GERAÇÃO DE COLUNAS EM PROGRAMAÇÃO INTEIRA APLICADA À SÍNTESE DE REDES

Marcos de Mendonça Passini

TESE SUBMETIDA AO CORPO DOCENTE DA COORDENAÇÃO DOS PROGRAMAS DE PÓS-GRADUAÇÃO DE ENGENHARIA 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. Nelson Maculan Filho, D. Habil.

Prof. Paulo Roberto Oliveira, D.Sc.

Prof. Oscar Porto, D.Sc.

Prof. Luiz Satoru Ochi, D.Sc.

Prof. Henrique Pacca Loureiro Luna, D.Sc.

RIO DE JANEIRO, RJ - BRASIL

SETEMBRO DE 2001

PASSINI, MARCOS DE MENDONÇA

Geração de Colunas em Programação Inteira Aplicada à Síntese de Redes [Rio de Janeiro] 2001

X, 67p. 29,7 cm (COPPE/UFRJ, D.Sc. Engenharia de Sistemas e Computação, 2001)

Tese - Universidade Federal do Rio de Janeiro, COPPE

1. Geração de Colunas em Programação Inteira
2. Síntese de Redes

I. COPPE/UFRJ II. Título (série)

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

GERAÇÃO DE COLUNAS EM PROGRAMAÇÃO INTEIRA APLICADA À SÍNTESE DE REDES

Marcos de Mendonça Passini

Setembro/2001

Orientador: Nelson Maculan

Programa: Engenharia de Sistemas e Computação

Este trabalho apresenta um método para a solução de problemas de programação linear inteira baseado em geração de colunas e a sua aplicação a problemas selecionados em síntese de redes.

O método baseia-se mormente nos princípios da decomposição de Dantzig-Wolfe, aparentando-se com a estratégia *branch and price* para solução de programas lineares inteiros.

As aplicações foram selecionadas dentre as mais relevantes em síntese e sobrevivência de redes de telecomunicações. Trata-se, portanto, de problemas de multifluxo com restrições de integralidade.

Apresentamos sucessivamente o método, as aplicações e sua relevância, os problemas com os modelos correspondentes e, finalmente, os resultados obtidos.

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

COLUMN GENERATION IN INTEGER PROGRAMMING APPLIED TO NETWORK DESIGN

Marcos de Mendonça Passini

September/2001

Advisor: Nelson Maculan

Department: Systems Engineering and Computing

This work presents a method, based in column generation, for solving integer programming problems, and its application to selected problems in network design.

The method is based mainly in the Dantzig-Wolfe principles of decomposition, and it relates to the *branch and price* strategy for linear integer programs.

Applications were selected amongst the most relevant in telecommunications networks design and survivability, problems of multicommodity flow with integrality constraints.

After introducing the method, we show the application and its relevance. So we show the problems and the corresponding models, and finally the numerical results.

Rezumo de la tezo prezentita al COPPE/UFRJ kiel parto de la postuloj necesaj por la akiro de Doktora Diplomo pro Scienco (D.Sc.)

PRODUKTADO DE KOLUMNOJ EN ENTJERA PROGRAMADO APLIKATA AL RETSINTEZO

Marcos de Mendonça Passini

Septembro/2001

Gvidanto: Nelson Maculan

Departemento: Enĝenierarto pri Sistemoj kaj Komputado

Ĉi tiu laboro prezentas metodon por la solvo de entjeraj linearaj programoj bazitaj sur produktado de kolumnoj kaj ties aplikoj al retsintezaj selektitaj problemoj.

La metodo baziĝas precipe sur la Dantzig-Wolfe-a principo de malkompono, kaj alkonformiĝas al la strategio *branch-and-price* por solvo de entjeraj linearaj programoj.

La alaplikoj estis selektitaj el inter la plej elstaraj en telekomunika retsintezo. Ili estas do problemoj pri multfluo kun entjerigaj kondiĉoj.

Ni prezentas sinsekve la metodon, la alaplikojn kaj ghia elstareco, la problemojn kun ghiaj modeloj kaj fine la rezultojn atingitajn.

À Deise
espelho de estranhas e raras paixões
luz de caminhos que se bifurcam e unificam
nos sorrisos da lua

Agradecimentos

Ao Professor Nelson Maculan, pelo convívio fecundo, pela amizade segura e, de modo especial, pela humanidade inspiradora;

ao André Brito, pela dedicação e disposição sempre presentes no trabalho conjunto;

aos colegas de projeto, Professor Adilson Xavier, Amir Coelho, André Brito, Antônio Silvério, Douglas Valiati e Rosa Figueiredo, pelo convívio, nem sempre pacífico, mas garantia de aprendizado profissional e pessoal permanente, gerador de trabalho e amizade;

à Laura Bahiense e ao Élder Macambira, pelos valiosos comentários em cima da hora;

ao Flávio Montenegro e ao Amir Coelho, pelos diálogos loucos e geniais e pelas tiradas, loucas e geniais;

aos colegas do Laboratório de Otimização, impossível citar todos, por compartilharmos um ambiente de trabalho acolhedor e produtivo;

ao Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq), pelo apoio financeiro;

aos meus pais, que sempre acreditaram.

Desvarío laborioso y empobrecedor el de componer vastos libros; el de explayar en quinientas páginas una idea cuya perfecta exposición oral cabe en pocos minutos.
BORGES, Ficciones.

Conteúdo

Introdução

Após o aparecimento do método "branch-and-bound" nos anos 60 a pesquisa na área de Programação Inteira centrou-se na obtenção de boas relaxações lineares, visando à envoltória convexa dos pontos inteiros viáveis, e deixando de lado o aperfeiçoamento de técnicas de busca na árvore combinatória.

Os últimos anos viram surgir a técnica chamada "branch-and-price". Como preconiza o princípio da decomposição de Dantzig-Wolfe, grupos de colunas são removidos do programa porque, possivelmente, as variáveis associadas a elas terão valor nulo na solução ótima. Obtida uma solução para a relaxação linear, sua otimalidade é verificada resolvendo-se um subprograma, chamado "problema do preço" (*pricing problem*), que busca identificar colunas que possam entrar na base, e o programa é então reotimizado com a nova coluna. Aplica-se este procedimento em programação inteira a cada passo do *branch-and-bound*; se uma coluna candidata não é encontrada e a solução atual não satisfaz as restrições de integralidade, procede-se à ramificação.

Neste trabalho estudamos a resolução de problemas relacionados a redes de telecomunicações envolvendo concepção de redes e roteamento de multifluxos. Especificamente, estudamos o problema da definição de anéis auto-regenerativos unidirecionais e o problema prático de sobrepor uns tais anéis a uma rede em malha pré-existente, constituindo este último o a que denominamos "Problema das Duas Arquiteturas". Os vários modelos apresentados fazem uso extensivo de variáveis de decisão, em geral associadas à escolha entre instalar ou não certos equipamentos na rede.

Sob o ponto de vista da engenharia, equipar uma rede usando duas arquiteturas distintas coexistentes foi uma inovação proposta pelo grupo de pesquisa do Centre National d'Etudes des Télécommunications — CNET

— da France Telecom. O estudo dos custos e da viabilidade, via modelagem matemática, e a solução do programa decorrente constituíram o presente trabalho, bem como outros a ele relacionados, totalizando dois artigos internacionais, duas teses de mestrado e uma de doutorado. Neste sentido, o trabalho é pioneiro.

Sob o ponto de vista da programação inteira, a idéia básica do *branch and price* estabelece que o programa seja decomposto e que o programa mestre guie a busca ao longo da árvore ora ramificando-a, ora gerando novas colunas. A geração de colunas, entretanto, deixa margem a muitas estratégias distintas que podem conformar-se tanto com um paradigma genérico, simplesmente resolvendo um programa linear inteiro, quanto com heurísticas *ad hoc* que envolvam conhecimento do problema tratado. Assim, no trabalho descrevemos uma estratégia geral para a solução de programas inteiros pelo algoritmo COLGER, apresentado em duas versões, e também mostramos algumas possíveis opções, heurísticas ou exatas, para o trato do programa secundário, reduzindo-lhe o número de soluções viáveis através de procedimentos e conhecimento específico do problema.

Este trabalho originou-se de uma cooperação entre o Laboratório de Otimização da COPPE/UFRJ e o CNET iniciada em 1995 e materializada como acordo de cooperação científica em 1998. O projeto do acordo teve como metas principais levantar e comentar a bibliografia sobre otimização combinatória aplicada a problema de redes, desenvolver técnicas de programação inteira que dessem conta dos problemas da definição de anéis e outros relacionados, bem como resolver o Problema das Duas Arquiteturas. Esse projeto, intitulado "Etude de la synthèse de réseaux de transmission à architecture mixte" foi coordenado pelo Prof. Nelson Maculan, da parte da COPPE, e pelo Dr. Abdel Lisser, da parte do CNET, e teve ainda como participantes Dr. Abilio Lucena, Prof. Adilson Xavier, Amir Coelho, André Brito e Gabriel Weintraub.

Concomitantemente com o desenvolvimento desta tese e fruto do mesmo esforço produziram-se dois artigos intitulados "Column Generation in Integer Linear Programming" [MPBL01b] e "Column Generation Method for Network Design" [MPBL01a]. Na data da defesa, o primeiro deles estava já aceito para publicação na revista francesa RAIRO - Operations Research e o segundo no prelo na coletânea Transportation and Network

Analysis: Current Analysis, da editora Kluwer holandesa.

Partes deste trabalho foram apresentadas nos congressos: Conférence Internationale en Recherche Opérationnelle, Marrakech, 1996 [LMP96c]; INFORMS Fall Meeting, Atlanta, 1996 [LMP96a]; XXIX Simpósio Brasileiro de Pesquisa Operacional, Salvador, 1997 [PML97]; IX Congreso Latino Americano de Investigación Operativa, Buenos Aires, 1998 [LMPW98]; INFORMS Fall Meeting, Philadelphia, 1999 [MPL99]; VIII Escuela Latino-Americana de Investigación Operativa, Mendes, 1999 [Pas96]; XXIII Congresso Nacional de Matemática Aplicada e Computacional, Santos, 2000 [BMP00]; X Congreso Latino Americano de Investigación de Operaciones y Sistemas, Cidade do México, 2000 [MPBL00].

O capítulo 1 traz uma breve introdução e histórico das técnicas de programação linear inteira. No 2 apresentamos uma revisão bibliográfica de geração de colunas em programação linear contínua e inteira, citando aplicações importantes. A seguir descrevemos nosso método para resolução de programas lineares inteiros usando geração de colunas, abordando desde um algoritmo geral de solução até questões mais específicas como solução viável inicial e resolução de nós da árvore *branch and bound* usando dualidade. Depois, no capítulo 3, descrevemos o cenário dos problemas de síntese de redes de telecomunicações e apresentamos dois problemas intimamente relacionados com o objeto deste estudo, bem como os modelos correspondentes. Em seguida apresentamos modelos já propostos para a solução do Problema das Duas Arquiteturas ao qual, no capítulo 4, é aplicada a técnica de programação inteira desenvolvida. Finalmente, mostramos algumas alternativas de solução do programa secundário, heurísticas e exatas, bem como os resultados numéricos obtidos.

Capítulo 1

Programação Inteira

Não pode haver uma linguagem mais universal e mais simples, mais livre de erros e obscuridades... mais digna de expressar as relações invariáveis das coisas naturais [do que a matemática].
FOURIER, Teoria Analítica do Calor.

Modelar a realidade usando números tem sido uma ocupação central no trabalho científico há séculos. Descrever um sistema físico de forma a prever-lhe o comportamento em função do tempo ou outros parâmetros é uma atividade particularmente fascinante e que tem intrigado os epistemólogos e filósofos da ciência. Ao perguntarmos por que a natureza parece se comportar "numericamente" levantamos questões outras, quais sejam, se realmente a natureza se modela pela matemática, até onde a matemática está inserida na estrutura do Universo, que tipo de matemática pode ser considerada "natural" e que tipo é criação humana etc. Os pitagóricos pretendiam que apenas os números inteiros explicassem o mundo, e desconcertaram-se com a descoberta da irracionalidade da diagonal do quadrado e do diâmetro do círculo. Kronecker, em pleno século dezenove, sustentava que "Deus criou os números inteiros, o resto é criação do homem."

Às mitologias antigas sucederam o conhecimento empírico não sistematizado, as pesquisas normalizadas não matemáticas ao estilo de Bacon e, finalmente, as festejadas descrições matematicamente perfeitas de Newton. Dos seguidores desses, talvez ressuscitando os antigos místicos

obscurantistas, ouvem-se repetidas as crenças aferradas em dogmas do matematicismo. "O Universo explica-se naturalmente pela matemática," dizem, "os números são a essência da natureza", propalam esses neopitagóricos, crendo num injustificado caráter absoluto ou divino dos números e desprezando a evidente relatividade e impermanência dos modelos matemáticos.

A ciência da Otimização assume essa relatividade das modelagens sem maiores sobressaltos. Por sua própria natureza, tendo por fim último a resolução de problemas, trabalha com modelos que não podem esperar até a próxima geração para serem resolvidos. Assim sendo, um modelo de otimização não é qualificado exclusivamente pela sua aderência ao problema modelado, mas, antes, à sua capacidade de fornecer uma resposta razoável num tempo razoável (a definição precisa de "razoável", neste caso, depende do problema em questão e do seu contexto).

1.1 Programa Inteiro

Um *programa inteiro* ou *problema de programação inteira* consiste numa função objetivo de variáveis inteiras a ser maximizada ou minimizada num subdomínio restrito por equações ou inequações. Quando a função objetivo ou as restrições apresentam também variáveis contínuas, pode-se falar em *programa inteiro misto*.[1]

Escrevemos o programa da forma a seguir,

$$\min\{cx : Ax \leq b, x \geq 0, x \in \mathbb{Z}^n\} \tag{1.1}$$

onde A é uma matriz real $m \times n$, $b \in \mathbb{R}^m$ e $c \in \mathbb{R}^n$.

O programa inteiro genérico é um problema intrinsecamente combinatório, pertencente à classe $\mathcal{NP}$ dos problemas *polinomiais não determinísticos*, significando que a verificação de sua solução pode-se realizar em tempo polinomial. Na verdade o problema de encontrar o ótimo de um programa inteiro é $\mathcal{NP}$*-árduo*, significando ser pelo menos tão difícil quanto qualquer outro problema $\mathcal{NP}$, incluindo-se aí alguns famosamente difíceis

[1]Se a função objetivo e as restrições forem lineares, para maior precisão dever-se-ia falar de *programação linear inteira*. Neste texto tratamos apenas de problemas lineares, ficando o termo "linear" subentendido.

como o Problema do Caixeiro Viajante. Nessas condições, as estratégias de resolução são variadas e elaboradas *ad hoc*. Não há uma única técnica universalmente bem sucedida como a do método simplex para programação linear contínua, especialmente se trata-se de problemas de grande porte. Sobre a classe $\mathcal{NP}$ e a teoria de complexidade em geral veja-se [CLR90].

1.2 Branch and Bound

A enumeração das soluções viáveis é a abordagem mais natural a problemas combinatórios e também a menos realizável na prática. Raramente conseguem-se enumerar soluções de programas inteiros com mais de 20 ou 30 variáveis.

A segunda solução mais imediata é a *enumeração implícita*, que busca realizar "podas" na árvore combinatória, deixando de explorar nós que sabidamente não melhorarão a solução atual. Essas técnicas são coletivamente conhecidas pelo nome *branch and bound*, que se refere ao procedimento de construção dos ramos da árvore e de busca de cotas.

Buscamos, neste texto, evitar a confusão normalmente ocorrida com as cotas "inferiores" e "superiores", cujos significados se invertem de acordo com a orientação do problema, min ou max. Destarte, seguimos a nomenclatura usada por Wolsey em [Wol98], preferindo os termos "cota primal" e "cota dual", que não sofrem a alteração citada e melhor explicitam a função da cota no problema. Assim, num problema de minimização, chamaremos *cota primal* a melhor solução viável já obtida ("cota superior") e *cota dual* o limite para a qualidade da solução ("cota inferior").

O problema 1.1 poderia ser escrito de maneira mais genérica da seguinte forma,

$$\min\{cx : x \in X\} \tag{1.2}$$

onde X seria o subconjunto de $\mathbb{Z}$ definido pelas inequações $Ax \leq b$ e $x \geq 0$. Definimos uma *relaxação* de 1.2 como um problema com a mesma função objetivo e um conjunto viável X' tal que $X \subseteq X'$. Assim, a solução de uma relaxação de um problema fornece sempre uma cota dual, enquanto qualquer solução viável do próprio problema fornece uma cota primal.

Seja um programa inteiro como o mostrado em 1.1:

$$(P) : \min\{cx : Ax \leq b, x \geq 0, x \in \mathbb{Z}^n\} \tag{1.3}$$

O procedimento de enumeração implícita básico para (P) inicia-se resolvendo sua *relaxação linear* $(\bar{P})$, que nada mais é que o próprio (P), com as restrições $x \in \mathbb{Z}^n$ substituídas por $x \in \mathbb{R}^n$, resultando num programa linear.

$$(\bar{P}) : \min\{cx : Ax \leq b, x \geq 0, x \in \mathbb{R}^n\} \tag{1.4}$$

O ótimo deste programa fornecerá a primeira cota dual para o programa inteiro.

Se a solução encontrada para $(\bar{P})$, $\mathsf{sol}(\bar{P}) = x^*$, for inteira, $x^* \in \mathbb{Z}^n$, então $\mathsf{sol}(P) = \mathsf{sol}(\bar{P}) = x^*$. Em caso contrário, existe um k tal que $x_k^* \notin \mathbb{Z}$. A árvore combinatória é então recursivamente construída, gerando-se os dois nós filhos de (P), (P_1) e (P_2).

$$(P_1) : \min\{cx : Ax \leq b, x \geq 0, x_k \leq \lfloor x_k^* \rfloor, x \in \mathbb{Z}^n\} \tag{1.5}$$

$$(P_2) : \min\{cx : Ax \leq b, x \geq 0, x_k \geq \lfloor x_k^* \rfloor + 1, x \in \mathbb{Z}^n\} \tag{1.6}$$

A poda ocorre por:

(i) solução inteira encontrada;

(ii) solução encontrada pior que a atual cota primal;

(iii) problema inviável.

Em qualquer dos três casos, nenhuma informação adicional sobre o problema pode ser obtida pela exploração dos nós descendentes.

Uma lista de nós a serem explorados é mantida pelo algoritmo; quando esta lista estiver vazia, o problema estará resolvido. As estratégias de busca dentro da árvore combinatória são variadas e constituem-se em parâmetros importantes a serem ajustados de acordo com as idiossincrasias do problema.

A primeira aparição de um algoritmo *branch and bound* para programação inteira ocorreu em 1960 no trabalho de Land e Doig [LD60]. Balas [Bal60] desenvolveu um algoritmo para variáveis binárias, restritas aos valores 0–1, usando testes simples para obter cotas duais e verificar viabilidade primal.

1.3 Relaxação

A necessidade de boas cotas para podas eficientes na enumeração implícita guiou a pesquisa à obtenção de boas relaxações. Ainda que não busquem diretamente obter a envoltória convexa, diversas relaxações são possíveis e utilizadas. Merece destaque a técnica popularizada por Held e Karp em [HK70] e [HK71] na solução do problema do caixeiro viajante. Esta técnica foi descrita de maneira geral para a programação inteira pela primeira vez por Geoffrion [Geo74]. Trata-se da aplicação à programação inteira da relaxação lagrangeana, que remove do problema grupos de restrições de modo a obter uma estrutura que permita boas abordagens. As restrições "complicantes" são então convertidas em penalidades com multiplicadores de Lagrange (variáveis duais) e adicionadas à função objetivo. Esta abordagem costuma oferecer boas relaxações e tem sido campo ativo de pesquisa.

1.4 Cortes

Após a introdução do *branch and bound* nos anos sessenta a pesquisa centrou-se na obtenção de boas relaxações lineares. Algumas delas tinham por objeto a envoltória convexa dos pontos inteiros viáveis, o poliedro cujos vértices fossem soluções inteiras viáveis do problema original. Isso normalmente é obtido através da introdução de restrições (cortes) segundo diversos algoritmo. O mais antigo procedimento registrado na literatura deve-se a Gomory [Gom58].

Na década de oitenta retornou o interesse pela elaboração da árvore combinatória com o aparecimento da técnica "branch and cut", que nada mais é que uma generalização do *branch and bound* usando a relaxação de restrições outras que não as de integralidade. Nessa abordagem,

selecionam-se para serem excluídas algumas classes de restrições de desigualdade válidas, preferencialmente aquelas que, espera-se, não estarão ativas na solução ótima. A cada nó da árvore, quando a relaxação linear produz uma solução inviável, resolve-se um subproblema, chamado "problema da separação" para restrições violadas. Destas, algumas são acrescentadas ao problema para eliminar a solução inviável. Se não for possível encontrar uma tal restrição violada, procede-se à ramificação (*branch*). O primeiro trabalho usando técnicas de corte ao longo da árvore combinatória deve-se a Crowder, Johnson e Padberg [CJP83].

Os últimos anos viram surgir a técnica chamada "branch and price", descrita em [BJN+98], baseada na geração de colunas nos nós da árvore *branch-and-bound*. Esta técnica será explanada em detalhes nos capítulos seguintes.

Capítulo 2

Geração de Colunas

A base constitui a parte inferior.
Suporta o fuste, que, por sua
vez, é encimado pelo capitel.
DELTA UNIVERSAL, coluna.

2.1 Revisão

As técnicas de geração de colunas apareceram pela primeira vez no início dos anos sessenta, como parte de métodos criados para a solução de programas lineares de grande porte, com grande número de variáveis. Em [DW60], Dantzig e Wolfe apresentaram essas técnicas no largamente conhecido algoritmo de decomposição, desenvolvido para contornar as limitações de memória dos computadores da época.

Gilmore e Gomory, em seus trabalho iniciais, [GG61] e [GG63], solucionaram problemas de corte, originalmente com variáveis inteiras, por geração de colunas, resolvendo um problema da mochila em cada iteração. Usou-se a geração de colunas para resolver a relaxação linear e o resultado foram excelentes cotas inferiores. Duas décadas após, Marcotte [Mar85] mostrou que, com freqüência, o ótimo inteiro é obtido arredondando-se o valor da solução linear.

A literatura apresenta diversas outras aplicações de geração de colunas desenvolvidas desde então, que colimaram relaxações mais justas para programas inteiros.

Mais recentemente apareceram novos métodos de geração de colunas para solução exata de programas inteiros. Os mais relevantes são, sem dúvida, os genericamente conhecidos como *branch and price*, que combinam *branch and bound* e geração de colunas para resolver a relaxação linear em cada nó com regras de ramificação *ad hoc*, elaboradas de forma a manter a estrutura do programa secundário tratável ao longo da árvore de busca. Os melhores resultados foram obtidos em problemas modeláveis como particionamento ou como cobertura. Dos métodos exatos podem-se derivar heurísticas usando apenas um subconjunto das colunas viáveis em cada nó ou mantendo apenas as geradas no nó raiz.

Em [DSD84], [DDS92], [RS94], [DDSS95], [BSL97] e [Lob98] encontram-se aplicações bem resolvidas de geração de colunas a problemas de roteamento de veículos com janelas de tempo. Desrosiers *et al.* modelaram o problema como particionamento de conjunto. Neste trabalho, as colunas são geradas por um algoritmo de caminhos mínimos modificado, que leva em conta as janelas de tempo. As regras de ramificação conservam essa estrutura de caminhos do subproblema. Nos casos em que não foi possível explorar a árvore por inteiro, obtiveram-se boas cotas inferiores. Uma abordagem similar foi usada em [DDS92] por Desrochers *et al.* Em [RS94], Ribeiro e Soumis apresentam um método de geração de colunas para resolver a relaxação linear de um problema de escalonamento de veículos com múltiplos depósitos. A solução obtida foi uma cota melhor que a então conhecida na literatura.

Em [DDSS95], Desrosiers *et al.* apresentaram uma revisão bastante clara dos problemas de escalonamento com restrições e roteamento de veículos. Neste trabalho observou-se que algoritmos ótimos baseados na decomposição Dantzig-Wolfe e esquemas de geração de colunas provaram ser as metodologias mais efetivas para esse tipo de problemas.

Bramel e Simchi-Levi [BSL97] pretenderam formalizar uma conclusão empírica advinda de aplicações quando se aplicam técnicas de geração de colunas a problemas modelados como o problema da cobertura. O método funciona bem quando a margem entre as soluções linear (relaxação) e inteira é pequena. Sobre o roteamento de veículos com janelas de tempo, os autores mostram que, assumindo-se certas condições sobre a distribuição dos consumidores, essa margem cai para zero à medida que cresce o nú-

mero desses consumidores. A geração de colunas foi aplicada ao problema do escalonamento de tripulações em trânsito urbano, usando-se cobertura e caminhos mínimos como subproblemas, em [DS89]. Em [VBJN94], Vance *et al.* apresentaram um algoritmo de decomposição para o problema de escala de tripulações aéreas, com cotas melhores que as obtidas até então. Também Gamache *et al.* [GSMD92] apresentaram uma heurística para a solução do problema da escala de tripulações aéreas, obtendo resultados favoráveis se comparados a problemas reais de grande porte. Loebel [Lob98] desenvolveu um método de geração de colunas para resolver a relaxação do problema de escalonamento de veículos em tráfego público, fornecendo soluções excelentes para casos reais.

Vance *et al.* [VBJN94] implementaram um algoritmo exato para o problema do corte unidimensional, propondo regras de ramificação que mantêm tratável a estrutura do problema em cada nó. Essas regras foram anteriormente propostas por Ryan e Foster em [RF81]. Em [Van98], a autora comparou dois algoritmos para o mesmo problema, baseados em duas formulações diferentes do problema mestre, apresentando regras de ramificação apropriadas a cada um. Em [dC98], Carvalho apresentou uma abordagem diferente para o caso genérico (não necessariamente binário) do problema do corte: propôs uma formulação de fluxo em arcos com restrições adicionais e a resolveu usando geração de colunas. O autor afirmou que o algoritmo funciona bem quando a diferença entre a primeira relaxação e o ótimo inteiro é menor que 1 (um); afirmou também que o método é sensível à largura dos rolos cortados. Vanderbeck [Van99] mostrou resultados de um algoritmo geração de colunas para os problemas do corte e do empacotamento.

Mehrotra e Trick, em [MT96], desenvolveram um método para a coloração de grafos usando uma formulação baseada em conjuntos independentes, que evitava a simetria até um certo ponto e usava regras de ramificação específicas, similares às apresentadas em [VBJN94]. Esta abordagem produziu uma heurística na qual as colunas eram geradas apenas no nó raiz. Em [JMN93], a geração de colunas serviu como esteio a heurísticas para o particionamento de grafos baseadas em relaxação linear. Bourjolly *et al.* [BLM97] obtiveram cotas inferiores para o problema do conjunto estável máximo usando geração de colunas inserida num método *branch and*

bound que não fazia uso explícito de programação linear.

Savelsbergh [Sav97] apresentou um algoritmo *branch and price* dirigido à obtenção de soluções inteiras ótimas para o problema geral da alocação, formulando-o como partição de conjuntos. O autor mostrou também que podem-se obter excelentes algoritmos aproximativos truncando a árvore de busca.

Em [vdAHvdV99], van den Akker *et al.* descreveram um método baseado em geração de colunas para resolver um problema de escalonamento de máquinas paralelas, obtendo uma excelente cota inferior e resolvendo algumas instâncias até a otimalidade. Hansen *et al.* [HJdA91] propuseram algoritmos primal e dual para programação inteira mista e usaram-nos para resolver o problema da satisfabilidade probabilística máxima.

Barnhart *et al.* [BJN+98] apresentaram uma visão geral das técnicas de geração de colunas para solução de problemas inteiros à otimalidade, juntamente com várias classes de problemas assim resolvidos com êxito. A proposta desse trabalho era generalizar idéias profícuas na solução de problemas especiais por geração de colunas.

Vanderbeck e Wolsey, [Van94] e [VW96], apresentaram um método exato que combinava *branch and bound* com geração de colunas, desenvolvendo um esquema de ramificação *ad hoc* e testando um algoritmo em três tipos de problema. Em [SVW98], a mesma abordagem foi aplicada à resolução de um problema de síntese de redes de telecomunicações. Em [Van00], o autor prosseguiu nessa mesma direção e propôs uma decomposição Dantzig-Wolfe baseada tanto na discretização do poliedro inteiro associado a um grupo de restrições quanto em esquemas de ramificação apropriados. Essas idéias foram testadas em problemas de corte.

Em [BHV00], um problema de multifluxo foi resolvido combinando-se geração de colunas e linhas (cortes). Desenvolveu-se um algoritmo composto por um algoritmo de *pricing*, regras de ramificação e adição de cortes, todos mutuamente compatíveis ao longo da árvore *branch and bound*. Geração de colunas e regras de ramificação apenas generalizaram trabalhos anteriores; entretanto, adicionam-se desigualdades de recobrimento elevadas[1] em cada nó. Até o presente momento, apareceram na literatura

[1] "Lifted cover inequalities" em inglês. O termo "lift", no sentido de "projeção" para um espaço de dimensão maior, ainda não tem tradução consagrada em português.

poucos trabalhos combinando geração de colunas e *branch and cut*.

O objetivo deste trabalho é a solução de alguns problemas específicos relacionados a multifluxo em redes, especialmente envolvendo concepção de redes e roteamento de multifluxos. Os modelos fazem uso extensivo de variáveis de decisão, em geral associadas à escolha entre instalar ou não certos equipamentos na rede. Esses problemas, por suas peculiaridades, encaixam-se bem numa estratégia de geração de colunas. A técnica usada é exposta a seguir, de forma sucinta.

2.2 Programação Linear com Variáveis Canalizadas

Consideremos o seguinte programa linear:

$$(LP): \text{minimizar} \quad z = \sum_{j=1}^{p} c_j x_j \tag{2.1}$$

$$\text{sujeito a} \quad \sum_{j=1}^{p} a_j x_j = b, \tag{2.2}$$

$$0 \leq x_j \leq d, j = 1, 2, \ldots, p. \tag{2.3}$$

onde $a_j \in \mathbb{R}^m$, $b \in \mathbb{R}^m$, $b \geq 0$, $d \in \mathbb{R}$, $c_j = f(a_j)$, onde $f : \mathbb{R}^m \to \mathbb{R}$. Neste trabalho supomos $c_j = c^T a_j + g$, $c \in \mathbb{R}^m$, $g \in \mathbb{R}$. Assumimos $c = 0$, $g = 1$ e $p \geq m$ sem perda de generalidade, a fim de simplificar a notação. Os resultados podem facilmente ser generalizados. Definimos também o conjunto $\mathcal{K}$ das colunas da matriz de (LP): $\mathcal{K} = \{a_1, a_2, \ldots, a_p\}$.

(LP) pode ser resolvido usando o método proposto por Dantzig mostrado em [Dan55], [Dan63], [Las70], [Chv83]. Nosso problema consiste em resolver (LP) por técnicas de geração de colunas [DW60].

2.3 Um Método de Geração de Colunas

2.3.1 O algoritmo COLGEN

Escrevemos (LP) novamente da seguinte forma:

$$(LP): \text{minimizar} \quad z = \sum_{j=1}^{p} x_j \tag{2.4}$$

$$\text{sujeito a} \quad \sum_{j=1}^{p} a_j x_j = b; \tag{2.5}$$

$$0 \leq x_j \leq d, j = 1, 2, \ldots, p. \tag{2.6}$$

Seja $B = (a_{B(1)}\ a_{B(2)}\ \ldots\ a_{B(m)})$ uma matriz $m \times m$ inversível e N uma matriz $m \times (p-m)$ formada pelas colunas a_j que não estão em B. Seja $x_B = (x_{B(1)}\ x_{B(2)}\ \ldots\ x_{B(m)})^T$ e x_N o vetor associado às colunas de N. Então (2.5) pode ser escrita $Bx_B + Nx_N = b$. Assim,

$$x_B = B^{-1}b - B^{-1}Nx_N. \tag{2.7}$$

Dividimos as colunas de N em dois conjuntos, construindo as matrizes O e D, da seguinte forma: estejam em O as colunas a_j associadas a $x_j = 0$ e em D as colunas associadas a $x_j = d$. Podemos então reescrever (2.7) da seguinte forma:

$$x_B = B^{-1}b - B^{-1}Ox_O - B^{-1}Dx_D. \tag{2.8}$$

Seja $\bar{x}_B = B^{-1}b$ e $\hat{x}_B = \bar{x}_B - B^{-1}D\bar{x}_D$, onde $\bar{x}_D = [d\ d\ \ldots\ d]^T$.

Se $0 \leq \hat{x}_{B(i)} \leq d$, $i = 1, 2, \ldots, m$, então B é solução básica viável de (LP), veja-se [Chv83].

Para identificar uma solução básica ótima de (LP), consideremos $u = e^T B^{-1}$, com $e = (1\ 1\ \ldots\ 1)^T \in \mathcal{R}^m$ (lembrando que os $c_j = 1$). Seja $z_j = ua_j$ e $\bar{z}_j = z_j - 1$. Suponhamos que $\bar{z}_j \leq 0$, com j associado a $x_j = d$. Isso acontecerá se (LP) não for vazio. Suponhamos ainda $\bar{z}_j < 0$ para $x_j = d$. Resolve-se então (LP) sem as colunas de O, obtendo uma nova D para a qual $\bar{z} \geq 0$ para todo $x_j = d$.

Consideremos o seguinte subproblema, chamado *oráculo:*

$$(SP): \text{maximizar} \quad ua - 1 \tag{2.9}$$

$$\text{sujeito a} \quad a \in \mathcal{K} \tag{2.10}$$

Seja a_k uma solução ótima de (SP) e $\mathsf{val}(SP) = ua_k - 1$, onde $\mathsf{val}(.)$ é o valor da função objetivo no ótimo de $(.)$.

Consideremos uma família de soluções opt_s de (SP), para $s = 1, 2, \ldots$ tais que

$$\mathsf{val}(\mathsf{opt}_s) \geq \mathsf{val}\left(\mathsf{opt}_{s+1}\right)$$

Então

$$\mathsf{val}(\mathsf{opt}_1) = \mathsf{val}(SP)$$

Definimos

$$I = \{1, 2, \ldots, p\} \tag{2.11}$$
$$I_O = \{j : a_j \in O\} \tag{2.12}$$
$$I_D = \{j : a_j \in D\} \tag{2.13}$$
$$I_B = \{j : a_j \in B\} \tag{2.14}$$

e consideramos o seguinte programa linear:

$$(LP, B, M) : \text{minimizar} \quad ub - \sum_{j \in M} (z_j - 1)\, x_j \tag{2.15}$$
$$\text{sujeito a} \quad x_B = \bar{x}_B - \sum_{j \in M} y_j x_j \tag{2.16}$$
$$0 \leq x_{B(i)} \leq d, i = 1, 2, \ldots, m \tag{2.17}$$
$$0 \leq x_j \leq d \tag{2.18}$$

onde $u = e^T B^{-1}$, $z_j = ua_j$ e $y_j = B^{-1}a_j$.

Dada uma solução básica viável B de (LP), apresentamos o algoritmo COLGEN:

início
    resolva(LP, B, I_D);
    $s := 0$;
1   $s := s + 1$;
    $t := \mathsf{val}(\mathsf{opt}_s) := u\bar{a} - 1$; *(oráculo)*

se $\bar{a} \in D$ vá para 1;
se $t \leq 0$ pare; *(solução ótima encontrada)*
$\bar{a} \in I_O$ entrará em B ou D; *(pivoteamento simplex)*
resolva(LP, B, I_D);
$s := 0$
vá para 1;
fim

Outra versão do algoritmo COLGEN foi proposta por Ribeiro, Minoux e Penna em [RMP89].

Quando o conjunto de todos os $a \in K$ puder ser descrito por

$$Ka \leq h, a \in \{0,1\}^m \tag{2.19}$$

onde $K \in \mathbb{R}^{q \times m}$ é uma matriz conhecida e $h \in \mathbb{R}^q$ é um vetor dado, poderemos apresentar uma versão diferente do algoritmo COLGEN, mostrada mais adiante.

2.3.2 Eliminando um ponto dado

Proporemos uma desigualdade linear para eliminar um ponto dado de um conjunto $\{0,1\}^m$. Seja $\bar{a} \in \{0,1\}^m$ um ponto dado. Desejamos construir uma desigualdade para eliminar apenas $\bar{a}$ em (2.19).

Assim, dado $\mathcal{B} = \{0,1\}^m$, como encontrar uma faceta da envoltória convexa de $\mathcal{B} - \{\bar{a}\}$?

Propomos considerarem-se todos os pontos de uma esfera centrada em $\bar{a}$ com raio 1:

$$\sum_{j=1}^{m} (a_j - \bar{a}_j)^2 = 1 \tag{2.20}$$

Consideremos o hipercubo cujos vértices são todos pontos de $\{0,1\}^m$. Todos os vizinhos de $\bar{a}$ nesse hipercubo pertencem à superfície da esfera (2.20). Assim, escrevendo

$$\sum_{j=1}^{m} (a_j - \bar{a}_j)^2 \geq 1 \tag{2.21}$$

excluiremos $\bar{a}$ incluindo todos os outros pontos de $\mathcal{B}$.
Expandindo o quadrado, chegamos a

$$\sum_{j=1}^{m}(a_j^2 - 2\bar{a}_j a_j + \bar{a}_j^2) \geq 1 \tag{2.22}$$

Como $a_j \in \{0,1\}$, $a_j^2 = a_j$. Assim, escrevemos

$$\sum_{j=1}^{m}(a_j - 2\bar{a}_j a_j + \bar{a}_j) \geq 1 \tag{2.23}$$

o que implica:

$$\sum_{j=1}^{m}(1 - 2\bar{a}_j)a_j \geq 1 - \sum_{j=1}^{m}\bar{a}_j. \tag{2.24}$$

Proposição I: *A restrição (2.21) elimina do conjunto $\mathcal{B}$ o ponto $\bar{a}$ e somente esse ponto.*

Demonstração I:
(i) Elimina o ponto $\bar{a}$:
Basta ver que $\bar{a}$ não satisfaz (2.24)

$$\sum_{j=1}^{m}(\bar{a}_j - \bar{a}_j)^2 = 0 < 1 \tag{2.25}$$

(ii) Não elimina outros:
Por outro lado, seja $\hat{a} \in \mathcal{B}$, $\hat{a} \neq \bar{a}$. Existe um k tal que $\hat{a}_k \neq \bar{a}_k$. Logo,

$$(\hat{a}_k - \bar{a}_k)^2 = 1 \rightarrow \sum_{j=1}^{m}(\hat{a}_k - \bar{a}_k)^2 \geq 1 \tag{2.26}$$

Proposição II: *O hiperplano*

$$\sum_{j=1}^{m}(1 - 2\bar{a}_j)a_j = 1 - \sum_{j=1}^{m}\bar{a}_j \tag{2.27}$$

é faceta da envoltória convexa de $\mathcal{B}$, ou seja, tem dimensão $m-1$.

Demonstração II:

Vamos demonstrar que a superfície contém m pontos afim-independentes.

Inicialmente, observamos que a expressão $\sum_{j=1}^{m}(a_j - \bar{a}_j)^2$ dá o número de coordenadas em que a difere de $\bar{a}$. Observamos também que os pontos vizinhos de $\bar{a}$ no hipercubo são os que diferem de $\bar{a}$ em apenas uma coordenada, para atenderem a (2.27). Vamos chamá-los de a^i, constituindo o conjunto $\{a^1, a^2, \ldots, a^m\}$ de forma a que $a_j^j = \bar{a}_j$ e $a_i^j \neq \bar{a}_j, j \neq i$, ou seja, a^i é o vizinho de $\bar{a}$ que se diferencia de $\bar{a}$ apenas na coordenada i.

Como se trata de m vizinhos, temos nos a^i precisamente m pontos distintos que atendem a (2.27). Vamos a seguir mostrar que eles são afim-independentes.

Para tanto, consideremos os vetores $(a^i - \bar{a})$. Pela definição dos a^i, sabemos que os $(a^i - \bar{a})$ são vetores com todas as coordenadas nulas, exceto i, que pode ser 1 ou -1. O mesmo pode-se dizer dos vetores $(\bar{a} - a^i)$. Assim, observamos que os vetores definidos por

$$(a^i - a^1) = (a^i - \bar{a} + \bar{a} - a^1) = (a^i - \bar{a}) + (\bar{a} - a^1), i = 2, \ldots, m \qquad (2.28)$$

têm nulas todas as coordenadas, exceto i e 1, constituindo um conjunto de $m - 1$ vetores linearmente independentes e paralelos ao hiperplano (2.27), assim definidos pelos pontos a^i que são, portanto, afim-independentes e (2.27) é faceta da envoltória convexa de $\mathcal{B}$.

Outra demonstração da mesma propriedade, bem como interessantes generalizações desse tipo de cortes podem ser vistos em [MM01] e [MMS01].

2.3.3 Um novo algoritmo COLGEN

Utilizando a faceta definida, apresentamos um algoritmo COLGEN especial.

início

1 resolva (LP, B, I_D);
resolva $(SP) \cap \sum_{j=1}^{m}(1 - 2\bar{a}_j)a_j \geq 1 - \sum_{j=1}^{m}\bar{a}_j, \forall \bar{a} \in D$;
val$(SP) := u\hat{a} - 1$; *(oráculo)*
se val$(SP) \leq 0$ pare; *(solução ótima encontrada)*

â entra em B ou D; *(pivoteamento simplex)*
vá para 1;
fim

2.3.4 Solução básica viável inicial

A fim de encontrar uma solução básica viável para (LP) temos que resolver o seguinte programa linear usando o algoritmo COLGEN:

$$(P_1) : \text{minimizar} \quad e^T w \tag{2.29}$$

$$\text{sujeito a} \quad \sum_{j=1}^{p} a_j x_j + w = b, \tag{2.30}$$

$$0 \leq x_j \leq d, j = 1, 2, \ldots, p, w \geq 0. \tag{2.31}$$

Como $b \geq 0$, observamos que $w = b$ e $x_j = 0, j = 1, 2, \ldots, p$ é solução básica viável de (P_1).

Se $\mathsf{val}(P_1) > 0$, então (LP) não tem solução viável. Uma solução básica viável de (LP) estará associada à solução de (P_1) para a qual $\mathsf{val}(P_1) = 0$.

2.4 Resolução de Programas Lineares Inteiros

Definimos um programa linear inteiro da seguinte forma:

$$(IP) : \text{minimizar} \quad z = \sum_{j=1}^{p} x_j \tag{2.32}$$

$$\text{sujeito a} \quad \sum_{j=1}^{p} a_j x_j = b; \tag{2.33}$$

$$0 \leq x_j \leq d, j = 1, 2, \ldots, p; \tag{2.34}$$

$$x_j \in \mathbb{Z}, j = 1, 2, \ldots, p. \tag{2.35}$$

Observe-se que $(IP) = (LP) \cap \{x_j \in \mathbb{Z}, j = 1, 2, \ldots, p\}$, onde $\mathbb{Z}$ é o conjunto dos números inteiros.

A fim de resolver (IP) aplicar-se-ão técnicas de *branch and bound*, veja-se [Dak65] e [NW88]. O algoritmo de geração de colunas será aplicado em cada nó do esquema enumerativo.

Inicialmente resolve-se (LP). Se a solução ótima for inteira ter-se-á resolvido (IP); caso contrário, haverá um k tal que $\hat{x}_{B(k)}$ não é inteiro. Então proceder-se-á à ramificação neste nó resolvendo dois programas lineares:

- $(LP) \cap \{x_{B(k)} \leq \lfloor \hat{x}_{B(k)} \rfloor\}$ e
- $(LP) \cap \{x_{B(k)} \geq \lfloor \hat{x}_{B(k)} \rfloor + 1\}$.

Supondo que i seja o nó corrente na árvore *branch and bound*, o programa linear associado assume a seguinte forma:

$$(LP_i) : \text{minimizar} \quad \sum_{j=1}^{p} x_j \tag{2.36}$$

$$\text{sujeito a} \quad \sum_{j=1}^{p} a_j x_j = b; \tag{2.37}$$

$$\alpha_j \leq x_j \leq \beta_j, j \in S_i; \tag{2.38}$$

$$0 \leq x_j \leq d, j \in I - S_i, \tag{2.39}$$

onde S_i é o conjunto de índices associados a todas as colunas geradas, incluindo as colunas em B, até o nó i. Pode haver algum $j \in S$ para o qual $\alpha_j = 0$ ou $\beta_j = d$.

Uma possível solução básica viável inicial para (LP_i) é uma matriz B que verifique $\alpha_{B(i)} \leq \bar{x}_{B(i)} \leq \beta_{B(i)}$, com $i = 1, 2, \ldots, m$, $\bar{x}_j = \alpha_j$ ou $\bar{x}_j = \beta_j$ se $j \in S_i - I_B$ e $\bar{x}_j = 0$ se $j \in I - S_j$.

Consideremos agora uma solução básica viável de (LP_i) associada a uma solução básica ótima de

$$(LXP_i) : \text{minimizar} \quad \sum_{j \in S_i} x_j \tag{2.40}$$

$$\text{sujeito a} \quad \sum_{j \in S_i} a_j x_j = b; \tag{2.41}$$

$$\alpha_j \leq x_j \leq \beta_j, j \in S_i. \tag{2.42}$$

de modo que:

- $\bar{z}_j = 0$ se $j \in I_B$;

- $\bar{z}_j \leq 0$ se $\bar{x} = \alpha_j$ e $j \in S_i - I_B$;
- $\bar{z}_j \geq 0$ se $\bar{x} = \beta_j$ e $j \in S_i - I_B$.

Se (LP_i) é folha da árvore *branch and bound*, procede-se ao *backtrack*; caso contrário, escolhemos um $k \in I_B$ tal que $\hat{x}_B(k)$ não é inteiro, ramificando da seguinte forma:

- $LP_{i+1} := (LP_i) \cap \{x_{B(k)} \leq \lfloor \hat{x}_{B(k)} \rfloor\}$, e
- $LP_{i+2} := (LP_i) \cap \{x_{B(k)} \geq \lceil \hat{x}_{B(k)} \rceil\}$

Podemos também considerar apenas as colunas já geradas até o nó i:

- $LXP_{i+1} := (LXP_i) \cap \{x_{B(k)} \leq \lfloor \hat{x}_{B(k)} \rfloor\}$, e
- $LXP_{i+2} := (LXP_i) \cap \{x_{B(k)} \geq \lceil \hat{x}_{B(k)} \rceil\}$

É importante notar que uma solução básica ótima de (LXP_i) é dual-viável de LXP_{i+1} e LXP_{i+2}. Assim, iniciando o algoritmo dual simplex com uma solução básica de ótima de (LXP_i), podemos facilmente resolver LXP_{i+1} e LXP_{i+2}(veja-se [Chv83], página 153; [Del74] ou [MFL99]).

Infelizmente, LXP_{i+1} pode ser vazio mesmo que LP_{i+1} não o seja. O mesmo pode ocorrer com LXP_{i+2} com relação a LP_{i+2}. Assim sendo, quando LXP_{i+1} não for vazio, ter-se-á dele uma solução básica ótima que será solução básica inicial para LP_{i+1}, o qual será resolvido usando-se o algoritmo COLGEN. O mesmo será feito a LP_{i+2}, a partir de uma solução básica ótima de LXP_{i+2}.

Se, porém, LXP_{i+1} não tiver solução viável, LP_{i+1} terá que ser resolvido iniciando-se o simplex com uma solução artificial. Uma variável artificial r é introduzida na linha k conforme a seguir:

$$x_{B(k)} + r = \bar{x}_{B(k)} - \sum_{j \in I_O} y_{kj} x_j - \sum_{j \in I_D} y_{kj} x_j \tag{2.43}$$

Seja $x_{B(k)} = \lfloor \hat{x}_{B(k)} \rfloor$. Então

$$\hat{r} = \bar{x}_{B(k)} - \sum_{j \in I_O} y_{kj} \alpha_j - \sum_{j \in I_D} y_{kj} \beta_j - \lfloor \hat{x}_{B(k)} \rfloor > 0 \tag{2.44}$$

Como $a_{B(k)}$ está na base B, tem-se $y_B(k) = B^{-1}a_{B(k)} = e_k$. Se considerarmos $I_D := I_D \cup \{B(k)\}$ e $\beta_{B(k)} = \lfloor \hat{x}_{B(k)} \rfloor$, então pode-se resolver o programa linear seguinte usando o algoritmo COLGEN:

$$
\begin{aligned}
(AP_1) : \text{minimizar} \quad & r && (2.45)\\
\text{sujeito a} \quad & \sum_{j=1}^{p} a_j x_j + a_{B(k)} r = b; && (2.46)\\
& \alpha_j \leq x_j \leq \beta_j, j \in S_i; && (2.47)\\
& 0 \leq x_j \leq d, j \in I - S_i; && (2.48)\\
& r \geq 0 && (2.49)
\end{aligned}
$$

B é solução básica primal viável de (AP_1). Se $\mathsf{val}(AP_1) = 0$ teremos uma solução inicial básica viável para LP_{i+1}; caso contrário LP_{i+1} é vazio e o nó $i+1$ será fechado, procedendo-se ao *backtrack*.

Para a solução de LP_{i+2} quando LXP_{i+2} for vazio usaremos os mesmos resultados apresentados acima. Consideramos $I_O := I_O \cup \{B(k)\}$ e $\alpha_{B(k)} = \lceil \hat{x}_{B(k)} \rceil$ e definimos o seguinte programa linear:

$$
\begin{aligned}
(AP_2) : \text{minimizar} \quad & -r && (2.50)\\
\text{sujeito a} \quad & \sum_{j=1}^{p} a_j x_j + a_{B(k)} r = b; && (2.51)\\
& \alpha_j \leq x_j \leq \beta_j, j \in S_i; && (2.52)\\
& 0 \leq x_j \leq d, j \in I - S_i; && (2.53)\\
& r \geq 0 && (2.54)
\end{aligned}
$$

B será também solução básica primal viável de (AP_2). Usando COLGEN, encontrar-se-á $\mathsf{val}(AP_2)$. Se $\mathsf{val}(AP_2) = 0$, ter-se-á uma solução básica viável para LP_{i+2}; caso contrário LP_{i+2} é vazio.

Capítulo 3

Problemas em Síntese de Redes

... ainda vislumbrava durante o sono a
matriz, a brilhante esteira de lógica
desdobrando-se pelo vazio sem cor...
W. GIBSON, Neuromancer.

3.1 Relevância

A comunicação à distância baseada em impulsos elétricos, inaugurada com o telégrafo em 1837, tem se beneficiado continuamente do progresso tecnológico desde então. Naturalmente, o passar dos anos acompanha o barateamento e popularização dos recursos disponíveis. Hoje, falar à distância é algo muito mais presente no dia-a-dia do homem comum do que jamais poderiam imaginar Morse ou Marconi.

Sejam quais sejam, esses benefícios que a telecomunicação hauriu do avanço da tecnologia terminam sempre por apresentar novos problemas, visto que o aumento da disponibilidade de recursos sempre gera novas formas de utilização e atinge maior número de pessoas. Assim, ao mesmo tempo em que se tornam mais eficientes e baratos, os meios de telecomunicação também aumentam em capacidade e complexidade, criando problemas novos e diferentes na sua gestão. Essa gestão apresenta seu maior desafio ao enfrentar-se o crescimento acelerado da demanda por quantidade e qualidade de serviços.

A demanda por quantidade, ou capacidade, de serviços decorre da crescente disponibilidade de meios dando azo ao surgimento contínuo de

novas e diferentes aplicações, no que se amplia o conjunto de usuários potenciais.

A demanda por qualidade, por sua vez, surge de duas formas: na variedade de usos em novas aplicações, algumas mesmo inimagináveis pouco tempo antes, e na necessidade crescente de eficiência e confiabilidade.

A todos esses pontos, a resposta tem sido uma só: a integração dos serviços em meios de transmissão unificados. Esse mostra ser o melhor caminho para atender à demanda por quantidade porque possibilita economia de escala, quando vários serviços são prestados através de um mesmo meio, por exemplo, voz, dados e vídeo-conferência numa mesma fibra ótica, atendendo a uma clientela heterogênea.

Também a integração dos meios de transmissão atende à demanda por eficiência favorecendo a gestão integrada de diferentes serviços. Isso ocorre porque torna-se possível dividir a banda de transmissão em níveis hierárquicos de multiplexação, os quais se comunicam entre si por protocolos pré-estabelecidos e universais. Isso possibilita, a uma empresa operadora de redes óticas de longa distância oferecer diversos serviços, operados através de suas redes, de forma transparente para o usuário, seja ele um consumidor final ou um outro prestador de serviços.

A busca por essa unificação de meios tem ocupado os recursos de pesquisa disponíveis nos últimos tempos. Procura-se uma definição de protocolos eficientes e aceitáveis universalmente, procura-se a configuração dos meios de transmissão que os capacite tanto à eficiência necessária à diminuição contínua de custos como à necessidade de atenderem-se aumentos de demanda que podem surgir, e certamente surgirão, no futuro próximo. O mais recente padrão internacional de multiplexação é denominado SDH[1]. O SDH possibilitou o empacotamento de contêineres virtuais oriundos de padrões diferentes, como o americano T1 de 1,5 Mb/s [2] e o europeu E1 de 2 Mb/s [3], em módulos unificados de trânsito internacional denominados STM-n, onde n atualmente varia de 1 (155 Mb/s) até 64 (10

[1] *Synchronous Digital Hierarchy* — Hierarquia Digital Síncrona — versão internacional do americano SONET

[2] Megabits por segundo

[3] Adotado no Brasil

Gb/s).

Além da grande integração de meios, outro problema enfrentado pelas empresas do setor é a disponibilidade dos meios de transmissão e, conseqüentemente, a confiabilidade do serviço prestado.

O tema da sobrevivência (*survivability*) de redes tem sido objeto de continuado estudo nos meios acadêmicos. Esses estudos têm servido de base a grandes melhorias em serviços diversos, não apenas os de telecomunicações, mas também todos os que lidem de alguma forma com a idéia de rede, quais sejam, transportes, abastecimento de água, distribuição de energia elétrica e outros. As pesquisas buscam a previsão do comportamento da rede em situações críticas diversas como falhas de ligações ou picos de demanda, buscando-se então a configuração com melhor relação custo-benefício, dadas as restrições de confiabilidade mínima.

Apresentaremos a seguir dois problemas de grande relevância no campo das telecomunicações e que se relacionam com os problemas de geração de colunas sobre os quais tencionamos trabalhar. Trata-se do "Problema da Rede em Malha" e do "Problema da Definição de Anéis". Esses problemas associam-se às duas arquiteturas usadas atualmente nas redes de telecomunicações, desde o nível urbano até os *backbones* internacionais.

3.2 O Problema da Rede em Malha

O modelo mais geral de uma rede de telecomunicações obedece ao paradigma de multifluxo, conforme definido em [GM79]. São dados uma rede, representada por um grafo não direcionado $\mathcal{G}(\mathcal{V}, \mathcal{E})$, e um conjunto $\mathcal{D}$ de comodidades, ou demandas. Uma boa introdução ao projeto de redes em malha pode ser visto em [Min87]. Lisser *et al.*([LSV95]) lida com a sobrevivência de redes.

Em $\mathcal{G}$ têm-se $\mathcal{V}$, o conjunto de nós, que podem representar centrais telefônicas, centros de roteamento ou nós de redes *backbone*, e $\mathcal{E}$, as ligações entre elas, que podem ser cabos de fibra ótica, cabos de cobre, enlaces de rádio ou outros. Podem ser também especificadas as capacidades de fluxo máximo de cada nó em $\mathcal{V}$ e de cada aresta $\mathcal{E}$.

Os modelos clássicos de fluxo em rede sempre representam a demanda como uma função $d : \mathcal{V} \times \mathcal{V} \rightarrow \mathbb{R}$ ou $\mathbb{Z}$, ou seja, um valor dado para todo par

(i, j) de nós. Em certas aplicações a demanda associada a um par de nós deve ser toda alocada a um único caminho, em outras pode ser dividida em valores inteiros, e em outras dividida de forma contínua. Chamamos esses padrões, respectivamente, de "sem divisão", "divisão inteira" e "divisão contínua". Em alguns casos específicos, porém, essa demanda já é dividida *a priori* em pacotes que devem ser tratados distintamente, ou seja, cada um desses pacotes pode ser alocado a um caminho distinto, mas não pode ser dividido em mais de um caminho. Esse é o que chamamos de "divisão pré-definida". Nesse caso a idéia da função d deve ser generalizada especificando-se, para cada elemento $d \in \mathcal{D}$, uma origem ori_d, um destino des_d e um valor val_d. Na verdade essas demandas não têm uma orientação, de vez que sempre é alocada a mesma capacidade de transmissão nos dois sentidos[4]. Quando se fala em origem e destino entenda-se que está sendo atribuída uma orientação arbitrária para as demandas, de forma a tornar possível a modelagem. Basta adotar alguma relação de ordem sobre o conjunto dos nós da rede, e assim orientar cada demanda como existindo do nó de maior ordem para o de menor ordem.

Uma solução do problema consiste numa associação de cada demanda d a um caminho no grafo $\mathcal{G}$. Se for o caso, essas associações devem levar em conta a capacidade das arestas, somando o fluxo total associado a cada aresta, e a capacidade dos nós, somando o fluxo total recebido e emitido por cada nó. Como as demandas não são ordenadas no problema real, sempre que uma rota é determinada para um par origem–destino na solução, interprete-se que o fluxo seguirá essa rota nos dois sentidos.

Se uma associação demanda-caminho for encontrada para todos os $d \in \mathcal{D}$, então existe uma solução de custo zero. Normalmente esse não é o caso, a restrição de capacidade das arestas ou dos nós estará ativa. Abre-se então a possibilidade de se aumentarem as capacidades das arestas ou dos nós, a um custo dado. O problema então transforma-se em um problema de otimização, em que se deseja minimizar o custo total dos aumentos de capacidade de arestas e nós. Essas expansões geralmente não são contínuas, levando o problema a ter cem por cento de variáveis inteiras. Há diversas maneiras de lidar com essa dificuldade, conforme será visto.

A seguir apresentamos o modelo:

[4]Tecnicamente diz-se que é uma rede *full duplex*

consideremos o grafo $\mathcal{G}(\mathcal{V}, \mathcal{D})$ e seu análogo orientado, $\mathcal{G}(\mathcal{V}, \mathcal{A})$, onde

$$[i,j] \in \mathcal{E} \Rightarrow (i,j), (j,i) \subset \mathcal{A}$$

Sejam dados:

- C_i = custo unitário de expansão do nó i;
- C_u = custo unitário de expansão da aresta u;
- P_i = capacidade pré-instalada no nó i;
- P_u = capacidade pré-instalada na aresta u;
- $\mathsf{ori}_d, \mathsf{des}_d, \mathsf{val}_d$ origem, destino e valor da demanda d, $\forall d : \mathsf{ori}_d < \mathsf{des}_d$;

Por simplicidade, adotem-se as convenções:

- $\mathsf{ori}_a, \mathsf{des}_a$: nós origem e extremidade do arco a;
- $u = [i,j] \in \mathcal{E} \Rightarrow u^+ \in \mathcal{A}, u^+ = (i,j)$
- $u = [i,j] \in \mathcal{E} \Rightarrow u^- \in \mathcal{A}, u^- = (j,i)$

E as variáveis:

- $v_i \in \mathbb{Z}$:= expansão do nó i
- $v_u \in \mathbb{Z}$:= expansão da aresta u
- $z_a^d \in \mathbb{Z}$:= quantidade da demanda d que passará pelo arco a

Assim, o problema do fluxo em malha (PFM) pode ser escrito

$$(PFM) : \min \sum_{i \in \mathcal{V}} C_i v_i + \sum_{u \in \mathcal{E}} C_u v_u \tag{3.1}$$

sujeito às restrições de capacidade dos nós

$$\sum_{d \in \mathcal{D}} \sum_{j \in \mathcal{V}} z_{(i,j)}^d + z_{(j,i)}^d \leq P_i + v_i, \forall i \in \mathcal{V} \tag{3.2}$$

às restrições de capacidade das arestas

$$\sum_{d=1}^{\mathcal{D}} z^d_{u^+} + z^d_{u^-} \leq P_u + v_u, \forall u \in \mathcal{E}\left(u^-, u^+ \in \mathcal{A}\right) \tag{3.3}$$

e às de balanço de fluxo

$$\sum_{i \in \mathcal{V}} z^d_{(\mathsf{ori}_d, i)} - z^d_{(i, \mathsf{ori}_d)} = \mathsf{val}_d, \forall d \in \mathcal{D} \tag{3.4}$$

$$\sum_{i \in \mathcal{V}} z^d_{(i,j)} - z^d_{(j,i)} = 0, \left\{ \begin{array}{c} i \neq \mathsf{ori}_d \\ j \neq \mathsf{des}_d \end{array} \right\}, \forall d \in \mathcal{D} \tag{3.5}$$

O problema assim apresentado não é exatamente um problema difícil, especialmente em se considerando que a relaxação linear fornece uma boa cota inferior, veja-se [Pas96]. No caso particular das capacidades iniciais nulas, esse limite é obtido em tempo polinomial e corresponde ao ótimo, desde que os valores val_d das demandas $d \in \mathcal{D}$ sejam inteiros, veja-se [Hu69]. Baseando-se nessa possibilidade pode-se considerar a princípio as capacidades nulas e observar a rede que será construída segundo o modelo. Freqüentemente as capacidades preexistentes aparecerão lá, garantindo uma solução ótima em tempo polinomial.

A arquitetura de rede em malha apresenta duas dificuldades, uma na modelagem e uma especificamente tecnológica.

A primeira é o problema das rotas de proteção. As rotas usualmente determinadas e associadas a cada demanda estão sujeitas a falhas de nó ou falhas de aresta, por contingência de utilização, limitações do equipamento, falhas humanas, etc. Como lidar com essas falhas de forma precisa é uma questão complexa, que envolve estudos estocásticos, em se considerando as probabilidades de falha com proteção, com dupla proteção etc. Os projetistas do setor contornam essas dificuldades usualmente provendo um único caminho alternativo para cada rota, disjunto em arestas e, quando possível, também em nós, do caminho principal. Em caso de falha da linha principal, a rota alternativa, ou *linha de proteção*, é usada. Esse detalhe acrescenta grande dificuldade ao problema em si. A determinacao de uma rota para cada demanda é um problema de estrutura bastante conhecida e de resolução mais fácil. A determinação de caminhos de proteção, porém, é muito mais difícil, devido à exigência de disjunção da rota principal.

Com relação à tecnologia, os nós de uma rede em malha necessitam de um equipamento chamado DXC (*Digital Cross Connect*). Esse equipamento tem a função de *hub*, ou seja, de conectar qualquer aresta incidente com qualquer outra aresta incidente. Tanto a capacidade quanto o custo desse equipamento são altos, de vinte a trinta vezes mais caro que uma expansão de aresta, mas sua capacidade só pode ser aumentada por adição de módulos até certo ponto. A dificuldade na expansão de uma rede baseada em DXC surge exatamente aí. Se essa capacidade máxima for atingida, a compra de um novo equipamento geralmente mostra-se inviável e, mesmo que seja realizada, deixaria uma grande capacidade ociosa no novo equipamento. Por isso, na maior parte dos modelos de expansão de rede *não* se considera a possibilidade de expansão de nó. Da mesma forma, nos problemas que tratamos também não abrimos essa possibilidade. Nos modelos iniciais, porém, ela foi incluída por generalidade.

3.3 O Problema da Definição de Anéis

A solução para o problema das linhas de proteção foi encontrado na adoção da arquitetura em anéis. O anel auto-regenerável, ou SHR (*Self Healing Ring*), provê linhas de proteção na sua própria estrutura. Um SHR pode ser *uni* ou *bidirecional*.

A estrutura SHR mais simples é o anel unidirecional composto por duas fibras, ou linhas, paralelas, chamadas *linha principal* e *linha de proteção*, vide 3.1 Os fluxos nessas linhas correm em sentidos opostos. O fluxo do nó A para o nó genérico C corre normalmente no sentido horário, pelo caminho ABC, e o fluxo de C para A no mesmo sentido, pelo caminho CDA, sempre na *linha principal*. Em caso de falha simples de um nó ou arco, o fluxo principal é interrompido apenas num sentido, assim, quando o sinal de B deixa de chegar em A, por exemplo, por falha no arco BD, A seleciona o sinal equivalente que corre na outra fibra, a *linha de proteção*, pelo caminho BCA, e há tempo para que um reparo seja feito. Como todos os sinais atravessam todo o anel, a soma de todas as demandas alocadas ao anel não pode exceder as capacidade das linhas que o compõem.

Num anel bidirecional o fluxo corre nos dois sentidos, vide 3.2. O fluxo de A

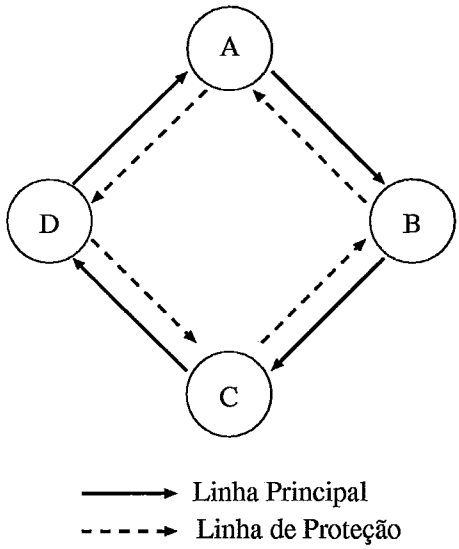

Figura 3.1: Anel auto-regenerativo unidirecional

para B e o fluxo de B para A percorrem o mesmo caminho em sentidos opostos, possivelmente o caminho mais curto AB O caminho complementar é selecionado para ambos em caso de falha. A capacidade de um anel bidirecional limita não a soma dos fluxos em cada arco, mas o máximo deles. Também num anel deste tipo podem-se fazer escolhas de roteamento, o que não é possível no unidirecional.

Uma vez que a restauração é imediata e independe de planejamento, uma rede baseada em anéis é bastante robusta e de manutenção mais fácil que uma malha. O problema da definição dos anéis, entretanto, é um problema combinatório difícil, e que tem ocupado muita capacidade de pesquisa.

O custo de uma rede em anéis é dado pela instalação de equipamentos multiplexadores nos nós, que fazem a ligação nó-anel. O custo do anel físico, isto é, das fibras, por diversas razões, não é considerado aqui.

A modelagem mais simples para uma rede de anéis independentes, ou seja, que não se comunicam entre si, define um anel unidirecional como um subconjunto de nós e, observada a capacidade do anel, tenta alocar as demandas com um custo mínimo. Essa modelagem, entretanto, apresenta algumas limitações. Em primeiro lugar, as variáveis binárias associadas a cada par anel-demanda podem facilmente crescer acima do tratável. Em

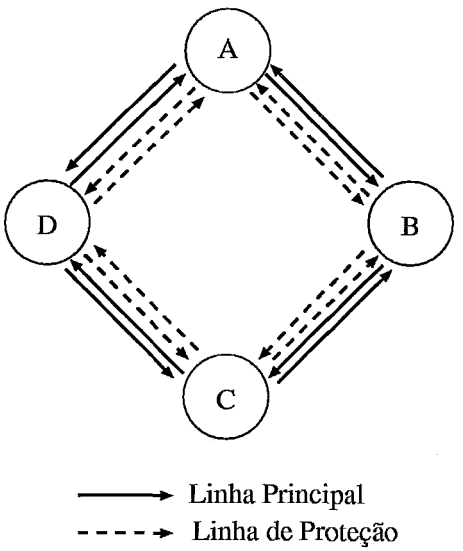

Figura 3.2: Anel auto-regenerativo bidirecional

segundo lugar, algumas restrições técnicas importantes como o limite do comprimento máximo do anel e a distância máxima entre nós vizinhos no anel não é considerada, visto que o traçado não é fornecido pelo modelo.

Outras modelagens buscam construir fisicamente o anel. Isso pode ser feito sobre um fluxo "virtual" unitário que parte de um nó "dummy" acrescentado ao grafo, ou através ciclos-suporte predefinidos, com todas as restrições técnicas aplicadas.

Uma variação importante deste problema é o dos anéis comunicantes ou *emaranhado de anéis*, que significa que um determinado fluxo entre dois nós pode viajar por mais de um anel. Essa variação é especialmente importante quando se consideram redes em anéis "puras", isto é, sem outras arquiteturas coexistentes.

Apresentamos aqui um modelo para definição de anéis unidirecionais estanques numa rede vazia, publicado em [SVW98]. Apesar da elegância trata-se de um modelo teórico apenas porque, para redes reais, o número de variáveis binárias do tipo y_k^d, que atribuem uma demanda d a um anel k, é excessivamente alto. Para estudos em redes pequenas, e para complementar outros modelos, entretanto, ele é bastante útil.

Sejam dados:

- K = número máximo de anéis;

- $\mathcal{D}$ = conjunto de demandas;
- $\mathsf{ori}_d, \mathsf{des}_d, \mathsf{val}_d$ = origem, destino e valor da demanda d, $\forall d : \mathsf{ori}_d < \mathsf{des}_d$
- P_k = capacidade do anel k
- C_i = custo de um multiplexador instalado no nó i

E as variáveis:

- $y_k^d = \begin{cases} 1, \text{se a demanda } d \text{ é alocada ao anel } k \\ 0, \text{caso contrário} \end{cases}$
- $x_k^i = \begin{cases} 1, \text{se é instalado um multiplexador no nó } i \\ 0, \text{caso contrário} \end{cases}$

$$(PDA) : \min \sum_{i=1}^{k} C_i x_k^i \tag{3.6}$$

sujeito às restrições

$$y_k^d \geq x_k^{\mathsf{ori}_d}, \forall d \in \mathcal{D}, k = 1, \ldots, K \tag{3.7}$$

$$y_k^d \geq x_k^{\mathsf{des}_d}, \forall d \in \mathcal{D}, k = 1, \ldots, K \tag{3.8}$$

$$\sum_{k=1}^{K} y_k^d = 1, \forall d \in \mathcal{D} \tag{3.9}$$

$$\sum_{d \in \mathcal{D}} \mathsf{val}_d y_k^d \leq P_k, k = 1, \ldots, K \tag{3.10}$$

das quais 3.7 e 3.8 forçam a instalação de equipamentos nos nós origem e destino de cada demanda, 3.9 aloca cada demanda a um e somente um anel e 3.10 limita o montante de fluxo alocado a cada anel segundo sua capacidade.

Os trabalhos de Vachani *et al.*([VSK96]) e Sutter *et al.*([SVW98]) apresentam estudos sobre definição de anéis.

3.4 O Problema das Duas Arquiteturas

Nos últimos anos, os trabalhos sobre dimensionamento e sobrevivência de redes centraram-se nas arquiteturas de anéis e em malha, tratando dos dois modelos separadamente.

Atualmente muitas empresas constroem redes SDH baseadas em anéis, mas há muitas redes em malha em operação e ainda algumas sendo construídas, obedecendo às necessidades de cada mercado e às realidades dos diferentes cenários, o que, como é usual, leva a soluções distintas.

Nas áreas centrais das grandes metrópoles, normalmente repleta de bancos, instituições financeiras, empresas de telecomunicações etc, aparecem os chamados *cachos*[5], grupos de usuários que habitualmente "falam" entre si. No grafo de chamadas, onde cada arco corresponde a uma chamada entre usuários, os cachos identificam-se como cliques ou quase-cliques. Ao mesmo tempo que esse tráfego concentra-se nas áreas centrais, as áreas periféricas da mesma rede podem apresentar tráfego pequeno. Se a região metropolitana em questão já é servida por uma rede em malha, é possível dar conta do crescimento de demanda através da superposição de anéis sobre a malha existente. Esta idéia, que chamamos que *Solução das Duas Arquiteturas* ou *Problema das Duas Arquiteturas*, foi introduzida em [LMP96d] e [LMP96b] e estudada em [Pas96] e [Bri99]. Uma vez identificados os nós que formam o cacho, buscamos construir os anéis para interconectá-los. Essa solução pode ser economicamente interessante como alternativa para aumentar a capacidade da rede. Essa abordagem pode ainda ser uma boa opção para se evitarem os altos custos dos equipamentos *cross-connect* usados nas redes ponto-a-ponto, uma vez que os anéis necessitam apenas de multiplexadores *add-drop* (ADMs).

O problema é quase uma união dos dois anteriores. São dadas as demandas, as capacidades em malha já instaladas, se for o caso, e busca-se atender as demandas expandindo arestas ou criando-se anéis no cacho. A função objetivo é a soma dos custos das expansões de arestas e dos multiplexadores que comporão os anéis. As restrições de capacidade usuais

[5]na literatura em inglês o termo aparece como *cluster*. Em textos técnicos franceses usa-se o termo *grappe*, que significa "cacho" (de frutas ou flores). Assim, essa tradução pareceu-nos a mais apropriada.

como aresta, nó e anel são aplicáveis.

Vale observar que, para tratar com eficiência do Problema das Duas Arquiteturas, é preciso ter uma modelagem eficiente para cada um dos problemas apresentados, e uma maneira simples e direta de conectar os dois.

3.5 Anéis como Subconjuntos de Nós

O primeiro modelo com que se buscou resolver problema das duas arquiteturas foi descrito detalhadamente em [Pas96]. Trata-se na verdade uma junção dos modelos de malha e anéis anteriormente descritos. Os anéis criados aparecem na solução como subconjuntos de nós, ou seja, o modelo não fornece um traçado para os mesmos ao longo do grafo.

Nesta modelagem, como na da rede em malha, as variáveis z_a^d, que associam a demanda d ao arco a, podem ser relaxadas sem grandes problemas. A dificuldade aparece devido às variáveis binárias y_k^d, que associam a demanda d ao anel k. Da mesma forma como o modelo para rede em anéis, este modelo é bastante sensível ao parâmetro K, que determina o número máximo de anéis. Isso deve-se à variável binária y_k^d, que existe para cada par $(anel_k, demanda_d)$. Assim, se a rede em anéis é apenas um apêndice da rede em malha, caso em que se permite a criação de poucos anéis, o modelo roda sem problemas. Para resultados mais ambiciosos, porém, as dificuldades combinatórias vão tornando-se intransponíveis rapidamente.

O modelo completo segue a seguir.

Criamos as variáveis $s_i, i \in \mathcal{V}$, definidas por

- $s_i = \begin{cases} 1, \text{ se o nó } i \text{ pertence a algum anel;} \\ 0, \text{ caso contrário.} \end{cases}$

A função objetivo é simplesmente a soma da expansão de arestas de 3.1 e da instalação de multiplexadores nos anéis como em 3.6. Não se considera o custo de expansão de nós, pelas razões já expostas:

$$(P2A) : \min \sum_{u \in \mathcal{E}} C_u v_u + \sum_{i=1}^{k} C_i x_k^i \tag{3.11}$$

Sujeito às restrições 3.2, 3.3 e 3.10, de capacidade de nós, arestas e anéis, às de balanço de fluxo genéricas 3.4 e às de balanço de fluxo na origem 3.5, estas últimas modificadas para excluir os fluxos que tenham sido alocados aos anéis, tornando-se

$$\sum_{i \in \mathcal{V}} z^d_{(\mathsf{ori}_d, i)} - z^d_{(i, \mathsf{ori}_d)} = \mathsf{val}_d - \mathsf{val}_d \sum_{k=1}^{K} y^d_k, \forall d \in \mathcal{D} \tag{3.12}$$

Como é necessário alocar equipamentos multiplexadores nos nós que se ligam a um anel temos, correspondendo a 3.7 e 3.8, as restrições 3.13 e 3.14 abaixo:

$$x^i_k \leq s_i, \forall i \in \mathcal{V}, k = 1, \ldots, K \tag{3.13}$$

$$-1 + (s_{\mathsf{ori}_d} + s_{\mathsf{des}_d}) \leq \sum_{k=1}^{K} y^d_k \leq 3 - (s_{\mathsf{ori}_d} + s_{\mathsf{des}_d}), \forall d \in \mathcal{D} \tag{3.14}$$

3.6 Anéis Construídos com Fluxo Virtual

Este modelo busca tratar de uma questão deixada em aberto pelo modelo anterior, qual seja, o traçado dos anéis, podendo lidar com restrições técnicas importantes como comprimento máximo dos anéis e distância máxima entre nós sucessivos num anel. Os primeiros resultados com esta formulação apareceram em [LMPW98]. Dada a sua complexidade, porém, grandes instâncias do Problema das Duas Arquiteturas ainda não podem ser resolvidas eficientemente.

Chamamos aqui de cacho o grupo de nós que se ligam a algum anel, recebendo para isso um multiplexador. A idéia básica é do modelo é, dados os nós candidatos a entrarem no cacho, tomamos um nó boneco [6] $\odot$ auxiliar e criamos um par de arcos entre o nó $\odot$ e cada um dos candidatos. Depois injetamos um "fluxo virtual"[7], numericamente igual a 1 (um), em $\odot$ e a obrigamos a sair também em $\odot$. Assim, com as restrições de

[6] Também chamado *dummy node*.

[7] Assim chamado para evitar confusão com o fluxo "real", originário da demanda entre os nós.

fluxo adequadas, garantimos que o "fluxo virtual" determinado por essa demanda siga um caminho único, fornecendo o traçado do anel.

Dados

- $\mathcal{G}(\mathcal{V}, \mathcal{E})$ um grafo não-orientado;
- $F_{ij} > [i,j] \in \mathcal{D}, \mathcal{D} = \{[i,j] \,|\, i < j, i \in \mathcal{V}, j \in \mathcal{V}\}$ a demanda entre os nós i e j;
- C_i, o custo de cada multiplexador instalado em $i \in \mathcal{V}$;
- C_u, o custo do incremento unitário na capacidade da aresta $u \in \mathcal{E}$
- P_k, a capacidade de cada anel $k \in \{1, 2, \ldots, K\}$, onde K é um limite superior do número de anéis sobre o cacho;
- P_i, a capacidade máxima do nó $i \in \mathcal{V}$;
- $P_u \geq 0$, a capacidade existente da aresta $u \in \mathcal{E}$.
- $\text{dist}_a > 0$, o comprimento do arco a se $a \in \mathcal{A}$ ou aresta a se $a \in \mathcal{E}$;
- MaxAnel> 0, o comprimento máximo de cada anel;
- MaxAdj> 0, a distância máxima entre i e j adjacentes dentro de um mesmo anel;
- $\mathcal{V}_c \subseteq \mathcal{V}$, conjunto dos nós candidatos ao cacho;
- MaxCacho, número máximo de nós no cacho;

Dados auxiliares

- $\mathcal{G}_a = (\mathcal{V}, \mathcal{A})$, um grafo orientado, onde $\mathcal{A} = \{(i,j), (j,i) \,|\, [i,j] \in \mathcal{E}\}$;
- $\mathcal{G}_c = (\mathcal{V}_c, \mathcal{A}_c)$, onde $\mathcal{A}_c = \{(i,j), (j,i) \,|\, [i,j] \in \mathcal{E} \text{ e } i,j \in \mathcal{V}_c\}$, um subgrafo orientado associado ao cacho;
- $\mathcal{G}_\odot = (\mathcal{V}_\odot, \mathcal{A}_\odot)$, onde $\mathcal{V}_\odot = \mathcal{V}_c \cup \{\odot\}$ e $\mathcal{A}_\odot = \mathcal{A}_c \cup \{(\odot, i), (i, \odot) \,|\, i \in \mathcal{G}_c\}$, um grafo sobre o qual são formados os anéis.

Variáveis

- $s_i = \begin{cases} 1 \text{ se o nó } i \text{ pode receber multiplexador (pertence ao cacho);} \\ 0 \text{ caso contrário.} \end{cases}$
- $x_k^i = \begin{cases} 1 \text{ se um multiplexador está instalado no nó } i \text{ no anel } k; \\ 0 \text{ caso contrário.} \end{cases}$
- $w_k^a = \begin{cases} 1 \text{ se o arco } a \in \mathcal{A}_\odot \text{ pertence ao anel } k; \\ 0 \text{ caso contrário.} \end{cases}$
- $v_u \in \mathbb{R}_+$, a capacidade a ser instalada sobre a aresta $u \in \mathcal{E}$;
- $z_a^d \in \mathbb{R}_+$, o fluxo sobre o arco $a \in \mathcal{A}$ associado à demanda d;
- $y_{lk}^a \in \mathbb{R}_+$ o fluxo virtual sobre o arco $a \in \mathcal{A}_\odot$ associado à demanda virtual que vai de $\odot \in \mathcal{G}_\odot$ a $l \in \mathcal{G}_c$ sobre o anel k.

Variáveis auxiliares

Definimos aqui duas variáveis auxiliares t_k^a e p_k^a , $a \in \mathcal{A}, k = 1, \ldots, K$, da seguinte forma:

- $t_k^{ij} \in \mathbb{R}_+, t_k^{ij} = x_k^i x_k^j, i, j \in \mathcal{V}_c$
- $p_k^a \in \mathbb{R}_+, p_k^a = w_k^{(\text{ori}_a, \odot)} w_k^{(\odot, \text{des}_a)}$

O domínio de t_k^a e de p_k^a é $\mathbb{R}_+$, mas na verdade elas assumirão sempre valores em $\{0, 1\}$, podendo ser interpretadas da seguinte forma:

- $t_k^{ij} = \begin{cases} 1 \text{ se ambos os nós } i \text{ e } j \text{ estão no anel } k; \\ 0 \text{ caso contrário.} \end{cases}$
- $p_k^a = \begin{cases} 1 \text{ se o arco } a \in \mathcal{A}_c \text{ é alocada ao anel } k; \\ 0 \text{ caso contrário.} \end{cases}$

Essa interpretação de p_k^a auxilia a enxergar o processo de construção dos anéis. Injetamos o fluxo virtual de valor unitário num nó *dummy* $\odot$, que deve necessariamente sair nesse mesmo nó. Como o fluxo não pode ser dividido, percorrerá um único caminho, formando um ciclo que começa e termina em $\odot$. Exigiremos adicionalmente que o último nó visitado pelo fluxo antes de $\odot$, digamos o nó α, seja vizinho (i.e., tenha um

arco conectando-o) ao primeiro visitado *depois* de $\odot$, digamos β. Assim, quando o ciclo for fechado, bastará remover dele os arcos $(\odot, \beta)$ e $(\alpha, \odot)$ e fechá-lo com (α, β), que teremos um ciclo dentro do grafo $\mathcal{G}_c$ original. A variável p_k^a faz exatamente esse controle, associando ao anel k um tal arco $(\alpha, \beta) \in \mathcal{A}_c$ que resulte de um produto da forma $w_k^{(\mathsf{ori}_a, \odot)} w_k^{(\odot, \mathsf{des}_a)}$.

Já t_k^{ij} será usada na composição dos balanços de fluxo, possibilitando descontar as demandas alocadas a anéis.

A não-linearidade dessas definições pode ser contornada por definições alternativas, conforme será visto na seção 3.6.

Restrições

Número máximo de nós no cacho (3.15) e instalação de multiplexadores nos nós (3.16):

$$\sum_{i \in \mathcal{V}_c} s_i \leq \mathsf{MaxCacho}, s_i \in \{0,1\}, i \in \mathcal{V}_c \tag{3.15}$$

$$x_k^i \leq s_i, i \in \mathcal{V}_c, k = 1, \ldots, K \tag{3.16}$$

As restrições 3.17 e 3.18 realizam o balanço do fluxo virtual:

$$\sum_{\beta} y_{\odot\beta}^{lk} - \sum_{\beta} y_{\beta\odot}^{lk} = x_k^l, l \in \mathcal{V}_c, k = 1, \ldots, K \tag{3.17}$$

$$\sum_{\beta} y_{(\alpha,\beta)}^{lk} - \sum_{\beta} y_{(\beta,\alpha)}^{lk} = 0, \alpha \in \mathcal{V}_c - \{l\}, l \in \mathcal{V}_c, k = 1, \ldots, K \tag{3.18}$$

3.19, 3.20 e 3.21 estão associadas à unicidade e conexidade dos anéis.

$$\sum_{\beta \in \mathcal{V}_c} w_k^{(\odot,\beta)} = \sum_{\beta \in \mathcal{V}_c} w_k^{(\beta,\odot)} \leq 1, \alpha \in \mathcal{V}_c, k = 1, \ldots, K \tag{3.19}$$

$$\sum_{\beta \in \mathcal{V}_c} w_k^{(\alpha,\beta)} = \sum_{\beta \in \mathcal{V}_c} w_k^{(\beta,\alpha)} = x_k^\alpha, \alpha \in \mathcal{V}_c, k = 1, \ldots, K \tag{3.20}$$

$$y_{lk}^a \leq w_k^a, l \in \mathcal{V}_c, a \in \mathcal{A}_\odot, k = 1, \ldots, K \tag{3.21}$$

As restrições 3.22 a 3.24 asseguram respectivamente que dois nós sucessivos num anel nunca ultrapassarão a distância máxima MaxAdj e que o comprimento total de cada anel ficará aquém do limite MaxAnel:

$$\begin{aligned}
\mathsf{dist}_a w_a^k &\leq \mathsf{MaxAdj}, a \in \mathcal{A}_c, k = 1, \ldots, K && (3.22)\\
\mathsf{dist}_a p_k^a &\leq \mathsf{MaxAdj}, a \in \mathcal{A}_c, k = 1, \ldots, K && (3.23)\\
\sum_{a \in \mathcal{A}_c} \mathsf{dist}_a \left(w_a^k + p_a^k\right) &\leq \mathsf{MaxAnel}, k = 1, \ldots, K && (3.24)
\end{aligned}$$

As restrições 3.25 e 3.26 realizam o balanço do fluxo (real) nos nós da malha:

$$\begin{aligned}
\sum_{\beta} z^d_{(\alpha,\beta)} - \sum_{\beta} z^d_{(\beta,\alpha)} &= 0, \alpha \in \mathcal{V} - \{\mathsf{ori}_d, \mathsf{des}_d\}, d \in \mathcal{D} && (3.25)\\
\sum_{\beta} z^d_{(\mathsf{ori}_d,\beta)} - \sum_{\beta} z^d_{(\beta,\mathsf{ori}_d)} &= \mathsf{val}_d - \mathsf{val}_d \sum_{k=1}^{K} t_k^d, d \in \mathcal{D} && (3.26)
\end{aligned}$$

As restrições de capacidade seguem-se quase idênticas ao modelo da seção 3.5. 3.27, 3.28 e 3.29 tratam respectivamente das capacidades de anéis, nós e arestas. Mais uma vez não cogitamos de expansões de nós:

$$\begin{aligned}
\sum_{d \in \mathcal{D}} \mathsf{val}_d y_k^d &\leq C_k, k = 1, \ldots, K && (3.27)\\
\sum_{\beta \in \mathcal{V}} \sum_{d \in \mathcal{D}} z^d_{(\alpha,\beta)} + z^d_{(\beta,\alpha)} &\leq P_\alpha, \alpha \in \mathcal{V} && (3.28)\\
\sum_{[i,j] \in \mathcal{D}} \left(z^{ij}_{(\alpha,\beta)} + z^{ij}_{(\beta,\alpha)}\right) &\leq P_{[\alpha,\beta]} + v_{[\alpha,\beta]}, [\alpha,\beta] \in \mathcal{E} && (3.29)
\end{aligned}$$

Função Objetivo

A função objetivo apresenta uma soma dos custos totais de expansão de arestas e de instalação de multiplexadores:

$$\min \sum_{k=1}^{K} \sum_{\alpha \in \mathcal{V}_c} C_\alpha x_\alpha^k + \sum_{u \in \mathcal{E}} C_u v_u \qquad (3.30)$$

Linearização

Produtos entre variáveis binárias podem ser linearizados, eventualmente com a introdução de variáveis auxiliares. Aqui mostramos o que pode ser feito no caso das variáveis t_k^a e p_k^a, definidas como produtos. Propomos substituir as definições dadas anteriormente pelas seguintes restrições:

$$
\begin{aligned}
p_k^{(\alpha,\beta)} &\leq w_k^{(\alpha,\odot)} && (3.31)\\
p_k^{(\alpha,\beta)} &\leq w_k^{(\odot,\beta)} && (3.32)\\
w_k^{(\alpha,\odot)} + w_k^{(\odot,\beta)} - 1 &\leq p_k^{(\alpha,\beta)} && (3.33)\\
p_k^{(\alpha,\beta)} &\geq 0 && (3.34)\\
(\alpha,\beta) \in \mathcal{A}_c &\quad , \quad k = 1,\ldots,K &&\\
t_k^{ij} &\leq x_k^i && (3.35)\\
t_k^{ij} &\leq x_k^j && (3.36)\\
x_k^i + x_k^j - 1 &\leq t_k ij && (3.37)\\
t_k^{ij} &\geq 0 && (3.38)\\
i \neq j, \{i,j\} \subset \mathcal{V}_c &\quad , \quad k = 1,\ldots,K &&
\end{aligned}
$$

assim temos uma formulação 100% linear.

3.7 Anéis Construídos a Partir de Ciclos Suporte Pré-Gerados

Trata-se da mais profícua dentre as três abordagens aqui apresentadas, que busca o traçado dos anéis a partir de ciclos presentes no grafo. Como a conexidade[8] das redes de telecomunicações não é alta, tipicamente abaixo de 2, o número de ciclos também não é excessivamente grande. Uma redução considerável pode ser obtida aplicando-se as restrições habituais de comprimento máximo, número máximo de nós participantes e distância máxima entre nós vizinhos, veja-se [Bri99]. O capítulo 4 trata da utilização dessa abordagem na geração de colunas.

[8] razão entre o número de nós e o número de arestas do grafo

Capítulo 4

Problema e Resultados

O problema com os problemas —
raciocinava Gossage — é que,
uma vez resolvidos, deixam de
ser problemas. Decididamente,
jamais ganharei a vida com isso.
GERSHWALD, O Sicofanta.

4.1 Modelo Matemático

O modelo matemático a ser proposto é derivado dos modelos apresentados no capítulo 3. Consideramos um grafo suporte no qual se estuda a ampliação de capacidade da rede em malha e também a implantação de anéis. Definimos os dados, as restrições e a função objetivo para o problema.

Os trabalhos citados que lidaram com este problema modelaram anéis como subconjuntos de nós, sem considerar seu traçado real ao longo da rede. Isto pode levar a anéis inviáveis por causa de algumas restrições técnicas, como comprimento máximo do anel e distância máxima entre nós vizinhos. Assim, introduzimos o conceito de *ciclo suporte*: um ciclo no grafo da rede atuando como esteio para um anel. Cada anel possui um ciclo suporte e todos os nós pertencentes a um anel devem pertencer a seu ciclo suporte.

Nas definições a seguir, considere-se o *cacho* como sendo o subconjunto de nós efetivamente pertencentes a um ou mais anéis na solução final. Nos

dados do problema é definido o subconjunto de nós candidatos a estarem no cacho, veja-se $\mathcal{V}_c$a seguir.

Dados

- $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ grafo suporte não orientado;
- $F_{ij} > 0, [i,j] \in \mathcal{D}, \mathcal{D} = \{[i,j] | i < j, i \in \mathcal{V}, j \in \mathcal{V}\}$, demanda entre os nós i e j;
- $C_i > 0$, custo de cada multiplexador instalado em $i \in \mathcal{V}$;
- Capacidades a serem instaladas em cada aresta: algum múltiplo de $2, 34, 155$ tendo como custo $C^{(2)} > 0, C^{(34)} > 0, C^{(155)} > 0$ respectivamente. Supomos $\frac{C^{(155)}}{155} \leq \frac{C^{(34)}}{34} \leq \frac{C^{(2)}}{2}$;
- CapAnel, capacidade de cada anel;
- $P_i > 0$, capacidade máxima do nó $i \in \mathcal{V}$, relativa à malha;
- dist_{ij}>0, comprimento de cada aresta $[i,j] \in \mathcal{E}$;
- MaxAnel > 0, comprimento máximo de cada anel;
- $K \in \mathbb{Z}_+$, número máximo de nós num anel;
- $\mathcal{V}_c \subseteq \mathcal{V}$, conjunto de vértices candidatos a estarem no cacho;
- $T_u \geq 0$, capacidade existente (pré-instalada) na aresta $u \in \mathcal{E}$;
- a_k^{ij}, demanda entre os nós i e j satisfeita pelo anel k, $[i,j] \in \mathcal{D}$, $k = 1, 2, ..., K$ e K é o número total de anéis viáveis;
- $0 \leq f_{ij} \leq F_{ij}$, onde f_{ij} uma parte da demanda a ser roteada pelo anel;
- h_k, custo do anel k.

Dados Auxiliares

- $\mathcal{G}_a = (\mathcal{V}, \mathcal{A})$, grafo orientado correspondente a $\mathcal{G}$, onde $\mathcal{A} = \{(i,j),(j,i)|[i,j] \in \mathcal{E}\}$.
- $\mathcal{G}_c = (\mathcal{V}_c, \mathcal{A}_c)$, onde $\mathcal{A}_c = \{(i,j),(j,i)|[i,j] \in \mathcal{E} \text{ and } i,j \in \mathcal{V}_c\}$, subgrafo orientado associado ao cacho;
- $\mathcal{G}_\odot = (\mathcal{V}_\odot, \mathcal{A}_\odot)$, onde $\mathcal{V}_\odot = \mathcal{V}_c \cup \{\odot\}$ e $\mathcal{A}_\odot = \{(\odot,i),(i,\odot)|i \in \mathcal{V}_c\} \cup \mathcal{A}_c$, grafo sobre o qual formulamos os anéis;

Variáveis

- $t_i \in \{0,1\}$, $t_i = 1$ se o nó i for escolhido para pertencer a um ciclo suporte, $t_i = 0$ caso contrário.
- $r_i \in \{0,1\}$, $r_i = 1$ se o nó i for escolhido para receber um multiplexador, $r_i = 0$ caso contrário.
- $v_u^{(2)}, v_u^{(34)}, v_u^{(155)} \in \mathbb{Z}_+, u \in \mathcal{E}$, as capacidades de tipo 2, 34 e 155 a serem instaladas na aresta u.
- $z_{\alpha\beta}^{ij} \geq 0$, fluxo passante no arco $(\alpha,\beta) \in \mathcal{A}_c$ associado à demanda entre i e j.
- $y_{\alpha\beta}^{l} \geq 0$, fluxo passante no arco $(\alpha,\beta) \in \mathcal{A}_\odot$ associado ao fluxo virtal de $\odot \in \mathcal{V}_\odot$ a $l \in \mathcal{V}_c$ sobre um anel.
- $w_{\alpha\beta} \in \{0,1\}$, $w_{\alpha\beta} = 1$ se o arco $(\alpha\beta) \in \mathcal{A}_\odot$ pertence a um anel, $w_{\alpha\beta} = 0$ caso contrário.
- $p_{\alpha\beta} = w_{\alpha 0}.w_{0\beta}, \alpha \neq \beta, \alpha \neq 0, \beta \neq 0$.
- $q_{ij} \in \{0,1\}$, $q_{ij} = 1$ se há um fluxo de i a j por um anel e $q_{ij} = 0$ caso contrário, $i,\ j \in \mathcal{V}_c$ e $i < j$.
- $x_k \in Z_+$, o número de anéis k.

Restrições

$$\sum_{\beta} z_{i\beta}^{ij} - \sum_{\beta} z_{\beta i}^{ij} + \sum_{k=1}^{K} a_k^{ij} x_k \geq F_{ij},\ [i,j] \in \mathcal{D} \tag{4.1}$$

$$\sum_{\beta} z_{\alpha\beta}^{ij} - \sum_{\beta} z_{\beta\alpha}^{ij} = 0,\ \alpha \in \mathcal{V} - \{i,j\},\ [i,j] \in \mathcal{D} \tag{4.2}$$

$$\sum_{\beta} \sum_{[i,j]\in\mathcal{D}} z_{\alpha\beta}^{ij} + z_{\beta\alpha}^{ij} \leq P_\alpha,\ \alpha \in \mathcal{V} \tag{4.3}$$

$$\sum_{[i,j]\in\mathcal{D}} (z_{\alpha\beta}^{ij} + z_{\beta\alpha}^{ij}) \leq T_{\alpha\beta} + 2v_{\alpha\beta}^{(2)} + 34v_{\alpha\beta}^{(34)} + 155v_{\alpha\beta}^{(155)},$$
$$\alpha < \beta,\ [\alpha,\beta] \in \mathcal{E} \tag{4.4}$$

$$x_k \in Z_+,\ k = 1, 2, ..., K; \tag{4.5}$$

$$v_{\alpha\beta}^{(2)},\ v_{\alpha\beta}^{(34)},\ v_{\alpha\beta}^{(155)} \in \mathbb{Z}_+,\ [\alpha,\beta] \in \mathcal{E} \tag{4.6}$$

$$z_{\alpha\beta}^{ij} \geq 0,\ (\alpha,\beta) \in \mathcal{A}_c,\ [i,j] \in \mathcal{D} \tag{4.7}$$

As restrições (4.1) e (4.2) garantem que as demandas F_{ij} sejam satisfeitas. A desigualdade (4.3) limita a capacidade de cada vértice α. A expressão (4.4) representa a restrição de capacidade para a aresta $[\alpha,\beta]$. A restrição (4.6) e (4.7) define os conjuntos aos quais as variáveis de decisão devem pertencer.

Função Objetivo

$$\text{Minimizar} \sum_{k=1}^{K} h_k x_k + \sum_{[\alpha,\beta]\in\mathcal{E}} (C^{(2)} v_{\alpha\beta}^{(2)} + C^{(34)} v_{\alpha\beta}^{(34)} + C^{(155)} v_{\alpha\beta}^{(155)}) \tag{4.8}$$

Esta função objetivo representa a soma dos custos de construção dos anéis e expansão de arestas.

Minimizar (4.8) sob as restrições (4.1 – 4.7) constitui um problema de otimização combinatória extremamente difícil nas instâncias com que pretendemos lidar. Pode-se, entretanto, conceber o modelo simplificado a seguir.

4.2 Modelo Simplificado

A fim de criar um modelo mais simples, removemos a restrição de capacidade de nós (4.3) e supomos que $2v_{\alpha\beta}^{(2)} + 34v_{\alpha\beta}^{(34)} + 155v_{\alpha\beta}^{(155)} = v_{\alpha\beta}$ e $(C^{(2)}v_{\alpha\beta}^{(2)} + C^{(34)}v_{\alpha\beta}^{(34)} + C^{(155)}v_{\alpha\beta}^{(155)}) = Cv_{\alpha\beta}$, onde a variável contínua $v_{\alpha\beta}$ representa a expansão de capacidade da aresta $[\alpha, \beta]$ e C o custo unitário dessa expansão. O problema a ser resolvido pode então ser escrito da seguinte forma:

$$(P): \text{ minimizar} \sum_{k=1}^{K} h_k x_k + \sum_{[\alpha,\beta]\in\mathcal{E}} Cv_{\alpha\beta} \tag{4.9}$$

sob as restrições:

$$\sum_{\beta} z_{i\beta}^{ij} - \sum_{\beta} z_{\beta i}^{ij} + \sum_{k=1}^{K} a_k^{ij} x_k \geq F_{ij},\ [i,j] \in \mathcal{D} \tag{4.10}$$

$$\sum_{\beta} z_{\alpha\beta}^{ij} - \sum_{\beta} z_{\beta\alpha}^{ij} = 0,\ \alpha \in \mathcal{V} - \{i,j\},\ [i,j] \in \mathcal{D} \tag{4.11}$$

$$\sum_{[i,j]\in\mathcal{D}} (z_{\alpha\beta}^{ij} + z_{\beta\alpha}^{ij}) \leq T_{\alpha\beta} + v_{\alpha\beta},\ [\alpha,\beta] \in \mathcal{E} \tag{4.12}$$

$$x_k \in Z_+, k = 1,2,...,K; v_{\alpha\beta} \geq 0, [\alpha,\beta] \in \mathcal{E}; z_{\alpha\beta}^{ij} \geq 0, (\alpha,\beta) \in \mathcal{A}_c,\ [i,j] \in \mathcal{D} \tag{4.13}$$

Um primeiro passo para a solução de (P) é a resolução de sua relaxação linear, doravante chamada $(\bar{P})$. Introduzem-se ainda as variáveis de folga w_{ij} e s_{ij} respectivamente em 4.10 e 4.12. Assim, obtém-se:

$$(\bar{P}): \text{ minimizar} \sum_{k=1}^{K} h_k x_k + \sum_{[\alpha,\beta]\in\mathcal{E}} Cv_{\alpha\beta} \tag{4.14}$$

sob as restrições

$$\sum_{\beta} z_{i\beta}^{ij} - \sum_{\beta} z_{\beta i}^{ij} + \sum_{k=1}^{K} a_k^{ij} x_k - w_{ij} = F_{ij},\ [i,j] \in \mathcal{D} \tag{4.15}$$

$$\sum_{\beta} z_{\alpha\beta}^{ij} - \sum_{\beta} z_{\beta\alpha}^{ij} = 0,\ \alpha \in \mathcal{V} - \{i,j\},\ [i,j] \in \mathcal{D} \tag{4.16}$$

$$\sum_{[i,j]\in\mathcal{D}} (z^{ij}_{\alpha\beta} + z^{ij}_{\beta\alpha}) - v_{\alpha\beta} + s_{\alpha\beta} = T_{\alpha\beta},\ [\alpha,\beta] \in \mathcal{E} \tag{4.17}$$

$$x_k \in R_+,\ k = 1, 2, ..., K;\ v_{\alpha\beta} \geq 0,\ s_{\alpha\beta} \geq 0,\ [\alpha,\beta] \in \mathcal{E};$$

$$w_{ij} \geq 0,\ [i,j] \in \mathcal{D};\ z^{ij}_{\alpha\beta} \geq 0,\ (\alpha,\beta) \in \mathcal{A}_c,\ [i,j] \in \mathcal{D}. \tag{4.18}$$

Suponhamos que val(.) seja o valor da função objetivo no ponto ótimo de (.). O valor de val($\bar{P}$) fornecerá uma quota inferior para val(P). Resolveremos ($\bar{P}$) usando o método da geração de colunas, vejam-se [DW60], [GG61] e [GG63].

4.3 Geração de Colunas a_k

Nesta seção usamos as técnicas para geração de colunas em programação linear, vejam-se também [Chv83] e [Las70].

O programa linear ($\bar{P}$) pode ser resolvido pelo método do simplex. Seja u o vetor linha dos multiplicadores do simplex associados a uma solução primal-viável. Assim, $u^T \in \mathbb{R}^\delta$, onde δ é o número de linhas advindas das restrições (4.15) a a (4.17).

O vetor coluna a_k, no qual os elementos são iguais a a^{ij}_k nas restrições (4.15) e zero em todas as outras, pode ser conhecido *a priori* ou gerado se necessário.

A fim de saber se uma coluna a_k, no momento ausente da base, é candidata a entrar, deve-se calcular $\bar{h}_k = h_k - ua_k$ (custo reduzido associado à coluna a_k). Se $\bar{h}_k < 0$, então a_k entrará na nova base e necessitar-se-á encontrar a coluna a deixar a base (método primal simplex).

Assim, é possível determinar uma coluna a_l tal que

$$\bar{h}_l = h_l - ua_l = \min_{k=1,...,K}\{h_k - ua_k\}$$

Se $\bar{h}_l > 0$, nenhuma coluna a_k é candidata a entrar na base.

4.3.1 Um método automático

Sabe-se que uma coluna a_k está associada a um anel k. Destarte, passamos à modelagem de um anel viável k de modo a gerar uma coluna a_k.

Desenvolvemos um procedimento, apresentado em [LMPW98], que consiste em introduzir um nó boneco $\odot$ [1] no grafo $\mathcal{G}_c$. A partir desse nó constroem-se arcos ligando-o a cada nó candidato (esse processo constrói o grafo $\mathcal{G}_\odot$). Então um fluxo virtual é injetado e extraído em $\odot$. Prevenindo que se bifurque ou que cruze consigo mesmo, obrigamo-lo a seguir um único caminho através do subgrafo de candidatos. Naturalmente, o primeiro nó depois e o último antes de $\odot$ devem ser adjacentes, para que o ciclo se feche. Assim é criado um ciclo no grafo de candidatos.

Isto será feito sobre o grafo $\mathcal{G}_\odot$, de modo que incluem-se nos conjuntos $\mathcal{A}_c$ e $\mathcal{A}_\odot$ todos os arcos $(i,j) \notin \mathcal{A}_c, i \neq j, i,j \in \mathcal{V}_c$. A distância entre i e j para estes arcos será MaxAnel + 1.

Introduzimos as seguintes variáveis:

- $t_i \in \{0,1\}$, $t_i = 1$ se o vértice i é pertence a um ciclo suporte, $t_i = 0$ caso contrário;
- $y^l_{\alpha\beta} \geq 0$, fluxo virtual no arco $(\alpha,\beta) \in \mathcal{A}_\odot$ associado ao fluxo virtual de $\odot \in \mathcal{V}_\odot$ a $l \in \mathcal{V}_c$ por um anel;
- $w_{\alpha\beta} \in \{0,1\}$, $w_{\alpha\beta} = 1$ se o arco $(\alpha,\beta) \in \mathcal{A}_\odot$ pertence a um anel, $w_{\alpha\beta} = 0$ caso contrário;
- $p_{\alpha\beta} = w_{\alpha\odot}.w_{\odot\beta}, \alpha \neq \beta, \alpha \neq 0, \beta \neq 0$;
- $q_{ij} \in \{0,1\}$, $q_{ij} = 1$ se há um fluxo de i a j por algum anel e $q_{ij} = 0$ caso contrário, $i,j \in \mathcal{V}_c$ e $i < j$.

O modelo que constrói um anel é:

$$\sum_\beta y^l_{\odot\beta} - \sum_\beta y^l_{\beta\odot} = t_l,\ l \in \mathcal{V}_c, \tag{4.19}$$

$$\sum_\beta y^l_{\alpha\beta} - \sum_\beta y^l_{\beta\alpha} = 0, \alpha \in \mathcal{V}_c - \{l\},\ l \in \mathcal{V}_c, \tag{4.20}$$

$$\sum_\beta w_{\odot\beta} = \sum_\beta w_{\beta\odot} = 1, \tag{4.21}$$

$$\sum_\beta w_{\alpha\beta} = \sum_\beta w_{\beta\alpha} = t_\alpha,\ \alpha \in \mathcal{V}_c, \tag{4.22}$$

[1] vide seção 3.6

$$0 \leq y^l_{\alpha\beta} \leq w_{\alpha\beta}, l \in \mathcal{V}_c, (\alpha,\beta) \in \mathcal{A}_\odot, \tag{4.23}$$

$$p_{\alpha\beta} \geq 0,\ p_{\alpha\beta} \leq w_{\alpha\odot},\ p_{\alpha\beta} \leq w_{\odot\beta},\ (\alpha,\beta) \in \mathcal{A}_c, \tag{4.24}$$

$$w_{\alpha\odot} + w_{\odot\beta} - 1 \leq p_{\alpha\beta},\ (\alpha,\beta) \in \mathcal{A}_c, \tag{4.25}$$

$$w_{\alpha\beta} \in \{0,1\},\ (\alpha,\beta) \in \mathcal{A}_\odot, \tag{4.26}$$

$$\sum_{(\alpha,\beta)\in\mathcal{A}_c} \mathsf{dist}_{\alpha\beta} w_{\alpha\beta} + \sum_{(\alpha,\beta)\in\mathcal{A}_c} \mathsf{dist}_{\alpha\beta} p_{\alpha\beta} \leq \mathsf{MaxAnel}, \tag{4.27}$$

$$q_{ij} \in \{0,1\},\ q_{ij} \leq t_i,\ q_{ij} \leq t_j,\ i<j,\ i,j \in \mathcal{V}_c, \tag{4.28}$$

$$1 \geq r_i \geq q_{ij}, i<j,\ i,j \in \mathcal{V}_c, \tag{4.29}$$

$$1 \geq r_j \geq q_{ij}, i<j,\ i,j \in \mathcal{V}_c, \tag{4.30}$$

$$\sum_{i<j,\ i,j\in\mathcal{V}_c} f_{ij} q_{ij} \leq \mathsf{CapAnel}, \tag{4.31}$$

$$\sum_{i\in\mathcal{V}_c} t_i \leq K. \tag{4.32}$$

Isso posto, podemos escrever: $h_k = \sum_{i\in\mathcal{V}_c} C_i r_i$ e $a^{ij}_k = f_{ij} q_{ij}$.

Seja u^{ij} o elemento do vetor u associado à restrição (4.15) relacionada à demanda entre i e j, $[i,j] \in \mathcal{D}$ e $i,j \in \mathcal{V}_c$. Pode-se assim apresentar a função objetivo do problema que gerará um a_k da seguinte maneira:

$$h_k - u a_k = \sum_{i\in\mathcal{V}_c} C_i r_i - \sum_{i<j,\ i,j\in\mathcal{V}_c} u^{ij} a^{ij}_k = \sum_{i\in\mathcal{V}_c} C_i r_i - \sum_{i<j,\ i,j\in\mathcal{V}_c} u^{ij} f_{ij} q_{ij}$$

Assim deve-se minimizar

$$\sum_{i\in\mathcal{V}_c} C_i r_i - \sum_{i<j,\ i,j\in\mathcal{V}_c} u^{ij} f_{ij} q_{ij} \tag{4.33}$$

sob as restrições (4.19) a (4.32).

As restrições (4.19) a (4.27) e (4.32) definem os ciclos contidos no grafo que poderiam servir de suporte a anéis; em outras palavras, são o que chamamos de *ciclos viáveis*. (4.19) e (4.20) realizam o balanço de fluxo como exposto acima, se $t_i = 1$ então um fluxo virtual é injetado em $\odot$ para construir um ciclo. (4.21) garante que $\odot$ seja visitado apenas uma vez, (4.22) amarra a presença de um arco no ciclo com as variáveis t_i, (4.23) garante que um arco esteja presente se algum fluxo o atravessa. (4.24) e (4.25)

implementam em restrições lineares a condição $p_{\alpha\beta} = w_{\alpha\odot}.w_{\odot\beta}$; note-se que p apenas funciona como um w para o arco que "fecha" o anel entre o primeiro e o último nós, ambos ligados diretamente a $\odot$, conforme exposto. (4.27) limita o comprimento total do ciclo. Resumindo, as retrições de (4.19) a (4.27) definem o ciclo viável e as de (4.28) a (4.30) põem um anel sobre o ciclo de forma a minimizar a função objetivo (4.33), a qual fornecerá o custo reduzido da coluna associada ao anel gerado.

Se não houver nenhum $u^{ij} > 0$, então não haverá nenhuma coluna associada aos anéis com custo reduzido negativo.

Para $|\mathcal{V}| = 30$, o número de varíaveis 0–1 ultrapassa dois mil. A fim de reduzir esse número podem-se promover algumas modificações no modelo, o qual, todavia, permanecerá muito difícil de resolver em circunstâncias práticas. Assim sendo, outras técnicas devem ser pesquisadas.

4.3.2 Geração de ciclos viáveis

Um anel é formado por um ciclo no grafo e uma lista de nós *ativos*. São chamados assim os nós onde são efetivamente instalados equipamentos, ou seja, onde efetivamente há custo. Os outros nós são apenas passagem. Os ciclos devem ser menores ou iguais a MaxAnel. Assim, propusemos que alguns deles fossem gerados por métodos de busca no próprio grafo.

O método que usamos encontra todos os circuitos elementares sobre um grafo orientado. No grafo $\mathcal{G}$, esse procedimento tem complexidade $O(|\mathcal{V}|+[|C|+1]|\mathcal{A}|)$, onde C é o conjunto de circuitos elementares no grafo $\mathcal{G}$, veja-se [SL75].

O problema real aqui tratado apresenta-se sobre grafos esparsos. Os comprimentos dos ciclos suporte não ultrapassam MaxAnel. Graças a essa condição, puderam-se gerar todos os ciclos para os dois exemplos onde $|\mathcal{V}| = 7$ e $|\mathcal{V}| = 10$. No primeiro caso, o número de ciclos viáveis é da ordem de 15, no segundo caso, 165. Veja-se a esse respeito [Bri99].

4.3.3 Um método semi-automático

Conforme já foi frisado, o custo reduzido associado a uma coluna tem a forma $h = \sum_i C_i r_i - \sum_i \sum_j u^{ij} a^{ij}$. Pretende-se obter uma coluna associada a um custo reduzido negativo. Dados $C_i > 0$, $r_i \geq 0$, $a^{ij} \geq 0$, para h

ter valor negativo é necessário que $u^{ij} > 0$. Sejam $\bar{\mathcal{D}} = \{[i,j]|\ u^{ij} > 0\}$ e $\bar{\mathcal{V}} = \{i,j|\ [i,j] \in \bar{\mathcal{D}}\}$. Se $\bar{\mathcal{D}} = \phi$, nenhuma coluna associada a um anel fará parte da nova base de $(\bar{P})$.

Geram-se ciclos suporte *a priori* de forma a que o comprimento total seja inferior a MaxAnel. Dentre esses ciclos toma-se um que contenha os vértices de $\bar{\mathcal{V}}$. Suponha-se $|\bar{\mathcal{V}}| = p$. Seja $s(1) < s(2) < \cdots < s(p-1) < s(p)$ e $\bar{\mathcal{V}} = \{s(1),\ s(2),\ \ldots,s(p-1),\ s(p)\}$.

Suponhamos que o vetor u esteja associado a uma base primal viável de $(\bar{P})$ tal que os custos reduzidos das colunas correspondentes às variáveis $z^{ij}_{\alpha\beta}$ e $s_{\alpha\beta}$ sejam maiores ou iguais a zero.

O problema que gera uma coluna associada a um anel pode ser escrito como se segue:

$$(PA):\ \text{minimizar} \sum_{i=1}^{p} C_{s(i)} r_{s(i)} - \sum_{i=1}^{p-1} \sum_{j=i+1}^{p} u^{s(i)s(j)} a^{s(i)s(j)}$$

sob as restrições:

$$M_{s(i)} = \sum_{j=i+1}^{p} a^{s(i)s(j)},\ i = 1,2,...,p-1$$

$$0 \leq M_{s(i)} \leq (\sum_{j=i+1}^{p} f_{s(i)s(j)}) r_{s(i)},\ i = 1,2,...,p-1$$

$$\sum_{i=1}^{p-1} M_{s(i)} \leq \mathsf{CapAnel}$$

$$0 \leq a^{s(i)s(j)} \leq f_{s(i)s(j)} r_{s(j)},\ i = 1,2,...,p-1,\ j = i+1, i+2, ..., p$$

$$r_{s(i)} \in \{0,1\},\ i = 1,2,...,p.$$

Se não houver nenhum ciclo para suportar $\bar{\mathcal{V}}$, poder-se-á buscar um conjunto $\mathcal{D}_0 \subseteq \bar{\mathcal{D}}$, tal que exista um ciclo viável para $\mathcal{V}_0 = \{i,j|\ [i,j] \in \mathcal{D}_0\}$. Se $\mathsf{val}(PA) \geq 0$ and $\bar{\mathcal{V}} = \mathcal{V}_0$ então foi encontrada uma solução ótima para $(\bar{P})$.

Quando $\mathsf{val}(PA) \geq 0$ e $\bar{\mathcal{V}} \neq \mathcal{V}_0$, pode-se fazê-lo de outra forma, a ser detalhada posteriormente. Se $\mathsf{val}(PA) < 0$, a coluna gerada entrará na nova base e proceder-se-á uma nova iteração do simplex.

4.3.4 Outro método

Pode-se também supor a existência de um ciclo viável e, com os mesmos $\bar{\mathcal{D}}$ e $\bar{\mathcal{V}}$ definidos acima, resolver o problema (PA) e obter como solução $\bar{r}_i \in \{0,1\}$, $i = 1,2,...,n$. Se verificar $\mathsf{val}(PA) \geq 0$, então foi encontrada uma solução ótima para $(\bar{P})$.

Se, por outro lado, $\mathsf{val}(PA) < 0$, verificaremos a existência de um ciclo viável para suportar $\mathcal{V}_1 = \{i \mid \bar{r}_i = 1\}$ de uma das duas maneiras seguintes:

- Todos os ciclos viáveis já foram listados numa etapa anterior. Se não houver nenhum ciclo para suportar $\mathcal{V}_1$, esta solução será eliminada introduzindo-se a seguinte restrição em (PA):

$$\sum_{i=1}^{n}(1-2\bar{r}_i)r_i \geq 1 - \sum_{i=1}^{n}\bar{r}_i. \tag{4.34}$$

  A desigualdade (4.34) elimina o ponto $\bar{r} \in \{0,1\}^p$ do conjunto $\{0,1\}^p$, veja-se seção 2.3.2.

- Os ciclos viáveis não foram ainda listados. Note-se que $\bar{\mathcal{V}}_1 = \mathcal{V} - \mathcal{V}_1$ e considere-se $\alpha \in \mathcal{V}_1$. Há que se resolver o seguinte problema:

$$(VC): \text{ minimizar } \sum_{(i,j)\in\mathcal{A}} \mathsf{dist}_{ij} w_{ij}$$

  sob as restrições

$$\sum_j y_{\alpha j} - \sum_j y_{j\alpha} = |\mathcal{V}_1| - 1$$

$$\sum_j y_{ij} - \sum_j y_{ji} = -1,\ i \in \mathcal{V}_1 - \{\alpha\}$$

$$\sum_j y_{ij} - \sum_j y_{ji} = 0,\ i \in \bar{\mathcal{V}}_1$$

$$\sum_j w_{ij} = \sum_j w_{ji} = 1,\ i \in \mathcal{V}_1$$

$$\sum_j w_{ij} = \sum_j w_{ji},\ i \in \bar{\mathcal{V}}_1$$

$$0 \leq y_{ij} \leq (|\mathcal{V}_1| - 1)w_{ij},\ (i,j) \in \mathcal{A}$$

$$\sum_{(i,j)\in\mathcal{A}} w_{ij} \leq K$$

$$w_{ij} \in \{0,1\}.$$

Este problema encontra um ciclo em $\mathcal{G}$ contendo os vértices de $\mathcal{V}_1$, com comprimento mínimo.

Se val$(VC) \leq$ MaxAnel, então as soluções $\bar{w}_{ij}$, $(i,j) \in \mathcal{A}$ de (VC), definirão um ciclo viável para $\mathcal{V}_1$. Se val$(VC) >$ MaxAnel, então não há ciclos para $\bar{\mathcal{V}}$. A solução $\bar{r}_i$, $i = 1,2,...,p$ será eliminada introduzindo-se em (VC) a restrição (4.34).

4.3.5 Heurística para geração de colunas

Um algoritmo produzindo para (PA) uma solução viável $\hat{r}_i$, $i \in \bar{\mathcal{V}}$ e $\hat{a}^{ij}$, $[i,j] \in \bar{\mathcal{D}}$ poderá fornecer uma coluna candidata a entrar na nova base de $(\bar{P})$; para isso deverá haver um ciclo suporte para $\hat{r}_i$, $i \in \bar{\mathcal{V}}$ e $\sum_{i\in\bar{\mathcal{V}}} C_i\hat{r}_i - \sum_{[i,j]\in\bar{\mathcal{D}}} u^{ij}\hat{a}^{ij} < 0$.

Veja-se a seguir o algoritmo heurístico para geração de colunas:

$u^{i(1)j(1)} f_{i(1)j(1)} \geq u^{i(2)j(2)} f_{i(2)j(2)} \geq ... \geq u^{i(p)j(p)} f_{i(p)j(p)} > 0$;
$\hat{h} := 0$; $\bar{h} := C_{i(1)} + C_{j(1)} - u^{i(1)j(1)} f_{i(1)j(1)}$;
$D := \{[i(1), j(1)]\}$; $V := \{i(1), j(1)\}$; T $:= f_{i(1)j(1)}$;
para $k = 2$ a p faça
  se (T $+ f_{i(k)j(k)} \leq$ CapAnel) e $(i(k) \in V)$ e $(j(k) \in V)$ então
    $\hat{h} := \bar{h} - u^{i(k)j(k)} f_{i(k)j(k)}$;
  se (T $+ f_{i(k)j(k)} \leq$ CapAnel) e $(i(k) \in V)$ e $(j(k) \notin V)$ então
    $\hat{h} := \bar{h} + C_{j(k)} - u^{i(k)j(k)} f_{i(k)j(k)}$;
  se (T $+ f_{i(k)j(k)} \leq$ CapAnel) e $(i(k) \notin V)$ e $(j(k) \in V)$ então
    $\hat{h} := \bar{h} + C_{i(k)} - u^{i(k)j(k)} f_{i(k)j(k)}$;
  se (T $+ f_{i(k)j(k)} \leq$ CapAnel) and $(i(k) \notin V)$ e $(j(k) \notin V)$ então
    $\hat{h} := \bar{h} + C_{i(k)} + C_{j(k)} - u^{i(k)j(k)} f_{i(k)j(k)}$;
  se $(\bar{h} < 0)$ e $(\hat{h} > \bar{h})$ então
    pare (toma-se a solução associada a $\bar{h}$);
  $\bar{h} := \hat{h}$;
  T:= T $+ f_{i(k)j(k)}$;
  $V := V \cup \{i(k), j(k)\}$; $D := D \cup \{[i(k), j(k)]\}$;

fim

Se $\bar{h} \geq 0$, então não há coluna com custo reduzido negativo. Caso contrário, $\mathcal{V}$ e $\mathcal{D}$ definem um anel potencial, e será preciso encontrar um ciclo que o suporte.

4.4 Solução Viável para (P)

Obter boas soluções viáveis é sempre importante num problema de otimização combinatória. Para isso, podem-se construir heurísticas *ad hoc*, baseadas em conhecimento específico do problema, ou pode-se tentar desenvolver um esquema aproximativo através do qual seja possível determinar *a priori* a distância, no pior caso, entre a solução encontrada por este esquema e o ótimo, veja-se [CM94].

Escolhemos desenvolver um método *ad hoc*. Inicialmente, consideraram-se os maiores ciclos viáveis, e a eles atribuíram-se as demandas a fim de construírem-se os anéis. Para alguns pares de nós pode não haver nenhum anel no grafo $\mathcal{G}$; neste caso a demanda em questão deverá ser enviada pela rede em malha, veja-se [Bri99].

4.5 Método *Branch and Price*

A fim de resolver o modelo matemático com variáveis inteiras, consideramos métodos da classe *branch and price*, veja-se [BJN+98] e [MPBL01b].

Uma vez resolvido $(\bar{P})$, verificam-se os valores obtidos pelas variáveis x_k. Se todos forem inteiros, ter-se-á uma solução ótima para (P). Caso contrário, haverá uma variável x_l tal que $\bar{x}_l \notin \mathbb{Z}$ no ótimo de $(\bar{P})$.

A seguir descreve-se o método *branch and price* proposto para a solução de (P).

Para iniciar o algoritmo, consideram-se dois problemas descedentes de $(\bar{P})$:

- $(\bar{P}_1) := (\bar{P}) \cap \lfloor \bar{x}_l \rfloor$;
- $(\bar{P}_2) := (\bar{P}) \cap \lfloor \bar{x}_l \rfloor + 1$;

onde $\lfloor \alpha \rfloor$ representa o maior inteiro menor ou igual a α.

A fim de resolver $(\bar{P}_1)$, pode-se usar o método dual simplex, veja-se [Dan63] e [Chv83], porque a base da solução ótima de $(\bar{P})$ é dual viável de $(\bar{P}_1)$ e $(\bar{P}_2)$.

Inicia-se o procedimento resolvendo $(\bar{P}_1)$ pelo dual simplex considerando apenas as colunas já geradas para a solução de $(\bar{P})$; este problema chamar-se-á $(\bar{P}_1')$. Resolvê-lo é tarefa simples, veja-se [Chv83], página 157, ou [MFL99].

Até este ponto, as colunas foram geradas desconsiderando-se outras colunas viáveis. Com isso pode ocorrer de $(\bar{P}_1')$ ser inviável, mesmo que $(\bar{P}_1)$ não o seja. Neste caso, será usado um método primal simplex de duas fases, veja-se [MPBL01b]. Uma vez resolvido $(\bar{P}_1')$, buscamos resolver $(\bar{P}_1)$ pelo primal simplex com geração de colunas. Para tanto faz-se necessário eliminar a coluna associada à variável x_l a fim de impedir seu retorno à base. Essa eliminação é feita introduzindo-se uma restrição do tipo 4.34 em (PA), onde os $\bar{r}_i, i = 1, 2, \ldots, p$, definem a coluna associada a x_l.

A resolução de $(\bar{P}_2)$ é semelhante à de $(\bar{P}_1)$, mas não é necessário eliminar a coluna associada a x_l.

Para os descendentes de $(\bar{P}_1)$ e $(\bar{P}_2)$, os comentários precedentes se aplicam.

Um problema do tipo $(\bar{P}_1)$ estará fechado quando não houver mais descendentes: seja sua solução ótima uma solução viável de (P), seja $\mathsf{val}(\bar{P}_i)$ maior ou igual à melhor solução viável de (P), seja $(\bar{P}_i)$ vazio.

4.6 Resultados Numéricos

Apresentamos resultados numéricos para duas instâncias do problema, ambas redes de áreas metropolitanas européias. O programa foi escrito na linguagem C, usando a biblioteca XPRESS-MP. Todos os experimentos realizaram-se numa máquina Pentium III 450 MHz. A tabela mostra algumas informações sobre as redes e os problemas respectivos.

Os custos adotados ao longo dos experimentos foram 1(um) para um nó ativo num anel e 2(dois) para uma expansão unitária de capacidade de aresta, na malha. Isto é, na verdade, uma aproximação para os custos reais de um multiplexador, usado para conectar um nó a um anel, e um par de

Capacidade nos anéis (nós)	**12**	**16**
Custo de uma rede pura malha	50	50
Solução contínua com anéis	10.067	8.740
Tempo de CPU (segundos)	9	8
Solução inteira com anéis	12	10
Colunas geradas	25	24
Colunas ativas	2	2
Custo total dos anéis	10	10
Custo total da malha	2	0
Tempo de CPU (segundos)	1	1
Ótimo (B&P)	11	10
Colunas geradas	43	50
Colunas ativas	3	2
Custo total dos anéis	11	10
Custo total da malha	0	0
Nós abertos	3	4
Nós fechados	3	5
Tempo de CPU (segundos)	10	7

Tabela 4.1: Resultados na rede de 7 nós

terminais a serem instalados em SDXC's, em nós da malha.

Resolvendo (P) na rede de 10 nós, obtivemos um valor de 152 para a função objetivo, o que significa 72 expansões de aresta. Em seguida iniciamos o gerador de colunas, que gerou 88 novas colunas com custo reduzido negativo e baixou o custo da solução para 24,403. Adicionando apenas estas colunas a (P) iniciou-se o *branch and bound* para buscar uma solução inteira, e obtivemos o valor 29. Nesta solução, das 88 colunas geradas, 7 estavam relacionadas a variáveis não-nulas, o que significa 7 anéis com capacidade 12 na solução final. Todas as expansões de malha caíram a zero.

Uma vez obtido os resultados do *branch and bound* com as colunas geradas, iniciamos o método *branch and price* descrito na seção 4.5. Em ambos os casos o ótimo foi encontrado.

Consideramos dois valores diferentes para as capacidades dos anéis. Estes resultados, juntamente com os da rede de 7 nós, estão relatados nas tabelas 4.1 e 4.2.

Capacidade dos anéis (nós)	**12**	**16**
Custo de uma rede pura malha	152	152
Solução contínua com anéis	24.403	21.958
Tempo de CPU (segundos)	48	57
Solução inteira com anéis	29	27
Colunas geradas	88	78
Colunas ativas	7	6
Custo total dos anéis	29	27
Custo total da malha	0	0
Tempo de CPU (segundos)	6	3
Ótimo (B&P)	27	24
Colunas gerada	4450	11104
Colunas ativas	6	6
Custo total dos anéis	27	24
Custo total da malha	0	0
Nós abertos	281	436
Nós fechados	540	685
Tempo de CPU (horas)	2	2,5

Tabela 4.2: Resultados na rede de 10 nós

Conclusão

Ao longo deste trabalho, vimos que na solução dos programas inteiros que modelavam casos reais não se pode dispensar a concorrência de informações diretamente do problema real. O procedimento genérico de programação inteira não foi a técnica mais rápida e nem garantia de encontrar o ótimo. Concluímos que problemas de síntese de redes com muitas restrições de integralidade ainda estão longe de contarem com um método genérico e universalmente eficiente.

O uso da geração de colunas em programação inteira tem-se mostrado profícuo em diversos campos. Quando certas estruturas do problema aparecem no modelo de forma distinta, de forma a poderem ser destacadas, como é o caso dos anéis, podemos associar essas estruturas a colunas de maneira fácil e direta. Outros problemas de síntese de redes podem também ser abordados assim, em particular o problema da definição de redes de acesso, apresentado em [PdFB+99] e [PBV+99], que trata de uma rede hierarquizada em dois níveis, um dos quais é composto de anéis.

Trabalhos posteriores a seguir esta linha podem investigar a complexidade do algoritmo COLGEN para o caso médio, sugerir novas estratégias para selecionar a variável pivô da ramificação ou verificar em que tipos de grafos a pré-geração de ciclos permanece uma opção viável.

Bibliografia

[Bal60] Egon Balas. An addictive algorithm for solving linear programs with 0–1 variables. *Operations Research*, 13:517–546, 1960.

[BHV00] Cynthia Barnhart, Christopher Hanne, and Pamela H. Vance. Using branch-and-price and cut to solve origin destination integer multicommodity flow problems. *Operations Research*, 46:316–329, 2000.

[BJN+98] Cynthia Barnhart, Ellis L. Johnson, George L. Nemhauser, Martin W. P. Savelsbergh, and Pamela H. Vance. Branch-and-price: Column generation for solving huge integer programs. *Operations Research*, 46:316–329, 1998.

[BLM97] Jean Marie Bourjolly, Gilbert Laporte, and Helene Mercure. A combinatorial column generation algorithm for the maximun stable set problem. *Operations Research Letters*, 20:21–29, 1997.

[BMP00] José André M. Brito, Nelson Maculan, and Marcos M. Passini. Um modelo de otimização para dimensionamento de uma rede de telecomunicações. In *XXIII Congresso Nacional de Matemática Aplicada e Computacional*, Santos, SP, 2000.

[Bri99] José André de Moura Brito. Um modelo de otimização para dimensionamento de uma rede de telecomunicações. Tese de mestrado, COPPE, Universidade Federal do Rio de Janeiro, Rio de Janeiro, Brasil, 1999.

[BSL97] Julien Bramel and David Simchi-Levi. On the effectiveness of set covering formulations for the vehicle routing problem with time windows. *Operations Research*, 45:295–301, 1997.

[Chv83] Vasec Chvátal. *Linear programming*. W. H. Freeman and Company, New York / San Francisco, USA, 1983.

[CJP83] H. Crowder, E.L. Johnson, and M.W. Padberg. Solving large scale zero-one linear programming problems. *Operations Research*, 31:803–834, 1983.

[CLR90] Thomas H. Cormen, Charles E. Leiserson, and Ronald L. Rivest. *Introduction to Algorithms*. The MIT Press / McGraw Hill Book Company, 1990.

[CM94] Ruy Eduardo Campello and Nelson Maculan. *Algoritmos e Heurísticas: Desenvolvimento e Avaliação de Performance*. EDUFF, Niterói, Brasil, 1994.

[Dak65] R. Dakin. A tree search algorithm for mixed integer programming problems. *Computer Journal*, 8:250–255, 1965.

[Dan55] George B. Dantzig. Upper bounds, secondary constraints and block triangulary in linear programming. *Econometrica*, 23:174–183, 1955.

[Dan63] George B. Dantzig. *Linear Programming and Extensions*. Princeton University Press, New Jersey, USA, 1963.

[dC98] J.M. Valério de Carvalho. Exact solution of cutting stock problems using column generation and branch-and-bound. *International Transactions in Operational Research*, 5(1):35–43, 1998.

[DDS92] Martin Desrochers, Jacques Desrosiers, and Marius Solomon. A new optimization algorithm for the vehicle routing problem with time windows. *Operations Research*, 40(2):342–353, 1992.

[DDSS95] J. Desrosiers, Y. Dumas, M.N. Solomon, and F. Soumis. Time constrained routing and scheduling. In M.O. Ball, T.L. Magnanti, C.L. Monma, and G.L. Nemhauser, editors, *Network Routing*, volume 8 of *Handbooks in Operations Research and Management Science*, pages 35–139. INFORMS - North Holland, 1995.

[Del74] Jacques Delorme. *Contributions à la Résolution du Problème de Recouvrement: Méthode de Troncature*. Docteur-ingenieur dissertation, Université Paris VI, Paris, France, 1974.

[DS89] Jacques Desrochers and François Soumis. A column-generation approach to the urban transit crew scheduling problem. *Transportation Science*, 23:1–13, 1989.

[DSD84] Jacques Desrosiers, François Soumis, and Martin Desrochers. Routing with time-windows by column generation. *NETWORKS*, 14:545–565, 1984.

[DW60] G.B. Dantzig and P. Wolfe. Decomposition principle for linear programming. *Operations Research*, 8:101–111, 1960.

[Geo74] A.M. Geoffrion. Lagrangean relaxation for integer programming. *Mathematical Programming Study*, 2:82–114, 1974.

[GG61] P.C. Gilmore and R.E. Gomory. A linear programming approach to the cutting stock problem. *Operations Research*, 9:849–859, 1961.

[GG63] P.C. Gilmore and R.E. Gomory. A linear programming approach to the cutting stock problem - part II. *Operations Research*, 11:863–888, 1963.

[GM79] Michel Gondran and Michel Minoux. *Graphes et algorithmes*. Eyrolles, Paris, France, 1979.

[Gom58] R.E. Gomory. Outline of an algorithm for integer solutions to linear programs. *Bulletin of the American Mathematical Society*, 64:275–278, 1958.

[GSMD92] Michel Gamache, François Soumis, Gerald Marquis, and Jacques Desrosiers. A column generation approach for large-scale aircrew rostering problems. *Operations Research*, 48(2):247–263, 1992.

[HJdA91] Pierre Hansen, Brigitte Jaumardi, and Marcus Poggi de Aragão. Un algorithme primal de programmation linéaire généralisée pour les programmes mixtes. *C. R. Acad. Sci. Paris*, série I, 313:557–560, 1991.

[HK70] M. Held and R.M. Karp. The traveling salesman problem and minimum spanning tree. *Operations Research*, 18:1138–1162, 1970.

[HK71] M. Held and R.M. Karp. The traveling salesman problem and minimum spanning tree: Part ii. *Mathematical Programming*, 1:6–25, 1971.

[Hu69] T.C. Hu. *Integer Programming and Network Flows*. Addison-Wesley, Inc., 1969.

[JMN93] E. Johnson, A. Mehrotal, and G.L. Nemhauser. Min-cut clustering. *Mathematical Programming*, 62:133–152, 1993.

[Las70] Leon S. Lasdon. *Optimization Theory for Large Systems*. Macmillan, New York, USA, 1970.

[LD60] A.H. Land and A.G. Doig. An automatic method for solving discrete programming problems. *Econometrica*, 28:497–520, 1960.

[LMP96a] Abdel Lisser, Nelson Maculan, and Marcos M. Passini. A mathematical model for solving design problems of telecommunications networks. In *INFORMS Fall Meeting*, Philadelphia, USA, 1996.

[LMP96b] Abdel Lisser, Nelson Maculan, and Marcos M. Passini. A mathematical model for solving design problems of telecommunications networks. In *INFORMS Fall Meeting*, Atlanta, USA, October 1996.

[LMP96c] Abdel Lisser, Nelson Maculan, and Marcos M. Passini. Modeling and solving design problems of telecomunication network using two different architectures. In *CIRO — Conférence Internationale en Recherche Opérationnelle*, Marrakech, Maroc, 1996.

[LMP96d] Abdel Lisser, Nelson Maculan, and Marcos M. Passini. Modelling and solving design problems of telecommunications network using two different architectures. In *Conférence Internationale en Recherche Opérationnelle*, Marrakech, Maroc, June 1996.

[LMPW98] Abdel Lisser, Nelson Maculan, Marcos M. Passini, and Gabriel Weintraub. A new model for the 2-architecture problem in telecommunications network design. In *IX CLAIO - Congreso Latino Americano de Investigación Operativa*, Buenos Aires, Argentina, 1998.

[Lob98] Andreas Lobel. Vehicle scheduling in public transit and lagrangean pricing. *Management Science*, 44(12):1637–1649, 1998.

[LSV95] A. Lisser, R. Sarkassian, and J.-P. Vial. Survivability in transmission telecommunications networks. Technical report, NT/PAA/ATR/ORI, CNET, France Telecom, April 1995. A ser publicado em Investigación Operativa.

[Mar85] Odile Marcotte. The cutting stock problem and integer rounding. *Mathematical Programming*, 13:82–92, 1985.

[MFL99] Nelson Maculan, Márcia Fampa, and Abilio Lucena. Otimização linear e inteira. Notas de aula — COPPE/Universidade Federal do Rio de Janeiro, 1999.

[Min87] Michel Minoux. Network synthesis and dynamic network optimization. *Annals of Discrete Mathematics*, 31:283–324, 1987.

[MM01] Nelson Maculan and Elder M. Macambira. Planos-de-corte esféricos e cilíndricos aplicados a problemas de programação linear inteira 0–1. Working paper, 2001.

[MMS01] Nelson Maculan, Elder M. Macambira, and Luidi Simonnetti. Resoluçãao do problema do caixeiro viajante simétrico com o emprego de planos-de-corte geométricos. Working paper, 2001.

[MPBL00] Nelson Maculan, Marcos M. Passini, José André M. Brito, and Irene Loiseau. Column generation in linear integer programming. In *X CLAIO — Congresso Latino Iberoamericano de Investigación de Operaciones y Sistemas*, Ciudad de Mexico, 2000. ALIO.

[MPBL01a] Nelson Maculan, Marcos M. Passini, José André M. Brito, and Abdel Lisser. Column generation method for network design. In P. Marcotte and M. Gendreau, editors, *Transportation and Network Analysis: Current Analysis*. Kluwer Academic Publishers, Holanda, 2001.

[MPBL01b] Nelson Maculan, Marcos M. Passini, José André M. Brito, and Irene Loiseau. Column generation in integer linear programming. *RAIRO - Operatons Research*, 2001. A ser publicado.

[MPL99] Nelson Maculan, Marcos M. Passini, and Abdel Lisser. Solving design problem of telecommunications network using column generation. In *INFORMS Fall Meeting*, Philadelphia, USA, 1999.

[MT96] Anuj Mehrotra and Michael Trick. A column generation approach for graph coloring. *INFORMS Journal on Computing*, 8(4), 1996.

[NW88] George L. Nemhauser and Laurence A. Wolsey. *Integer and Combinatorial Optimization*. John Wiley & Sons, Inc., 1988.

[Pas96] Marcos de Mendonça Passini. Um modelo de otimização combinatória para o dimensionamento de uma rede urbana de telecomunicações. Tese de mestrado, COPPE, Universidade Federal do Rio de Janeiro, 1996.

[PBV+99] Marcos M. Passini, Amir Coelho Barros, Douglas Valiati, Rosa Maria de Figueiredo, José André M. Brito, Nelson Maculan, and Adilson Elias Xavier. Dimensionamento de redes de acesso. Relatório do projeto FORTrans — COPPE/EMBRATEL, 1999.

[PdFB+99] Marcos M. Passini, Rosa Maria de Figueiredo, Amir Coelho Barros, José André M. Brito, Douglas Valiati, Nelson Maculan, Adilson Xavier, and Antônio Silvério. Dimensionamento de redes de acesso. In *XXX SBPO — Simpósio Brasileiro de Pesquisa Operacional*, Juiz de Fora, MG, 1999.

[PML97] Marcos M. Passini, Nelson Maculan, and Abdel Lisser. Modelagem e solução de problemas em projeto de redes de telecomunicações usando duas arquiteturas. In *XXIX SBPO — Simpósio Brasileiro de Pesquisa Operacional*, Salvador, BA, 1997.

[RF81] D.M. Ryan and B.A. Foster. An integer programming approach to scheduling. In *Computer Scheduling of Public Transport Urban Passanger Vehicle and Crew Scheduling*. North Holland, 1981.

[RMP89] Celso C. Ribeiro, Michel Minoux, and M.C. Penna. An optimal column-generation-with-ranking algorithm for very large scale partitioning problems in traffic assignment. *European Journal of Operational Research*, 41:232–239, 1989.

[RS94] Celso C. Ribeiro and François Soumis. A column generation approach to the multiple-depot vehicle scheduling problem. *Operations Research*, 42:41–52, 1994.

[Sav97] Martin Savelsbergh. A branch-and-price algorithm for the generalized assignment problem. *Operations Research*, 46:831–841, 1997.

[SL75] J.L. Szwarcfiter and P.E. Lauer. A search strategy for the elementary cycles of a directed graph. *BIT*, 16:192–204, 1975.

[SVW98] Alain Sutter, François Vanderbeck, and Laurence Wolsey. Optimal placement of add/drop multiplexers: Heuristic and exact algorithms. *Operations Research*, 46(5):719–728, September-October 1998.

[Van94] François Vanderbeck. *Decomposition and Column Generation for Integer Programming*. Thése de doctorat en sciences appliquées, Université Catholique de Louvain, Louvain, Belgique, 1994.

[Van98] Pamela H. Vance. Branch-and-price algorithms for the one-dimensional cutting stock problem. *Computational Optimization and Applications*, 9:211–228, 1998.

[Van99] François Vanderbeck. Computational study of a column generation algorithm for bin packing and cut stocking problems. *Mathematical Programming*, 86:565–594, 1999.

[Van00] François Vanderbeck. On Dantzig-Wolfe decomposition in integer programming and ways to perform branching in a branch-and-price algorithm. *Operations Research*, 48(1):111–128, 2000.

[VBJN94] Pamela H. Vance, Cynthia Barnhart, Ellis L. Johnson, and George L. Nemhauser. Solving binary cutting stock problems by column generation and branch-and-bound. *Computational Optimization and Applications*, 3:111–130, 1994.

[vdAHvdV99] J.M. van den Akker, J.A. Hoogeveen, and S.L. van de Velde. Parallel machine scheduling by column generation. *Operations Research*, 47(6):862–872, 1999.

[VSK96] R. Vachani, A. Shulman, and P. Kubat. Multicommodity flows in ring networks. *INFORMS Journal on Computing*, 8:235–242, 1996.

[VW96] François Vanderbeck and Laurence Wolsey. An exact algorithm for IP column generation. *Operations Research Letters*, 19:151–159, 1996.

[Wol98] Laurence A. Wolsey. *Integer Programming*. John Wiley & Sons, Inc., 1998.