{n_1} \times \Re^{n_2} \rightarrow \Re$ e $f : \Re^{n_1} \times \Re^{n_2} \rightarrow \Re$ são as funções objetivo do líder (nível superior) e do seguidor (nível inferior), respectivamente. Finalmente, $G : \Re^{n_1} \times \Re^{n_2} \rightarrow \Re^{m_1}$ e $g : \Re^{n_1} \times \Re^{n_2} \rightarrow \Re^{m_2}$ são as restrições do líder (nível superior) e do seguidor (nível inferior), respectivamente.

Do ponto de vista histórico, a programação em dois níveis pode ser vista como um jogo de versão estática e não cooperativo proposto por Stackelberg [50] no contexto de desequilíbrio dos mercados econômicos. O modelo de Stackelberg envolve duas empresas, uma líder e uma seguidora. O líder é aquele que toma as decisões em primeiro lugar e o seguidor, conhecendo a decisão do primeiro, reage de acordo com a sua *função de reação*.

No problema de otimização (3.4) temos um líder, que detém a vantagem da primeira escolha, e seleciona sua estratégia $\mathbf{x}$ de tal forma a minimizar a função objetivo $F(\cdot)$. Para isto, ele considera a reação do seguidor, que é caracterizada pela função de reação $f(\cdot)$. Este problema pode não estar bem definido quando a resposta do seguidor não é única (ver por exemplo [3]). É comum substituir a formulação de Stackelberg pelo problema (3.5). Ambas formulações são equivalentes quando a resposta do seguidor é única.

A programação em dois níveis foi inicialmente estudada na década de 70 por Bracken e McGill [7, 8, 9]. Nos anos 80, os problemas começaram a receber maior atenção, quando esforços significativos foram concentrados na compreensão dos conceitos fundamentais associados com a programação em dois níveis e vários algoritmos foram propostos.

3.2.2 Métodos Numéricos

Devido à hierarquia inerente, problemas de programação em dois níveis são intrinsecamente difíceis, com soluções locais. A maioria dos algoritmos concentram-se nos casos mais simples de programação de dois níveis, que têm boas propriedades,

como restrições e/ou função objetivo linear, quadrática ou convexa. Os métodos tradicionais para resolver modelos de programação em dois níveis (equação (3.5)) baseiam-se na reformulação do problema, transformando-o em um com um único nível. Para tal, incorpora-se informações do nível inferior no nível superior. Em particular, quando o nível inferior é convexo, substitui-se o nível inferior pelas suas condições de KKT(Karush-Kuhn-Tucker)[1] [55].

Um algoritmo de ponto interior [23] foi proposto considerando a reformulação KKT do problema de programação em dois níveis .

Ao longo do tempo, vários algoritmos, por exemplo, método de enumeração, métodos de região de confiança e método de penalidade, foram propostos para problemas de programação em dois níveis, com a suposição de que todas as funções são convexas e duas vezes diferenciáveis [39, 41, 1, 45, 2].

Outros métodos para resolver o problema de programação em dois níveis são os de *branch-and-bound*, de descida e algoritmos evolucionários apresentados em [55].

Em [36], os autores ilustram que as soluções da maioria dos métodos tradicionais são dependentes do conhecimento de espaço de pesquisa e não são suficientes para resolver problemas de grande ponte.

Para uma discussão geral sobre a programação em dois níveis, consulte o *survey* de Vicente e Calamai [55] e e o livro de Shimizu [46].

O modelo proposto nesta tese corresponde a um problema de programação em dois níveis convexo. O objetivo do algoritmo que apresentaremos é encontrar o ponto de mínimo global do problema. Um algoritmo genético será proposto para resolver o problema de programação em dois níveis.

[1] As condições Karush-Kuhn-Tucker(também conhecido como as condições Kuhn-Tucker ou as condições KKT) são condições necessárias para uma solução ser ótima na programação não-linear.

3.2.3 Uma Aplicação ao Problema de Roteamento Ótimo por Múltiplas Rotas

Para se fazer a estimação da matriz de tráfego no contexto proposto nesta tese, é necessário levar em consideração o efeito do roteamento ótimo por múltiplas rotas. Podemos então empregar o esquema da programação em dois níveis, da seguinte forma (Figura 1.4(b)):

- no nível superior, temos o problema da estimação da matriz de tráfego $\mathbf{x}$;
- no nível inferior, temos o roteamento ótimo por múltiplas rotas;
- a *função de reação* do roteamento ótimo por múltiplas rotas tem como entrada a matriz de tráfego $\mathbf{x}$, e como saída a matriz de roteamento $\mathbf{A(x)}$ e o tráfego nos canais $\mathbf{y(x)}$.

No próximo capítulo as idéias aqui expostas serão formalizadas, e uma descrição precisa do modelo proposto será apresentada.

3.3 O Método dos Mínimos Quadrados

O método dos mínimos quadrados baseia-se para aproximar um ponto observado $\hat{y}$ em um conjunto C conexo e fechado sobre C na minimização da seguinte função objetivo:

$$S = \sum_{i=1}^{N} (\hat{y}_i - y_i)^2 = (\hat{\mathbf{y}} - \mathbf{y})^T (\hat{\mathbf{y}} - \mathbf{y}) \tag{3.6}$$

que corresponde a calcular a projeção ortogonal de $\hat{y}$ sobre o conjunto C.

Os resíduos são, por definição, dados por $\hat{y}_i - y_i$. Uma generalização do método dos mínimos quadrados consiste em associar um peso w_i para cada resíduo i. Temos então a seguinte função objetivo:

$$S = \sum_{i=1}^{N} w_i (\hat{y}_i - y_i)^2 \tag{3.7}$$

Resta-nos especificar a relação entre a solução de sistemas indeterminados e o método dos mínimos quadrados. Considere o sistema $\mathbf{Ax} = \mathbf{y}$, onde o número de linhas de $\mathbf{A}$ é menor que o número de colunas. Uma maneira de encontrar uma solução é usando mínimo quadrado que consiste em resolver o seguinte problema

$$\min_x \|\mathbf{y} - \mathbf{Ax}\|^2 = \min_x (\mathbf{y} - \mathbf{Ax})^T (\mathbf{y} - \mathbf{Ax}) \tag{3.8}$$

3.4 Ajuste Proporcional Iterativo

O ajuste proporcional iterativo (Iterative Proportional Fitting, IPF) é um procedimento matemático originalmente desenvolvido para combinar as informações de dois ou mais conjuntos de dados. Este procedimento pode ser usado para garantir que os dados de uma tabela são consistentes com o total de linhas e colunas da tabela obtidos de outras fontes. Essa técnica foi proposta por Deming e Stephan[14] como um método para estimar probabilidades de célula numa tabela.

A relação entre o problema de estimação da matriz de tráfego e do algoritmo IPF é que o fluxo OD $\mathbf{x}$ é um vetor não negativo. Então, ele pode ser convertido em uma função de probabilidade, após assumir que o volume total do tráfego é conhecido, e, assim, a equação de observação define o conjunto convexo e restrito naturalmente [33].

O algoritmo IPF para a estimação da matriz de tráfego é dado a seguir

Algoritmo 1 O algoritmo IPF para a estimação da matriz de tráfego

Inicialização Dados um ponto inicial $\hat{\mathbf{x}}$, a matriz de roteamento $\mathbf{A}$ e o tráfego nos canais medido $\mathbf{y}$.

Passo 1

$$\textbf{for} \quad i = 1, \cdots, M \tag{3.9}$$

$$\alpha = \sum_j A_{ij} x_j / y_i \tag{3.10}$$

$$x_j^+ = x_j/\alpha \quad \textbf{for all } j \textbf{ with } A_{ij} > 0 \tag{3.11}$$

Passo 2 Se $\|x^+ - x\| \leq \epsilon$, volta ao Passo 1.

3.5 O Algoritmo Genético

3.5.1 Introdução

Essencialmente, algoritmos genéticos (AGs) são algoritmos de busca que simulam o processo de evolução natural, aplicando a idéia darwiniana de seleção [20]. Para tanto, procede-se da seguinte maneira: inicialmente gera-se aleatoriamente uma população inicial. Cada membro da população representa uma possível solução do problema, e é avaliado de acordo com alguma medida de adaptação (e.g. função de aptidão). Os indivíduos com alto grau de adaptabilidade têm mais oportunidade de se reproduzir, através da mutação e cruzamento (crossover). Esta técnica segue a lei de genética natural onde os mais aptos têm maior probabilidade de sobreviver. Então, novas soluções são geradas e soluções inadequadas na população são descartas. Com isso, uma nova população é formada. A nova população é utilizada como entrada para a próxima iteração do algoritmo. Assim, este processo é repetido até uma solução satisfatória seja encontrada.

Os Algoritmos Genéticos (AGs) diferem dos métodos tradicionais de busca e otimização, principalmente em quatro aspectos: [20]

1. AGs estão baseados em uma codificação do conjunto das soluções possíveis.

2. AGs trabalham com uma população de soluções e não com um único ponto.

3. AGs não necessitam de nenhum conhecimento derivado do problema, apenas de uma forma de avaliação da qualidade de cada ponto amostrado, enquanto solução do problema.

4. AGs utilizam regras de transição probabilísticas e não determinísticas.

A estrutura básica do algoritmo genético é mostrada na figura 3.3:

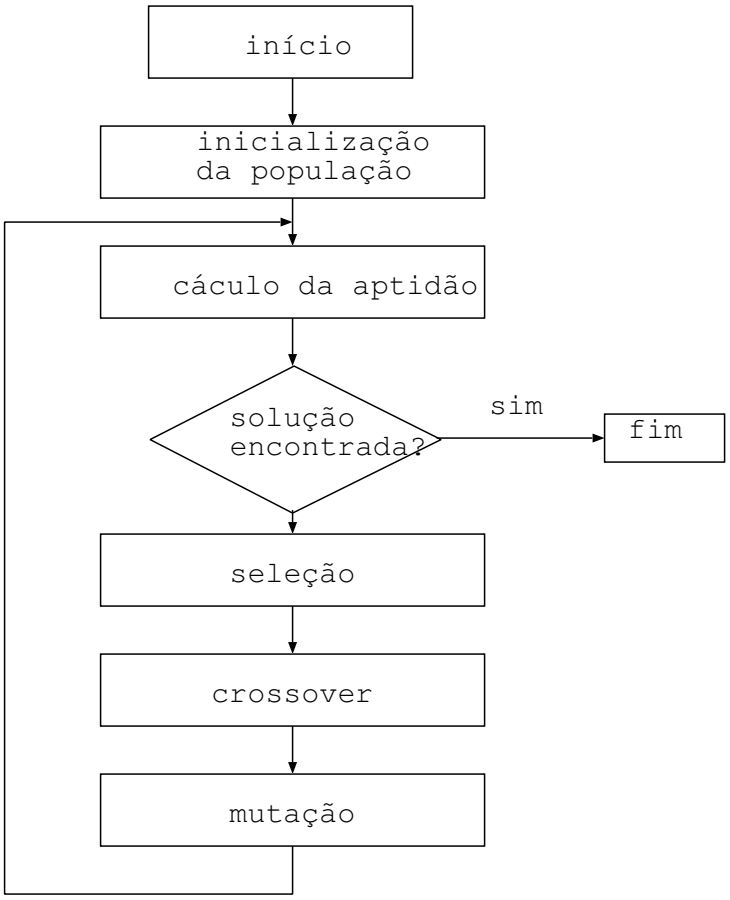


Figura 3.3: A estrutura básica do algoritmo genético

3.5.2 Elementos básicos

Representação

A representação das soluções é um aspecto fundamental para os algoritmos genéticos e deve descrever o espaço de busca em termos de sua características. Geralmente, a representação binária dos cromossomas é utilizada. Cada cromossomo

é constituído por uma sequência de bits. Os outros tipos de representação são: números reais, inteiros, grupamento de inteiro e baseadas em ordem.

População

Primeiramente, precisa-se definir o tamanho da população. o tamanho da população influencia o desempenho e a eficiência dos algoritmos genéticos. Uma grande população normalmente permite uma maior diversidade de soluções. Mas, para se trabalhar com grandes populações, são necessários maiores recursos computacionais, ou maior tempo de processamento do problema. Então, deve-se escolher o tamanho da população buscando um ponto de equilíbrio entre diversidade da população e tempo de computação.

Avaliação a aptidão (fitness)

Cada individuo da população é avaliado no início de cada geração. A avaliação permite ao algoritmo genético determinar sua proximidade à solução ótima do problema. Uma vez selecionada a função de avaliação, é construída a função de aptidão que melhor representa o problema e tem como objetivo oferecer uma medida da aptidão de cada indivíduo. A diferença entre a função objetivo e a função de aptidão é a função objetivo fornece uma medida de qualidade em relação a um problema específico e é restrita a um indivíduo. A função de aptidão transforma essa medida numa grandeza que representa a opotunidade de reprodução. algumas vezes a função de aptidão pode ser o mesmo da função objetivo. Porém, isso nem sempre possível.

Seleção

A função de seleção é selecionar quais indivíduos participarão do processo de reprodução, ou seja, selecionar os indivíduos que sofrerão alterações ou mesmo passar diretamente para a próxima geração. Esta escolha pode ser feita de tal forma que os indivíduos mais adaptados que apresentam um valor da função fitness maior,

tenham maior chance de reprodução. Há diversos métodos de implementação deste processo de seleção. Os métodos mais conhecidos são descritos a seguir:

1. Roleta. Cada indivíduo da população é representado em uma fração de uma roleta, proporcionalmente ao seu índice de aptidão (fitness). Com isso, os indivíduos que possuem alta aptidão ocuparão uma porção maior enquanto os indivíduos que possuem uma aptidão menor ocuparão uma porção menor. A roleta será girada tantas vezes quanto o tamanho de população. Em cada giro da roleta é selecionado um indivíduo para participar do processo de geração da nova população.

2. Torneio. Normalmente escolhe-se dois indivíduos aleatoriamente e retorna o melhor deles em termos dos valores da função de fitness.

3. Elitismo. Uma cópia sem alterações da melhor solução da geração é passada diretamente para a nova população, garantindo que o melhor indivíduo não seja perdido de uma geração para outra.

Cruzamento (Crossover)

O operador de cruzamento tem a função de recombinar as características dos pais durante a reprodução, permitindo que as próximas gerações herdem estas características. Por isso, o operador é baseado na troca de partes dos cromossomos, formando-se novos cromossomos. Existem vários tipos de cruzamento. Os mais utilizados são discutidos a seguir:

1. Cruzamento com um ponto de corte. O ponto onde ocorre o cruzamento é selecionado aleatoriamente. Com isso, o cromossomo dos pais é dividido em duas partes. A primeira parte do cromossomo de um pai é combinada com a segunda parte do outro, e a segunda parte do pai como a primeira parte do outro, formando assim novos indivíduos. A figura a seguir demonstra um exemplo.

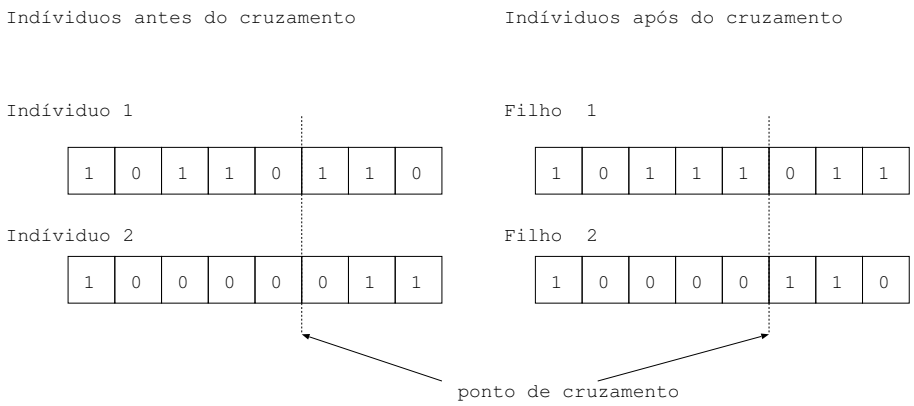


Figura 3.4: Cruzamento com um ponto de corte

2. Cruzamento com n pontos de corte. Similar ao o cruzamento com um ponto de corte, podem ser realizados cortes em mais de um ponto.

3. Cruzamento uniforme. No cruzamento uniforme, é criada aleatoriamente uma máscara que indica os genes dos dois cromossomas que vão ser trocados. No processo de criação, estabelece-se todas as posições da máscara analisando seus valores, quando o valor da posição corrente for 1, o gene do primeiro pai nesta mesma posição da máscara será copiado para o descendente. Caso contrário, o gene do segundo pai será copiado.

Mutação

A mutação tem o objetivo de melhorar a diversidade das características dos indivíduos da população e permite que novas informações que não estiveram apresentadas por nenhum dos indivíduo sejam introduzidas. A mutação consiste na alteração do valor de um ou mais genes de um indivíduos, resultando em um novo indivíduo. No caso de binário troca-se de 0 para 1 e vice-versa (Figura 3.5).

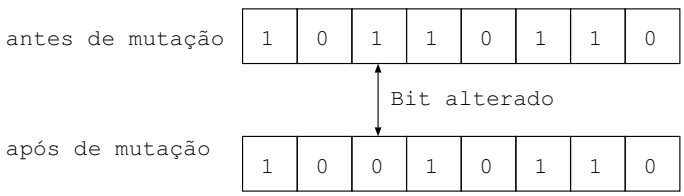


Figura 3.5: Mutação

3.5.3 Um Exemplo Simples

A fim de melhor exemplificar o funcionamento do algoritmo genético, apresentamos um exemplo bem simples de otimização. Este exemplo foi extraído do livro de Goldberg. [20]

Consideramos o problema de maximizar a função $f(x) = x^2$, respeitando as restrições de que x tem que estar no intervalo entre 0 e 31 e ser inteiro.

$$\max \quad f(x) = x^2 \tag{3.12}$$

$$\text{sujeita a} \quad 0 \leq x \leq 31$$

$$\text{x é inteiro} \tag{3.13}$$

Conforme discutido anteriormente, o primeiro passo é codificação da variável x em binário. para representar os 32 possíveis soluções, é necessário usar cadeiras binárias de 5 bits que variam entre 00000 e 11111. o número binário 00000 corresponde ao 0 e 11111 corresponde 31.

O tamanho da população é escolhido quatro neste exemplo. A primeira população pode ser gerada aleatoriamente. Seja a população inicial igual a:

01101, 11000, 01000 e 10011

A avaliação de indivíduos será feita antes do processo de seleção. A função de aptidão precisa-se representar a opotunidade de reprodução, então neste exemplo, a função de aptidão é a própria função objetivo $f(x) = x^2$. O método de seleção, Roleta, é usado neste exemplo. Cada indivíduo i tem uma probabilidade de seleção p_s de acordo com a sua aptidão(*fitness*) f_i. Esse processo se inicia com a soma de todas as aptidões $\sum f_i$ dos indivíduos da atual população e, em seguida, calcula-se a porcentagem $f_i / \sum f_i$ para cada indivíduo i. A Tabela 3.6 mostra os resultados.

A roleta(Figura 3.6) então é girada quatro vezes, pode-se ter, por exemplo, a seguinte escolha onde o indivíduo 2 obteve duas cópias, os indivíduos 1 e 4 uma, enquanto que o indivíduo 3 não obteve nenhuma cópia.

número	indivíduo	x	$f_i = f(x)$	$f/\sum f$
1	01101	13	169	0.14
2	11000	24	576	0.49
3	01000	8	64	0.06
4	10011	19	361	0.31
Soma	-	-	1170	1
Média	-	-	293	0.25
Máximo	-	-	576	0.49

Tabela 3.6: Os resultados obtidos pelo método da roleta

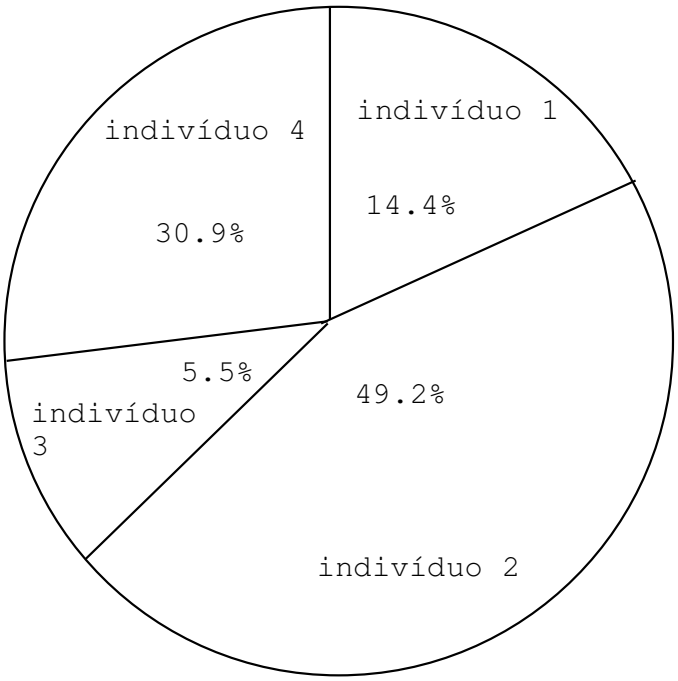


Figura 3.6: Roleta

1°-giro: indivíduo 1

2°-giro: indivíduo 2

3°-giro: indivíduo 2

4°-giro: indivíduo 4

Uma outra maneira de verificar a potencialidade da roleta é calcular o número

esperado de cópias para cada indivíduo, encontrado na penúltima coluna da tabela 3.7. A última mostra o número de cópias realmente obtido por cada indivíduo.

número	indivíduo	x	$f_i = x^2$	Prob. Seleção $f/\sum f$	Núm.Esp de cópias	Número de cópias
1	01101	13	169	0.14	0.58	1
2	11000	24	576	0.49	1.97	2
3	01000	8	64	0.06	0.22	0
4	10011	19	361	0.31	1.23	1
Soma	-	-	1170	1.00	4.00	4
Média	-	-	293	0.25	1.00	1
Máximo	-	-	576	0.49	1.97	2

Tabela 3.7: Os resultados no processo de seleção

Neste exemplo, consideramos a probabilidade de cruzamento $p_c = 0.9$ e a probabilidade de mutação $p_m = 0.001$, A seguir, passamos para os processos de cruzamento e mutação. Existe varias maneiras para escolher indivíduos. Neste exemplo, consideramos que os indivíduos 1 e 2 foram escolhidos para formar o primeiro par enquanto que os indivíduos 3 e 4 o segundo. O ponto de cruzamento é sorteado na posição 4 para o primeiro e na posição 2 para o segundo. Após o cruzamento, obtivemos os novos indivíduos que estão na ultima coluna da tabela 3.8.

Finalmente, aplicamos a mutação. Como a probabilidade de mutação escolhida $p_m = 0.001$, então há poucas chances de ocorrer mutação nessa geração.

casal	corte entre os locus	indivíduo	novo indivíduo
1	4 e 5	0110|1	0100|0
		1100|0	1100|1
2	2 e 3	11|000	11|011
		10|011	10|000

Tabela 3.8: Após o cruzamento

Após a seleção, cruzamento e mutação, a nova população é gerada. Observamos que na tabela 3.9 tanto o valor máximo quanto a média melhoraram na nova população. A função objetivo média aumentou de 293 para 439 e a máxima aumentou de 576 para 729. Estas circunstancias não são um golpe de sorte, devido ao processo randômico. Na verdade, o melhor indivíduo da primeira geração (11000) recebe duas cópias devido a sua performance alta e acima da média. Quando esta combina-se aleatoriamente com o segundo melhor indivíduo (10011) e é cruzada na posição 2 (aleatoriamente), uma das cadeiras resultantes (11011) tem melhor performance que as duas que a deram origem [20].

número	indivíduo	x	$f_i = f(x)$	
1	01100	12	144	
2	11001	25	625	
3	11011	27	729	
4	10000	16	256	
Soma	-	-	1754	
Média	-	-	439	
Máximo	-	-	729	

Tabela 3.9: Geração nova

Capítulo 4

Matriz de Tráfego para Roteamento pelo Caminho mais curto

Neste capítulo, tratamos o problema de estimação da matriz de tráfego quando o roteamento empregado é roteamento pelo caminho mais curto. Os conceitos e propriedades de otimização aqui usados podem ser encontrados no [24].

4.1 Incerteza nos Dados SNMP

Uma vez que a estimativa inicial do modelo de gravidade, $\mathbf{x}_g$, é calculada a partir de (2.9), é muito pouco provável que a estimativa inicial satisfaça à equação $\mathbf{Ax} = \mathbf{y}$. Como está escrito em [43], os dados $\mathbf{y}$ medidos via SNMP têm muitas limitações. Dados perdidos (SNMP utiliza o protocolo de transferência inconfiável de dados UDP), dados incorretos (através de pobres implementações nos roteadores) e um intervalo de amostragem longo (cinco minutos é típico) são exemplos destas limitações. Portanto, os dados medidos via SNMP têm erros e sua confiabilidade pode variar.

No modelo de gravidade apresentado no capítulo 2, erros nos dados medidos via SNMP não são considerados. Porém mesmo que esses erros sejam pequenos, poderiam trazer grandes erros na estimação da matriz de tráfego.

Uma maneira de resolver este problema é formular o problema de mínimos quadrados a seguir,

$$\min_{\mathbf{x}} \quad f(\mathbf{x}) = (\mathbf{x} - \mathbf{x}_g)^T(\mathbf{x} - \mathbf{x}_g) + (\mathbf{A}\mathbf{x} - \hat{\mathbf{y}})^T\mathbf{W}^{-1}(\mathbf{A}\mathbf{x} - \hat{\mathbf{y}}) \tag{4.1}$$

onde o vetor $\hat{\mathbf{y}}$ fornece o tráfego medido nos canais via SNMP, $\mathbf{W}^{-1}$ é uma matriz de peso que é simétrica definida positiva. No presente trabalho, consideramos $\mathbf{W}^{-1}$ como uma matriz de identidade.

Proposição 1 *O Problema* (4.1) *tem uma única solução* $\boldsymbol{x}$ *que pode ser expressa como*

$$\boldsymbol{x} = \boldsymbol{x}_g + \boldsymbol{A}^T(\boldsymbol{A}\boldsymbol{A}^T + \boldsymbol{W})^{-1}(\hat{\boldsymbol{y}} - \boldsymbol{A}\boldsymbol{x}_g) \tag{4.2}$$

Prova: como a função objetivo de (4.1) *é diferenciável e estritamente convexa, o problema* (4.1) *tem uma única solução caracterizada por*

$$\nabla f(\boldsymbol{x}) = 0 \tag{4.3}$$

De (4.3)*, obtemos*

$$(\boldsymbol{x} - \boldsymbol{x}_g) - \boldsymbol{A}^T\boldsymbol{W}^{-1}(\hat{\boldsymbol{y}} - \boldsymbol{A}\boldsymbol{x}) = \boldsymbol{0} \tag{4.4}$$

que implica,

$$(\boldsymbol{I} + \boldsymbol{A}^T\boldsymbol{W}^{-1}\boldsymbol{A})\boldsymbol{x} = \boldsymbol{x}_g + \boldsymbol{A}^T\boldsymbol{W}^{-1}\hat{\boldsymbol{y}} \tag{4.5}$$

Como $\boldsymbol{W}^{-1}$ *é simétrica definida positiva,* $\boldsymbol{A}^T\boldsymbol{W}^{-1}\boldsymbol{A}$ *é também simétrica definida positiva. Então* $\boldsymbol{I}+\boldsymbol{A}^T\boldsymbol{W}^{-1}\boldsymbol{A}$ *é simétrica definida positiva e, portanto, não singular*[1]*. Isso significa que o seu inverso existe e*

$$\boldsymbol{x} = (\boldsymbol{I} + \boldsymbol{A}^T\boldsymbol{W}^{-1}\boldsymbol{A})^{-1}(\boldsymbol{x}_g + \boldsymbol{A}^T\boldsymbol{W}^{-1}\hat{\boldsymbol{y}}) \tag{4.6}$$

[1]De fato, $\mathbf{I} + \mathbf{A}^T\mathbf{W}^{-1}\mathbf{A}$ é simétrica definida positiva pois $\mathbf{x}^T(\mathbf{I} + \mathbf{A}^T\mathbf{W}^{-1}\mathbf{A})\mathbf{x} = \|\mathbf{x}\|^2 + \|\mathbf{W}^{1/2}\mathbf{A}\mathbf{x}\|^2 \geq 0$ e é zero somente se $\mathbf{x} = 0$. Mais ainha se $\mathbf{I}+\mathbf{A}^T\mathbf{W}^{-1}\mathbf{A}$ for singular, significa que existe $\mathbf{x} \neq 0$ tal que $(\mathbf{I}+\mathbf{A}^T\mathbf{W}^{-1}\mathbf{A})\mathbf{x} = 0$, portanto $\mathbf{x}^T(\mathbf{I}+\mathbf{A}^T\mathbf{W}^{-1}\mathbf{A})\mathbf{x} = 0$ o que é um absurdo.

O Lema de Inversão de Matrizes [52]

$$(\boldsymbol{M}+\boldsymbol{UBV})^{-1} = \boldsymbol{M}^{-1} - \boldsymbol{M}^{-1}\boldsymbol{U}(\boldsymbol{B}^{-1}+\boldsymbol{VM}^{-1}\boldsymbol{U})^{-1}\boldsymbol{VM}^{-1} \tag{4.7}$$

pode ser usado em (4.6) *considerando* $\boldsymbol{M}=\boldsymbol{I}$, $\boldsymbol{U}=\boldsymbol{A}^T$, $\boldsymbol{B}=\boldsymbol{W}^{-1}$ *e* $\boldsymbol{V}=\boldsymbol{A}$

$$(\boldsymbol{I}+\boldsymbol{A}^T\boldsymbol{W}^{-1}\boldsymbol{A})^{-1} \;=\; \boldsymbol{I}-\boldsymbol{A}^T(\boldsymbol{AA}^T+\boldsymbol{W})^{-1}\boldsymbol{A} \tag{4.8}$$

Logo, considerando $\boldsymbol{I}=(\boldsymbol{AA}^T+\boldsymbol{W})^{-1}(\boldsymbol{AA}^T+\boldsymbol{W})$,*obtemos*

$$\begin{aligned}
\boldsymbol{x} &= (\boldsymbol{I}-\boldsymbol{A}^T(\boldsymbol{AA}^T+\boldsymbol{W})^{-1}\boldsymbol{A})(\boldsymbol{x}_g+\boldsymbol{A}^T\boldsymbol{W}^{-1}\hat{\boldsymbol{y}}) \\
&= \boldsymbol{x}_g+\boldsymbol{A}^T\boldsymbol{W}^{-1}\hat{\boldsymbol{y}}-\boldsymbol{A}^T(\boldsymbol{AA}^T+\boldsymbol{W})^{-1}(\boldsymbol{Ax}_g+\boldsymbol{AA}^T\boldsymbol{W}^{-1}\hat{\boldsymbol{y}}) \\
&= \boldsymbol{x}_g+\boldsymbol{A}^T(\boldsymbol{AA}^T+\boldsymbol{W})^{-1}(\boldsymbol{AA}^T+\boldsymbol{W})\boldsymbol{W}^{-1}\hat{\boldsymbol{y}}-\boldsymbol{A}^T(\boldsymbol{AA}^T+\boldsymbol{W})^{-1}(\boldsymbol{Ax}_g+\boldsymbol{AA}^T\boldsymbol{W}^{-1}\hat{\boldsymbol{y}}) \\
&= \boldsymbol{x}_g+\boldsymbol{A}^T(\boldsymbol{AA}^T+\boldsymbol{W})^{-1}\big((\boldsymbol{AA}^T+\boldsymbol{W})\boldsymbol{W}^{-1}\hat{\boldsymbol{y}}-\boldsymbol{Ax}_g-\boldsymbol{AA}^T\boldsymbol{W}^{-1}\hat{\boldsymbol{y}}\big) \\
&= \boldsymbol{x}_g+\boldsymbol{A}^T(\boldsymbol{AA}^T+\boldsymbol{W})^{-1}\big(\boldsymbol{AA}^T\boldsymbol{W}^{-1}\hat{\boldsymbol{y}}+\hat{\boldsymbol{y}}-\boldsymbol{Ax}_g-\boldsymbol{AA}^T\boldsymbol{W}^{-1}\hat{\boldsymbol{y}}\big) \qquad (4.9)
\end{aligned}$$

Finalmente, temos,

$$\boldsymbol{x} \;=\; \boldsymbol{x}_g+\boldsymbol{A}^T(\boldsymbol{AA}^T+\boldsymbol{W})^{-1}(\hat{\boldsymbol{y}}-\boldsymbol{Ax}_g)$$

4.2 Não-negatividade das Variáveis

A solução (4.2) não garante a não-negatividade dos elementos da matriz de tráfego. Em [58], O Ajuste Proporcional Iterativo (IPF), como sugerido em [10], é utilizado para garantir a não-negatividade. A estimativa de tráfego matriz, solução de (2.13), com a substituição de zero nos elementos negativos da matriz, é usada como um ponto inicial no processo iterativo, ou seja, este ponto inicial é a projeção da solução de (2.13) sobre o conjunto $\mathbf{R}_+^N = \{x \in \mathbf{R}^N : x_i \geq 0, i = 1, 2, \ldots, N\}$.

É relatada em [58] que o processo IPF prossegue sucessivamente para refinar a estimativa e que só leva algumas iterações para reduzir erros nas equações de restrição ao ponto em que eles são insignificantes na prática. Note-se que o IPF também é baseado no sistema linear $\mathbf{Ax} = \mathbf{y}$. Assim, os erros no $\mathbf{y}$ também têm

grande influência sobre o exatidão das estimativas da matriz de tráfego, quando o procedimento IPF é utilizado.

Dado que (4.1) é um problema de otimização de mínimos quadrados, a maneira direta para garantir a não-negatividade, seria colocar a restrição de não-negatividade no problema. Temos,

$$\begin{aligned} \min \quad & f(\mathbf{x}) = (\mathbf{x}-\mathbf{x}_g)^T(\mathbf{x}-\mathbf{x}_g) + (\mathbf{A}\mathbf{x}-\hat{\mathbf{y}})^T\mathbf{W}^{-1}(\mathbf{A}\mathbf{x}-\hat{\mathbf{y}}) \qquad (4.10) \\ \text{sujeito a:} \quad & \mathbf{x} \geq 0 \end{aligned}$$

Note-se que a solução de (4.2) não é a solução do problema acima, uma vez que a não-negatividade não está sendo garantida . Portanto, o método de projeção de gradiente é usado para resolver este problema.

4.3 O Método de Projeção do Gradiente

Seja $\mathbf{C} \subset \mathbf{R}^n$ um conjunto convexo e fechado. Dado $\mathbf{x} \in \mathbf{R}^n$, a projeção Euclideana de $\mathbf{x}$ no conjunto $\mathbf{C}$ é $P_{\mathbf{C}}(\mathbf{x}) = \arg\min_{\mathbf{v}\in\mathbf{C}} \frac{1}{2}\|\mathbf{x}-\mathbf{v}\|^2$. Em outras palavras, o ponto $P_{\mathbf{C}}(\mathbf{x})$ é o ponto mais próximo em $\mathbf{C}$ do ponto $\mathbf{x}$ no sentido de norma euclideana. Em nosso caso, $\mathbf{C}$ é o conjunto dado por, $\mathbf{R}^N_+ = \{x \in \mathbf{R}^N : x_i \geq 0, i = 1, 2, \ldots, N\}$. Então, o i-ésimo componente de $P_{\mathrm{C}}(\mathbf{x})$ é, $P_{\mathbf{C}}(\mathbf{x})_i = \max(x_i, 0), \quad i = 1, 2, \ldots, N$.

Algoritmo da Projeção do Gradiente

O método gera uma sequência de pontos viáveis do problema (4.10). No ponto atual, $\mathbf{x}_k \geq 0$, a direção $\mathbf{p}_k = -\nabla f(\mathbf{x}_k)$ é considerada[2]. Buscamos o novo ponto $\mathbf{x}$ a partir de $\mathbf{x}_k$, ou seja, $\mathbf{x} = \mathbf{x}_k + \tau_k\mathbf{p}$ com $\tau_k \geq 0$. Como $F(\tau) = f(\mathbf{x}_k + \tau\mathbf{p}_k)$ é uma

[2]Se $\mathbf{p}_k = 0$, então o ponto $\mathbf{x}_k$ minimiza a função estritamente convexa $f(\mathbf{x})$ e portanto é a solução de (4.10). Caso contrário, $\mathbf{p}_k$ é uma direção de descida desde $\mathbf{x}_k$. Neste caso existe $T > 0$ tal que $f(\mathbf{x}_k + t\mathbf{p}_k) < f(\mathbf{x}_k)$ para todo t ∈ (0, T).

função estritamente convexa em t, existe $\tau_k \geq 0$ tal que

$$\min_{\tau \geq 0} f(\mathbf{x}_k + \tau \mathbf{p}_k) = f(\mathbf{x}_k + \tau_k \mathbf{p}_k); \tag{4.11}$$

Então, projetamos este novo ponto $\mathbf{x}$ sobre o conjunto não-negativo para garantir a sua viabilidade, $\mathbf{x}_{k+1} = P_{\mathbf{R}^N_+}(\mathbf{x})$. O algoritmo inexato continua até $\mathbf{p}$ ser próximo de zero ou $\mathbf{x}_{k+1}$ próximo de $\mathbf{x}_k$.

O algoritmo é dado a seguir,

Algoritmo 2 Gradient Projection Method

Passo 1 Inicialização.$k = 0; \mathbf{x}_0 \geq 0$ (solução inicial não-negativa);

Passo 2 Iterações de projeção de gradiente

$$\mathbf{p}_k = -\nabla f(\mathbf{x}_k); \tag{4.12}$$

Se $\mathbf{p}_k = 0$ termina o algoritmo com resposta $\mathbf{x}_k$.

Passo 3

$$\tau_k = \arg\min_{\tau \geq 0} f(\mathbf{x}_k + \tau \mathbf{p}_k); \tag{4.13}$$

$$\mathbf{x}_{k+1} = P_{\mathbf{R}^N_+}(\mathbf{x}_k + \tau_k \mathbf{p}_k); \tag{4.14}$$

$$k = k + 1; \tag{4.15}$$

Passo 4 Se $\max_i \mid x^i_{k+1} - x^i_k \mid \leq \epsilon$, termina o algoritmo com resposta $\mathbf{x}_k$. Senão, volta ao passo 2.

No algoritmo 2, o passo ótimo τ_k em (4.13) pode ser determinado explicitamente.

De fato, temos a seguinte propriedade.

Proposição 2 *Se $\nabla f(\boldsymbol{x}_k) \neq 0$, então $\tau_k > 0$ e pode ser expresso por*

$$\tau_k = -\frac{\boldsymbol{p}_k^T(\boldsymbol{x}_k - \boldsymbol{x}_g) + (\boldsymbol{A}\boldsymbol{p}_k)^T \boldsymbol{W}^{-1}(\boldsymbol{A}\boldsymbol{x}_k - \hat{\boldsymbol{y}})}{\boldsymbol{p}_k^T \boldsymbol{p}_k + (\boldsymbol{A}\boldsymbol{p}_k)^T \boldsymbol{W}^{-1}(\boldsymbol{A}\boldsymbol{p}_k)} \tag{4.16}$$

Prova:

Consideramos o problema

$$\min_{\tau \geq 0}[F(\tau) = f(\boldsymbol{x}_k + \tau \boldsymbol{p}_k)] \tag{4.17}$$

Como $F(\tau)$ *é uma função estritamente convexa e* $\boldsymbol{p}_k$ *é uma direção de descida resulta que existe* $T > 0$ *tal que*

$$F(\tau_k) = \min_{\tau \geq 0} F(\tau) \leq F(t) < F(0) \quad para \quad todo \quad t \in (0, T] \tag{4.18}$$

Portanto, resulta $\tau_k > 0$.

Considere o lagrangeano do problema (4.11)

$$L(\tau, \mu) = F(\tau) - \mu\tau \quad com \quad \mu \geq 0 \tag{4.19}$$

Logo, as condições de otimalidade (τ_k, μ_k) *são*

$$\frac{\partial L(\tau_k, \mu_k)}{\partial \tau} = F^{'}(\tau_k) - \mu_k = 0 \tag{4.20}$$

$$\mu_k \tau_k = 0 \tag{4.21}$$

$$\mu_k \geq 0, \tau_k \geq 0 \tag{4.22}$$

Como a solução $\tau_k > 0$ *deve ter* $\mu_k = 0$ *portanto*

$$F'(\tau_k) = 0 \tag{4.23}$$

Desenvolveremos $f(\boldsymbol{x}_k + \tau \boldsymbol{p}_k)$ *para calcular* $F'(\tau)$.

$$\begin{aligned} f(\boldsymbol{x}_k + \tau \boldsymbol{p}_k) &= (\boldsymbol{x}_k + \tau \boldsymbol{p}_k - \boldsymbol{x}_g)^T (\boldsymbol{x}_k + \tau \boldsymbol{p}_k - \boldsymbol{x}_g) + (\boldsymbol{A}(\boldsymbol{x}_k + \tau \boldsymbol{p}_k) - \hat{\boldsymbol{y}})^T \boldsymbol{W}^{-1} (\boldsymbol{A}(\boldsymbol{x}_k + \tau \boldsymbol{p}_k) - \hat{\boldsymbol{y}}) \\ &= (\boldsymbol{x}_k - \boldsymbol{x}_g)^T (\boldsymbol{x}_k - \boldsymbol{x}_g) + 2\tau \boldsymbol{p}_k^T (\boldsymbol{x}_k - \boldsymbol{x}_g) + \tau^2 \boldsymbol{p}_k^T \boldsymbol{p}_k + (\boldsymbol{A}(\boldsymbol{x}_k - \hat{\boldsymbol{y}}))^T \boldsymbol{W}^{-1} (\boldsymbol{A}(\boldsymbol{x}_k - \hat{\boldsymbol{y}})) \\ &\quad + 2\tau (\boldsymbol{A}\boldsymbol{p}_k)^T \boldsymbol{W}^{-1} (\boldsymbol{A}\boldsymbol{x}_k - \hat{\boldsymbol{y}}) + \tau^2 (\boldsymbol{A}\boldsymbol{p}_k)^T \boldsymbol{W}^{-1} (\boldsymbol{A}\boldsymbol{p}_k) \\ &= f(\boldsymbol{x}_k) + 2\tau \boldsymbol{p}_k^T (\boldsymbol{x}_k - \boldsymbol{x}_g) + \tau^2 \boldsymbol{p}_k^T \boldsymbol{p}_k + \\ &\quad 2\tau (\boldsymbol{A}\boldsymbol{p}_k)^T \boldsymbol{W}^{-1} (\boldsymbol{A}\boldsymbol{x}_k - \hat{\boldsymbol{y}}) + \tau^2 (\boldsymbol{A}\boldsymbol{p}_k)^T \boldsymbol{W}^{-1} (\boldsymbol{A}\boldsymbol{p}_k) \end{aligned} \tag{4.24}$$

logo

$$\begin{aligned} F'(\tau) &= 2\big(\boldsymbol{p}_k^T(\boldsymbol{x}_k - \boldsymbol{x}_g) + (\boldsymbol{A}\boldsymbol{p}_k)^T \boldsymbol{W}^{-1}(\boldsymbol{A}\boldsymbol{x}_k - \hat{\boldsymbol{y}})\big) + \\ &\quad 2\tau\big(\boldsymbol{p}_k^T\boldsymbol{p}_k + (\boldsymbol{A}\boldsymbol{p}_k)^T \boldsymbol{W}^{-1}(\boldsymbol{A}\boldsymbol{p}_k)\big) \end{aligned} \tag{4.25}$$

Logo, de (4.23) *e* (4.25) *resulta* (4.16).

Proposição 3 *Seja* $\boldsymbol{x}_k \geq 0$ *e* $\tau_k > 0$. *Então,* $\boldsymbol{x}_{k+1} = P_{\boldsymbol{R}_+^N}(\boldsymbol{x}_k + \tau_k\boldsymbol{p}_k) = \boldsymbol{x}_k$ *se e somente se* $\boldsymbol{x}_k$ *é a solução do problema* (4.10).

Prova: Pela propriedade de projeção, tem-se que um ponto $\boldsymbol{P}$ *de* $\boldsymbol{R}_+^N$ *é a projeção do ponto* $\boldsymbol{x}_k + \tau_k\boldsymbol{p}_k$ *se e somente se*

$$(\boldsymbol{y} - \boldsymbol{P})^T(\boldsymbol{x}_k + \tau_k\boldsymbol{p}_k - \boldsymbol{P}) \leq 0 \quad \forall \boldsymbol{y} \in \boldsymbol{R}_+^N \tag{4.26}$$

Se $\boldsymbol{x}_k$ *é solução do problema* (4.10) *então a derivada direcional de* f *em* $\boldsymbol{x}_k$ *na direção* $\boldsymbol{d} = \boldsymbol{y} - \boldsymbol{x}_k, \quad \forall \boldsymbol{y} \in \boldsymbol{R}_+^N$ *é não negativa:*

$$f'(\boldsymbol{x}_k, \boldsymbol{y} - \boldsymbol{x}_k) = \lim_{t \to 0^+} \frac{f(\boldsymbol{x}_k + \boldsymbol{t}(\boldsymbol{y} - \boldsymbol{x}_k)) - f(\boldsymbol{x}_k)}{t} \geq 0 \quad \forall \boldsymbol{y} \in \boldsymbol{R}_+^N \tag{4.27}$$

Como a função f *é diferenciável tem-se que*

$$0 \leq f'(\boldsymbol{x}_k, \boldsymbol{y} - \boldsymbol{x}_k) = (\boldsymbol{y} - \boldsymbol{x}_k)^T \nabla f(\boldsymbol{x}_k) \quad \forall \boldsymbol{y} \in \boldsymbol{R}_+^N \tag{4.28}$$

Logo

$$\begin{aligned} 0 &\geq -\tau_k(\boldsymbol{y} - \boldsymbol{x}_k)^T \nabla f(\boldsymbol{x}_k) \\ &= (\boldsymbol{y} - \boldsymbol{x}_k)^T\big(-\tau_k \nabla f(\boldsymbol{x}_k)\big) \\ &= (\boldsymbol{y} - \boldsymbol{x}_k)^T\big(\boldsymbol{x}_k - \tau_k \nabla f(\boldsymbol{x}_k) - \boldsymbol{x}_k\big) \\ &= (\boldsymbol{y} - \boldsymbol{x}_k)^T\big((\boldsymbol{x}_k + \tau_k\boldsymbol{p}_k) - \boldsymbol{x}_k\big) \quad \forall \boldsymbol{y} \in \boldsymbol{R}_+^N \end{aligned} \tag{4.29}$$

Portanto, por (4.26), *dever ser* $\boldsymbol{x}_k$ *a projeção de* $\boldsymbol{x}_k + \tau_k\boldsymbol{p}_k$ *sobre o conjunto* $\boldsymbol{R}_+^N$, *ou seja,* $\boldsymbol{x}_k = P_{\boldsymbol{R}_+^N}(\boldsymbol{x}_k + \tau_k\boldsymbol{p}_k) = \boldsymbol{x}_{k+1}$.

Para provar a outra direção, suponha agora que $\boldsymbol{x}_k = \boldsymbol{x}_{k+1}$. *Então, pela desigualdade* (4.26), *caraterizando a projeção usando* $\boldsymbol{x}_k = \boldsymbol{x}_{k+1}$ *e* $\boldsymbol{p}_k = -\nabla f(\boldsymbol{x}_k)$ *resulta*

$$\begin{aligned} 0 &\geq (\boldsymbol{y} - \boldsymbol{x}_{k+1})^T(\boldsymbol{x}_k + \tau_k \boldsymbol{p}_k - \boldsymbol{x}_{k+1}) \\ &= (\boldsymbol{y} - \boldsymbol{x}_k)^T(\boldsymbol{x}_k + \tau_k \boldsymbol{p}_k - \boldsymbol{x}_k) \\ &= (\boldsymbol{y} - \boldsymbol{x}_k)^T(-\tau_k \nabla f(\boldsymbol{x}_k)) \quad \forall \boldsymbol{y} \in \boldsymbol{R}_+^N \end{aligned} \tag{4.30}$$

Logo

$$f'(\boldsymbol{x}_k, \boldsymbol{y} - \boldsymbol{x}_k) = (\boldsymbol{y} - \boldsymbol{x}_k)^T \nabla f(\boldsymbol{x}_k) \geq 0 \quad \forall \boldsymbol{y} \in \boldsymbol{R}_+^N \tag{4.31}$$

Portanto, $\boldsymbol{x}_k$ *é a solução do problema.*

Proposição 4 *A seqüência gerada pelo método de projeção do gradiente converge à solução deste problema quando* $\epsilon = 0$ *(algoritmo exato).*

Prova: Como a função objetivo do problema (4.10) *é não negativa, quadrática e estritamente convexa, é uma aplicação direta das Proposições 3.4 e 3.5 dadas em [6].*

4.4 Resultados para o Problema A

4.4.1 Uso de Dados Sintéticos

Consideramos uma topologia pequena de 4 nós, mostrado na Figura 4.1. Esta topologia também é usada em [37]. Todos os pares OD e seus tráfegos podem ser enumerados. Isso é útil para ilustrar como os métodos se comportam.

Dados sintéticos são muito úteis para avaliar o desempenho das técnicas para a estimação da matriz de tráfego. Isto se deve a que podemos estudar o comportamento das técnicas sob avaliação com relação às matrizes de tráfego completas, ao invés das matrizes de tráfego parciais.

Os dados sintéticos também permitem avaliar melhor os erros gerados pelas diferentes técnicas.

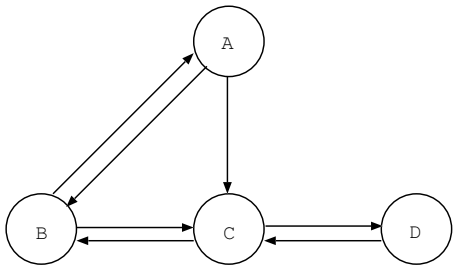


Figura 4.1: Topologia de uma rede de 4 nós

Em geral, nos estudos e nas avaliações comparativas existentes na literatura, as matrizes de tráfego geradas sinteticamente são baseadas nas hipóteses fortes que respeitam à distribuição das demandas de tráfego dos pares origem-destino (OD) [37, 53, 10]. Por exemplo, é comum assumir que as demandas OD são distribuídas de acordo com a distribuição Gaussiana ou de Poisson. Em nossos experimentos de dados sintéticos, consideramos a matriz de tráfego gerada com distribuição de Poisson. Esta matriz também é usada e analisada em [37].

A matriz de tráfego sintética é obtida através de geração uniforme dos parâmetros de Poisson Λ_i para cada par OD i em um intervalo $[100, 500]$. Então, as demandas de tráfego de Poisson são geradas de acordo com $x_i \sim Poisson(\lambda_i)$. Roteamos esta matriz de tráfego na topologia de 4 nós para obter o tráfego nos canais sintéticos. Para avaliar o impacto dos erros de medição SNMP, introduzimos um vetor de erro ϵ. O vetor de erro é formado pela multiplicação $\mathbf{y}$ com um vetor de ruído, $\varepsilon = \mathbf{y} * N(0, \phi)$, onde $*$ denota o produto de Hadamard [3], $N(0, \phi)$ é um vetor e seus elementos são gerados aleatoriamente por distribuição Normal com média 0 e variância ϕ. Em resumo, os tráfegos nos canais são gerados adicionando ruído branco ao $\mathbf{y}$, isso é, $\hat{y} = y + \varepsilon$. Finalmente, a matriz de tráfego a priori é gerada a partir do modelo de gravidade (2.9). O seu erro relativo médio (MRE) é cerca de 33%, e por isso esta matriz **não** é uma boa matriz a priori. Note que, se a matriz a priori é muito boa (perto da solução final), o processo de avaliação não seria adequado.

Para avaliar os nossos métodos, precisamos saber a precisão de estimação da matriz de tráfego. Primeiro, consideramos a métrica MRE, que é definida como

[3] $a, b \in R^n, a * b = (a_1 b_1, ...a_n b_n)$

$$MRE = \frac{1}{N_T} \sum_{i:x_i>T} \left| \frac{\hat{x}_i - x_i}{x_i} \right| \tag{4.32}$$

onde x_i é o elemento verdadeiro da matriz de tráfego e $\hat{x}_i$ é a estimativa correspondente. N_T é o número dos elementos da matriz que são maiores do que um limite T. Neste experimento, quando a matriz de tráfego é gerada com distribuição de Poisson e a diferença entre o maior e o menor valor não é grande, então, consideramos todos os elementos da matriz de tráfego na avaliação MRE .

A tabela 4.1 compara a matriz de tráfego estimada obtida pelo método de tomogravidade com a nossa abordagem. Observamos que a estimativa do método de tomogravidade para o tráfego entre par CB é um valor negativo. Após o processo IPF, esse valor é ajustado para zero aproximadamente. Note-se que este valor conduz a um grande erro. Com nossa abordagem, o valor da estimativa é 53,51, que é consideravelmente melhor do que a estimativa do método de tomogravidade. Também pode ser observado que, quando o método de tomogravidade é usado, o MRE diminui um pouco após IPF. Neste exemplo, o nosso método só precisa de uma única etapa, e não precisa entrar em um processo iterativo. O seu MRE é menor do que o obtido pelo método de tomogravidade.

4.4.2 Utilização de Dados Reais

Usamos matrizes de tráfego de 6 meses da Rede Abilene[4]

(Figura 4.2) recolhidas por Zhang [57].

A matriz de roteamento e a solução obtida pelo modelo de gravidade também são fornecidas para a comunidade de pesquisa científica. Realizamos dois experimentos em duas séries temporais diferentes de matrizes de tráfego. Cada série temporal consiste de 500 matrizes de tráfego medidos durante 5 minutos cada uma. O tempo inicial da primeira série temporal é 01/03/2004 e da segunda 31/07/2004. O tráfego nos canais pode ser obtido através do cálculo do produto da matriz de tráfego e

[4]Rede Abilene é a rede de backbone de Internet2 com alta performance.

TM			Tomogravidade		Our Method
OD Pair	Original TM	prior TM (gravity)	TM before IPF	TM after IPF	TM
AB	318	241.26	260.28	248.77	165.81
AC	289	344.40	221.17	228.91	192.42
AD	312	142.48	111.44	383.52	312.57
BA	294	256.79	538.94	292.91	223.33
BC	292	574.77	378.	258.95	238.51
BD	267	237.79	376.12	376.12	322.22
CA	305	350.82	165.18	147.95	224.70
CB	289	550.09	-48.50	0.81	53.51
CD	324	324.86	322.17	322.17	288.71
DA	283	157.52	308.52	298.68	246.45
DB	277	247.01	183.04	160.81	163.47
DC	291	352.60	396.69	428.76	275.10
	MRE	33.33%	29.84%	29.42%	26.88%

Tabela 4.1: Comparação com o método de tomogravidade

da matriz de roteamento **A**. Para avaliar a nossa abordagem, simulamos o ruído descrito na secção 4.4.1.

O tráfego mais importante a ser estimado é o maior tráfego de pares OD. A sua exatidão tem um grande impacto para as tarefas de engenharia de tráfego, como balanceamento de carga ou análise de falhas. Neste experimento, podemos escolher o valor de N_T (definido em (4.32)) como 5.000.000 no sentido que o tráfego considerado consiste aproximadamente 85% do tráfego total.

As figuras 4.3(a) e 4.3(b) mostram a função de distribuição percentual cumulativa empírica das estimativas das matrizes de tráfego para os dois métodos que comparamos usando os dados destas datas 01/03/2004 e 31/07/2004. Por exemplo, na Figura 4.3(a), podemos ver que apenas cerca de 50% das matrizes de tráfego estimadas pelo método de tomogravidade tem erros que são menores do que 35%, enquanto 70% da matrizes de tráfego estimadas pela nossa abordagem tem erros menores que 35%. A figura 4.3(b) mostra comportamento semelhante. Se estamos considerando os erros menores do que 35%, apenas 30% da matrizes de tráfego es-

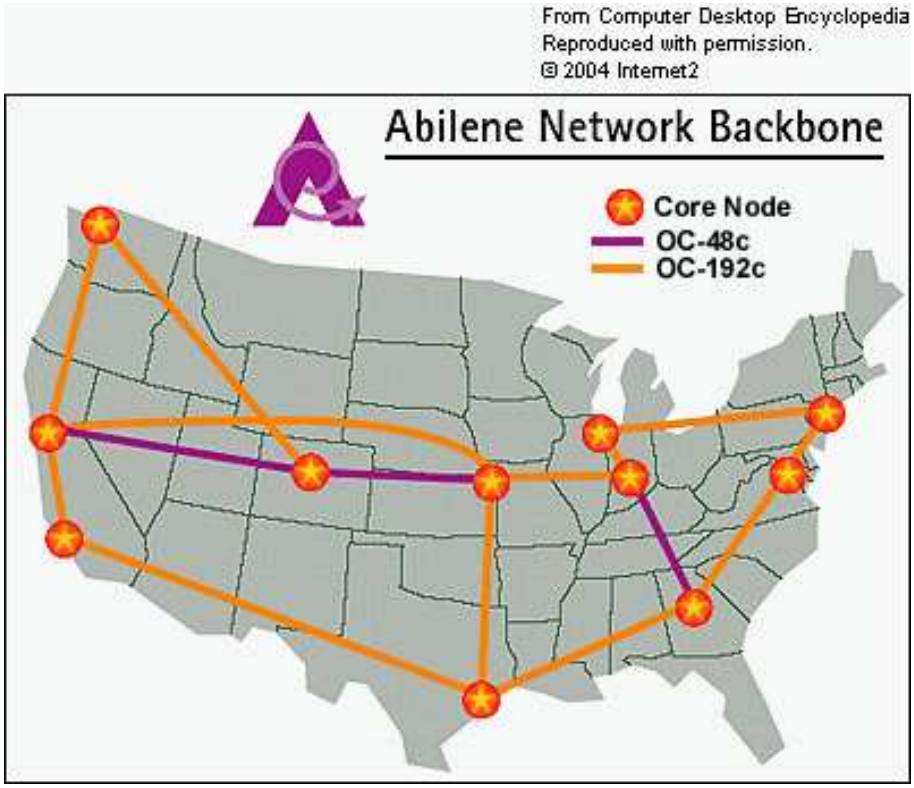
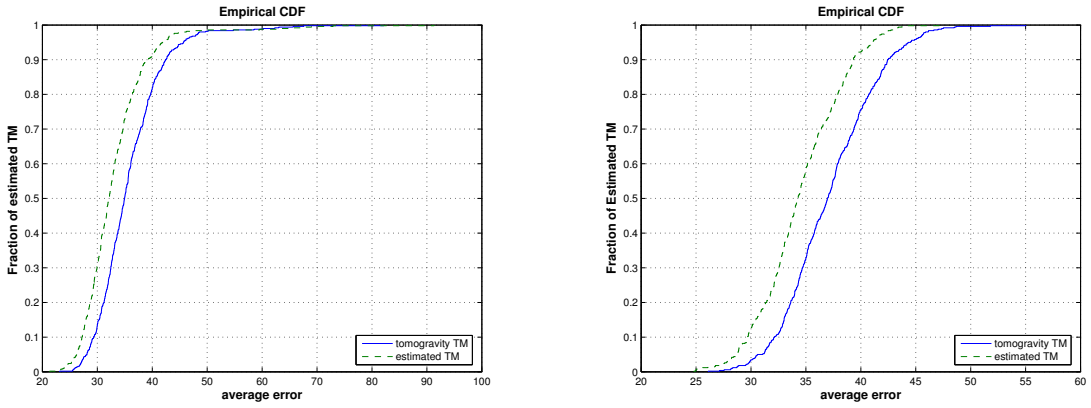


Figura 4.2: Rede Abilene

timadas pelo método de tomogravidade estão sob este limite, em comparação com 60% da matrizes de tráfego estimadas pelo nosso método. Segundo essas figuras, podemos concluir que o nosso método tem melhor desempenho comparado com o método de tomogravidade, mostrando a vantagem da incorporação de $\mathbf{Ax} - \hat{\mathbf{y}}$ à função objetivo.

A continuação, o nosso trabalho é avaliar o impacto de ruído de diferentes níveis em medição SNMP sobre a estimação da matriz de tráfego. Em particular, o nível de ruído ϕ é o que foi utilizado na definição de $\varepsilon = \mathbf{y} * N(0, \phi)$. A Figura 4.4 mostra a comparação de erro relativo médio (MRE), com o método de tomogravidade com ruído de diferentes níveis para uma matriz de tráfego determinada. O erro de medição SNMP é sinteticamente gerado pela distribuição Normal com média 0 e variância ϕ. É razoável que o MRE aumente com o nível de ruído ϕ, como mostrado na figura.

A figura 4.5(a) mostra a diferença entre erro relativo médio (MRE) do método de



(a) 01/03/2004 (b) 31/07/2004

Figura 4.3: A função de distribuição cumulativa empírica das estimativas das matrizes de tráfego

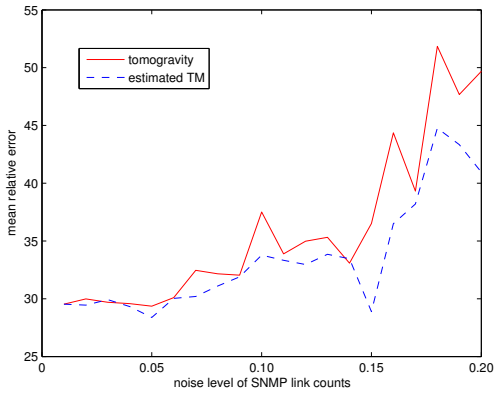
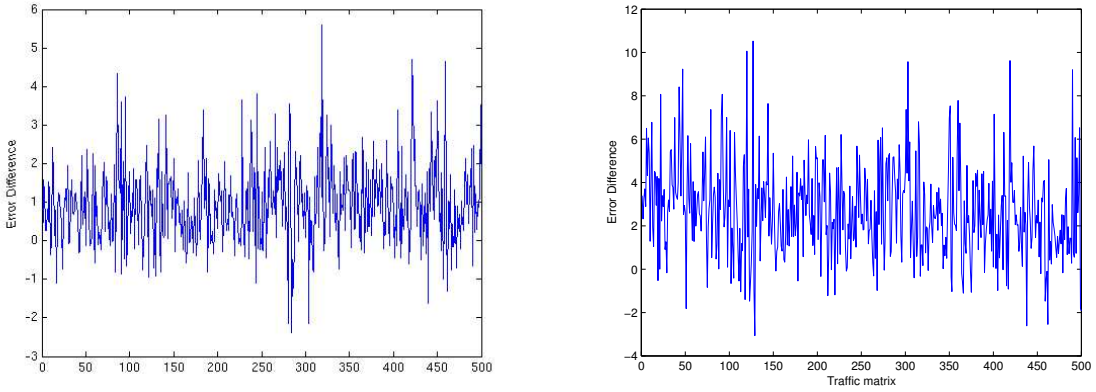


Figura 4.4: Comparação com o método de tomogravidade com ruído de diferentes níveis para uma matriz de tráfego determinada

tomogravidade e o erro relativo médio (MRE) do nosso método para 500 matrizes de tráfego, quando o nível de ruído é 0,05. Vemos que 411 (82,2%) são valores positivos. Isto significa que, na maioria dos casos, o nosso método tem um MRE menor. O valor médio desta diferença é 0,93. Isso é, quando o nível de ruído é 0,05, o nosso método tem uma MRE que é de cerca de 0,93% menor do que o método de tomogravidade. A figura 4.5(b) mostra os resultados quando o nível de ruído é 0,1. Neste caso, 453 (90,6%) são valores positivos, e consequentemente, o nosso método tem um MRE menor . O valor médio desta diferença é 2,69. Isto significa que, quando o nível de ruído é 0,1, o nosso método tem um MRE que é de cerca de

2,69% menor do que o método de tomogravidade.



(a) 01/03/2004 e o nível de ruído = 0.05 (b) 31/07/2004 e o nível de ruído = 0.1

Figura 4.5: Comparação com o método de tomogravidade com ruído de diferentes níveis

A Tabela 4.3 mostra o valor médio das diferenças de MRE com diferentes níveis de ruído para 500 matrizes de tráfego. A tabela mostra que a precisão do nosso método aumenta com o nível de ruído, como esperado.

nível de ruído	0.01	0.03	0.05	0.09	0.11	0.13	0.15
diferença de MRE	0.0592	0.3431	0.8665	1.5372	2,2174	2.9455	3.7584

Tabela 4.2: Diferenças de MRE para diferentes níveis de ruído

4.4.3 Complexidade do algoritmo de projeção do gradiente

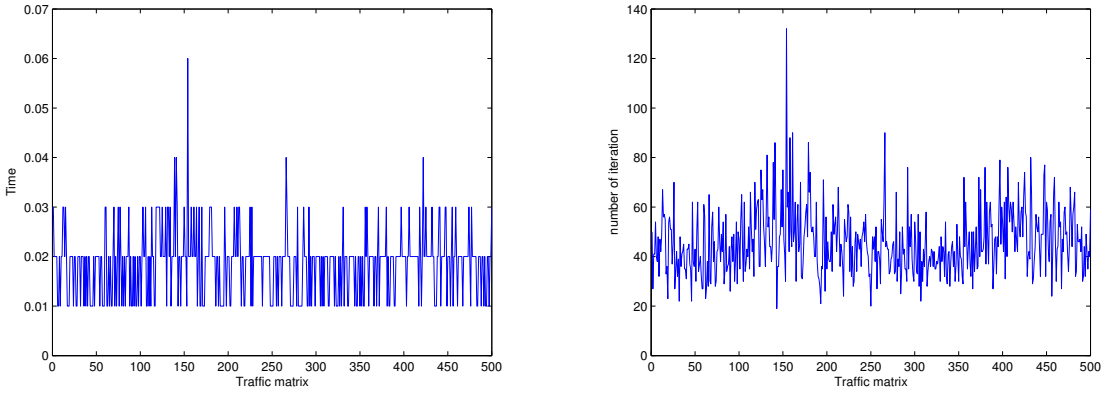
Nesta seção, analisamos a complexidade de tempo e número de iterações do algoritmo de projeção do gradiente para estimação da matriz de tráfego.

O nosso ambiente de experimentos é:

Processador	Pentium IV 3.20GHZ
Sistema operacional	Ubuntu 7.10
Ferramenta de programação	Matlab 7.0.0

Tabela 4.3: Ambiente de experimentos

A figura 4.6(a) mostra o tempo de execução em segundos para rodar o nosso algoritmo 2 resolvendo (4.10) para estimar 500 matrizes de tráfego da rede Abilene. Lembramos que a nossa matriz de tráfego é uma matriz de 144 por 1. O tempo de execução está no intervalo [0.01, 0.06] segundos estando a maioria no intervalo [0.01, 0.03] segundos. Da figura 4.6(b), observamos que na maioria dos casos, o número de iterações do nosso algoritmo está no intervalo [20, 60].



(a) Tempo de execução (b) Número de iterações

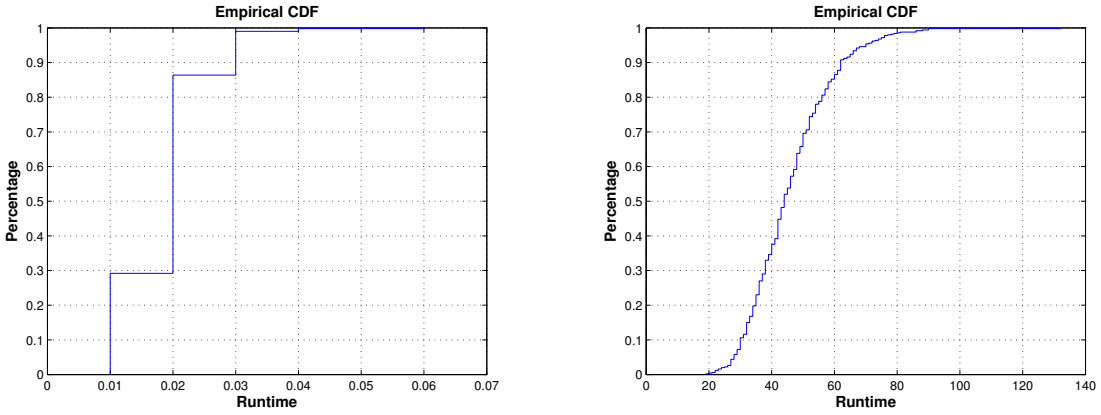
Figura 4.6: Complexidade de tempo e número de iterações do algoritmo proposto

Mais detalhadamente, na figura 4.7 tem-se em (a) a função de distribuição percentual cumulativa empírica do tempo de execução e em (b) a função de distribuição percentual cumulativa empírica do número de iterações. Por exemplo, na figura 4.7(a), podemos ver que cerca de 90% dos casos tem tempo de execução inferior a 0.03 segundos e na figura 4.7(b), cerca de 85% dos casos tem número de iterações inferior a 60.

Comparação com o método de tomogravidade

Lembramos que o método de tomogravidade funciona em três etapas: uma estimativa inicial $\mathbf{x}_g$ é calculada, e, em seguida, o problema de otimização (2.13) é resolvido. Finalmente, o algoritmo IPF é usado para eliminar os elementos negativos da matriz tráfego.

Na segunda etapa, o algoritmo e sua rotina para resolver o problema de otimi-



(a) Tempo de execução (b) Número de iterações

Figura 4.7: Função de distribuição percentual cumulativa empírica do tempo de execução e número de iteração

zação de mínimos quadrados (2.13) são fornecidos em [58]. O algoritmo é muito simples porque a sua rotina inclui apenas seis linhas no Matlab. Segundo Zhang et al. [58], para todas as matrizes que foram estimadas, o tempo de execução do algoritmo em um processador Ultrasparc-II 336 MHz(rodando Solaris 7) foi inferior a 2 segundos.

Na terceira etapa, o algoritmo IPF é usado para eliminar os elementos negativos da matriz tráfego se for necessário. Também é relatado em [58] que o algoritmo IPF leva somente algumas iterações para convergir. Mas a rotina do algoritmo IPF dele não é fornecido em [58].

Os autores consideram o método de tomogravidade como um método rápido.

No método proposto, usamos a matriz de tráfego a priori obtido pelo modelo de gravidade, o tempo de execução para obter a matriz de tráfego a priori é o mesmo que o usado na primeira etapa do método de tomogravidade. Segundo [58], a rotina do algoritmo da segunda etapa do método de tomogravidade é fornecido e a rotina do algoritmo IPF na terceira etapa não é fornecido.

Então, para comparar com o método de tomogravidade, comparamos o tempo de execução da segunda etapa do método de tomogravidade com o do algoritmo proposto. A figura 4.8 mostra a função de distribuição cumulativa empírica do

tempo de execução da segunda etapa do método de tomogravidade e do algoritmo proposto para estimar 500 matrizes de tráfego da rede Abilene. Podemos ver que o tempo de execução da segunda etapa do método de tomogravidade é pouco menor que o do algoritmo proposto, mas os dois são da mesma ordem de grandeza. Mas o método de tomogravidade precisa mais uma etapa que usa o algoritmo IPF para eliminar os elementos negativos caso necessário enquanto o algoritmo proposto já termina. Então, concluímos que o método proposto também é um método rápido.

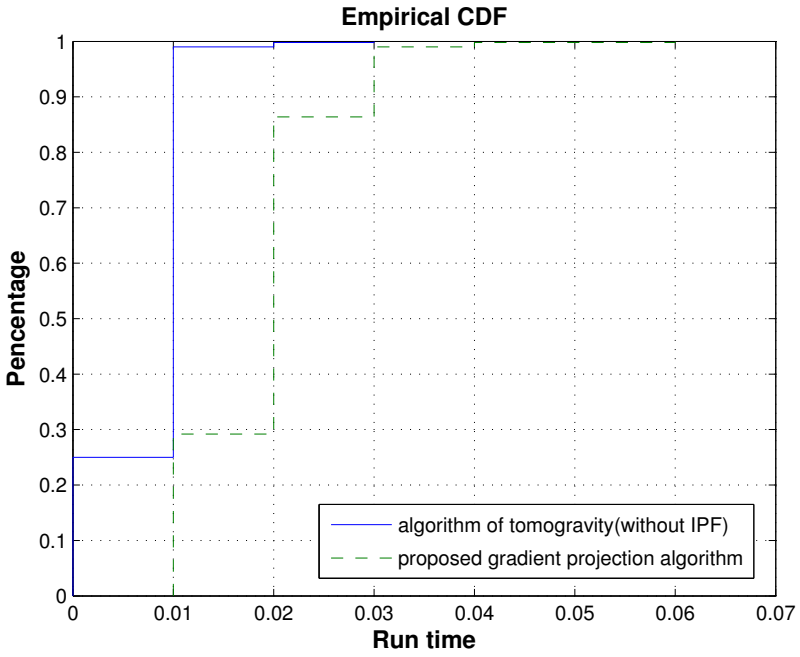


Figura 4.8: Comparação com o algoritmo usado na segunda etapa do método de tomogravidade

Capítulo 5

Matriz de Tráfego para Roteamento Ótimo por Múltiplas Rotas

Neste capítulo, nós tratamos o problema de estimação da matriz de tráfego quando o roteamento empregado é roteamento ótimo por múltiplas rotas. Primeiramente apresentamos o modelo proposto baseado nos fundamentos teóricos apresentados no capítulo 3. A formulação corresponde a um problema de dois níveis: no nível superior, fazemos a estimação da matriz de tráfego usando o método dos mínimos quadrados. No nível inferior, modelamos o roteamento ótimo por múltiplas rotas.

Posteriomente, dois algoritmos são usados para resolver o problema de programação em dois níveis (Figura 5.1). Os resultados são mostrados no final do capítulo.

5.1 O Sistema

Considere uma rede envolvendo $R = |\mathbf{R}|$ nós, e $M = |\mathbf{M}|$ canais, sendo C_a a capacidade do canal a. Um par origem-destino n é um par ordenado $(s, d) \in \mathbf{R} \times \mathbf{R}$ e $N = |\mathbf{N}|$ é o número de todos os pares origem-destino. O grafo $G = (\mathbf{R}, \mathbf{M})$ caracteriza a rede estudada. Um par origem-destino n está relacionado a n_Z rotas,

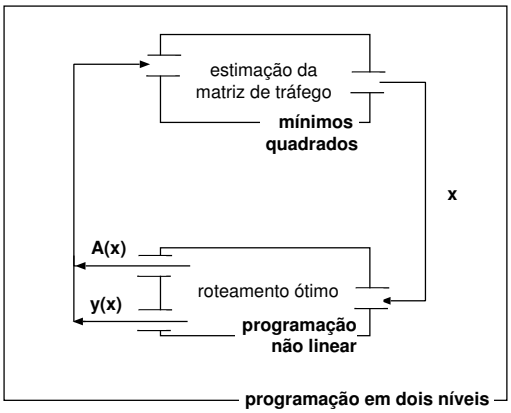


Figura 5.1: O método de estimação da matriz de tráfego proposto.

$n_{\mathbf{Z}} = \{\Gamma_n^1, \ldots, \Gamma_n^{nz}\}$,[1]onde cada rota é um subconjunto de $\mathbf{M}$.[2] A fonte do par origem destino n transmite dados usando a taxa x_n. O roteamento ótimo por múltiplas rotas irá automaticamente determinar as taxas $x_{n,1}, \ldots, x_{n,nZ}$, onde $x_{n,z}$ corresponde ao tráfego injetado pelo par origem destino n na rota Γ_n^z.

A função indicadora δ_a^{ij} é definida da seguinte forma:

$$\delta_a^{ij} = \begin{cases} 1, \text{se a rota } j \text{ do par OD } i \ (\Gamma_i^j) \text{ passa pelo canal } a \\ 0, \text{caso contrário} \end{cases}$$

Seja $\mathbf{A}$ a matriz de roteamento, onde $0 < A_{mn} \leq 1$ se existe uma rota Γ_n^z tal que $m \in \Gamma_n^z$ e $x_{n,z} > 0$, e $A_{mn} = 0$ caso contrário. Mais precisamente,

$$A_{mn} = \frac{\sum_z x_{n,z} \delta_m^{nz}}{x_n} \tag{5.1}$$

Finalmente, $\boldsymbol{C} = (C_1, \ldots, C_M)$, e $\mathbf{x} = (x_1, \ldots, x_N)$.

Tipicamente, o administrador da rede conta com as medições de tráfego feitas em cada um dos canais da rede, ou seja, com um vetor $\hat{\mathbf{y}} \leq \boldsymbol{C}$, ou seja, $y_a \leq C_a, a = 1, ..., M$. Note que $\hat{\mathbf{y}} = \mathbf{y} + \boldsymbol{\eta}$, onde $\boldsymbol{\eta}$ corresponde aos erros de medição, e $\mathbf{y}$ é o vetor que caracteriza o tráfego real passando pelos canais.

[1] Γ_a^bepresenta a b-ésimo rota do a-ésimo par OD

[2] Generalizamos aqui o modelo apresentado no primeiro capítulo.

Exemplo

Na Figura 5.2(a) ilustramos um sistema através de um grafo. Note que o canal entre os nós A e C é unidirecional, enquanto que todos os outros canais são bidirecionais.

Temos um total de 7 canais. Na Figura 5.2(b) ilustramos todos os pares OD. Temos um total de 12 pares origem destino.

Os parâmetros deste exemplo são:

$R = 4$

$M = 7$ $\quad \mathbf{M} = \{$ AB, BA, AC, BC, CB, CD, DC $\}$

$N = 12$ $\quad \mathbf{N} = \{$ AB, AC, AD, BA, BC, BD, CA, CB, CD, DA, DB, DC $\}$

$\Gamma_1^1 =$ AB	$\Gamma_1^2 =$ ACB
$\Gamma_2^1 =$ AC	$\Gamma_2^2 =$ ABC
$\Gamma_3^1 =$ ACD	$\Gamma_3^2 =$ ABCD
$\Gamma_4^1 =$ BA	
$\Gamma_5^1 =$ BC	$\Gamma_5^2 =$ BAC
$\Gamma_6^1 =$ BCD	$\Gamma_6^2 =$ BACD
$\Gamma_7^1 =$ CA	
$\Gamma_8^1 =$ CB	
$\Gamma_9^1 =$ CD	
$\Gamma_{10}^1 =$ DCBA	
$\Gamma_{11}^1 =$ DCB	
$\Gamma_{12}^1 =$ DC	

Por exemplo, Γ_1^1 é a primeira rota do par origem-destino AB. o número do par origem-destino AB é 1.

Devido à topologia do exemplo, existem duas rotas disponíveis para cada um dos pares origem destino 1, 2, 3, 5 e 6. Para cada um dos demais pares OD há apenas uma rota disponível.

5.2 Modelo de Dois Níveis

Ilustramos na Figura 5.3 (uma extensão da Figura 5.1, apresentada no início deste capítulo) o modelo proposto. O administrador do sistema, conhecendo $\hat{\mathbf{y}}$ e tendo em mãos uma estimativa *a priori* para a matriz de tráfego, $\hat{\mathbf{x}}$, pode estimar

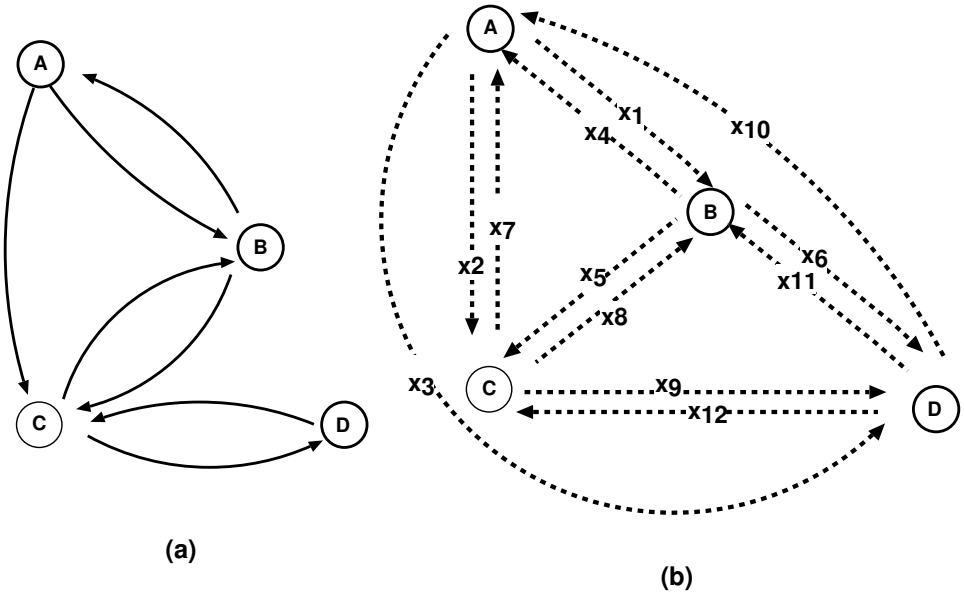


Figura 5.2: Exemplo ilustrativo do sistema considerado

a matriz de roteamento $\mathbf{A}$. O seu objetivo final é obter $\mathbf{x}$, a matriz de tráfego, a partir da matriz de tráfego *a priori* $\hat{\mathbf{x}}$ e do tráfego nos canais $\hat{\mathbf{y}}$ (Figura 5.3).

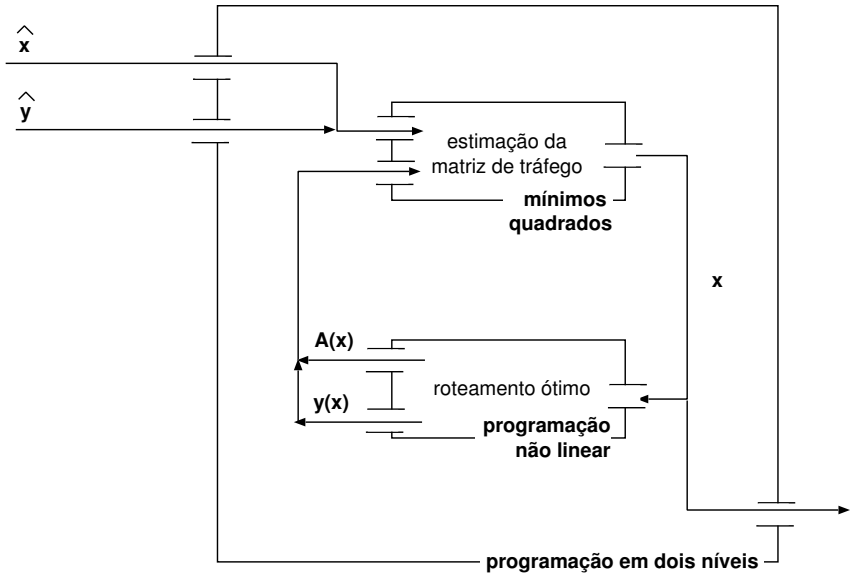


Figura 5.3: Versão detalhada do método de estimação da matriz de tráfego proposto.

Recordando, o esquema proposto é o da programação em dois níveis, onde os níveis são considerados na figura 5.3:

- no nível superior, o problema da estimação da matriz de tráfego $\mathbf{x}$ que corresponde a um problema de mínimos quadrados;

- no nível inferior, o roteamento ótimo por múltiplas rotas que corresponde a um problema de programação não linear.

5.2.1 Nível Superior

No nível superior computamos a estimativa da matriz de tráfego $\mathbf{x}$.

Um dos parâmetros do nível superior é a matriz de tráfego *a priori* $\hat{\mathbf{x}}$. Esta matriz pode ser obtida a partir de estimativas realizadas por exemplo por terceiros ou a partir de modelos como o da gravidade, apresentado na Seção 2.2.2 ou dos dados do tráfego medido NetFlow (seção 2.3.6) ou do modelo de conexão independente (seção 2.3.5). A matriz de tráfego *a priori* $\hat{\mathbf{x}}$ serve como uma aproximação inicial para $\mathbf{x}$. Neste trabalho, a matriz de tráfego *a priori* é obtida pelo modelo de gravidade.

O segundo parâmetro do nível superior é o tráfego medido nos canais, $\hat{\mathbf{y}}$.

Tanto $\hat{\mathbf{x}}$ quanto $\hat{\mathbf{y}}$ envolvem erros, caracterizados respectivamente por $\boldsymbol{\xi}$ e $\boldsymbol{\eta}$. Temos as seguinte equações:

$$\hat{\mathbf{x}} = \mathbf{x} + \boldsymbol{\xi} \tag{5.2}$$

$$\hat{\mathbf{y}} = \mathbf{y} + \boldsymbol{\eta} \tag{5.3}$$

É comum assumir que $\boldsymbol{\xi}$ e $\boldsymbol{\eta}$ têm média zero, ou seja, $E[\boldsymbol{\xi}] = 0$, $E[\boldsymbol{\eta}] = 0$.

Inspirados no método dos mínimos quadrados (Seção 3.3), vamos procurar o vetor $\mathbf{x}$ que minimiza a distância entre os dados medidos e as estimações. Mais precisamente, vamos minimizar a soma dos quadrados dos resíduos $\boldsymbol{\xi}$ e $\boldsymbol{\eta}$. A função objetivo do nível superior é

$$\min_{\mathbf{x}} (\hat{\mathbf{x}} - \mathbf{x})^T (\hat{\mathbf{x}} - \mathbf{x}) + (\hat{\mathbf{y}} - \mathbf{y})^T (\hat{\mathbf{y}} - \mathbf{y}) \tag{5.4}$$

A função objetivo (5.4) é convexa nas variáveis $\mathbf{x}$ e $\mathbf{y}$. Portanto, no nível superior temos um problema de programação convexa.

5.2.2 Nível Inferior

No nível inferior caracterizamos o roteamento ótimo por múltiplas rotas. Este é modelado pelo problema 3.2, que reproduzimos abaixo por conveniência:

$$\begin{aligned} \min_{\mathbf{y},x_{ij}} \quad & \sum_{a\in\mathbf{M}} \frac{y_a}{C_a - y_a} \\ \text{sujeito a} \quad & \sum_{j\in i_\mathbf{Z}} x_{ij} = x_i, \quad i \in \mathbf{N} \\ & \sum_{\substack{i\in\mathbf{N} \\ j\in i_\mathbf{Z}}} x_{ij}\delta_a^{ij} = y_a, \quad a \in \mathbf{M} \\ & x_{ij} \geq 0, \quad i \in \mathbf{N}, j \in i_\mathbf{Z} \\ & 0 \leq y_a < C_a, \quad a \in \mathbf{M} \end{aligned} \tag{5.5}$$

Os parâmetros deste nível são:

- uma estimativa da matriz de tráfego, $\mathbf{x}$;
- a topologia da rede, caracterizada pela função δ;
- a função de custo dos canais.

A saída é o conjunto de elementos x_{ij}, $i \in \mathbf{N}, j \in i_\mathbf{Z}$, além de $\mathbf{y} = (y_1, \cdots, y_a, \cdots, y_M)$.

Assumimos que a função de custo dos canais, relacionando o custo de um canal com o atraso (*delay*) experimentado pelos pacotes que nele trafegam, é [27]:

$$l_a(y_a) = \frac{y_a}{C_a - y_a} \tag{5.6}$$

Esta função de custo é extremamente popular. Um de seus atrativos consiste no fato de que ela é estritamente convexa para $y_a < C_a$. O roteamento ótimo por múltiplas rotas passa a ser um problema de programação convexa [40].

5.2.3 Modelo Proposto

Os problemas tratados no nível superior e inferior (equações (5.4) e (5.5)) são convexos. A fim de combiná-los, usamos um esquema de programação convexa em

dois níveis.

Combinando os dois problemas, a estimação da matriz de tráfego em redes que suporta o roteamento ótimo por múltiplas rotas é feita pelo seguinte conjunto de equações:

$$\begin{aligned} \min_{\mathbf{x},\mathbf{y}} \quad & f(\mathbf{x},\mathbf{y}) = (\hat{\mathbf{x}}-\mathbf{x})^T(\hat{\mathbf{x}}-\mathbf{x}) + (\hat{\mathbf{y}}-\mathbf{y})^T(\hat{\mathbf{y}}-\mathbf{y}) \\ \text{sujeito a} \quad & \mathbf{x} \geq 0 \end{aligned}$$

onde $\mathbf{y}$ é, por sua vez, obtida a partir de

$$\begin{aligned} \min_{\mathbf{y},x_{ij}} \quad & g(\mathbf{x},\mathbf{y}) = \sum_{a\in\mathbf{M}} \frac{y_a}{C_a - y_a} \\ \text{sujeito a} \quad & \sum_{j\in i_\mathbf{Z}} x_{ij} = x_i, \quad i \in \mathbf{N} \\ & \sum_{\substack{i\in\mathbf{N} \\ j\in i_\mathbf{Z}}} x_{ij}\delta_a^{ij} = y_a, \quad a \in \mathbf{M} \\ & x_{ij} \geq 0, \quad i \in \mathbf{N}, \quad j \in i_\mathbf{Z} \\ & 0 \leq y_a < C_a, \quad a \in \mathbf{M} \end{aligned} \tag{5.7}$$

Para maior clareza, é conveniente reescrever o problema da seguinte forma:

$$\begin{aligned} \min_{\mathbf{x},\mathbf{A}} \quad & (\hat{\mathbf{x}}-\mathbf{x})^T(\hat{\mathbf{x}}-\mathbf{x}) + (\hat{\mathbf{y}}-\mathbf{A}\mathbf{x})^T(\hat{\mathbf{y}}-\mathbf{A}\mathbf{x}) \\ \text{sujeito a} \quad & \mathbf{x} \geq 0 \end{aligned}$$

onde $\mathbf{A}$ de elemento A_{ai} é, por sua vez, obtida a partir de

$$\min_{\mathbf{A}} \quad \sum_{a\in\mathbf{M}} l_a(\textstyle\sum_i A_{ai}x_i) = \displaystyle\sum_{a\in\mathbf{M}} \frac{\sum_i A_{ai}x_i}{C_a - \sum_i A_{ai}x_i} \tag{5.8}$$

$$\begin{aligned} \text{sujeito a} \quad & \sum_{j\in i_\mathbf{Z}} x_{ij} = x_i, \quad i \in \mathbf{N} \\ & \sum_{\substack{i\in\mathbf{N} \\ j\in i_\mathbf{Z}}} x_{ij}\delta_a^{ij} = y_a, \quad a \in \mathbf{M} \\ & x_{ij} \geq 0, \quad i \in \mathbf{N}, \quad j \in i_\mathbf{Z} \\ & 0 \leq y_a < C_a, \quad a \in \mathbf{M} \\ \text{onde} \quad & A_{ai} = \frac{\sum_j x_{ij}\delta_a^{i,j}}{x_i}, \quad i \in \mathbf{N}, \quad j \in i_\mathbf{Z} \quad a \in \mathbf{M} \end{aligned}$$

Por conveniência, reproduzimos a seguir parte da tabela de notação:

$\mathbf{N}$	=	conjunto de pares origem-destino
$\mathbf{M}$	=	conjunto de canais
N	=	número de pares origem-destino
M	=	número de canais
x_i	=	tráfego do par origem-destino i, $i = 1, \ldots, N$
x_{ij}	=	tráfego na rota j do par OD i
y_a	=	tráfego no canal a, $a = 1, \ldots, M$
l_a	=	função de custo associada ao canal a
C_a	=	capacidade do canal a
$\mathbf{x}$	=	$(x_1, x_2, \ldots, x_N)^T$ vetor de tráfego entre os pares OD (matriz de tráfego)
$\mathbf{y}$	=	$(y_1, y_2, \ldots, y_M)^T$ tráfego nos canais
$\hat{\mathbf{x}}$	=	$(\hat{x_1}, \hat{x_2}, \ldots, \hat{x_N})^T$ matriz de tráfego *a priori*
$\hat{\mathbf{y}}$	=	$(\hat{y_1}, \hat{y_2}, \ldots, \hat{y_M})^T$ tráfego medido nos canais
δ_a^{ij}	=	1, se a rota j do par OD i passa pelo canal a, e 0 caso contrário

A formulação do problema da estimação da matriz de tráfego pelo conjunto de equações (5.7) é a principal contribuição deste capítulo. As formulações (5.7) e (5.8) são iguais. A primeira segue os moldes do conjunto de equações (3.5), enquanto que a segunda é obtida a partir da primeira empregando-se a relação $\mathbf{y} = \mathbf{Ax}$ (denotamos $\mathbf{A}(\mathbf{x})$ por $\mathbf{A}$ para não sobrecarregar a notação).

O nível superior de (5.8), inspirada no método dos mínimos quadrados, foi descrito na Seção 5.2.1. O nível inferior foi descrito na Seção 5.2.2. As restrições do nível inferior de (5.8) correspondem ao roteamento ótimo por múltiplas rotas (Seção 3.1, equação 3.2), e a equação final exibida em (5.8) é equivalente à (5.1).

É importante observar que o problema do nível inferior tem apenas um solução em $\mathbf{y}$, porque a função objetivo é estritamente convexa no conjunto viável. Um problema de programação em dois níveis é não-convexo porque o seu conjunto viável, dado pelas soluções do problema do nível inferior, é não-convexo. Não-convexidade implica a existência de soluções locais e, por isso, pode ser difícil de encontrar a solução ótima global. .

O Modelo Proposto versus os Modelos Existentes

Conforme mencionado no Capítulo 1 (Figura 1.3), os modelos de estimação da matriz de tráfego pré-existentes na literatura dividem-se em duas classes, em função do esquema de roteamento subjacente ao modelo. Há aqueles que assumem que o mecanismo de roteamento empregado é o roteamento pelo caminho mais curto (e.g. OSPF), enquanto outros, primordialmente advindos da literatura de transportes, assumem usuários egoístas (*selfish routing*).

Os modelos que assumem o roteamento pelo caminho mais curto foram descritos no Capítulo 2. Resta-nos, portanto, descrever as diferenças entre o modelo proposto e aqueles presentes na literatura de transportes.

Tanto o modelo proposto quanto os modelos presentes na literatura de transportes fazem uso da programação em dois níveis. Eles diferem, entretanto, nos seguintes

aspectos:

- os modelos presentes na literatura de engenharia de transportes modelam no nível inferior o *selfish routing*, e procuram um ponto de equilíbrio para o sistema[56]. Mais precisamente, buscam um ponto tal que nenhum usuário tenha incentivos unilaterais para mudar sua estratégia, ou seja, um equilíbrio de Nash, e a partir daí inferem a matriz OD.[3] Já no modelo proposto, modelamos no nível inferior o *roteamento ótimo por múltiplas rotas* , e não buscamos um ponto de equilíbrio, mas sim um ótimo global do sistema;

- os modelos advindos da literatura de engenharia de tráfego consideram que entre cada par origem-destino podem existir múltiplas rotas sendo usadas concomitantemente, mas assumem que isto só ocorrerá caso o custo delas seja igual [47]. Já em cenários envolvendo o *roteamento ótimo por múltiplas rotas* podem existir concomitantemente múltiplas rotas entre cada par origem destino *mesmo que o custo delas seja distinto*, desde que o uso de múltiplas rotas seja necessário para atingir-se o ótimo global do sistema.

5.3 Um Algoritmo Genético para a Estimação da Matriz de Tráfego

Os algoritmos traditionais para resolver a programação em dois níveis têm limitações devido a não-convexidade e não-diferenciabilidade [3]. Como já foi dito no capítulo 3, a maioria dos algoritmos que abordam este problema, acham um solução local, em vez de uma solução ótima global.

A motivação da utilização de algoritmos genéticos para a estimação da matriz de tráfego é devido à sua simplicidade, poucas restrições de problema e paralelismo. Eles trabalham com uma população de solução e não como uma solução única. Além disso, eles são bastante robustos no tratamento de problemas não-convexos e não-diferenciáveis, como o problema de programação em dois níveis.

[3]Na literatura de engenharia de transportes, a matriz de tráfego é chamada de matriz OD.

O algoritmo começa com uma população $\{x_1^{(0)}, \ldots, x_N^{(0)}\}$. No processo iterativo, dado $x_i^{(k)}(k = 0, 1, \ldots)$, $y_i^{(k)}$ é calculado por resolver o problema do nível inferior (problema de roteamento ótimo). A função objetivo superior é avaliada em cada par $(x_i^{(k)}, y_i^{(k)})$ obtendo uma seleção da população, que é usada para gerar uma nova população $x_i^{(k+1)}$. A proposta de algoritmo genético é dada a seguir.

Algoritmo 3 O Algoritmo Genético para Estimação da Matriz de Tráfego

Passo 0 Segundo a recomendação por Goldberg [20], selecionar os valores dos parâmetros: tamanho de população, método de seleção, probabilidade de crossover, probabilidade de mutação e número de geração.

Passo 1. Gerar aleatoriamente uma população inicial de N de $\mathbf{x}_i^{(k)}$. $i \in N$. Definir k = 1.

Passo 2. Resolver o problema de nível inferior (problema de roteamento ótimo por múltiplas rotas) com cada fixo $\mathbf{x}_i^{(k)}$ e encontrar a correspondente variável de nível inferior $\mathbf{y}_i^{(k)}$.

Passo 3. Retornar o problema de nível superior e calcular a função de fitness para cada $\mathbf{x}_i^{(k)}$. com $\mathbf{y}_i^{(k)}$

Passo 4. A seleção por torneio (Tournament Selection) é utilizada para selecionar os indivíduos adequados $\mathbf{x}_i^{(k)}$. Executar a operação de crossover com probabilidade P_c. Executar a operação de mutação com probabilidade P_m. Usar elitismo estratégia: A melhor solução é copiada para a população na próxima geração. Isso é feito para evitar perder a melhor solução encontrada até agora. Depois gerar uma nova população $\mathbf{x}^{(k+1)}$.

Passo 4. Se k = número máximo de gerações, o algoritmo termina. Senão, $k = k+1$. Retornar para o Passo 2.

5.4 Resultados Computacionais

A topologia que consideramos aqui também é aquela de 4 nós, mostrada na figura 5.4. O mecanismo de roteamento é o roteamento ótimo por múltiplas rotas.

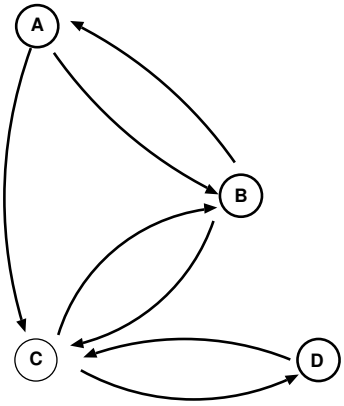


Figura 5.4: Topologia de 4 nós

Geramos a matriz de tráfego sintética de distribuição Poisson. É assumido que a capacidade de cada canal é de 2500 unidades. Conforme descrito anteriormente, a matriz de tráfego a priori também é gerada usando o modelo de gravidade. O tráfego nos canais é obtido por resolver o problema de roteamento ótimo por múltiplas rotas. Simulamos o ruído do tráfego nos canais para obter o tráfego nos canais medido.

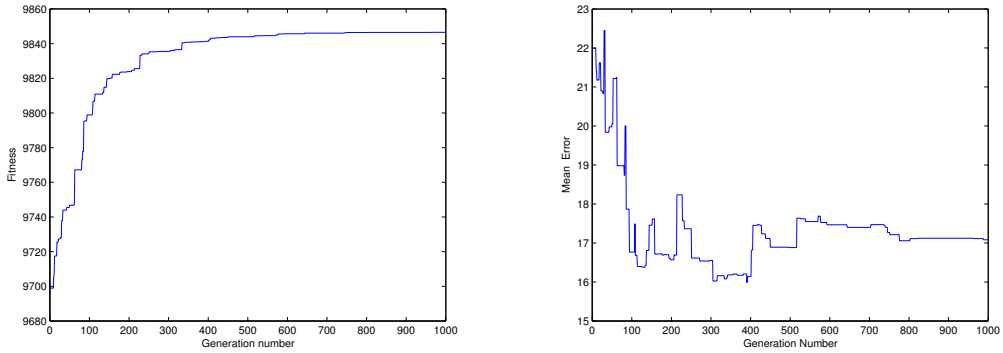
Os valores dos parâmetros são escolhindo seguindo a recomendação de Goldberg. [20]

- tamanho de população = 50
- método de seleção = Torneio
- probabilidade de crossover = 0.6
- probabilidade de mutação = 0.033

O número de iterações é escolhido com 1000 neste experimento.

Lembra-se que a função de aptidão representa a opotunidade de reprodução. O indivíduo que tem o valor mais elevado da função de aptidão tem maior opotunidade de reprodução. Então neste problema, a relação entre a função objetivo e a função de fitness é definida como, $f(x) = C_{max} - O(x)$, onde $f(x)$ é a função de fitness, e $O(x)$ é o valor da função objetivo do problema de nível superior. C_{max} é escolhido

como $10,000$ neste experimento para garantir o valor da $f(x) = C_{max} - O(x)$ seja positivo.



(a) Fitness (b) Mean Error

Figura 5.5: Algoritmo Genético para Estimação da Matriz de Tráfego

A convergência do algoritmo é mostrada na figura 5.5(a). Pode ser visto que o algoritmo obtém bons resultados rapidamente nas primeiras 200 gerações. Em seguida, ela começa a convergir lentamente. Após 1000 gerações, o fitness da solução ótima é 9483, que aproxima o fitness da solução ótima ideal , 10000. Portanto, podemos aproveitar esta solução sub-ótima como a solução final do problema de estimação da matriz de tráfego. .

A Figura 5.5(b) mostra o erro médio da solução em cada geração. Podemos observar que o menor valor da função objetivo do nível superior, obtido pelo algoritmo genético, não implica necessariamente uma melhor estimativa da matriz de tráfego. Isto pode ser explicado pelo erro na estimativa da matriz de tráfego, que depende não somente do algoritmo genético, mas também da função objetivo do nível superior e da matriz de tráfego a priori. Mais importante ainda, a função objetivo do nível superior (5.4), que minimiza a distância entre a matriz de tráfego estimada e a matriz de tráfego a priori, é uma aproximação para alcançar a matriz de tráfego real. Obviamente, um valor menor da função objetivo (5.4) significa que a matriz de tráfego estimada aproxima-se da matriz de tráfego a priori, mas pode não ser próxima da matriz de tráfego real.

A comparação dos elementos da matriz de tráfego real e da matriz de tráfego estimada é mostrada na figura 5.6. A linha sólida diagonal mostra igualdade, enquanto

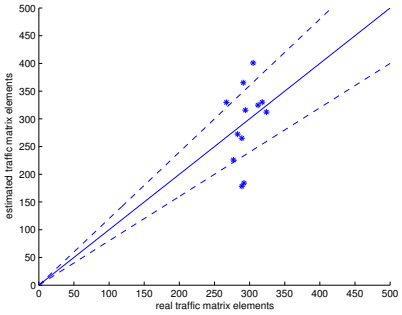


Figura 5.6: Comparação da matriz de tráfego real com a matriz de tráfego estimada

que a linha traçada mostra ±20% erros. Podemos ver que, entre os 12 elementos da matriz de tráfego(12 pontos na figura 5.6), os erros de 7 elementos são menores de 20%. Os erros dos outros 3 elementos estão próximos de 20%. Concluímos então que o algoritmo funciona bem para a estimação de matriz de tráfego.

5.5 Complexidade

5.5.1 Complexidade do algoritmo genético proposto

Nesta seção, analisamos a complexidade do algoritmo genético para estimação da matriz de tráfego. O tempo total T pelo algoritmo genético é,

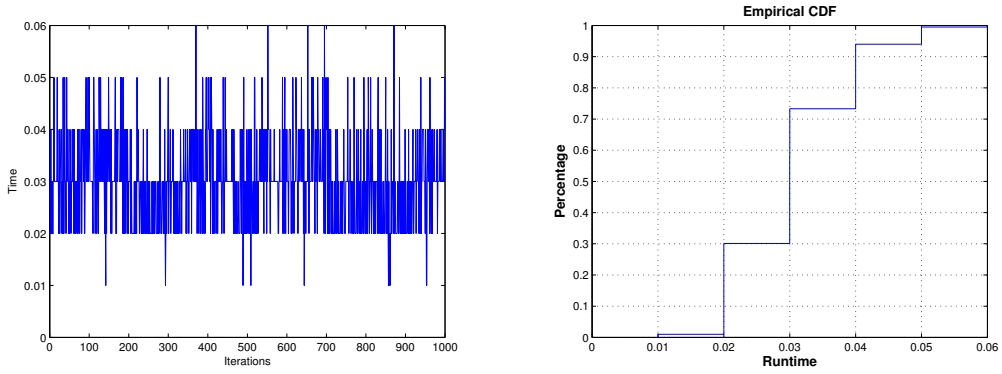
$$T = N_i * (N * T_{op} + T_{ob} + T_{go}) \tag{5.9}$$

onde N_i é o número de gerações e N é o tamanho de população que são definidos no algoritmo; T_{op} é o tempo levado pelo problema de roteamento ótimo; T_{ob} é o tempo levado no cálculo da função objetivo do nível superior e T_{go} é o tempo levado pelas operações genéticas (seleção, cruzamento, mutação e elitismo).

Agora consideramos o tempo levado pelo problema de roteamento ótimo T_{op}. Usamos a função *fmincon* do MATLAB[4] para resolver o problema de roteamento

[4]MATLAB é um "software"interativo de alta performance voltado para o cálculo numérico.

ótimo. Este procedimento usa o método da programação quadrática sequencial (SQP). Neste método, a função resolve um sub-problema de programação quadrática a cada iteração. Uma estimativa da Hessiana da Lagrangiana é renovada a cada iteração usando a fórmula Broyden-Fletcher-Goldfarb-Shanno (BFGS). É considerada uma busca linear usando uma função de mérito.[35]



(a) Tempo de execução (b) Percentagem

Figura 5.7: Tempo de execução do roteamento ótimo

A figura 5.7(a) mostra o tempo de execução em segundos em um computador de Pentium IV 3.6GHz para resolver o roteamento ótimo para 1000 casos. A nossa matriz de tráfego é uma matriz de 12 por 1. O tempo de execução está no intervalo [0.02, 0.06] segundos estando a maioria dos casos no intervalo [0.02, 0.04] segundos. Mais detalhadamente, a figura 5.7(b) mostra a função de distribuição percentual cumulativa empírica do tempo de execução. Por exemplo, na figura 4.6(b) podemos ver que cerca de 92% dos casos tem tempo de execução inferior a 0.04 segundos.

Comparado com T_{ob} e T_{go}, T_{op} é muito maior do que os outros dois porque T_{op} é o tempo levado para resolver o problema de roteamento ótimo.

$$T_{op} \gg T_{ob}, T_{go} \tag{5.10}$$

Então, o tempo levado pelo algoritmo genético é dado por

$$T \approx N_i * N * T_{op} \tag{5.11}$$

Finalmente, concluímos que o tempo levado pelo algoritmo genético é proporcional ao número de geração N_i, o número de população N e o tempo levado para resolver o problema de roteamento ótimo T_{op}.

5.5.2 Vantagens e desvantagens do algoritmo proposto

Na última seção, estudamos a complexidade do tempo do algoritmo genético. Agora, mostramos as vantagens e desvantagens do algoritmo proposto e a comparação com os algoritmos tradicionais para a programação de dois níveis. A Tabela 5.1 mostra que o algoritmo genético pode encontrar uma solução sub-ótima. O custo de tempo é alto porque em cada iteração ele resolve o problema de roteamento ótimo para cada indivíduo de população.

	custo de tempo	ótimo global	ótimo local
algoritmo genético	alto	sub-ótimo global	não
algoritmos tradicionais(caso geral)	-	não	sim

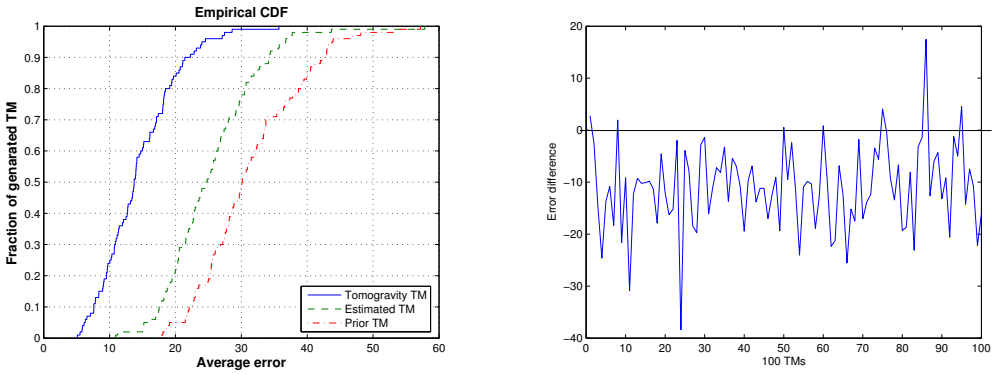
Tabela 5.1: Comparação dos algoritmos

5.6 Aplicação do método proposto às redes que suportam o roteamento pelo caminho mais curto

Sabemos que hoje em dia a Internet não suporta o roteamento ótimo por múltiplas rotas. A maioria dos ambientes de redes são baseados no roteamento pelo caminho mais curto. Nesta seção, queremos saber o que vai acontecer se usarmos o nosso método para estimar a matriz de tráfego nas redes que suportam o roteamento pelo caminho mais curto. Lembremos que o nosso método é baseado no roteamento ótimo por múltiplas rotas.

Os experimentos feitos são baseados na topologia de 4 nós (figura 5.4). O roteamento é assumido como o roteamento pelo caminho mais curto. 100 matrizes

de tráfego sintéticas são geradas por distribuição uniforme. A matriz de tráfego a priori é obtida pelo modelo de gravidade.



(a) (b)

Figura 5.8: Comparação entre o método de tomogravidade e o nosso método nos ambientes que suportam o roteamento pelo caminho mais curto

A Figura 5.8(a) mostra a função de distribuição cumulativa da matriz de tráfego para o método de tomogravidade e para o nosso método. Podemos observar que o método de tomogravidade tem performance melhor do que o nosso método. A razão é simples, o método de tomogravidade é destinado a funcionar nos ambientes que suportam o roteamento pelo caminho mais curto, enquanto que o nosso método é baseado nos ambientes que suportam o roteamento ótimo por múltiplas rotas. Também podemos observar que os erros da matriz de tráfego obtida pelo nosso método são menores que os da matriz de tráfego a priori.

A Figura 5.8(b) mostra a diferença entre o erro médio relativo do método de tomogravidade e de nosso método para 100 matrizes de tráfego. Na maioria dos casos (93 de 100), o nosso método tem uma maior erro médio relativo Porém, em alguns casos (7 de 100), o nosso método tem uma menor erro médio relativo. Se o desempenho de dois métodos são mais ou menos iguais quando a diferença de erro é maior do que -5%, então, o número dos casos é de 22.

As conclusões são as seguintes:

- O desempenho do método de tomogravidade é melhor do que o nosso método, pois ele é baseado no roteamento pelo caminho mais curto;

- Os erros da matriz de tráfego estimada com o nosso método, são menores do que os da matriz de tráfego a priori.

- O método de tomogravidade, na maioria dos casos, tem uma MRE menor do que o nosso método. Mas, em alguns casos específicos, o nosso método tem uma menor erro médio relativo ou desempenho semelhante. estes casos podem ser os casos que a maioria dos tráfegos roteam pela uma única rota.

Capítulo 6

Conclusão e Trabalhos Futuros

Neste capítulo, apresentamos um sumário dos resultados obtidos. Além disso, discutimos algumas das possíveis melhorias e extensões.

6.1 Conclusão

Neste trabalho, realizamos estudos visando estimar a matriz de tráfego em uma rede de telecomunicação. A primeira contribuição é que descobrimos a relação entre problema de estimação da matriz de tráfego e os roteamentos da rede. Descobrimos que o mecanismo de roteamento sendo adotado na rede é o fator crucial para a estimação da matriz de tráfego. As abordagens tradicionais para a estimação da matriz de tráfego são baseadas em que o protocolo de roteamento adotado é um protocolo de roteamento pelo caminho mais curto tradicional, como o OSPF. Este protocolo de roteamento é o mais usado atualmente na Internet. Os outros protocolos de roteamento, dentre os quais destaca-se o roteamento ótimo por múltiplas rotas, diferem dos protocolos de roteamento pelo caminho mais curto tradicionais no objetivo de distribuir o tráfego na rede, de tal modo que o custo total imposto aos usuários seja minimizado.

Consideramos o problema de estimação da matriz de tráfego quando os roteamentos, sendo adotados na rede, são de dois tipos:

- roteamento pelo caminho mais curto;
- roteamento ótimo por múltiplas rotas;

Roteamento pelo caminho mais curto

Propomos uma abordagem que formula o problema da estimação da matriz de tráfego como um problema de otimização com restrições de não-negatividade. Um método da projeção é usado para resolvê-lo. Experimentos são feitos com dados sintéticos e dados reais obtidos na Rede Abilene. Mostramos que nossa abordagem estima a matriz de tráfego mais precisamente do que o método de tomogravidade (*tomogravity*).

Roteamento ótimo por múltiplas rotas

Nesta tese propomos um modelo original para estimar a matriz de tráfego, quando o mecanismo de roteamento empregado é o roteamento ótimo por múltiplas rotas. Propomos um modelo que pode ser usado para estimar a matriz de tráfego quando o mecanismo de roteamento empregado é o roteamento ótimo por múltiplas rotas. Formulamos o problema como um problema de programação em dois níveis. No nível superior, empregamos o método dos mínimos quadrados para estimar a matriz de tráfego a partir das informações advindas do nível inferior; no nível inferior, modelamos o roteamento ótimo por múltiplas rotas. Um algoritmo genético foi proposto para resolver o problema. O algoritmo genético começa com uma população aleatoriamente gerada. No processo iterativo do algoritmo genético, após resolver o problema do nível inferior (problema de roteamento ótimo), a função objetivo superior é avaliada. Após usar as operações genéticas (seleção, cruzamento, mutação), a nova população é gerada. A solução vai ser encontrada quando o algoritmo converge. Numericamente, avaliamos o desempenho dos dois algoritmos em uma rede envolvendo quatro nós (a mesma topologia foi usada por Medina et al. [37]). Alguns resultados foram obtidos e verificamos também que a estimativa depende fortemente da matriz de tráfego *a priori*.

Vamos comparar com a função objetivo a seguir

$$(\hat{\mathbf{y}} - \mathbf{A}\mathbf{x})^T(\hat{\mathbf{y}} - \mathbf{A}\mathbf{x}) \tag{6.1}$$

A segunda alterativa é considerar uma generalização da função objetivo. Esta função objetivo consiste em considerar matrizes de peso $\mathbf{U}$ e $\mathbf{V}$:

$$(\hat{\mathbf{x}} - \mathbf{x})^T\mathbf{U}^{-1}(\hat{\mathbf{x}} - \mathbf{x}) + (\hat{\mathbf{y}} - \mathbf{y})^T\mathbf{V}^{-1}(\hat{\mathbf{y}} - \mathbf{y}) \tag{6.2}$$

Outra alterativa é considerar a função objetivo do problema linear considerado na seção 2.2.1, proposto em [21] e dada por

$$\sum_{j=1}^{N} w_j x_j \tag{6.3}$$

onde w_j é um peso para o par OD j, e implica em maximizar a carga de tráfego que a rede suporta. Duas estratégias de definição dos pesos freqüentemente utilizadas na literatura são: (i) pesos unitários, os quais não atribuem qualquer prioridade entre os fluxos, e (ii) pesos baseados na distância da origem ao destino medida em número de saltos.

Note que as novas funções objetivo (6.1) e (6.3) não dependem da matriz de tráfego *a priori* $\hat{\mathbf{x}}$. Um trabalho futuro consiste em estudar o modelo com estas novas funções objetivos no nível superior.

6.1.1 Algoritmos para resolver programação em dois níveis

Lembramos que os métodos tradicionais para resolver modelos de programação em dois níveis (equação (3.5)) baseiam-se na reformulação do problema, transformando-o em um com um único nível. Para tal, incorpora-se informações do nível inferior no nível superior. Em particular, quando o nível inferior é convexo, substitui-se o nível inferior pelas suas condições KKT(Karash-Kuhn-Tucker) [55]. Obtém-se então um problema da seguinte forma:

Sejam $\mathbf{x} \in \Re^{n_1}$, $\mathbf{y} \in \Re^{n_2}$.

$$\begin{array}{lll} \min\limits_{\mathbf{x},\mathbf{y}} & F(\mathbf{x},\mathbf{y}) & (6.4) \\ \text{sujeita a} & G(\mathbf{x},\mathbf{y}) \leq 0 & \end{array}$$

e condições KKT adicionais impostas pelo nível inferior

Lembramos que no nosso modelo proposto (5.7), o problema do nível inferior é estritamente convexo. Por isso, o problema do nível inferior tem uma única solução.

Além de usar o algoritmo genético, pesquisaremos outros algoritmos que resolvem problema de dois níveis baseados na reformulação (6.4).

Apêndice A

Código Fonte

Neste apêndice apresentamos a listagem do programa MATLAB utilizado nesta tese. As rotinas são rotinas de três algoritmos propostos nesta tese: (1) algoritmo de projeção do gradiente, (2) algoritmo iterativo proposto na qualificação e (3) algoritmo genético. Começamos apresentando a rotina do algoritmo de projeção do gradiente para estimar a matriz de tráfego quando o roteamento é roteamento pelo caminho mais curto.

A.1 Algoritmo de Projeção do Gradiente

A.1.1 Código Principal

```
% Traffic Matrix Estimation: Main Code of Gradient Projection Method
%
% Input:
% A:  the routing matrix
% x:  real traffic data
% y:  link traffic data with errors
% xg:  the prior traffic matrix obtained by gravity model
%
% Output:
```

```
% t6:  the estimated traffic matrix obtained by tomograticvy method
% t11:  the estimated traffic matrix obtained by gradiente projection method
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

clear all;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% real traffic data and routing matrix of Abilene Network

load X19
load A2.txt

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
A = zeros(54,144);

[m,n] = size(A2);

for k = 1 : m
    i = A2(k,1);
    j = A2(k,2);
    A(i,j) = A2(k,3);
end;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

realx = zeros(144,1)
simplegravity = zeros(144,1)
simpletomogravity = zeros(144,1);
generalgravity = zeros(144,1);
generaltomogravity = zeros(144  ,1);

threshold = 5000000;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

for time = 1 : 500  % number of experiments
```

```
% real TM and Gravity TM

for i =  1 : 144
    realx(i) = X19(time, (i-1)*5+1 );
    simplegravity(i) = X19(time, (i-1)*5+2 );
    simpletomogravity(i) = X19(time, (i-1)*5+3 );
    generalgravity(i) = X19(time, (i-1)*5+4 );
    generaltomogravity(i) = X19(time, (i-1)*5+5 );
 end;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% real link traffic and its errors
index = 1;

for deviation = 0: 0.15 :0

x = A* realx;

x = x + x .* normrnd(0,deviation,54,1);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

option = 1   % for diferent experiments

deviation2 = 0 % noise level for diferent experiments

switch option

case  1
    tg = simplegravity  %simple gravity model
case  2
    tg = generalgravity %generalize gravity model
case  3
    tg = realx + normrnd(0,deviation2,144,1) %random model
end;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
% save data for Matlab funcion
y =x;
assignin('base','tg', tg);
assignin('base','A', A);
assignin('base','y', y);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

w = 1  % weight

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% Least Square Method of Tomogravity
t = wlse(A,x, tg,w)
t1(:,time)= t;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% IPF of Tomogravity Method
[iter,t6] = ipf(A, t, x)
iteration(:,time)  = iter;
t62(:,time) =t6;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% Gradient Projection Method
t11= grad2(ones(144,1),tg, y, A, @objfun3);
t112(:,time) = t11;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% Error Calulation
average_tg = calulaterror(realx, tg, threshold);
averagetg(time) =average_tg;

[average_t,error_t] =  calulaterror(realx, t1(:,time), threshold);
averaget(time) =average_t;
```

```
errort(:,time) = error_t;

[average_t6,error_t6] = calulaterror(realx, t62(:,time), threshold);
averaget6(time) =average_t6;
errort6(:,time) = error_t6;

[average_t11,error_t11] = calulaterror(realx, t112(:,time), threshold);
averaget11(time) =average_t11;
errort11(:,time) = error_t11;


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

index = index +1;

end; % end of deviation
end; % end of time

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

A.1.2 Código Fonte do Algoritmo de Projeção do Gradiente

```
function [x] = grad2(x0,xg,y,A,f)

xc = x0;  %initialization

[fc,gc] = feval(f,xc);

gc2 = -gc; % Gradient

fc2 = fc;

num = 1;

while ( fc2 <= fc & norm(gc2) >= 1  & ( num <= 10000 ) ) % Stop Criterion
```

```
[fc,gc] = feval(f,xc);

gc2 = -gc;

dao = - (gc2'*( xc- xg) + (A*gc2)'*(A*xc- y))/ (gc2'*gc2 + (A*gc2)'*(A*gc2));

xc = max(xc+ dao*gc2, 0);

[fc2,gc] = feval(f,xc);

gc2 = -gc;

num = num + 1;

end

x =  xc; % Return function value
```

A.1.3 Código Fonte do Método Tomogravity

```
% weighted least-squares estimate of the TM
% Input:
% A: matrix A in constraints x=A*t
% x: vector x in constraints x=A*t
% tg: initial gravity model solution
% w: weight vector
% Output:
% t: estimated traffic matrix (as a vector)
% that minimizes |(t-tg)./w|
% among all t that minimize |A*t-x|

function [t] = wlse(A,x,tg,w)

% equivalently transform x=A*t into
% xw=Aw*tw, where tw=(t-tg)./w
xw = x - A*tg;
```

```
[r, c] = size(A);
repmat(w', r, 1);
Aw = A .* w;

% solve tw=Aw*tw by computing the pseudo-
% inverse of matrix Aw (through svd)
tw = pinv(full(Aw)) * xw;

% transform tw back to t
t = tg + w .* tw;
```

A.1.4 Código do Ajuste Proporcional Iterativo

```
function [iter,x] =ipf(A,u,Y)

iteration = 10000;
error = 0.001;

[ J, I ] = size(A);

for  j = 1 : I

    if u( j ) < 0

        u(j) = 1;

    end

end


initial = u;

for k = 1 : iteration
```

```
    old = u ;

    for j = 1 : J

        alpha = A( j , :)* u / Y(j);

        for i = 1 : I
            if A(j , i) > 0
            u(i) = u(i)/alpha;
            end;

        end;

    end ;

    if norm ( u - old)  < error
    break;
    end;

end

iter = k

%initial

x = u;
```

A.2 Algoritmo Iterativo

A.2.1 Código Principal

```
% Traffic Matrix Estimation: A New Approach Using Multipath Optimal Routing
%
% Input:
%   initialx:   A priori traffic matrix
```

```
%   initialy:   Traffic measured in the links
%   realx:  The 'real'  traffic matrix
%   C:      Link capacity
% Output:
%   inputx: The estimated traffic matrix
%   outputx:    The estimated traffic in each  path
%   A:      Routing matrix


global initialx initialy C A;


% ********   Initialization  **********************
disp('*****************************************************************');

C=2500;
eval = [0;0;0;0;0;0;0;0;0;0;0;0];
realx =[318;289;312;294;292;267;305;289;324;283;277;291];
initialx=[300;300;300;300;300;300;300;300;300;300;300;300];
initialx2=[400;400;400;400;400;400;400;400;400;400;400;400];
initialy=[0;0;0;0;0;0;0];
noise = 0.01;
yold = or2(realx);
initialy(1)=yold(1)+yold(4)+yold(17);
initialy(2)=yold(2)+yold(3)+yold(5);
initialy(3)=yold(6)+yold(8)+yold(10)+yold(11)+yold(14);
initialy(4)=yold(4)+yold(7)+yold(9)+yold(17);
initialy(5)=yold(2)+yold(11)+yold(12)+yold(14)+yold(15);
initialy(6)=yold(5)+yold(9)+yold(10)+yold(13)+yold(17);
initialy(7)=yold(14)+yold(15)+yold(16);
initialy;


randy=rand(7,1) % Noise
initialy= initialy+noise*initialy.*randy


inputx =realx;           % Experiment 1
```

```
%inputx = initialx       % Experiment 2
%inputx = initialx2      % Experiment 3

% **********Main Iteration ***************

for i = 1: 100

    outputx=or2(inputx)  %  Lower level (optimal routing)


    % Computing the Routing Matrix ...

    A=[outputx(1)/inputx(1),outputx(4)/inputx(2),
       outputx(17)/inputx(3),0,0,0,0,0,0,0,0,0;...
       outputx(2)/inputx(1),outputx(3)/inputx(2),
       outputx(5)/inputx(3),0,0,0,0,0,0,0,0,0;...
       0,0,0,outputx(6)/inputx(4),outputx(8)/inputx(5),
         outputx(10)/inputx(6),outputx(11)/inputx(7),0,
         0,outputx(14)/inputx(10),0,0;...
       0,outputx(4)/inputx(2),outputx(17)/inputx(3),
       0,0,outputx(7)/inputx(5),outputx(9)/inputx(6),0,0,0,0,0;...
         outputx(2)/inputx(1),0,0,0,0,0,
         outputx(11)/inputx(7),outputx(12)/inputx(8),
       0,outputx(14)/inputx(10),outputx(15)/inputx(11),0;...
       0,0,(outputx(5)+outputx(17))/inputx(3),0,
       0,(outputx(9)+outputx(10))/inputx(6),
         0,0,outputx(13)/inputx(9),0,0,0;...
         0,0,0,0,0,0,0,0,0,outputx(14)/inputx(10),
       outputx(15)/inputx(11),outputx(16)/inputx(12)]


    inputx=wls2(inputx);   % Upper Level (weighted least square)

    if ( max(abs(eval-inputx))< 0.01)  % Computing the error
        display(i);
        break;
        else
        display(i);
```

```
        eval= inputx;
    end  % end if

end  % end for
```

A.2.2 Código Fonte do Nível Superior

```
% Upper level Problem ---------- Least square estimation
% Input:
%   inputx: Old traffic matrix
%   initialy:   Traffic measured in the links
%   A:      Routing matrix
% Output:
%      xnew:     New traffic matrix

function [xnew]= wls2(inputx)
global A initialy
xnew = inv(eye(12)+A'*A)*(inputx + A'*initialy); % equation (4.15)
disp('function wls2 ends here');
```

A.2.3 Código Fonte Referente ao Nível Inferior

```
% Lower nivel Problem -- Multipath Optimal Routing
% Input:
%     inputx. Routing matrix
% Output;
%     output: Traffic in the paths



function [outputx] = or2(inputx)

% The fmincon is a Matlab function to solve an optimization problem,
% with constraints given by a system Ax = b.
% The parameters for fmincon are:
```

```
% — objective function
% — initial point
% — not relevant
% — not relevant
% — constraints (A)
% — constraints (x)
% — constraints (b)
% — not relevant

[outputx, fval] = fmincon(@objfun2,...
        [300;300; 100; 100; 100; 100; 100; 100; 100;...
         100; 100; 100; 100; 100; 100; 100;100],[],[],...
            [1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0;...
             0 0 1 1 0 0 0 0 0 0 0 0 0 0 0 0;...
             0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 1;...
             0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0;...
             0 0 0 0 0 0 1 1 0 0 0 0 0 0 0 0;...
             0 0 0 0 0 0 0 0 1 1 0 0 0 0 0 0;...
             0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0;...
             0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0;...
             0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0;...
             0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0;...
             0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0;...
             0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0],...
         inputx,[0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0],[]);

fval % shows in the screen the final value of the optimization

disp('function optimal routing ends here');
```

A.2.4 Código Fonte da Função Objetivo do Nível Inferior

A função objetivo é dada por:

```
% Objective function for the multipath optimal routing
```

```
% Input:
%     x: Traffic in the paths
%     C: Capacity  of the links
% Output:
%     f: Objective function for the multipath optimal routing (avg delay)

function f = objfun2(x)
global C
f = (x(1)+x(4)+x(17))/(C-x(1)-x(4)-x(17)) +...
    (x(2)+x(3)+x(5))/(C-x(2)-x(3)-x(5)) +...
    (x(6)+x(8)+x(10)+x(11)+x(14))/(C-x(6)-x(8)-x(10)-x(11)-x(14))+...
    (x(4)+x(7)+x(9)+x(17))/(C-x(4)-x(7)-x(9)-x(17)) +...
    (x(2)+x(11)+x(12)+x(14)+x(15))/(C-x(2)-x(11)-x(12)-x(14)-x(15))+...
    (x(5)+x(9)+x(10)+x(13)+x(17))/(C-x(5)-x(9)-x(10)-x(13)-x(17)) +...
    (x(14)+x(15)+x(16))/(C-x(14)-x(15)-x(16));
```

A.3 Algoritmo Genético

A.3.1 Código Principal

```
% Traffic matrix estimation in optimal routing networks: Genetic Algorithm
%
% Input:
%   initialx:   A priori overlay traffic matrix
%   initialy:   Traffic measured in the links
%   realx:      The 'real'  overlay traffic matrix
%     realy:      The 'real'  link traffic data
%   C:          Link capacity
% Output:
%   inputx:     The estimated overlay traffic matrix
%   outputx:    The estimated overlay traffic in each path
%   A:          Overlay routing matrix
```

```
clear all

global initialx initialy realy C ;
global Debug;
global TestValue;
global Xupperbound Xlowerbound;
global FunctionValue;

t = cputime;

Debug = 0;
TestValue = 0;
FunctionValue = 2 ;
TopologyValue = 1 ;

% ******************** Initialization *********************************
disp('*****************************************************************');
disp('*****************************************************************');
disp('*****************************************************************');
disp('*****************************************************************');
disp('*****************************************************************');

C=2500;

eval = [0;0;0;0;0;0;0;0;0;0;0;0];

realxValue = 5

switch realxValue
   case 0 %Constant TM
   realx =[ 318; 289; 312; 294; 292; 267; 305; 289; 324; 283; 277; 291];

   case 1 %Uniform TM
   realx = unifrnd(100,500,12,1);

   case 2
   % To genarate real traffic matrix using bimodal model
   Randomx11 =  normrnd( 150, 20, 12, 1 );
```

```
   Randomx12 =  normrnd( 400, 20, 12, 1 );

   % Resets the generator to a different state each time
   rand('state',sum(100*clock));

   for j = 1 : 12

     Temp = rand(1);
     if Temp < 0.8
        realx( j ) = Randomx11( j );
     else
        realx( j ) = Randomx12( j );
     end
   end

   realx = realx';

   case 3 %Normal TM

   mu = unifrnd(100,500,1,1);

   realx = normrnd(mu,40,12,1);

   case 4 %Poisson TM

   mu = unifrnd(100,500,1,1);
   realx = poissrnd(mu, 12, 1);

   case 5  %Constant TM

   realx = [ 300; 300; 300; 300; 300; 300; 300; 300; 300; 300; 300; 300];

end



initialy=[0;0;0;0;0;0;0];
```

```
noiselevel = 0.01;



xrota  = or2(realx);

% underlayx is AB, AC, AD, BA, BC, BD, CA, CB, CD, DA, DB, DC.
% the same topology with underlay
% It will be changed when the topology is changed

realy(1) = xrota(1) + xrota(4) + xrota(17);
realy(2) = xrota(2) + xrota(3) + xrota(5);
realy(3) = xrota(6) + xrota(8) + xrota(10) +...
               xrota(11) + xrota(14);
realy(4) = xrota(4) + xrota(7) + xrota(9) +...
               xrota(17);
realy(5) = xrota(2) + xrota(11) + xrota(12) +...
               xrota(14) + xrota(15);
realy(6) = xrota(5) + xrota(9) + xrota(10) +...
               xrota(13) + xrota(17);
realy(7) = xrota(14) + xrota(15) + xrota(16);



realy =  transpose(realy);


Ain =  realy(3);
Aout = realy(1) + realy(2);

Bin =  realy(1) + realy(5);
Bout = realy(3) + realy(4);

Cin =  realy(2) + realy(4) + realy(7);
Cout = realy(5) + realy(6);;

Din  = realy(6);
Dout = realy(7);
```

```
In = Ain + Bin + Cin + Din;
Out = Aout + Bout + Cout + Dout;

total = In;

% To Genarte Gravity TM
tg(1) = Ain *  Bout/total;
tg(2) = Ain *  Cout/total;
tg(3) = Ain *  Dout/total;

tg(4) = Bin *  Aout/total;
tg(5) = Bin *  Cout/total;
tg(6) = Bin *  Dout/total;

tg(7) = Cin *  Aout/total;
tg(8) = Cin *  Bout/total;
tg(9) = Cin *  Dout/total;

tg(10) = Din *  Aout/total;
tg(11) = Din *  Bout/total;
tg(12) = Din *  Cout/total;


option = 2
deviation2 = 100

if (option == 1)
    initialx = tg'
else
    initialx = realx + normrnd(0,deviation2,12,1);
end


% Noise
randy = rand(7,1); % Noise
initialy = realy+noiselevel*realy.*randy;
```

```
inputx = initialx       % Experiment 2



% Some parameters of the Bilevel Genetic Algorithm
PopulationSize = 50 ; %The population of the colony
MaxGeneration = 1000 ; %The maximal generation
ProCro = 0.6 ; %The probability of crossover
ProMut = 0.00333 ; %The probability of mutation

boundlevel = 0.5;

%Bounds
Xupperbound =  (1 + boundlevel ) .* initialx;

Xlowerbound =  (1 - boundlevel ) .* initialx;

Vmax = 10000 ; %

Vmin = 0 ; %



% FunctionMode ≥ 0 find the maximum of the function
% FunctionMode < 0 find the minimum of the function
FunctionMode = -1 ; %

%Chromosome Length

ChromosomeLen = 10 ;

ChromosomeLength = 12*ChromosomeLen;

% The total length of chromosome
```

```
for j = 1 : 12
    X(j,:) = zeros ( 1 , PopulationSize );
end


% Generate the first generation of the population

% Resets the generator to a different state each time
rand('state',sum(100*clock));


for j = 1 : PopulationSize
    for k = 1 : ChromosomeLength
    Temp = rand(1);
    if Temp < 0.5
    Population( j ) . Chromosome( k ) = 0;
    else
    Population( j ) . Chromosome( k ) = 1;
    end % End of if
    end % End of for k

    for i = 1 : 12
    X( i, j ) = DecodeChromosome(Population( j).Chromosome,
    ChromosomeLen*(i-1)+1 , ChromosomeLen );
    end % End of for i

Population( j ).Value = 0 ;
Population( j ).Fitness = 0 ;

end % End of for j

for j = 1 : 12
X(j, :) = Xlowerbound(j) + ( Xupperbound(j)-Xlowerbound(j) )
* X(j, :)/(power(2,ChromosomeLen) - 1) ;
end

BestValue = zeros( 1 , MaxGeneration ) ;
AverageValue = zeros( 1 , MaxGeneration ) ;
```

```
disp('***************Iterations begin here*****************');

for Gen = 1 : MaxGeneration
% Calculate the value according to the function which we want to find its maximum

for j = 1 : PopulationSize
TempValue = CalculateObjectValue( Population( j ).
Chromosome , ChromosomeLen);
Population( j ).Value = TempValue;
TestValue = 0;
end % End of for - j

%Calculate fitness value
for j = 1 : PopulationSize
TempValue = CalculateFitnessValue( Population( j ).Value ,
Vmax,Vmin,FunctionMode);
Population( j ).Fitness = TempValue;
end % End of for - j

% Find out the best and worst individual of this generation
BestValue( Gen ) = Population( 1 ).Value ;

for j = 1 : PopulationSize

    if Population( j ).Value < BestValue( Gen )
    BestValue( Gen ) = Population( j ).Value;

    end % End of if

end % End of for j

Len = ChromosomeLen ;

% Find out the best individual of this generation
BestFitness( Gen ) = Population( 1 ).Fitness ;
Bestindex = 1 ;
```

```
for j = 1 : PopulationSize

    if Population( j ).Fitness > BestFitness( Gen )
    Bestindex = j;
    BestFitness( Gen ) = Population( j ).Fitness;

    end % End of if

end % End of for

% Find out the worst individual of this generation
WorstFitness( Gen ) = Population( 1 ).Fitness ;
Worstindex = 1 ;

for j = 1 : PopulationSize

    if Population( j ).Fitness < WorstFitness( Gen )
    Worstindex = j;
    WorstFitness( Gen ) = Population( j ).Fitness;

    end % End of if

end % End of for

% Keep the best individual
for k = 1 : 12*ChromosomeLen
Bestindividual( Gen ).Chromosome( k ) = Population( Bestindex ).
Chromosome( k );
end % End of k

Bestindividual( Gen ).Fitness = Population( Bestindex ).Fitness;

% Elitist
if ( Gen > 1 )

      if ( BestFitness ( Gen ) < BestFitness( Gen - 1 ) )

      for k = 1 : 12 * ChromosomeLen
```

```
      Population ( Worstindex ).Chromosome( k )  =
     Bestindividual( Gen - 1 ) . Chromosome( k );
      end % End of for

      Population ( Worstindex ).Fitness =
                      Bestindividual( Gen - 1 ).Fitness;

      Bestindex = Worstindex ;

      BestFitness ( Gen ) = BestFitness ( Gen - 1) ;

      for k = 1 : 12 * ChromosomeLen
      Bestindividual( Gen ).Chromosome( k ) =
              Population( Bestindex).Chromosome( k ) ;
      end % End of k

      Bestindividual( Gen ).Fitness = Population( Bestindex ).Fitness;


    end % End of if


end % End of if



TempX1 = CalculateVector( Population( Bestindex ).Chromosome, ChromosomeLen);

RMSE1 = sqrt( transpose(TempX1 - realx)* (TempX1 - realx)/24 );

RErrorX1 = ( TempX1 - realx ) ./ realx ;

RMSRE1 = sqrt( transpose(RErrorX1)* RErrorX1/24 );

ErrorTempX1  = abs( 100 * ( TempX1 - realx ) ./ realx );

BestX1( Gen , : ) = TempX1;
```

```
ErrorX1( Gen , : ) = ErrorTempX1;

AverageErrorX1( Gen ) = sum( ErrorTempX1 ) / 12;


RMSEX1( Gen , : ) = RMSE1;

RMSREX1( Gen , : ) = RMSRE1 ;

Sum = 1 ;
for j = 1 : PopulationSize
Sum = Sum + Population( j ).Value;
end
AverageValue( Gen ) = Sum / PopulationSize ;

% Reproduce a chromosome by proportional selection
Sum = 1 ;
CFitness = zeros ( 1 , PopulationSize ); % The cumulative fitness value

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Calculate cumulative fitness
for j = 1 : PopulationSize
Sum = Sum + Population( j ).Fitness ;
end % End of for - j

for j = 1 : PopulationSize
CFitness( j ) = Population( j ).Fitness / Sum ;
end % End of for - j
for j = 2 : PopulationSize
CFitness( j ) = CFitness( j - 1 ) + CFitness( j ) ;
end % End of for - j

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Select operation
for j = 1 : PopulationSize
RandP = rand( 1 ) ;
Index = 1;
% CFitness( Index ) = 2;
```

```
while RandP > CFitness( Index )
Index = Index + 1 ;
end % End of while

NewPopulation( j ) = Population( Index ) ;

end % End of for — j

for j = 1 : PopulationSize
Population( j ) = NewPopulation( j ) ;
end % End of for — j

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Crossover two chromosome
IndexPair = zeros( 1, PopulationSize);
for j = 1 : PopulationSize
IndexPair( j ) = j;
end % End of for — j

for j = 1 : PopulationSize
Point = floor ( (PopulationSize — j — 1) * rand( 1 ) );
Temp = IndexPair( j ) ;
IndexPair( j ) = IndexPair( j + Point );
IndexPair( j + Point ) = Temp;
end % End of for — j

for j = 1 : 2 : ( PopulationSize — 1 )
RandP = rand( 1 ) ;
if RandP < ProCro
Point = floor ( 12 * rand( 1 ) * ChromosomeLen ) ;
if Point == 0
Point = 1;
end

for k = Point : (12 * ChromosomeLen — Point + 1 )
Temp = Population ( IndexPair( j ) ) . Chromosome( k ) ;
Population ( IndexPair( j ) ) . Chromosome( k ) =
```

```
Population ( IndexPair( j + 1) ) . Chromosome( k ) ;
Population ( IndexPair( j + 1) ) . Chromosome( k ) = Temp ;
end % End of for - k



end % End of if
end % End of for - j
% Crossover two chromosome
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Mutation of a chromosome
for j = 1 : PopulationSize
for k = 1 : 12*ChromosomeLen
RandP = rand( 1 ) ;
if RandP < ProMut

if Population( j ) . Chromosome( k ) == 1
Population( j ) . Chromosome( k ) = 0 ;
else
Population( j ) . Chromosome( k ) = 1 ;
end% End of if

end% End of if
end % End of for - k


end % End of for - j


Gen

disp('********the jth iteration ends here*******');

end % End of for - Gen
```

```
elapes_time = cputime – t;

BestFitness

BestValue

AverageValue

BestX1

ErrorX1

TErrorX1 = transpose( ErrorX1 );

AverageErrorX1 = sum( TErrorX1 ) / 12;



BestOne =  1  ;
for j = 1 : length( AverageErrorX1 )

    if ( AverageErrorX1( j )  )  ≤  ( AverageErrorX1(BestOne)  )
    BestOne = j ;
    end % End of if

end % End of for j

BestOne2 =  1  ;
for j = 1 : length( RMSEX1 )

    if  RMSEX1( j )  ≤  RMSEX1( BestOne2 )
    BestOne2 = j ;
    end % End of if

end % End of for j

BestOne3 =  1  ;
for j = 1 : length( RMSREX1 )
```

```
    if  RMSREX1( j )  ≤  RMSREX1( BestOne3 )
    BestOne3 = j ;
    end % End of if

end % End of for j
```

Referências Bibliográficas

[1] Aiyoshi, E., e Shimizu, K. A solution method for the static constrained Stackelberg problem via penalty method. *IEEE Transactions on Automatic Control 29*, 12 (1984), 1111–1114.

[2] Amouzegar, M. A. A global optimization method for nonlinear bilevel programming problems. *IEEE transactions on systems, man, and cybernetics part b: cybernetics 29*, 6 (December 1999), 771–777.

[3] Bard, J. *Practical Bilevel Optimization Algorithm and Applicati-ons.Nonconvex Optimization and Its Applications. Vol. 30.* Kluwer Academics, 1998.

[4] Bertsekas, D. A class of optimal routing algorithms for communication networks. In *5th International Conference Computer Communication* (Atlanta, GA, 1980), pp. 71–76.

[5] Bertsekas, D., e Gallager, R. *Data Networks.* Prentice Hall, 1992.

[6] Bertsekas, D. P., e Tsitsiklis, J. N. *Parallel and Distributed Computation, Numerical Methods.* Prentice Hall, New Jersey, 1989.

[7] Bracken, J., e McGill, J. Mathematical programs with optimization problems in the constraints. *Operations Research 21* (1973), 37–44.

[8] Bracken, J., e McGill, J. Defense applications of mathematical programs with optimization problems in the constraints. *Operations Research 22* (1974), 1086–1096.

[9] Brackenand, J., e McGill, J. Production and marketing decisions with multiple objectives in a competitive environment. *Journal of Optimization Theory and Applications 24* (1978), 449–458.

[10] Cao, J., Davis, D., VanderWeil, S., e Yu, B. Time-varying network tomography: Router link data. *Journal of the American Statistical Association 95* (2000), 1063–1075.

[11] Cao, J., Wiel, S. V., Yu, B., e Zhu, Z. A scalable method for estimating network traffic matrices. *Technical Report, Private Communications: Bell Labs* (2000).

[12] Cisco Whitepaper. *Introduction to Cisco IOS NetFlow - A Technical Overview*, 2006.

[13] Cremer, M., e Keller, H. A new class of dynamic methods for the identification of origin-destination flows. *Transportation Research, Part B 21* (1987), 117–132.

[14] Deming, W., e Stephan, F. On a least squares adjustment of a sampled frequency table when the expected marginal totals are known. *Annals of Mathematical Statistics* (1940), 427–444.

[15] Erramill, V., Crovella, M., e Taft, N. An independent-connection model for traffic matrices. In *Proc. of the ACM SIGCOMM Internet Measurement Conf.(IMC 2006)* (Rio de Janeiro, Brazil, 2006), pp. 251–256.

[16] Eum, S., Murphy, J., e Harris, R. A fast accurate lp apporach for traffic matrix estimation. In *Proc.of 19th Int'1 Teletraffic Congress(ITC19)* (Beijing, 2005), pp. 243–252.

[17] Fortz, B., e Thorup, M. Internet traffic engineering by optimizing OSPF weights. In *IEEE INFOCOM (IEEE, Piscataway, NJ, 2000)* (2000), pp. 519–528.

[18] Fratta, L., Gerla, M., e Kleinrock, M. Flow deviation method: An approach to store-and-forward communication network design. *Network 3* (march 1973), 97–133.

[19] Gintis, H. *Game Theory Evolving*. Princeton University Press, Princeton, NJ, 2000.

[20] Goldberg, D. E. *Genetic algorithms in search, optimization and machine learning*. Addison-Wesley, Massachusetts, 1989.

[21] Goldschmidt, O. ISP backbone traffic inference methods to support traffic engineering. In *Internet Statistics and Metrics Analysis(ISMA) Workshop* (San Diego, CA, December 2000).

[22] Gunnar, A., Johansson, M., e Telkamp, T. Traffic matrix estimation on a large IP backbone: a comparison on real data. In *Proc.of the ACM SIGCOMM Internet Measurement Conf.(IMC 2004)* (2004), pp. 149–160.

[23] Herskovits, J., e Leontiev, A. An interior point technique for solving bilevel programming problems. In *Mathematical Programming in Rio. A Conference in Honour of Nelson Maculan* (Buzios, Rio de Janeiro, 2003), vol. 1, pp. 12–20.

[24] Izmailov, A., e Solodov, M. *Otimização-Volume 1 Condições de Otimalidade, Elementos de Análise Convexa e de Dualidade*. IMPA, Rio de Janeiro, 2005.

[25] Juva, I., Vaton, S., e Virtamos, J. Quick traffic matrix estimation based on link count covariances. In *Proc.of the IEEE Int'1 Conf. on Communications(ICC)* (Istanbuls:IEEE Communication Society, June 2006), pp. 603–608.

[26] Kay, S. *Fundamentals of Statistical Signal Processing: Estimation Theory*. Prentice Hall, 1993.

[27] Kleinrock, L. *Communication Nets: Stochastic Message Flow and Delay*. McGraw-Hill, New York, 1964.

[28] KLEINROCK, L. *Queueing Systems volume 2.* Wiley-Interscience, 1976.

[29] KRUITHOF, J. Telefoonverkeersrekening. *De Ingenieur 52*, 8 (1937), E15–E25.

[30] KRUPP, R. Properties of Kruithof's projection method. *The Bell System Technical Journal 58*, 2 (February 1979), 517–538.

[31] KUROSE, J., E ROSS, K. *Computer Networking : A Top-Down Approach Featuring the Internet*, 3rd ed. Addison Wesley, 2004.

[32] LAKHINA, A., PAPAGIANNAKI, K., CROVELLA, M., DIOT, C., KOLACZYK, E., E TAFT, N. Structural analysis of network traffic flows. In *ACM Sigmetrics* (New York, June 2004).

[33] LIANG, G., TAFT, N., E YU, B. A fast lightweight approach to origin-destination IP traffic. *IEEE Tran., Special issue on Networking and Information Theory* (2006).

[34] LIANG, G., E YU, B. Maximum pseudo likelihood estimation in network tomography. *IEEE Trans. on Signal Processing 51*, 8 (2003), 2043–2053.

[35] THE MATH WORKS. *Optimization toolbox for use with Matlab- User's Guide Version 2*, 2001.

[36] MATHIEU, R., PITTARD, L., E ANANDALINGAM, G. Genetic algorithm based approach to bi-level linear programming. *Operations Research 28*, 1 (1994), 1–21.

[37] MEDINA, A., TAFT, N., SALAMATIAN, K., BHATTACHARYYA, S., E DIOT, C. Traffic matrix estimation: Existing techniques and new directions. In *Sigcomm'02* (Pittsburgh, USA, August 2003).

[38] MENASCHÉ, D. S. *Controle de Congestionamento: Uma Abordagem Dinâmico-Populacional via Teoria dos Jogos Evolucionários.* Dissertação de mestrado, Universidade Federal do Rio de Janeiro/COPPE Sistemas, Rio de Janeiro, May 2005.

[39] Muu, L. D., e Quy, N. V. A global optimization method for solving convex quadratic bilevel programming problems. *Journal of Global Optimization 26*, 2 (June 2003), 199–219.

[40] Ouorou, A., Mahey, P., e Pjvial, I. A survey of algorithms for convex multicommodity flow problems. In *Informs* (Pittsburgh, USA, January 2000), vol. 46, pp. 126–147.

[41] Outrata, J. V. On the numerical solution of a class of Stackelberg problems journal. *Mathematical Methods of Operations Research 34*, 4 (July 1990), 255–277.

[42] Papagiannaki, K., Taft, N., e Lakhina, A. A distributed approach to measure traffic matrices. In *ACM Internet Measurement Conference* (Taormina, Italy, October 2004), pp. 161 – 174.

[43] Roughan, M., Thourp, M., e Zhang, Y. Traffic engineering with estimated traffic matrices. In *Proceedings of the USENIX/ACM Internet Measurement Conference* (Miami, USA, October 2003), pp. 248–258.

[44] Roughgarden, T. *Selfish Routing*. PhD thesis, Cornell University, May 2002. `http://www.cs.cornell.edu/timr/`.

[45] Shimizu, K., e Aiyoshi, E. A new computational method for Stackelberg and min-max problems by use of a penalty method. *IEEE Transactions on Automatic Control 26*, 2 (1981), 460–466.

[46] Shimizu, K., Ishizuka, Y., e Bard, J. F. *Nondifferentiable and Two-Level Mathematical Programming*. Springer, 1997.

[47] Soule, A., Lakhina, A., Taft, N., Papagiannaki, K., Salamatian, K., Nucci, A., Crovella, M., e Diot, C. Traffic matrices: balancing measurements, inference and modeling. In *ACM Sigmetrics 2005*, pp. 362–373.

[48] Soule, A., Nucci, A., Leonardi, E., Cruz, R., e Taft, N. How to identify and estimate the largest traffic matrix elements in a dynamic enviroment. In *ACM Sigmetrics* (New York, June 2004), pp. 73–84.

[49] Soule, A., Salamatian, K., Nucci, A., e Taft, N. Traffic matrix tracking using Kalman filters. In *ACM Sigmetrics: Large Scale Networks Inference (LSNI) Workshop 2005*, pp. 1–8.

[50] Stackelberg, H. *The Theory of Market Economy*. Oxford, UK, 1952.

[51] Tebaldi, C., e West, M. Bayesian inference of network traffic using link count data. *Journal of the American Statistical Association 93*, 442 (1998), 557–576.

[52] Tylavsky, D. J., e Sohie, G. R. L. Generalization of the matrix inversion lemma. vol. 74, pp. 1050–1052.

[53] Vardi, Y. Estimating source-destination traffic intensities from link data. *Journal of the American Statistical Association* (march 1996), 365–377.

[54] Vaton, S., e Gravey, A. Network tomography: an iterative bayesian analysis. In *Proc.ITC 18* (Berlin, Germany, August 2003).

[55] Vicente, L., e Calamai, P. Bilevel and multilevel programming: A bibliography review. *Journal of Global Optimization 5* (1994), 291–306.

[56] Wardrop, J. Some theoretical aspects of road traffic research. In *Institute of Civil Engineers PartII* (London, 1952), vol. 1, pp. 325–378.

[57] Zhang, Y. 6 months of Abilene traffic matrices. `http://www.cs.utexas.edu/users/yzhang/research/AbileneTM/`.

[58] Zhang, Y., Roughan, M., Duffield, N., e Greenberg, A. Fast accurate computation of large-scale IP traffic matrices from link data. In *ACM Sigmetrics* (San Diego, CA, 2003), pp. 206–217.

[59] Zhang, Y., Roughan, M., Lund, C., e Donodo, D. An information theoretic approach to traffic matrix estimation. In *ACM Sigcomm* (Karlsruhe, Germany, August 2003).

[60] Zhao, Q., Ge, Z., Wang, J., e Xun, J. Robust traffic matrix estimation with imperfect information: making use of multiple data sources. *ACM SIGMETRICS Performance Evaluation Review 34*, 1 (2006), 133–144.