O Problema de Steiner em Grafos:
uma Abordagem Poliédrica

Carlos Eduardo Ferreira

DISSERTAÇÃO APRESENTADA AO
INSTITUTO DE MATEMÁTICA E ESTATÍSTICA DA
UNIVERSIDADE DE SÃO PAULO
PARA OBTENÇÃO DO GRAU DE
MESTRE EM
MATEMÁTICA APLICADA

Área de Concentração: **Ciência da Computação**
Orientadora: **Prof^a. Dr^a. Yoshiko Wakabayashi**

— São Paulo, Dezembro de 1989 —

Agradecimentos

À Prof^a. Dr^a. Yoshiko Wakabayashi, por sua excelente orientação, estímulo, dedicação, presença e especialmente por sua amizade;

Aos professores e colegas do MAC, em especial Cristina, Coelho, Dilma e Maria Angela pelo apoio e auxílio nesta "reta final";

Aos meus pais, Marinho, Sergio e Irene pela ajuda e afeição;

Aos meus alunos, pela tolerância aos meus erros e mau humor

Meu MUITO OBRIGADO.

PS. Em tempo, gostaria de agradecer J. Steiner, sem o qual este trabalho não poderia ser realizado.

Resumo

Dado um grafo G , o problema de determinar um subgrafo conexo de G que seja de custo mínimo e contenha um determinado subconjunto de vértices, é conhecido como o *Problema de Steiner em Grafos*. Este é o tema central deste trabalho, sobre o qual apresentamos uma resenha dos resultados da literatura: suas origens na Geometria, sua complexidade computacional, alguns dos algoritmos desenvolvidos para resolvê-lo, algumas de suas principais variantes e aplicações práticas. Investigamos este problema sob uma abordagem poliédrica e desenvolvemos um algoritmo baseado em planos-de-corte. Apresentamos uma implementação deste algoritmo e discutimos o seu desempenho quando aplicado a instâncias reais e algumas cujos dados foram gerados aleatoriamente.

Abstract

Given a graph G , the problem of determining a minimum cost connected subgraph that spans a given subset of nodes, is known as the *Steiner Problem in Graphs*. In this work we present a survey on this problem discussing the following topics: historical background, computational complexity, exact and approximate solution methods, polynomially solvable cases, some of its major variants and real world applications. We also investigate this problem using a polyhedral approach and give a partial description of the facial structure of a polyhedron we associate with it. Using these results we design and implement a linear programming based cutting plane algorithm. The practical performance of this algorithm is discussed both for random and real data.

O Problema de Steiner em Grafos: uma Abordagem Poliédrica.

Conteúdo

Introdução	3
1 Preliminares e Notações	6
1.1 Teoria dos Grafos	6
1.2 Teoria de Poliedros	9
1.3 Observações	11
2 O Problema de Steiner	12
2.1 O Problema de Steiner na Geometria	12
2.2 O Problema de Steiner em Grafos	15
2.2.1 Algumas aplicações	18
2.2.2 Complexidade Computacional	20
2.2.3 Algumas Variantes	24
3 Métodos de Solução : um Breve Resumo	26
3.1 Algoritmos Exatos	27
3.1.1 Busca na árvore de soluções	27
3.1.2 Enumeração explícita	32
3.1.3 Algoritmo baseado em Programação Dinâmica	35
3.1.4 Abordagens usando Programação Inteira	37

3.2	Métodos heurísticos	44
3.2.1	Heurística de caminhos mínimos	45
3.2.2	Heurística de árvore geradora mínima	49
3.2.3	Heurística de contração	51
3.3	Reduções no tamanho das instâncias	57
3.4	Casos polinomiais	62
3.4.1	2-Árvores	62
3.4.2	Redes de Halin	66
3.4.3	Grafos Fortemente Cordais	68
4	Abordagem Poliédrica	71
4.1	Escolha do Politopo	72
4.2	Facetas do Politopo $P_5(G, Z)$	78
4.2.1	Facetas de Corte	78
4.2.2	Facetas W-induzidas	81
4.2.3	Facetas de Partição	83
4.3	Facetas do Politopo $P_3(G, Z)$	87
4.4	Descrição do Algoritmo e Resultados Computacionais	92
4.4.1	Descrição do algoritmo	92
4.4.2	Resultados Computacionais	95
	Comentários Finais	105

Introdução

Desde a formulação do problema das pontes de Königsberg, ainda no século XVIII, a Teoria dos Grafos tem motivado centenas de pesquisadores a estudar problemas interessantes e com várias aplicações práticas. Alguns desses problemas parecem quebra-cabeças: conseguimos descrevê-los para qualquer pessoa em poucos minutos e esta pode passar o resto de sua vida tentando encontrar uma solução. A parte mais interessante desta teoria é que ela fornece entretenimento e desafio tanto para aqueles que descrevem grafos como “uma porção de bolinhas ligadas por risquinhos”, como para os que os definem de forma extremamente formal.

Entre os problemas que surgem na Teoria dos Grafos, existem os chamados problemas de otimização combinatória que são da seguinte forma: *Dada uma função custo (ou benefício) definida sobre um conjunto finito S , encontre, se existir, um elemento desse conjunto que minimiza (ou maximiza) a função dada.* É claro que os elementos de S não são dados explicitamente, senão o problema seria trivial desde que a função seja fácil de ser computada. Assim, um tal problema só é interessante se assumirmos que o conjunto S é definido implicitamente, por exemplo, especificando-se que propriedades seus elementos devem satisfazer. Uma outra característica destes problemas é que, feita a especificação, em geral é fácil gerar os elementos de S , ou mesmo decidir se um determinado elemento está ou não em S .

Então, onde estaria a dificuldade do problema? A dificuldade está no *número de elementos* de S . Apesar de finito, S pode ser muito grande, a ponto de desencorajar qualquer tentativa de solução que seja do tipo: gerar todos os elementos e escolher o melhor. Fica claro que (mesmo informalmente) estamos assumindo aqui o ponto de vista do cientista da computação, para quem importa, não apenas encontrar uma maneira de se resolver o problema, mas encontrar algoritmos eficientes para isso. Entre os problemas de otimização combinatória, os problemas de conexidade em grafos constituem uma classe muito interessante e com diversas aplicações práticas. Como exemplos de problemas de conexidade, podemos citar o problema do caminho mínimo, o problema de determinar uma árvore geradora de custo mínimo e o *Problema de Steiner em Grafos*. Este último problema pode ser assim definido: *dado um grafo G , determinar um subgrafo conexo de custo mínimo que contém um determinado subconjunto de vértices.*

Parece uma estranha conspiração que para uma boa parte dos problemas de

otimização combinatória que têm várias aplicações práticas não se conhecem algoritmos eficientes para resolvê-los e nem mesmo se espera encontrar tais algoritmos. Não é o que acontece com o problema do caminho mais curto nem com o problema da árvore geradora mínima, pois para estes existem algoritmos eficientes. Infelizmente, o *Problema de Steiner em Grafos* confere à classe dos problemas de conexidade o *status* de difícil, já que não se conhece nenhum algoritmo eficiente para resolvê-lo. Como veremos neste trabalho, o problema é $\mathcal{NP}$ -difícil, o que nos dá pouca esperança de encontrarmos um algoritmo eficiente para resolvê-lo. Este problema tem suas origens na Geometria em um problema cuja primeira referência nos leva ao século XVII e ao grande matemático Fermat. Essa formulação usando linguagem da Teoria dos Grafos data de 1971 e desde então diversos algoritmos e aplicações têm sido apresentados na literatura. É surpreendente encontrar aplicações deste problema nas mais variadas áreas do conhecimento, tais como projetos de redes em engenharia, inferência de consultas a bases de dados e teoria de evolução das espécies.

Como dissemos, desde sua primeira formulação em 1971, diversos algoritmos têm sido apresentados para o *Problema de Steiner em Grafos* e mesmo algumas resenhas bastante completas já foram apresentadas na literatura. Como veremos, as mais diferentes abordagens já foram aplicadas ao problema: *branch and bound*, programação dinâmica, heurísticas, busca de classes de grafos para as quais existem algoritmos polinomiais, etc.. Em nosso trabalho, além de tentar contribuir com uma nova resenha dos diversos resultados sobre o problema, procuramos apresentar uma abordagem diferente, implementar um algoritmo baseado nessa nova abordagem e discutir sua aplicabilidade em problemas reais.

A abordagem que utilizamos é a *abordagem poliédrica*. O desenvolvimento recente de algoritmos com bastante sucesso baseados nessa estratégia para problemas como o *Problema do Caixeiro Viajante* nos animou a fazer tal estudo. A busca de facetas do politopo associado ao problema, que, como veremos neste trabalho, é a essência desta abordagem, é uma atividade que fascina, nos transformando em detetives curiosos, ávidos em descobrir pistas arduamente escondidas pelo problema.

Outro aspecto interessante da abordagem poliédrica é que nela a teoria e a prática se unem, interagem e se completam. A parte prática, isto é, o desenvolvimento de um algoritmo não se concretiza sem o respaldo da parte teórica: a descoberta de facetas é importante para conseguirmos resolver os problemas. Por outro lado, a implementação prática nos devolve subsídios para que não retornemos à mesa de trabalho de mãos vazias, mas com resultados que, se soubermos interpretar corretamente, podem nos fornecer novas facetas bem como revelar outras propriedades de nosso problema.

Nosso objetivo ao iniciarmos este trabalho foi estudar o problema, motivados pela sua aparente simplicidade que se contrapõe à extrema dificuldade de resolução que este apresenta aos que o atacam, esperando não apenas desenvolver mais um novo algoritmo, mas principalmente aprender novos conceitos, novas técnicas, e sobretudo tentar aprender a fazer pesquisa.

Este trabalho está dividido em quatro capítulos: no primeiro apresentamos alguns resultados básicos da Teoria dos Grafos e Teoria de Poliedros e fixamos algumas notações que utilizaremos. No segundo capítulo fazemos uma apresentação do problema: um breve histórico, algumas aplicações práticas, algumas de suas principais variantes e sua complexidade computacional. A seguir, no terceiro capítulo, apresentamos uma resenha “personalizada” dos resultados conhecidos sobre o problema. Dizemos que nossa resenha é personalizada pois apesar de ser uma resenha bastante extensa dos resultados publicados, a escolha destes resultados foi baseada em gosto pessoal aliado à preocupação de cobrir as diversas abordagens existentes na literatura. No quarto e último capítulo, apresentamos a abordagem poliédrica que desenvolvemos para o problema, e discutimos alguns resultados computacionais que obtivemos, tanto no caso de dados aleatórios, como no caso de dados reais.

Capítulo 1

Preliminares e Notações

Neste capítulo resumimos alguns conceitos e estabelecemos a notação a ser utilizada em todo o trabalho. Para o leitor não familiarizado com os assuntos aqui tratados, indicamos alguns textos onde discussões mais completas podem ser encontradas.

1.1 Teoria dos Grafos

A maior parte dos conceitos aqui mencionados são básicos e podem ser encontrados no livro texto de Bondy e Murty [7] ou Bollobás [6].

Um **grafo** $G = (VG, EG)$ consiste de um conjunto finito e não vazio VG de elementos chamados **vértices** e um conjunto finito EG de elementos chamados **arestas**. A cada aresta está associado um par não ordenado de vértices, chamados suas **pontas**. Para sermos mais exatos, deveríamos definir G como sendo uma tripla (VG, EG, ψ) onde ψ é uma função que associa a cada elemento de EG um par não ordenado de VG . Por simplicidade, omitiremos ψ e para especificar que a uma aresta e está associado um par $\{u, v\}$ (ie, $\psi(e) = \{u, v\}$), usaremos a notação $e = \{u, v\}$ ou $e = uv$. Note que isso é um abuso de notação, pois podemos ter $e = uv, f = uv$ e $e \neq f$. Tal simplificação, porém, não causará confusão nos contextos onde será usada. Dizemos que uma aresta $e = \{u, v\} \in EG$ **incide** nos vértices u e v e também que u e v são **adjacentes**. O **grau** de um vértice v é o número de arestas incidentes em v . Para cada aresta $\{v, v\}$ de G esta será contada duas vezes no grau de v . Duas arestas são chamadas **múltiplas** se têm as mesmas pontas. Uma aresta com duas pontas iguais é chamada **laço**. Costuma-se representar um grafo da seguinte forma: os seus vértices são representados por pontos ou círculos e suas arestas por linhas ligando os pares de pontos que representam suas pontas. Por exemplo, se

$$G = (VG, EG), VG = \{u, v, w\}, EG = \{a_1, a_2, a_3, a_4\}, \\ a_1 = uv, a_2 = uu, a_3 = uv \text{ e } a_4 = vw$$

uma possível representação de G está na figura 1.1.

Figura 1.1

Um grafo G é **simples** se EG não contém laços e nem arestas múltiplas. Note que no caso de grafos simples, a notação convencionada não causa confusão já que cada aresta e pode ser identificada com o par $\{u, v\}$ de vértices ao qual está associada. Assim, neste caso, podemos simplesmente nos referir à “aresta uv ”. Além disso, é conveniente pensar em EG como sendo um conjunto de pares não ordenados de elementos distintos de VG .

Se $A \subseteq VG$, denotamos por $E(A)$ o conjunto das arestas com ambas as pontas em A , ou seja,

$$E(A) := \{e \in EG : e = uv, \quad u, v \in A\}.$$

Se $A, B \subseteq VG$, $A \cap B = \emptyset$, denotamos por $[A : B]$ o conjunto das arestas com uma ponta em A e outra em B , ou seja,

$$[A : B] := \{e \in EG : e = uv, \quad u \in A, v \in B\}.$$

Tal conjunto de arestas é chamado de **corte que separa A e B** . No caso particular em que $A = \{v\}$ e $B = VG - \{v\}$, denotamos $I(v) := [\{v\}, VG - \{v\}]$.

Se os grafos $H = (VH, EH)$ e $G = (VG, EG)$ são tais que $VH \subseteq VG$ e $EH \subseteq EG$ então diremos que H é um **subgrafo** de G . Dado $W \subseteq VG$, o subgrafo $H = (W, E(W))$ de G , denotado por $G[W]$, é chamado de **subgrafo induzido** por W . Se $A \subseteq EG$, o subgrafo de G que possui A como conjunto de arestas e cujo conjunto de vértices consiste dos vértices que são pontas das arestas em A também é chamado de subgrafo induzido por A e é denotado por $G[A]$. Para simplificar nossa notação, quando denotamos um grafo com um determinado símbolo (por exemplo, T), denotamos seu conjunto de vértices e arestas pelo mesmo símbolo precedido de V e E , respectivamente (por exemplo, VT e ET).

Dado $W \subseteq VG$, $G - W$ denota o grafo obtido de G removendo-se os vértices de W , ou seja, é o subgrafo $G[VG - W]$ de G . Se $W = \{v\}$, por simplicidade usaremos $G - v$ ao invés de $G - \{v\}$. Da mesma forma, dado $A \subseteq EG$, $G - A$ denota o subgrafo $G[EG - A]$. Também usamos $G - e$ para $G - \{e\}$. Se $u, v \in VG$ e $e = uv \notin EG$ então denotamos por $G + e$ o grafo H tal que $VH = VG$ e $EH = EG \cup \{e\}$.

A **contração** de uma aresta $e = uv$ em um grafo G é a operação que consiste em remover primeiramente a aresta e de G e depois identificar os vértices u, v em um novo

vértice w . O grafo resultante é denotado G/e . O grafo G/e tem $|EG| - 1$ arestas, cada aresta f de $G - e$ cujas pontas são diferentes de u e v são mantidas, se uma ponta de f é um dos vértices u ou v então esta ponta é substituída por w , se $f = uv$ então f se torna um laço. Veja um exemplo de contração de aresta na figura 1.2. Se $K \subseteq EG$ então G/K denota o grafo obtido de G contraindo-se uma a uma as arestas em K .

Figura 1.2

Um grafo G é **completo** se G é simples e para todo par de vértices $u, v \in VG$, $u \neq v$, existe $e \in EG$ tal que $e = uv$.

Uma sequência $C = (v_0, e_1, v_1, e_2, \dots, e_k, v_k)$ ($k \geq 1$) de elementos de um grafo G , onde $v_i \in VG$ para $i = 0, \dots, k$, $e_i \in EG$ para $i = 1, \dots, k$ e $e_i = v_{i-1}v_i$ para $i = 1, \dots, k$ é chamada de **passeio** entre v_0 e v_k ou simplesmente **passeio**. Se todos os vértices em C são distintos, C é chamado de **caminho**. No caso em que $v_0 = v_k$ e $v_0, \dots, v_{k-1}$ são todos distintos C é chamado de **circuito**. Em todos os casos, o **comprimento** de C é o número de arestas na sequência.

Dada uma função c definida nas arestas de G , e dados dois vértices $u, v \in VG$, a **distância** de u a v em G é

$$\text{dist}(u, v) := \min \left\{ \sum_{e \in C} c(e) : C \text{ é um caminho entre } u \text{ e } v \text{ em } G \right\}.$$

Um grafo G é **conexo** se para cada par $u, v \in VG$ existe um caminho entre u e v em G . Um grafo com um único vértice é conexo. Uma **componente** de um grafo G é um subgrafo conexo maximal de G . Uma aresta $e \in EG$ é uma **aresta de corte** (ou **ponte**) de G se $G - e$ tem mais componentes que G . De forma análoga, um vértice $v \in VG$ é um **vértice de corte** de G se $G - v$ tiver mais componentes que G . Numa floresta, um vértice de grau 1 é chamado **folha**. Um grafo G é **2-conexo** se quaisquer duas arestas estão contidas em um mesmo circuito.

Uma **árvore** é um grafo conexo em que toda aresta é aresta de corte. Uma **floresta** é um grafo no qual cada componente é uma árvore. Uma **árvore geradora** de um grafo G é um subgrafo T de G tal que $VT = VG$ e T é árvore.

Um **grafo dirigido** é definido de forma análoga a um grafo, porém, a cada aresta (aqui também chamado **arco**) é associado um par *ordenado* de vértices. Um grafo dirigido é **fortemente conexo** se para cada par u, v de vértices existe um caminho dirigido de u até v .

1.2 Teoria de Poliedros

Nesta seção apresentamos as principais notações e alguns resultados básicos sobre *Teoria de Poliedros*. O leitor interessado pode encontrar em [51] e [24] uma apresentação mais completa.

Dados os conjuntos finitos A, B denotamos por A^B o conjunto dos vetores com $|B|$ elementos indexados por B e com entradas em A , isto é, se $B = \{b_1, \dots, b_m\}$ então $A^B := \{(a_{b_1}, a_{b_2}, \dots, a_{b_m}), a_{b_i} \in A \text{ para } i = 1, \dots, m\}$. Por simplicidade, se $B = \{1, 2, \dots, m\}$ denotaremos A^B por A^m .

Se E é um conjunto e $A \subseteq E$ denotamos por x^A o **vetor característico** de A com relação a E , $x^A \in \{0, 1\}^E$, definido por:

$$x_e^A = \begin{cases} 1, & e \in A \\ 0, & \text{c.c.} \end{cases}$$

Se H é um subgrafo de um grafo G , o vetor característico de EH com relação a EG é denotado simplesmente por x^H e é referido também como o vetor característico de H .

Dado um vetor $x \in A^B$ e $K \subseteq B$ denotamos

$$x(K) := \sum_{e \in K} x_e.$$

Um vetor $x \in \mathbb{R}^m$ é uma **combinação linear** dos vetores $x_1, x_2, \dots, x_k \in \mathbb{R}^m$ se, para algum $\lambda = (\lambda_1, \dots, \lambda_k) \in \mathbb{R}^k$

$$x = \sum_{i=1}^k \lambda_i x_i.$$

Se λ é tal que $\sum_{i=1}^k \lambda_i = 1$, dizemos que x é **combinação afim**. Se adicionalmente $\lambda > 0$, dizemos que x é **combinação convexa** dos vetores $x_1, x_2, \dots, x_k$.

Dado $S \subseteq \mathbb{R}^m$, denotamos por $\text{conv}(S)$ o **fecho convexo** dos elementos de S , definido como o conjunto de todos os vetores de $\mathbb{R}^m$ que são combinação convexa de um conjunto finito de vetores de S . Definimos $\text{conv}(\emptyset) = \{0\}$.

Um conjunto finito não vazio $S \subseteq \mathbb{R}^m$ é chamado **linearmente independente** (**afim independente**) se não existe subconjunto $\{x_1, \dots, x_k\}$ de S e nenhum vetor $\lambda \in \mathbb{R}^k, \lambda \neq 0$ ($\lambda \neq 0$ e $\sum_{i=1}^k \lambda_i = 1$) tal que $\sum_{i=1}^k \lambda_i x_i = 0$.

O **posto afim** de um conjunto $S \subseteq \mathbb{R}^m$ ($pa(S)$) é a cardinalidade de um subconjunto afim independente maximal de S . A **dimensão** de um conjunto $S \subseteq \mathbb{R}^m$, denotada por $\dim(S)$ é definida por

$$\dim(S) := pa(S) - 1.$$

Se um conjunto $S \subseteq \mathbb{R}^m$ é tal que $\dim(S) = m$, dizemos que S tem **dimensão completa**.

Um conjunto $P \subseteq \mathbb{R}^m$ é chamado **poliedro** se P é solução de um sistema de inequações lineares, isto é,

$$P := \{x \in \mathbb{R}^m : Ax \leq b\}.$$

Um poliedro limitado é chamado **politopo**.

Seja $P \subseteq \mathbb{R}^m$ um poliedro. Uma inequação $ax \leq b$ é **válida** para P se

$$P \subseteq \{x \in \mathbb{R}^m : ax \leq b\}.$$

Um conjunto $F \subseteq P$ é uma **face** de P se existe uma inequação $ax \leq b$ válida para P tal que

$$F = \{x \in P : ax = b\}.$$

Uma face F é **própria** se $F \neq P$ e **não trivial** se $\emptyset \neq F \neq P$. Se $ax \leq b$ é uma inequação válida para P , então $P \cap \{x : ax = b\}$ é chamada de **face definida (ou induzida)** por $ax \leq b$.

Uma face não trivial F é chamada **faceta** se F não está contida em nenhuma outra face própria de P .

O resultado a seguir será utilizado várias vezes no capítulo 4.

Teorema 1 *Seja P um poliedro e F uma face não trivial de P . Então as seguintes afirmações são equivalentes:*

- (a) F é uma faceta de P ;
- (b) $\dim(F) = \dim(P) - 1$;
- (c) F contém $\dim(P)$ vetores afim independentes;
- (d) Se P é de dimensão completa e $ax \leq b$ é uma inequação válida para P tal que $F = \{x \in P : ax = b\}$, então para toda inequação válida $cx \leq d$ que induz uma face tal que $F \subseteq \{x \in P : cx = d\}$, existe $\alpha \in \mathbb{R}$ tal que $c = \alpha a$ (e portanto, $d = \alpha b$).

□

1.3 Observações

Além dos conceitos da teoria dos grafos e teoria dos poliedros, neste trabalho utilizamos conceitos de complexidade computacional, programação linear, *branch and bound*, programação dinâmica, relaxação lagrangeana, etc.

A razão de termos incluído apenas os tópicos das seções 1.1 e 1.2 é que na teoria dos grafos não existe uma terminologia padrão; assim, ao adotarmos uma pudemos ao mesmo tempo estabelecer a notação que usaremos. A seção 1.2 cumpre o papel de tornar este trabalho auto-contido em relação à abordagem poliédrica que desenvolvemos no capítulo 4.

Os conceitos não apresentados neste capítulo poderão ser encontrados nos textos [19], [45], [40] e [51].

Capítulo 2

O Problema de Steiner

Neste capítulo apresentamos o Problema de Steiner, começando com um histórico sobre sua origem na Geometria até chegar em sua formulação em linguagem de grafos, conhecida como *Problema de Steiner em Grafos*. Na seção 2.2 discutimos algumas características deste último problema, mostramos algumas de suas aplicações e variantes e analisamos sua complexidade computacional.

2.1 O Problema de Steiner na Geometria

O Problema de Steiner (cf. [36]) foi primeiramente proposto no século XVII por Fermat da seguinte forma:

Dadas três cidades, deseja-se construir uma rede de estradas de comprimento total mínimo de forma que se possa ir de qualquer cidade a qualquer outra através dessas estradas.

Figura 2.1

Mais formalmente, pode-se formulá-lo da seguinte maneira:

Dados três pontos A , B e C no plano euclidiano, deseja-se construir um sistema de segmentos de reta de comprimento total mínimo que interligue os pontos A , B e C . (Veja a figura 2.1.)

O problema acima proposto era citado em várias publicações no século XVII como um problema interessante, cuja solução era devida a Torricelli. No século XIX, o famoso geômetra alemão Jacob Steiner reuniu em um trabalho os resultados conhecidos sobre o problema. Assim, o problema proposto inicialmente por Fermat, passou a ser conhecido como o *Problema de Steiner*.

Apresentamos a seguir a solução deste problema conforme mencionada em [9].

Sem perda de generalidade, podemos considerar C como sendo o ponto tal que o ângulo $\hat{C}$ formado pelos segmentos que ligam esse ponto aos outros dois é máximo. Então temos dois casos:

Caso 1: $\hat{C} \geq 120^\circ$

Neste caso a solução é dada pelos segmentos $\overline{AC}$ e $\overline{BC}$. Veja a figura 2.2.

Figura 2.2

Caso 2: $\hat{C} < 120^\circ$

Neste caso, determine o ponto P em que os segmentos $\overline{AP}$, $\overline{BP}$ e $\overline{CP}$ formam três ângulos de 120° . A solução neste caso será dada por tais segmentos. Veja a figura 2.3.

Figura 2.3

É interessante notar que a solução do problema neste segundo caso, exigiu a introdução de um ponto que não estava na especificação inicial do problema. Chamaremos tais pontos de *facultativos* (na literatura são chamados de *Steiner points*).

A demonstração dos resultados acima, bem como a determinação de P com régua e compasso pode ser vista em [9].

Uma generalização do Problema de Steiner que surge de forma natural é a seguinte:

Dado um conjunto de pontos (cidades) $A_1, A_2, \dots, A_n$, deseja-se construir um sistema de segmentos de reta (rede de estradas) de comprimento total mínimo de tal forma que qualquer ponto esteja conectado a qualquer outro através deste sistema. Veja a figura 2.4.

Figura 2.4

Este problema é chamado por Courant e Robbins de *Problema da Rede de Ruas* (*Street Network Problem*), porém é mais conhecido como o *Problema de Steiner no Plano Euclidiano*, como chamaram Melzak [39] e Cockayne [8] em seus trabalhos no início da década de 60.

Melzak apresentou em [39] uma prova de que o conjunto de todas as soluções para o Problema de Steiner relativo a um conjunto qualquer de pontos podia ser obtido usando um conjunto finito de operações com régua e compasso. Porém, segundo Cockayne [8], algumas das provas daquele artigo continham erros e o algoritmo sugerido não estava claramente apresentado. Assim, em seu trabalho Cockayne apresenta um algoritmo para resolver o problema bem como faz algumas considerações sobre como este se comporta se aplicado a outros espaços topológicos. Infelizmente, tal algoritmo não é eficiente do ponto de vista computacional. Até hoje, vários algoritmos vêm sendo sugeridos (cf. [57] e [59]) porém os autores observam que os algoritmos exatos só são apropriados para instâncias pequenas (até 30 pontos).

Uma variante do Problema de Steiner no Plano Euclidiano foi sugerida por Hanan em [30] em que este usava a métrica retilinear ao invés da métrica euclidiana para definir as distâncias. Lembremos que dados dois pontos (x_1, y_1) e (x_2, y_2) , a distância retilinear d_r entre estes pontos é definida por:

$$d_r = |x_1 - x_2| + |y_1 - y_2|.$$

Neste caso, Hanan mostrou que o problema pode ser resolvido restringindo-se os possíveis pontos facultativos ao conjunto dos pontos formados pela intersecção das linhas horizontais e verticais que passam pelos pontos dados. Veja a figura 2.5. Tal variante é motivada pelas

Figura 2.5

diversas aplicações do problema em que se realizam preferencialmente ligações horizontais e verticais, tais como projetos de circuitos eletrônicos [30], redes de comunicação e de tráfego [18], redes de tubulações de gás e óleo [49] e redes de irrigação e drenagem [35].

Após discorrer sobre várias características do problema, Hanan sugere um algoritmo para resolver o *Problema de Steiner na Métrica Retilinear*. O algoritmo é baseado na busca exaustiva de todas as possibilidades de escolha de pontos facultativos. O próprio autor observa, entretanto, que tal algoritmo só é indicado para instâncias pequenas deste problema. Aliás, cabe aqui observar que não se conhece nenhum algoritmo eficiente (polinomial) que resolve o problema de Steiner na métrica retilinear. Na verdade, Garey e Johnson [20] mostraram que este problema é $\mathcal{NP}$ -completo. Devido às inúmeras aplicações práticas deste problema diversas heurísticas têm sido sugeridas. O leitor interessado pode encontrar em [53] uma discussão bastante completa destas heurísticas bem como de seus desempenhos computacionais.

Na próxima seção discutimos o *Problema de Steiner em Grafos* que é também uma generalização do problema original proposto por Fermat.

2.2 O Problema de Steiner em Grafos

Muitas vezes, encontramos aplicações do Problema de Steiner em que o conjunto de pontos facultativos é restrito a um conjunto finito de possibilidades (como já vimos ocorrer no caso em que a métrica adotada é a retilinear). Nesses casos, pode-se considerar que

são dados dois conjuntos de pontos: os pontos *obrigatórios* e os *facultativos*. Assim, se não considerarmos que estes pontos estão no plano, e ao invés da distância euclidiana, considerarmos uma função qualquer que se relaciona com o custo para se construir certas ligações possíveis entre esses pontos, surge naturalmente o seguinte problema:

Dado um grafo $G = (VG, EG)$ com uma função custo $c : EG \rightarrow \mathbb{R}$ e um conjunto $Z \subseteq VG$ de vértices (que chamaremos de vértices obrigatórios), queremos determinar um conjunto $A \subseteq EG$ tal que

- $G[A]$ contém um caminho entre cada par de vértices de Z e
- $\sum_{e \in A} c(e)$ é a menor possível.

Este é o chamado *Problema de Steiner em Grafos*, formulado pela primeira vez por Hakimi [28] bem como Dreyfus e Wagner [11] em 1971. Algumas vezes usamos a forma abreviada $PSG(G, Z)$ ou PSG para nos referirmos ao problema. Veja na figura 2.6 uma instância deste problema.

Figura 2.6

A primeira observação que ocorre é que se existir em G alguma aresta e tal que $c(e) < 0$, ela pode ser contraída pois certamente estará na solução ótima. Assim, se $K := \{e \in EG : c(e) \leq 0\}$, podemos tomar G/K o grafo obtido de G contraindo cada aresta de K com o cuidado adicional de que se a aresta a ser contraída incide em algum vértice obrigatório, o vértice resultante da contração da aresta também deve ser obrigatório. Veja na figura 2.7 o grafo da figura 2.6 após a contração das arestas de custo negativo. Não há, portanto, perda de generalidade supor que toda aresta tem custo positivo. É o que assumiremos de agora em diante. Além disso, é claro que os laços (de custo positivo) podem ser eliminados do grafo, visto que não podem estar em qualquer

solução ótima. Também arestas múltiplas podem ser eliminadas, ficando apenas uma aresta entre cada par de vértices (a de menor custo). Veja na figura 2.8 o grafo da figura 2.7 após a remoção de laços e arestas múltiplas.

Em vista das observações feitas, podemos supor que o grafo em consideração seja simples e a função custo seja positiva.

Outra observação é que se $|Z| = 2$, o problema é equivalente ao de determinar um caminho mais curto entre dois vértices num grafo. Para este problema, como sabemos, existem algoritmos eficientes (veja, por exemplo [10]). Se $|Z| = |VG|$ o problema é equivalente ao de determinar uma árvore geradora de custo mínimo num grafo. Para este problema também se conhecem algoritmos eficientes (veja, por exemplo [33] ou [47]). Então, a menos de menção em contrário, vamos assumir que $3 \leq |Z| \leq |VG| - 1$. Nas próximas subseções discutiremos algumas aplicações do problema, sua complexidade computacional e apresentaremos algumas de suas principais variantes.

2.2.1 Algumas aplicações

O *Problema de Steiner em Grafos* tem diversas aplicações nas mais diferentes áreas, tais como engenharia, teoria de evolução das espécies e inferência de consultas a bases de dados. Nesta subseção discutiremos algumas dessas aplicações.

Substituição de cabos convencionais por fibra ótica

Descrevemos a seguir um problema real muito interessante sobre conexidade de redes, apresentado por Grötschel e Monma em [23].

Trata-se de um projeto de redes de comunicação que surgiu nos EUA para substituir alguns cabos convencionais já existentes por fibras óticas. A fibra ótica possui muito maior confiabilidade na transmissão dos dados que os cabos convencionais, maior velocidade e capacidade de transmissão. Por outro lado, o custo da implantação da fibra ótica é muito maior que o dos cabos comuns. Porém, devido às boas características das fibras óticas, estas são necessárias principalmente nas comunicações entre os grandes centros. Assim as cidades americanas foram divididas em três grupos:

- As *cidades grandes*, para as quais se desejava no mínimo dois caminhos disjuntos com ligações de fibra ótica entre cada par de cidades;
- As *cidades médias*, para as quais se exigia no mínimo um caminho com ligações de fibra ótica entre cada par de cidades;
- As *cidades pequenas*, para as quais não se exigia que fossem servidas pela fibra ótica.

É fácil ver que no caso particular em que o conjunto de *cidades grandes* é vazio, o problema fica reduzido ao *Problema de Steiner em Grafos*. Na seção 4.3 discutimos o modelo geral estudado pelos autores bem como sua particularização ao caso do Problema de Steiner em Grafos.

Filogenia

De acordo com as teorias de evolução correntes, as espécies hoje existentes descendem de ancestrais comuns no passado. De acordo com a escola darwinista, alguns cientistas representam tais relacionamentos – chamados *filogenias* – através de árvores. Algumas espécies possuem um registro fóssil de seus antepassados que tem permitido fazer um estudo comparado entre as diversas espécies. Porém alguns desses registros parecem ter sido perdidos, causando frequentemente discórdias entre as diversas teorias.

Várias tentativas têm sido feitas para se construir filogenias que expliquem melhor as dúvidas hoje existentes. Uma delas é utilizar seqüências de proteínas ao invés de espécies para tentar descrever tais relacionamentos. A idéia é que a evolução da proteína nas espécies pode esclarecer melhor o processo de evolução. Tipicamente uma proteína é uma seqüência de aminoácidos. Cada aminoácido pode ser representado por uma tripla ordenada de nucleotídeos. Como existem quatro nucleotídeos (A, C, G e U), existem 64 triplas possíveis. Entretanto, apenas 20 destas têm sido encontradas na natureza.

Foulds e Graham [17] apresentam a seguinte abordagem para a construção de uma filogenia:

Toma-se um grafo completo com 20 vértices, onde cada vértice corresponde a um dos 20 aminoácidos existentes. O custo de cada aresta que liga dois aminoácidos é dado pelo número de nucleotídeos em que esses aminoácidos diferem. A seguir, o pesquisador escolhe uma determinada proteína existente nas diversas espécies que ele deseja relacionar (tipicamente, a proteína respiratória citocromo c, hemoglobina α ou β ou fibri-nopeptídeo α ou β) e toma como conjunto de vértices obrigatórios aqueles correspondentes aos aminoácidos das espécies em estudo. A solução do problema de Steiner para este grafo nos dá a filogenia que é chamada de “filogenia com máxima concordância”.

Inferência de consultas a bases de dados

Uma aplicação surpreendente do *Problema de Steiner em Grafos* está descrita no projeto de um sistema gerenciador de bases de dados “tolerante” e “cooperativo” [41].

Um dos objetivos deste sistema é tentar determinar, através de poucas informações dadas pelo usuário, que consulta ele deseja fazer. Tal procedimento é chamado de GUESS.

Para conseguir isso, o sistema constrói uma rede semântica representando a base de dados, utilizando o conhecimento das dependências funcionais existentes entre seus atributos. Para mostrar o que ocorre, vejamos um exemplo de uma base de dados contendo informações sobre compositores clássicos e suas obras e suponhamos que um usuário queira realizar a seguinte consulta:

Liste o nome e país de origem do compositor de “A Flauta Mágica”.

O projetista do sistema deseja que o GUESS receba esta frase e forneça uma consulta válida e sintaticamente correta à base de dados. O que o sistema faz é extrair algumas palavras-chaves da frase: nome, origem, compositor, “A Flauta Mágica” e considerar naquela rede semântica os vértices correspondentes a estas palavras como obrigatórios. A solução do *PSG* relativo a tal rede e conjunto de obrigatórios, proporciona ao sistema inferir uma consulta sintaticamente correta à base de dados. Nesse exemplo, poderia ser algo como:

retrieve Nome, Origem from Compositor, Composição
where Nome = Autor and Titulo = "A Flauta Mágica".

A resolução da instância do *PSG* visou, segundo o autor, minimizar o número de suposições do sistema. Ele observa que o subgrafo devolvido pelo sistema pode dar origem a diversas consultas e que tais ambigüidades seriam eliminadas interativamente com o usuário.

2.2.2 Complexidade Computacional

Nesta subseção analisamos a complexidade computacional do *Problema de Steiner em Grafos*. Mostramos que o problema de decisão associado a este problema é $\mathcal{NP}$ -completo, o que nos leva a concluir que o correspondente problema de otimização é $\mathcal{NP}$ -difícil. Algumas vezes, no restante do trabalho, por imprecisão de linguagem, diremos que o problema é $\mathcal{NP}$ -completo.

Apresentamos a seguir uma redução do *problema da Cobertura Exata* ao *Problema de Steiner em Grafos* que é basicamente a demonstração sugerida por Karp em [31], com uma pequena modificação.

Denotaremos por P_dSG , respectivamente PCE , o problema de decisão associado ao *PSG*, respectivamente ao *Problema da Cobertura Exata*, enunciados abaixo:

P_dSG :

Dado um grafo G com uma função custo $c : EG \rightarrow \mathbb{R}$, um subconjunto $Z \subseteq VG$ e $k \in \mathbb{R}$, encontrar (se existe) um subconjunto $A \subseteq EG$ tal que o subgrafo $G[A]$ contém um caminho entre cada par de vértices de Z e $\sum_{e \in A} c_e \leq k$.

PCE :

Dado um conjunto $U = \{u_1, u_2, \dots, u_t\}$ e uma família $\mathcal{F} = \{V_1, V_2, \dots, V_r\}$ de subconjuntos de U , encontrar (se existe) uma subfamília de $\mathcal{F}$ que seja uma partição de U .

Exemplo: Dados

$$U = \{1, 2, 3, 4, 5\} \text{ e} \\ \mathcal{F} = \{\{1, 2, 3\}, \{4, 5\}, \{1, 2, 4\}, \{1, 3, 5\}, \{2, 3\}, \{1\}\},$$

a subfamília $\{\{1\}, \{2, 3\}, \{4, 5\}\}$ é uma cobertura exata de U .

Teorema 2 P_dSG é $\mathcal{NP}$ -completo.

Dem: Inicialmente observamos que $P_dSG \in \mathcal{NP}$. De fato, é fácil ver que dado $A \subseteq EG$, podemos verificar em tempo polinomial ($O(|EG|)$) se $G[A]$ contém um caminho entre cada par de vértices de Z . Basta utilizar um algoritmo de busca (em largura ou profundidade). Também, podemos verificar em tempo $O(|EG|)$ se $\sum_{e \in A} c_e \leq k$. Com isso, fica claro que $P_dSG \in \mathcal{NP}$.

Agora vamos mostrar que PCE pode ser reduzido a P_dSG em tempo polinomial.

Dada uma instância $\mathcal{I}_C$ do PCE ,

$$U = \{u_1, u_2, \dots, u_t\} \text{ e} \\ \mathcal{F} = \{V_1, V_2, \dots, V_r\},$$

construa a seguinte instância $\mathcal{I}_S$ do P_dSG :

- Um grafo G com

$$VG := \{u_1, u_2, \dots, u_t\} \cup \{n_0\} \cup \{V_1, V_2, \dots, V_r\} \text{ e} \\ EG := \{n_0V_i : i = 1, \dots, r\} \cup \{u_iV_j : u_i \in V_j, i = 1, \dots, t, j = 1, \dots, r\}.$$

- Uma função $c : EG \rightarrow \mathbb{R}$ assim definida

$$c(e) := \begin{cases} |V_i| & \text{se } e \text{ é do tipo } n_0V_i \text{ para algum } i = 1, \dots, r \\ t & \text{caso contrário.} \end{cases}$$

- $Z := \{u_1, u_2, \dots, u_t, n_0\}$.
- $k := t^2 + t$.

Na figura 2.9 apresentamos a instância do P_dSG correspondente à instância do PCE dada no exemplo.

Provaremos que $\mathcal{I}_C$ tem solução se e somente se $\mathcal{I}_S$ tem solução.

Primeiramente vamos mostrar que se uma subfamília $\mathcal{V} = \{V_{i_1}, V_{i_2}, \dots, V_{i_l}\}$ de $\mathcal{F}$ é cobertura exata de U então o subgrafo $G[I(\mathcal{V})]$ é solução de $\mathcal{I}_S$, onde

$$I(\mathcal{V}) = \{e \in EG : e \text{ incide em algum vértice } V_{i_j} \text{ que pertence a } \mathcal{V}\}.$$

Como $\mathcal{V}$ é cobertura exata de U , existe em $G[I(\mathcal{V})]$ exatamente uma aresta incidente a cada vértice correspondente a um elemento de U (tais arestas têm custo total t^2). Como cada vértice correspondente a um membro de $\mathcal{V}$ está ligado a n_0 , existe um caminho entre n_0 e cada vértice correspondente a um elemento de U , e portanto, $G[I(\mathcal{V})]$

Na figura 2.10 as arestas mais escuras do grafo formam o subgrafo $G[I(\mathcal{V})]$ correspondente à cobertura exata $\mathcal{V} = \{\{1\}, \{2, 3\}, \{4, 5\}\}$ de U .

Tomemos $\mathcal{V} := \{V_j : n_0 V_j \in ET\}$. Como n_0 e os vértices correspondentes a elementos de U são vértices obrigatórios, existe em T um caminho entre n_0 e cada um desses vértices. Pela estrutura do grafo G , podemos concluir que $\mathcal{V}$ é uma cobertura de U e portanto $\sum_{V_j \in \mathcal{V}} |V_j| \geq t$.

Como

22

Figura 2.10

Assim,

$$t \geq \sum_{e \in ET \cap I(n_0)} c(e) = \sum_{V_j \in \mathcal{V}} |V_j| \geq t.$$

Logo, $\mathcal{V}$ é cobertura exata de U , e portanto é uma solução de $\mathcal{I}_C$.

□

Na redução apresentada por Karp [31] constatamos um pequeno deslize. A função $c : EG \rightarrow \mathbb{R}$ foi definida como descrevemos, exceto nas arestas incidentes a vértices correspondentes a elementos de U onde o autor sugeria custo 0. Além disso, foi considerado $k := t$. É fácil ver que com tal definição a seguinte instância $\mathcal{I}_C$ do PCE não tem solução, mas a correspondente instância do P_dSG tem solução:

$$U := \{1, 2, 3\} \text{ e } \mathcal{F} = \{\{1, 2\}, \{1, 3\}, \{2, 3\}\}.$$

Observe que a redução que apresentamos mostra que o problema permanece $\mathcal{NP}$ -completo mesmo quando aplicado a grafos bipartidos com bipartição $(Z, VG - Z)$. Note também que como t faz parte do tamanho da instância de PCE e na redução obtivemos arestas com custo de no máximo t , poderíamos subdividir em tempo polinomial cada aresta de custo s em s arestas de custo 1. Isto indica, como sugerem Garey e Johnson [19], que mesmo se todas as arestas do grafo têm custo unitário o problema continua $\mathcal{NP}$ -completo.

Na verdade, quando afirmamos na seção anterior que o problema de Steiner na métrica retilinear é $\mathcal{NP}$ -completo, ficou implícito que mesmo para grafos planares e bipartidos o PSG é $\mathcal{NP}$ -completo.

Por outro lado, existe algoritmo polinomial para o problema se fixarmos o número de vértices facultativos (veja a subseção 3.1.1) e também se fixarmos o número de vértices obrigatórios (veja a subseção 3.1.2). Na seção 3.4 apresentamos algumas classes de grafos para as quais o *Problema de Steiner* pode ser resolvido em tempo polinomial.

2.2.3 Algumas Variantes

O *Problema de Steiner em Grafos* tem algumas variantes bastante interessantes que comentaremos rapidamente, procurando fornecer algumas referências para o leitor interessado.

O Problema de Steiner em Grafos Dirigidos

O *Problema de Steiner em Grafos Dirigidos*, segundo Maculan [36], foi sugerido pela primeira vez por Nastansky, Selkow e Stewart [42] e pode ser definido da seguinte forma:

*Seja $G_d = (VG_d, AG_d)$ um grafo dirigido fortemente conexo, onde a cada arco $a \in AG_d$ está associado um custo positivo $c(a)$. Tome $s \in VG_d$ e $Z \subseteq VG_d - \{s\}$. Chamaremos s de **raiz**. O problema consiste, então, em achar caminhos dirigidos de s para cada vértice de Z de tal forma que o custo total das arestas pertencentes a esses caminhos seja mínimo.*

Se $|Z| = 1$ o problema se reduz à determinação de um caminho dirigido mais curto em um grafo dirigido. Para este problema conhecem-se algoritmos rápidos (veja, por exemplo, [10]). Quando $|Z| = |VG_d| - 1$ queremos determinar a menor árvore geradora dirigida com raiz em s . Neste caso temos um problema de intersecção de dois matróides, para o qual sabemos existir algoritmos polinomiais (veja [22]).

Quando $1 < |Z| < |VG_d| - 1$ o problema é $\mathcal{NP}$ -difícil. Veja a prova em [60].

Em [36] Maculan observa que o *PSG* pode ser transformado no Problema de Steiner em Grafos Dirigidos. No mesmo artigo, o autor faz uma resenha bastante completa sobre este problema. Um outro trabalho do mesmo autor que indicamos para o leitor interessado é o [37].

Um caso particular deste problema, quando $|Z| = 2$ foi estudado por Ball, Liu e Pulleyblank [3] através de uma abordagem poliédrica.

Problema de Steiner com pesos nos vértices

Segev em [50] introduz uma outra variante para o problema de Steiner em Grafos que descrevemos a seguir:

Seja $G = (VG, EG)$ um grafo com uma função custo $c : EG \rightarrow \mathbb{R}$ e uma função $w : VG \rightarrow \mathbb{R}$. Dado um conjunto $Z \subseteq VG$ queremos determinar um subgrafo T de G que possua um caminho entre cada par de vértices de Z e tal que

$$\sum_{v \in VT} w(v) + \sum_{e \in ET} c(e)$$

seja a menor possível.

Em seu trabalho, Segev mostra que o problema é $\mathcal{NP}$ -completo. O autor estuda o caso em que a função w definida para os vértices tem apenas valores negativos, ou seja, $w(v) < 0, \forall v \in VG$. Tal particularização é motivada, segundo o autor, pelo fato de que na maioria das aplicações a inclusão de vértices facultativos acarreta benefícios ao sistema.

Neste artigo, Segev apresenta alguns algoritmos para o problema baseados em relaxação lagrangeana, e também algumas heurísticas cujos resultados computacionais ele considera bastante satisfatórios.

O Problema da Floresta de Steiner

Duin e Volgenant [12] observam que em algumas aplicações práticas nem sempre é necessário que a solução seja um grafo conexo. Assim eles sugerem a seguinte variação do PSG:

Dado um grafo G com uma função custo $c : EG \rightarrow \mathbb{R}$, um subconjunto $Z \subseteq VG$ e um inteiro $1 \leq m < |Z|$, determinar um subgrafo de G com custo total mínimo que tenha no máximo m componentes, cada uma com pelo menos um vértice de Z .

O autor observa ainda que este problema pode ser generalizado associando uma função de benefício aos vértices. No mesmo trabalho, os autores sugerem uma abordagem para o problema baseada em relaxação lagrangeana.

Capítulo 3

Métodos de Solução : um Breve Resumo

O *Problema de Steiner em Grafos*, como já dissemos, foi primeiramente abordado por Hakimi [28] e Dreyfus e Wagner [11] em 1971. Desde então, diversos métodos de solução têm sido desenvolvidos para esse problema, motivados talvez por suas inúmeras aplicações.

Em 1987, Winter [57] publicou uma resenha dos diversos métodos de solução tratados na literatura: algoritmos exatos, métodos heurísticos e algoritmos polinomiais para classes particulares de grafos. Nesta resenha o autor faz referência a 92 outros trabalhos sobre o assunto.

Nesta nossa resenha, procuramos descrever os principais destes resultados, alguns já presentes na de Winter, sem, no entanto, detalhar minuciosamente cada um dos métodos. Vamos apresentar a idéia do algoritmo e descrever rapidamente o seu funcionamento, quase sempre fornecendo exemplos ilustrativos. A escolha, um tanto subjetiva, dos algoritmos foi motivada pelo desejo de abordar as mais diferentes técnicas aplicadas ao problema.

É interessante notar que as mais diferentes abordagens já foram aplicadas ao *Problema de Steiner em Grafos* sem que nenhum algoritmo “bom” tenha sido descoberto. Isso não causa surpresa já que, como vimos no capítulo anterior, o problema é $\mathcal{NP}$ -completo. Na verdade, este fato tem motivado pesquisas no sentido de encontrar classes relevantes de grafos para os quais o problema possa ser resolvido em tempo polinomial. Ultimamente muitas classes de tais grafos têm sido descobertas.

3.1 Algoritmos Exatos

O *Problema de Steiner em Grafos*, como vimos no capítulo anterior, é $\mathcal{NP}$ -completo. Assim, os algoritmos exatos que são conhecidos para o problema têm complexidade exponencial no tamanho da entrada.

Os métodos exatos para resolver o problema são, em última análise, algoritmos de busca exaustiva de todas as possíveis soluções. Apresentaremos nesta seção alguns destes algoritmos e discutiremos as suas complexidades computacionais.

3.1.1 Busca na árvore de soluções

Em [52], Shore, Foulds e Gibbons discutem inicialmente que nenhum bom algoritmo havia sido desenvolvido para o problema o que justificava o desenvolvimento de uma estratégia de busca exaustiva (*branch and bound*).

O método se baseia nas seguintes idéias:

Dada uma instância do $PSG(G, Z)$, observemos que:

1. Para cada aresta e do grafo, temos duas possibilidades:
 - ou e está em uma solução ótima e então podemos contrair tal aresta e seguir com o algoritmo;
 - ou e não está em nenhuma solução ótima, e aí tal aresta pode ser removida.
2. Utilizando a idéia acima poderíamos construir uma árvore binária com $2^{|EG|+1} - 1$ vértices. Associado a cada folha dessa árvore binária (veja a figura 3.1) temos um subgrafo do grafo original que pode ou não ser uma solução ótima para o $PSG(G, Z)$. Observe que a árvore tem $2^{|EG|}$ folhas.
3. Percorrendo-se cada folha dessa árvore procuramos por uma solução ótima do problema.

Observe que o tamanho dessa árvore cresce exponencialmente com o número de arestas do grafo e , portanto, se torna inviável criar toda a árvore se o grafo não for pequeno. Além disso, temos $|EG|!$ possíveis permutações na ordem em que as arestas podem ser escolhidas. Assim temos $|EG|!$ possíveis árvores distintas a verificar por uma solução ótima !!!! Como a busca em algumas dessas árvores pode ser mais produtiva no que nas outras, os autores sugerem então um algoritmo que corta (*bound*) alguns caminhos improdutivos, eliminando dessa forma alguns dos ‘galhos’ daquela árvore de tamanho exponencial. Para a escolha da aresta a ser utilizada em cada passo (*branching*), os autores sugerem também uma heurística.

Figura 3.1

Branching

A heurística sugerida pelos autores a fim de escolher qual aresta utilizar para o *branching*, isto é, para gerar dois ramos na árvore, está nas seguintes funções peso definida nos vértices obrigatórios do grafo:

- $w^* : Z \longrightarrow \mathbb{R}$
 $w^*(v) := \min\{c(e) : e \in I(v)\}.$
- $w^+ : Z \longrightarrow \mathbb{R}$
 $w^+(v) := \text{peso da segunda menor aresta que incide em } v.$
- $p : Z \longrightarrow \mathbb{R}$
 $p(v) := w^+(v) - w^*(v).$

Intuitivamente, a função p indica para cada vértice de Z o grau de “facilidade” para decidir qual aresta do grafo deve atingi-lo. Isto é, dado v , quanto maior for $p(v)$, mais fácil é a escolha da “melhor” aresta a incidir neste vértice.

No exemplo da figura 3.2, como $p(z_1) > p(z_2)$, intuitivamente, é mais fácil decidir para z_1 a escolha da ‘melhor’ aresta que deve incidir nesse vértice.

O algoritmo escolhe a cada passo um vértice para o qual a função p atinge o máximo e então seleciona para o *branching* a aresta de menor peso incidente a este vértice. No exemplo da figura 3.2, escolhe-se para o *branching* a menor aresta incidente a z_1 , ou seja, a aresta de peso 7.

Exemplo:

$$p(z_1) = 12 - 7 = 5$$

$$p(z_2) = 7 - 4 = 3$$

Figura 3.2

Depois de escolhida uma aresta e , devemos construir os grafos $G' = G - e$ e $G'' = G/e$ para prosseguir a busca. Tais construções podem ser feitas de forma eficiente conforme [52].

Bounding

O corte de alguns ramos da árvore está baseado no cálculo de limitantes inferiores (*lower bounds*) para esses vértices. Se ocorrer para algum vértice que seu limitante inferior é maior que o valor de alguma solução já encontrada, então podemos descartá-lo e também todos os seus descendentes.

Os limitantes inferiores utilizados pelo algoritmo são apresentados e demonstrados a seguir. Lembremos antes que

$$E(Z) := \{e \in EG : e \text{ tem ambas as pontas em } Z\}.$$

Sejam

$$b_1 = \sum_{v \in Z} \min\{c(e) : e \in I(v)\} \text{ e}$$

$$b_2 = \sum_{v \in Z} \min\{c(e) : e \in E(Z) \cap I(v)\} - \min\{c(e) : e \in E(Z)\}.$$

Teorema 3 Se z^* é o valor ótimo então,

$$z^* \geq \min\{b_1, b_2\}.$$

Dem: Tomemos uma solução ótima T^* de custo z^* . Sabemos que T^* é subgrafo de G e $Z \subseteq VT^* \subseteq VG$.

Caso 1: $VT^* = Z$.

Em cada vértice de Z incide uma aresta de ET^* , assim para todo $v \in Z$, e qualquer aresta e de T^* incidente a v , temos que

$$c(e) \geq \min\{c(f) : f \in E(Z) \cap I(v)\}.$$

Como $|ET^*| = |Z| - 1$,

$$\sum_{e \in ET^*} c(e) \geq \sum_{v \in Z} \min\{c(e) : e \in E(Z) \cap I(v)\} - \min\{c(e) : e \in E(Z)\} = b_2.$$

Caso 2: $VT^* \supset Z$.

Para cada vértice v de Z , sabemos que existe em T^* uma aresta a_v incidente a v tal que para todo par $u, w \in Z$, se $u \neq v$ então $a_u \neq a_w$. Claramente,

$$c(a_v) \geq \min\{c(e) : e \in I(v)\},$$

e portanto,

$$z^* \geq \sum_{v \in Z} c(a_v) \geq \sum_{v \in Z} \min\{c(e) : e \in I(v)\} = b_1.$$

Assim, $z^* \geq \min\{b_1, b_2\}$. $\square$

Na figura 3.3 vemos exemplos em que um dos limitantes coincide com o valor ótimo.

Figura 3.3

Na figura 3.4 ilustramos a execução do algoritmo para um exemplo. Observe que os vértices da árvore foram numerados de acordo com a ordem em que foram analisados. Os vértices assinalados com um X possuem limitante inferior maior que o valor de uma solução encontrada.

Exemplo da execução do algoritmo:

S : valor da solução atual

LB : valor do melhor limitante inferior da solução atual

Figura 3.4: Algoritmo de Busca na árvore de soluções

3.1.2 Enumeração explícita

Hakimi foi em 1971 [28] um dos primeiros a estudar o *Problema de Steiner em Grafos*. Nessa publicação pioneira Hakimi discute alguns dos casos particulares vistos no capítulo anterior e observa como este problema se relaciona com outros da Teoria de Grafos, tais como, *problema da cobertura mínima*, *problema do conjunto independente máximo*, *problema do clique máximo* e outros.

O autor sugere em seu artigo um algoritmo para resolver o problema. Na verdade, dois algoritmos são apresentados, porém o segundo é um melhoramento do primeiro (que aqui apresentamos). Nas suas considerações finais, o autor atesta que utilizando-se o algoritmo, o número de possibilidades que são testadas até obter uma solução ótima é muito grande para instâncias cujo conjunto Z de vértices obrigatórios tem mais de 9 vértices.

O algoritmo de Hakimi pode ser descrito da seguinte forma:

1. Para cada $W \subseteq VG$ tal que $W \supseteq Z$ calcule uma árvore geradora mínima de $G[W]$.
2. Tome como solução a menor entre todas essas árvores.

É fácil ver que este algoritmo tem complexidade polinomial em $|Z|$ e exponencial em $|VG - Z|$.

De fato, se denotarmos por $t(k)$ o tempo para calcular uma árvore geradora mínima para um grafo com k vértices, então o tempo total gasto T pelo algoritmo pode ser assim calculado:

$$\begin{aligned}
 T &= t(|Z|) + \binom{|VG - Z|}{1} t(|Z| + 1) + \dots + \binom{|VG - Z|}{|VG - Z|} t(|VG|) \\
 &\geq t(|Z|) \left(\binom{|VG - Z|}{1} + \binom{|VG - Z|}{2} + \dots + \binom{|VG - Z|}{|VG - Z|} \right) \\
 &= t(|Z|) 2^{|VG - Z|}.
 \end{aligned}$$

Portanto, o algoritmo não é computacionalmente eficiente. Veja na figura 3.5 uma simulação de sua execução.

Usando o mesmo princípio, Lawler descreve, em [34], um algoritmo polinomial em $|Z|$ e exponencial em $|VG - Z|$ que enumera todas essas árvores. Como veremos, em certos casos este algoritmo é mais eficiente que o anterior.

A idéia de Lawler consiste em construir a partir do grafo dado G , um grafo completo (sobre o mesmo conjunto de vértices) e uma função peso $p : EG \rightarrow R$ que

Exemplo: Considere o grafo G abaixo, onde $Z = \{b, c, e, g\}$. Denotando por $P(W)$ o custo de uma árvore geradora mínima de $G[Z \cup W]$ temos:

$$P(\phi) = +\infty \text{ (floresta)}$$

$$P(\{a\}) = +\infty$$

$$P(\{f\}) = 51$$

$$P(\{d\}) = 64$$

$$P(\{h\}) = 30$$

$$P(\{a, f\}) = 37$$

$$P(\{a, h\}) = 18\sqrt{}$$

$$P(\{a, d\}) = 52$$

$$P(\{f, h\}) = 32$$

$$P(\{f, d\}) = 45$$

$$P(\{d, h\}) = 39$$

$$P(\{a, f, h\}) = 18\sqrt{}$$

$$P(\{a, f, d\}) = 31$$

$$P(\{a, d, h\}) = 25$$

$$P(\{d, h, f\}) = 39$$

$$P(\{a, d, h, f\}) = 25$$

Figura 3.5: algoritmo de enumeração explícita

associa cada aresta uv a distância de u a v em G . Sabemos que essa função peso respeita a propriedade euclidiana, ou seja, para quaisquer vértices $u, v, w \in V$

$$p(uv) \leq p(uw) + p(wv)$$

Lawler mostra, então, que se a função custo respeita a propriedade acima o número de vértices facultativos a serem investigados pode ser reduzido. Discutimos esse fato na proposição abaixo.

Proposição 1 *Dado um grafo completo H em que a função peso $p : EH \rightarrow \mathbb{R}$ é tal que:*

1. $p(e) \geq 0 \forall e \in EH$
2. $p(uv) \leq p(uw) + p(wv) \quad \forall u, v, w \in VH$

então qualquer que seja $Z \subseteq VH$, o $PSG(H, Z)$ tem uma solução com no máximo $|Z| - 2$ pontos facultativos.

Dem: Tomemos uma solução ótima T para o problema com um número mínimo de arestas.

Sejam:

$$\begin{aligned} q &:= |VT - Z|, \\ x &:= \min\{\text{grau}(v) \text{ em } T : v \in VH - Z\}, \\ y &:= \min\{\text{grau}(v) \text{ em } T : v \in Z\}. \end{aligned}$$

Sabemos que a solução ótima T é árvore pois a função peso é positiva, e portanto:

$$q + |Z| - 1 = |ET| \geq \frac{qx + |Z|y}{2}. \quad (*)$$

Mas, $y \geq 1$, pois todo vértice obrigatório deve ser atingido e $x \geq 3$, pois no caso contrário, como a função p respeita a métrica euclidiana, teríamos uma solução ótima T' com $|ET'| < |ET|$. Logo

$$q + |Z| - 1 \geq \frac{3q + |Z|}{2},$$

donde $q \leq |Z| - 2$.

□

A proposição acima nos sugere o seguinte algoritmo de enumeração explícita:

1. a partir de G construa G' completo em que o custo de cada aresta corresponde à distância em G entre os vértices ligados por essa aresta (tempo necessário $O(|VG|^3|))$);
2. resolva o $PSG(G', Z)$ encontrando uma solução ótima T' ;
3. a partir de T' reconstrua uma solução para G , expandindo os caminhos mínimos de G para cada aresta de T' .

Se T_1 é o tempo para executar o passo (1), T_3 o tempo para executar o passo (3) e $t(k)$ o tempo para calcular uma árvore geradora mínima de um grafo com k vértices, o tempo total T gasto por este algoritmo é

$$T = T_1 + t(|Z|) + t(|Z| + 1) \binom{|VG| - |Z|}{1} + \dots + t(2|Z| - 2) \binom{|V| - |Z|}{|Z| - 2} + T_3.$$

Logo,

$$T \geq T_1 + T_3 + \sum_{k=0}^{|Z|-2} \binom{|VG| - |Z|}{k} t(|Z|).$$

Como sabemos que o passo (1) pode ser executado em tempo polinomial (veja, por exemplo [16]), e ainda que o passo (3) pode ser executado de forma eficiente através da escolha de uma estrutura adequada, então a complexidade do algoritmo é dominada pela expressão do passo (2). Como era de se esperar, o algoritmo de Lawler é mais eficiente que o algoritmo descrito por Hakimi, quando

$$|Z| - 2 \leq |VG| - |Z|,$$

ou seja,

$$|Z| < \frac{|VG| + 2}{2}.$$

O segundo algoritmo para o problema descrito por Lawler no mesmo texto é baseado em programação dinâmica e será discutido na próxima seção.

3.1.3 Algoritmo baseado em Programação Dinâmica

Dreyfus e Wagner sugeriram em [11] um algoritmo para o *Problema de Steiner em Grafos* baseado no princípio de programação dinâmica.

O princípio geral desta abordagem consiste em construir soluções ótimas de subproblemas de tamanho cada vez maior a partir de soluções ótimas de subproblemas menores, até atingir uma solução ótima para o problema original. Aplicando tal idéia ao Problema de Steiner em Grafos, Dreyfus e Wagner conseguiram um algoritmo polinomial em $|VG|$ e exponencial em $|Z|$. A versão do algoritmo que veremos aqui foi modificada por Lawler em [34]. Sem perda de generalidade, podemos supor que o grafo G e a função c com que estamos lidando são como na seção anterior (em que o custo de cada aresta corresponde à distância entre os vértices em que esta incide).

A idéia se baseia no fato de que construindo-se soluções para o problema aplicado a conjuntos de tamanho $1, 2, \dots, |Z| - 1$ podemos obter relações que nos permitem combinar tais soluções e obter uma solução ótima para $PSG(G, Z)$.

A seguir detalhamos um pouco mais o algoritmo:

Seja $\phi \neq K \subseteq Z$, $u \in VG - K$ e denotemos por

$T_{SP}(K + u)$: custo de uma solução ótima para $PSG(G, K + u)$.

$T_{SP_2}(K + u)$: custo de uma solução ótima como no caso anterior com a restrição adicional que o vértice u tenha grau pelo menos 2.

Com a restrição adicional de o vértice u não ser folha, podemos obter uma relação para $T_{SP_2}(K + u)$ a partir de T_{SP} da seguinte forma:

$$T_{SP_2}(K + u) = \min_{\emptyset \subset K' \subset K} \{T_{SP}(K' + u) + T_{SP}(K - K' + u)\}. \quad (*)$$

Resta apenas obter uma relação semelhante para $T_{SP}(K + u)$ e poderemos aplicar o método a partir de conjuntos pequenos e atingir Z .

Vamos analisar todos os casos para um vértice u em uma solução ótima T^* cujo custo é $T_{SP}(K + u)$:

1. $\text{grau}(u) \geq 2$ na árvore T^* .

Nesse caso, $T_{SP}(K + u) = T_{SP_2}(K + u)$.

2. u é folha na árvore T^* .

Aqui temos duas possibilidades quanto ao vizinho v de u na árvore T^* :

- (a) $v \notin Z$

Como u e v são vértices facultativos, $\text{grau}(v) \geq 3$ em T^* . Logo, $T_{SP}(K + u) = c(uv) + T_{SP_2}(K)$.

- (b) $v \in Z$,
Nesse caso,

$$T_{SP}(K + u) = c(uv) + T_{SP}(K).$$

Juntando os casos (1) e (2) , obtemos :

$$T_{SP}(K + u) = \min_{v \in I(u)} \left\{ T_{SP_2}(K + u), \min_{v \in VG} \{c(uv) + T_{SP}(K)\} \right\}. (**)$$

As equações (*) e (**) juntas com a convenção de que $T_{SP}(\phi) = 0$ (corresponde a uma árvore vazia), fornecem um meio de computar uma solução ótima com custo $T_{SP}(Z)$. Vamos analisar a complexidade computacional deste algoritmo.

A relação (*) tem sua complexidade T_1 associada a todas as possíveis escolhas para u , K e K' , ou seja, temos $|VG|$ escolhas para u e para cada vértice de $Z - u$ temos três escolhas possíveis (K' , $K - K'$, $Z - K$):

$$T_1 \leq O(|VG| 3^{|Z|}).$$

A complexidade T_2 associada a (**) está ligada a todas as possíveis escolhas para u e K e a minimização de todas as possíveis escolhas para v :

$$\begin{aligned} T_2 &\leq |VG| 2^{|Z|} |VG| \\ &\leq O(|VG|^2 2^{|Z|}). \end{aligned}$$

Assim, a complexidade T desse algoritmo é $O(|VG| 3^{|Z|} + |VG|^2 2^{|Z|} + |VG|^3)$ lembrando que o último termo ($|VG|^3$) é o tempo necessário para a construção do grafo de distâncias. Observe que o algoritmo resultou polinomial em $|VG|$ e exponencial em $|Z|$.

3.1.4 Abordagens usando Programação Inteira

Diversos autores sugerem a formulação do $PSG(G, Z)$ como problema de programação inteira (ou 0-1) e combinado com algum tipo de relaxação linear para obter limitantes inferiores na busca de soluções. Alguns dos trabalhos onde encontramos tal abordagem são [4], [5], [36], [40] (que usam relaxação de Lagrange para obter os limitantes) e [1] (que usa um procedimento de “geração de linhas”). Vamos discutir as várias abordagens de forma geral aprofundando-nos nos trabalhos acima citados.

As abordagens que utilizam Programação Inteira são baseadas nas seguintes idéias gerais:

1. Formula-se o problema a ser tratado como um problema de programação 0-1:

$$\begin{aligned} & \text{Min } \sum_{e \in EG} c(e)x(e) \\ & \text{s.a. } \begin{cases} A_1x \leq b_1 \\ A_2x = b_2 \\ x(e) \in \{0, 1\} \end{cases} \end{aligned}$$

2. Relaxa-se o problema obtido para um problema de programação linear substituindo-se as restrições de integralidade pelas restrições $x(e) \geq 0$ e $x(e) \leq 1 \forall e \in EG$ e então resolve-se o novo problema. Na verdade, do ponto de vista teórico existem algoritmos polinomiais para resolver um problema de programação linear, porém, na prática, por enquanto o algoritmo simplex (exponencial no pior caso) continua a ser utilizado por apresentar tempo médio melhor que aqueles (veja [51]).
3. Se a solução ótima x^* do problema relaxado for inteira, então x^* é solução ótima do problema 0-1. Caso contrário, podemos tomar x' tal que

$$x'(e) := \lceil x^*(e) \rceil, \forall e \in EG.$$

neste caso, cx' é um bom limitante para algoritmos de busca exaustiva (*branch and bound*) .

□

Vejamos a seguir algumas das formulações sugeridas na literatura para o *Problema de Steiner* e os respectivos limitantes.

Primeira formulação

A primeira formulação que apresentamos encontra-se no livro de Minoux [40] e é devida, segundo o autor, a Claus e Maculan. Para obter essa formulação, considere dada uma árvore T de G tal que $VT \supseteq Z$. Além disso, tome um vértice $s \in Z$, considere-o como raiz desta árvore e oriente as arestas da árvore saindo de s e caminhando em direção às suas folhas. Veja o exemplo na figura 3.6.

Observe que, na árvore dirigida T' assim construída, existe um caminho dirigido de s até cada um dos outros vértices de Z .

Tome o grafo dirigido G' obtido de G trocando-se cada aresta uv por dois arcos (um em cada direção) com o mesmo custo que uv tinha em G . Se tomarmos um vértice $s \in Z$ o problema então se reduz ao de achar capacidades $x^*(e) \in \{0, 1\}$ que devem ser atribuídas a cada arco de tal forma que um fluxo de valor 1 possa ser enviado de s a cada um dos outros elementos de Z e que minimizem $\sum_{a \in EG'} c(a)x(a)$ (veja a figura 3.7).

Para formular a idéia da existência de um fluxo de s aos outros vértices de Z usaremos algumas definições:

Figura 3.6

Figura 3.7

- para cada vértice $v \in Z - \{s\}$ seja b^v um vetor indexado por VG' assim definido:

$$b^v(w) = \begin{cases} 1 & , se w = v; \\ -1 & , se w = s; \\ 0 & , cc. \end{cases}$$

- para cada vértice $v \in Z - \{s\}$ seja φ^v um vetor indexado por EG (representando um fluxo de s a v).
- seja A a matriz de incidências do grafo dirigido G' .

Com isso o problema pode ser formulado da seguinte forma :

$$\begin{aligned} & Min \sum_{a \in EG'} c(a)x(a) \\ s.a. \quad & \begin{cases} A\varphi^v = b^v & , \forall v \in Z - \{s\} \\ \varphi^v \leq x & , \forall v \in Z - \{s\} \\ \varphi^v \geq 0 \\ x \in \{0, 1\}^{|EG|} \end{cases} \end{aligned}$$

O sistema de equações é para garantir que para cada $v \in Z - \{s\}$, φ^v é um fluxo de valor 1 de s a v .

O autor não demonstra os limitantes que podem ser obtidos quando relaxamos alguma das equações ou inequações que definem o sistema no estilo de relaxação de Lagrange, porém afirma que os resultados obtidos são muito bons: limitantes com apenas 5% de erro. Afirma, porém, que a qualidade dos limitantes cai sensivelmente quando o tamanho do problema cresce (50 a 100 vértices).

Segunda Formulação

Em 1984 Beasley [4] apresentou uma formulação bastante semelhante a que descrevemos acima:

Escolhemos um vértice $v \in Z$ e consideramos para cada aresta e e vértice $u \in Z$, $u \neq v$:

$$\begin{aligned} x(e) &:= \begin{cases} 1 & \text{se } e \text{ está na solução ótima} \\ 0 & \text{caso contrário;} \end{cases} \\ y_{eu} &:= \begin{cases} 1 & \text{se na solução ótima a aresta } e \text{ está no caminho dirigido de } v \text{ até } u \\ 0 & \text{caso contrário.} \end{cases} \end{aligned}$$

Com essas definições o problema pode ser assim formulado:

$$\text{Min } \sum_{a \in EG} c(a)x(a)$$

$$\text{s.a. } \begin{cases} x(e) \geq y_{eu} \quad \forall u \in Z - \{v\}, \forall e \in EG \\ |Z| - 1 \leq \sum_{e \in EG} x(e) \leq |VG| - 1 \\ \text{existe um caminho dirigido entre } v \text{ e } u \text{ para todo } u \in Z - \{v\} \\ \text{(expresso em termos das variáveis } y) \\ x(e) \in \{0, 1\} \end{cases}$$

Que pode ser lido como

1. todo vértice de Z deve ser atingível a partir de v e
2. o número de arestas na solução ótima deve ser pelo menos $|Z| - 1$ (árvore apenas com os obrigatórios) e no máximo $|VG| - 1$

Relaxando-se a primeira inequação de forma lagrangeana obtemos, utilizando o chamado procedimento do subgradiente, limitantes inferiores para realizar buscas mais eficientes na árvore de busca da solução.

Beasley fez uma implementação de seu algoritmo em computador e resolveu o PSG para instâncias com até 100 vértices e 200 arestas.

Terceira Formulação

Beasley [5], o mesmo autor da formulação anterior, desenvolveu em 1989 uma nova abordagem e melhorou seu algoritmo, chegando a resolver instâncias do $PSG(G, Z)$ com até 2500 vértices e 62500 arestas.

Apesar de a utilização de equipamentos distintos nos dois artigos não permitir uma comparação mais adequada (em 1984 o autor utilizou um CDC 7600 enquanto que em 1989 um CRAY X-MP), deve-se reconhecer que o algoritmo mais recente apresenta diversos melhoramentos:

1. Implementação de vários resultados de redução do tamanho da instância (que discutiremos na seção 3 deste capítulo), devidas a Balakrishnan e Patel [2].
2. Melhoramento do limitante inferior, obtido através de uma formulação baseada no *Problema da Árvore Geradora Mínima* de um grafo.

A idéia da formulação pode ser descrita como vemos a seguir. Escolha w em Z e construa a partir de G um grafo $G' = (VG', EG')$ onde

$$\begin{aligned} VG' &:= VG \cup \{v_0\}, v_0 \notin VG, \\ EG' &:= EG \cup \{v_0v : v \in VG - Z\} \cup \{v_0w\} \end{aligned}$$

e tome $c'(a) = \begin{cases} c(a) & a \in EG \\ 0 & \text{caso contrário.} \end{cases}$

Figura 3.8

Resolve-se, então, o problema de determinar uma árvore geradora mínima T' de G' (veja a figura 3.8) com a restrição adicional de que em T' todo vértice facultativo ligado a v_0 tenha grau 1. Chamaremos tal problema de *auxiliar*.

É fácil ver que uma solução ótima para o problema auxiliar aplicado a G' tem o mesmo valor que uma solução ótima de $PSG(G, Z)$ e também, que a partir de uma solução de um dos problemas obtemos a solução para o outro.

Assim, a formulação pode ser descrita como:

$$\begin{aligned} & \text{Min } \sum_{a \in EG} c(a)x(a) \\ \text{s.a. } & \begin{cases} \{e \in EG' : x(e) = 1\} \text{ define uma árvore geradora de } G' \\ x(v_0v) + x(vw) \leq 1 \quad \forall v \in VG - Z, \forall w \in VG \\ x(e) \in \{0, 1\} \quad \forall e \in EG' \end{cases} \end{aligned}$$

A segunda inequação é para garantir que na solução ótima os vértices facultativos ligados a v_0 tenham grau 1. É fácil também obter uma formulação em termos de inequações lineares para a primeira restrição. O limitante inferior obtido quando a segunda inequação é relaxada no estilo lagrangeano, permitiu ao autor obter os melhores resultados da literatura quanto ao tamanho da instância resolvida.

Quarta formulação

Essa formulação foi obtida por Aneja [1] em 1980 e de certa forma é a que mais se aproxima dos resultados que obtivemos em nossa abordagem poliédrica (discutida no

próximo capítulo).

A idéia do autor consiste em considerar todos os cortes que separam vértices obrigatórios no grafo G . Seja $x(e)$ um vetor de 0/1 indexado pelas arestas do grafo. Se cada corte tiver pelo menos uma de suas arestas na solução x , ou seja, uma aresta e tal que $x(e) = 1$, então x induz uma solução viável para o PSG : um subgrafo conexo que contém o conjunto de vértices obrigatórios.

A formulação é descrita a seguir. Sejam $C_1, C_2, \dots, C_\rho$ os cortes de G que separam vértices obrigatórios. Para toda aresta $e \in EG$ e $k = 1, 2, \dots, \rho$ tome

$$a_{ek} = \begin{cases} 1 & \text{se } e \in C_k \\ 0 & \text{caso contrário.} \end{cases}$$

Assim, o problema pode ser formulado como

$$\begin{aligned} & \text{Min } \sum_{e \in E} c(e)x(e) \\ \text{s.a. } & \begin{cases} \sum_{e \in EG} a_{ek}x(e) \geq 1 & k = 1, \dots, \rho. \\ x(e) \in \{0, 1\}. \end{cases} \end{aligned}$$

É fácil ver que uma solução para o problema acima proposto é solução ótima do $PSG(G, Z)$. No entanto, o número total de tais cortes cresce exponencialmente com o tamanho da instância, e, portanto, não podemos resolver o problema com todos os cortes. Aneja sugere, então, um procedimento de “geração de linhas” (geração de cortes) que escolhe quais cortes incluir no sistema.

Para isso o autor introduz a idéia de cobertura não redundante. Cada solução $x = (x_1, x_2, \dots, x_{|EG|})$ do problema pode ser associada a um conjunto $J := \{e : x(e) \neq 0\}$. Chamaremos J de *cobertura* do problema. Uma cobertura J é *não redundante* se nenhum subconjunto próprio de J é também uma cobertura.

Assim, tome $J = \{j_1, j_2, \dots, j_k\}$ uma cobertura não redundante associada a uma solução viável do problema e obtenha uma matriz $\bar{A}_{\rho \times k}$ (restrição da matriz $A_{\rho \times k}$)

$$\bar{a}_{ij} = \begin{cases} 1 & \text{se o corte } i \text{ contém a aresta } j \\ 0 & \text{caso contrário.} \end{cases}$$

Observe que tal matriz pode ser transformada, através de permutação de suas colunas, em :

$$\bar{A} = \begin{bmatrix} I_k \\ R \end{bmatrix}$$

Assim, J induz uma base para o sistema. Podemos então observar que uma condição necessária para que exista uma solução viável melhor que a solução x é que:

$$\sum_{j \in Q} x_j \geq 1$$

onde $Q = \{j \mid C_B \bar{A}_j - c(j) > 0\}$ e C_B é o vetor de custos associado às variáveis da base induzida por J .

Ou seja, se $Q = \emptyset$ temos uma solução ótima. Tais observações nos levam ao algoritmo:

1. Tome qualquer solução inicial x e associe a ela sua cobertura não redundante J .
2. Se o conjunto Q descrito acima for vazio, podemos parar e J nos dará uma solução ótima. Caso contrário, inclua a restrição $\sum_{j \in Q} x(j) \geq 1$ no problema e volte a iterar o processo.

Aneja descreve em seu artigo os vários procedimentos para obtenção do conjunto Q , o procedimento de geração de linhas de forma mais detalhada e ainda uma heurística para acelerar a convergência.

Os resultados por ele obtidos demonstram que o algoritmo se torna pouco eficiente com o aumento do tamanho do problema: dos 30 problemas com 40 ou 50 vértices testados, apenas 14 terminaram em 1 minuto de processamento. Em contrapartida, várias das instâncias resolvidas, mesmo com a relaxação linear, o autor obteve solução inteira. O próprio autor qualifica seus resultados computacionais como pouco satisfatórios.

3.2 Métodos heurísticos

Como vimos no capítulo anterior, Karp em [31] provou que o PSG é $\mathcal{NP}$ -completo, mostrando de certa forma, que o mesmo tem uma intratabilidade inerente. Porém, como dizem Garey e Johnson [19], muitos destes problemas são interessantes e importantes demais para nos contentarmos com uma resposta do tipo: “Não sei resolver”.

Os algoritmos exatos que vimos na seção anterior, têm complexidade exponencial, o que inviabiliza a sua aplicação prática a problemas de grande porte. Neste contexto, muitas vezes é importante apresentar algoritmos rápidos que nos dêem respostas próximas da solução ótima, assim como garantias de quão próximas elas estão. Descrevemos a seguir, alguns métodos heurísticos para o PSG e discutimos em cada caso a proximidade da solução heurística com relação à solução ótima.

3.2.1 Heurística de caminhos mínimos

Takahashi e Matsuyama em [54] desenvolveram um algoritmo de complexidade $O(|Z| |V|^2)$ para o $PSG(G, Z)$ que garante um limitante para a razão entre o valor devolvido pelo algoritmo e a solução ótima.

A seguir descrevemos o algoritmo:

1. Escolha arbitrariamente um vértice $v \in Z$, e tome uma árvore contendo apenas o vértice v .
2. A cada iteração, acrescente um novo vértice obrigatório à árvore T escolhendo aquele ligado a T pelo menor caminho possível P . Então tome $T := T + P$.
3. Se $VT \supseteq Z$ pare e retorne T .

Para se determinar a cada passo do algoritmo o vértice mais próximo da árvore T podemos, no início, usar o algoritmo de Dijkstra [10] para cada vértice de Z e então manter uma tabela que vai se atualizando a cada iteração. Usando-se o algoritmo de caminho mais curto para cada vértice obrigatório temos complexidade total de $O(|Z| |V|^2)$ que é o termo dominante da complexidade do algoritmo.

É claro que tal heurística não leva sempre à solução ótima (observe a figura 3.9).

Passemos a analisar quão próxima a solução heurística pode estar da solução ótima. Para isso, definamos $OTIMO$ como o valor da solução ótima e $APROX$ o valor aproximado obtido pela heurística. A seguir provaremos que

$$\frac{APROX}{OTIMO} \leq 2 \left(1 - \frac{1}{|Z|} \right).$$

Para demonstrarmos este resultado vejamos antes algumas notações:

- T_i denota a árvore da i -ésima iteração do algoritmo;
- P_i denota o menor caminho entre um vértice obrigatório $v_i \in Z - VT_i$ e T_i ;
- $\hat{c}_i(T_{i-1}, v_i) = \min\{\text{custo}(P_i) : v_i \in Z - VT_{i-1}\}$.

Com isso, $APROX = \sum_{i=2}^{|Z|} \hat{c}_i(T_{i-1}, v_i)$.

Teorema 4 Para todo grafo $G = (VG, EG)$ e $Z \subset VG$, $|Z| > 2$,

$$\frac{APROX}{OTIMO} \leq 2 \left(1 - \frac{1}{|Z|} \right).$$

Figura 3.9

Dem: A cada iteração i do algoritmo, tendo T_{i-1} , o vértice v_i é escolhido em $Z - VT_{i-1}$ de tal forma que este seja o vértice obrigatório mais próximo à árvore T_{i-1} . Assim,

$$\hat{c}(T_{i-1}, v_i) \leq \text{dist}(v, w) \quad \forall v \in VT_{i-1}, \quad \forall w \in Z - VT_{i-1} \quad (*)$$

Observe que dada uma solução ótima, obtemos uma enumeração dos vértices obrigatórios do grafo ao percorrer essa árvore em pré-ordem (veja a figura 3.10).

Figura 3.10

É fácil ver que dada uma enumeração $v_1, v_2, \dots, v_{|Z|}$, o custo total do segmento da árvore que leva de v_i a v_j é igual a $\text{dist}(v_i, v_j)$ em G (caso contrário, obteríamos uma árvore de custo menor).

Assim, $\text{dist}(v_1, v_2) + \text{dist}(v_2, v_3) + \dots + \text{dist}(v_{|Z|-1}, v_{|Z|}) + \text{dist}(v_{|Z|}, v_1) = 2 \text{ OTIMO}$

Mas, pela observação (*) temos que

$$\begin{aligned}
APROX &= \sum_{i=2}^{|Z|} \hat{c}(T_{i-1}, v_i) \\
&\leq \sum_{i=2}^{|Z|} \text{dist}(v_{\rho(i)}, v_{\rho(i)+1}) \quad (**) \\
&= 2 \text{OTIMO} - \text{dist}(v_k, v_{k+1}) \\
&\leq 2 \text{OTIMO} - \max_{k=1,2,\dots,|Z|-1} \{\text{dist}(v_k, v_{k+1})\} \\
&\leq 2 \text{OTIMO} - \frac{2\text{OTIMO}}{|Z|} \\
&= 2 \text{OTIMO} \left(1 - \frac{1}{|Z|}\right).
\end{aligned}$$

No ponto (**) devemos observar que existe uma permutação conveniente dos vértices que garante a inequação. O autor cita como exemplo dessa técnica o trabalho de Rosenkrantz [48].

□

Cabe observar que esse limite é justo, no sentido de que existem instâncias do problema para as quais o algoritmo dá uma solução aproximada tal que

$$\frac{APROX}{OTIMO} = 2 \left(1 - \frac{1}{|Z|}\right).$$

Observe a figura 3.11

Neste caso, $OTIMO = |Z|$, mas o algoritmo pode devolver $APROX = 2(|Z| - 1)$.

Figura 3.11

3.2.2 Heurística de árvore geradora mínima

Uma outra heurística para o problema, com um limite melhor do que o alcançado pela heurística da seção anterior, foi obtida por Kou, Markowsky e Berman em [32].

A idéia dos autores pode ser descrita como vemos a seguir:

1. Considere inicialmente o grafo completo $G_1 = (VG_1, EG_1)$ com $VG_1 = Z$ e uma função $c_1 : EG_1 \rightarrow \mathbb{R}$ onde $c_1(uv) = \text{dist}_G(u, v)$, $\forall u, v \in VG_1$;
2. Encontre em G_1 uma árvore geradora T_1 de custo mínimo;
3. Construa um subgrafo T de G estendendo cada aresta uv de T_1 a um caminho mínimo de u a v em G ;
4. Tome uma árvore geradora mínima T^* de T (observe que T pode ter circuitos);
5. Elimine de T^* as folhas que não são vértices de Z .

Observe que o algoritmo é rápido, pois o passo (1) requer tempo $O(|Z| |VG|^2)$ [16], o passo (2) requer $O(|Z|^2)$ [33], o passo (3) requer $O(|Z|)$ (basta que utilizemos uma estrutura de dados conveniente para armazenar os caminhos mínimos no passo (1)), o passo (4) requer $O(|VG|^2)$ e finalmente o passo (5) requer tempo $O(|VG|)$. O passo (1), portanto, é o que domina a complexidade computacional fazendo com que o algoritmo seja $O(|Z| |VG|^2)$. Observe um exemplo da execução do algoritmo na figura 3.12.

Observe no exemplo da figura 3.12, que se no passo (2) tivéssemos escolhido outra árvore geradora mínima a heurística não levaria a uma solução ótima. Veja na figura 3.13 que já no passo (3) obtemos uma solução cujo valor é 18 apesar de o valor ótimo ser 16.

Vamos novamente definir *OTIMO* para o valor ótimo e *APROX* para o valor devolvido pelo algoritmo.

Teorema 5 Dado $G = (VG, EG)$, $c : EG \rightarrow \mathbb{R}$ e $Z \subset VG$, $|Z| > 2$,

$$\frac{APROX}{OTIMO} \leq 2 \left(1 - \frac{1}{t}\right),$$

onde t é o número de folhas de uma árvore correspondente a uma solução ótima.

Dem: Seja T uma solução ótima para $PSG(G, Z)$ com t folhas. Como na demonstração do teorema anterior, observamos que podemos construir em T um passeio fechado C tal que cada folha só aparece uma vez e cada aresta de T é visitada duas vezes. Basta, para isso, tomar uma folha como raiz da árvore e enumerar suas folhas na ordem em que elas aparecem em pré-ordem. Observe a figura 3.14.

Com uma tal enumeração tome C o passeio fechado formado em T ,

$$P(f_1, f_2)P(f_2, f_3) \dots P(f_{t-1}, f_t)P(f_t, f_1),$$

onde $f_1, f_2, \dots, f_t$ são as folhas de T e $P(f_i, f_j)$ é o caminho que liga f_i a f_j em T .

Eliminando de C um caminho entre duas folhas que tem custo máximo, obtemos um passeio C' tal que

$$\begin{aligned} \text{custo}(C') &= \text{custo}(C) - \max(\text{custo}\{P(f_i, f_{i+1})\}) \\ &\leq \text{custo}(C) - \frac{\text{custo}(C)}{t} \\ &\leq \text{custo}(C) \left(1 - \frac{1}{t}\right). \end{aligned}$$

Mas, pela construção do algoritmo,

$$\text{custo}(C') \geq \text{custo}(\text{árvore geradora mínima de } G_1) \geq \text{APROX}.$$

Portanto,

$$\begin{aligned} \text{APROX} &\leq \text{custo}(C) \left(1 - \frac{1}{t}\right) \\ &\leq 2\text{OTIMO} \left(1 - \frac{1}{t}\right). \end{aligned}$$

□

Neste caso também, o limite é o melhor possível. Observe que para a classe de grafos da figura 3.15 temos

$$\frac{\text{APROX}}{\text{OTIMO}} = 2 \left(1 - \frac{1}{t}\right).$$

3.2.3 Heurística de contração

No artigo de Winter [57] estão descritas algumas outras heurísticas que são pequenas variações das descritas nos itens anteriores. A que veremos a seguir, no entanto, devida a Plesnik [46], trabalha com árvores geradoras mínimas e contrações de grafos.

Antes de descrevermos a idéia do método vejamos algumas definições:

Dado $v \in Z$, $vw \in EG$ e $r > 0$ dizemos que o vértice w está na **vizinhança de v de raio r** se $c(vw) \leq r$. Se $c(uw) > r$, definimos um ponto novo na aresta uw

Figura 3.15

e também dizemos que está na vizinhança de v de raio r . Denotaremos este conjunto de vértices e pontos nas arestas por $N_v(r)$. No seu artigo Plesnik não distingue se uma vizinhança “ultrapassa” ou não um vértice facultativo. Winter, em sua resenha toma esta restrição visando, nos parece, ir de encontro com as características do problema. Observe na figura 3.16 os pontos novos que surgiram no grafo.

Observe os pontos novos que surgiram no grafo para $N_v(3)$.

Figura 3.16

Duas vizinhanças $N_v(r_1)$ e $N_w(r_2)$ são **adjacentes** se tiverem algum vértice em comum ou $c(vw) = r_1 + r_2$ e nesse caso diremos que estas têm o mesmo ponto na aresta.

Uma **classe de vizinhanças** $\mathcal{C}_r$ para o problema é definida como o conjunto de todas as vizinhanças de vértices de Z , sendo que duas vizinhanças adjacentes são transformadas em uma só. Observe na figura 3.17 a classe de vizinhança $\mathcal{C}_3$.

Dada uma classe de vizinhanças $\mathcal{C}_r$ podemos obter um grafo contraído G' a partir de G seguindo os seguintes passos:

1. O conjunto de vértices VG' é obtido contraindo-se cada vizinhança em $\mathcal{C}_r$ a um

único vértice que é colocado também em Z' . Os vértices facultativos que não estão contidos em nenhuma vizinhança de C_r são também inseridos em VG' .

2. As arestas de G' e seus custos são dados pelas regras a seguir. Dada uma aresta uv de G , considere:
- (a) se u e v não estão em nenhuma vizinhança de C_r , a aresta uv é inserida em EG' com o mesmo custo que tinha em G .
 - (b) se u está em alguma vizinhança e v não está, a aresta é incluída e

$$c'(uv) = \begin{cases} c(uv) & u \notin Z \\ c(uv) - r & u \in Z \end{cases}$$

- (c) se u e v estão em vizinhanças de $\mathcal{C}_r$, a aresta é incluída e

$$c'(uv) = \begin{cases} c(uv) & u \notin Z, v \notin Z \\ c(uv) - r & u \in Z, v \notin Z \text{ ou } u \notin Z, v \in Z \\ c(uv) - 2r & u \in Z, v \in Z \end{cases}$$

- (d) Elimina-se do grafo todas os laços. Das arestas múltiplas podemos deixar apenas a de menor custo.

Observe o exemplo na figura 3.18.

O método de Plesnik está descrito a seguir:

1. Determine a aresta de menor custo do grafo que incide em um vértice obrigatório. Seja r seu custo. Construa $\mathcal{C}_r = (C_1, C_2, \dots, C_k)$. Construa árvores geradoras mínimas para cada $Z \cap C_i$. Observe que cada uma dessas árvores têm custo menor ou igual a $2r(|Z \cap C_i| - 1)$.
2. Construa a partir de G e $\mathcal{C}_r$ o grafo G' contraindo-se $\mathcal{C}_r$.

Figura 3.18

3. Se G' tiver apenas um vértice pare. Caso contrário, obtenha T' aplicando o mesmo processo recursivamente.
4. Obtenha de T' uma árvore para G , expandindo os vértices que representam vizinhanças das suas árvores geradoras construídas no passo 1. A expansão poderá adicionar ao custo da árvore um valor não superior a $2r(|Z| - 1)$.

Observe um exemplo da execução do algoritmo na figura 3.19.

O algoritmo tem complexidade $O(|VG|^3)$ desde que nenhum dos passos descritos exige tempo maior que $O(|VG|^2)$ e a recursão é executada no pior caso $O(|VG|)$ vezes (pois em cada iteração pelo menos um vértice será englobado em alguma vizinhança).

Para estudarmos o comportamento da solução obtida pelo algoritmo devemos fazer algumas observações:

Lema 1 *Seja T' a árvore obtida recursivamente. O valor da solução T obtida pelo algoritmo é tal que*

$$\text{custo}(T) \leq \text{custo}(T') + 2r(|Z| - 1)$$

□

Figura 3.19

Para verificarmos o lema basta observar que se o custo de uma aresta que liga vértices obrigatórios em T' é k , para obter a árvore T , o custo da aresta correspondente é, no máximo, $k + 2r$.

Lema 2 *Seja T^* uma solução ótima de $PSG(G, Z)$ e T'^* uma solução ótima de $PSG(G', Z')$. Então*

$$custo(T^*) \geq custo(T'^*) + (|Z| - 1)r,$$

e se $|Z'| \geq 2$

$$custo(T^*) \geq custo(T'^*) + |Z| r.$$

□

A demonstração segue por indução no número de elementos de $\mathcal{C}_r$.

Utilizando estes dois lemas obtemos um limite para o valor da solução obtida através desta heurística. Antes definamos como T_t a solução fornecida pela heurística na iteração t e T_t^* a solução ótima do problema desta iteração.

Teorema 6 *Seja $APROX$ o valor da solução obtida pelo algoritmo e $OTIMO$ o valor de uma solução ótima. Então*

$$\frac{APROX}{OTIMO} \leq 2.$$

Dem. Usando os Lemas $t - 1$ vezes temos:

$$APROX \leq custo(T_t) + 2 \sum_{i=1}^{t-1} r_i (|Z_i| - 1)$$

e

$$OTIMO \geq custo(T_t^*) + \sum_{i=1}^{t-1} r_i (|Z_i| - 1).$$

Daí

$$custo(T_t^*) + \sum_{i=1}^{t-1} r_i (|Z_i| - 1) \leq OTIMO \leq APROX \leq custo(T_t) + 2 \sum_{i=1}^{t-1} r_i (|Z_i| - 1).$$

Observando que $custo(T_t) \leq 2 custo(T_t^*)$, temos

$$OTIMO \leq APROX \leq 2 OTIMO.$$

□

3.3 Reduções no tamanho das instâncias

Os métodos de redução que estudaremos nesta seção foram desenvolvidos por Balakrishnan e Patel [2] e revistos no trabalho de Duin e Volgenant [13] e podem ser entendidos como procedimentos para

- determinar se algumas arestas e vértices devem ou não pertencer a uma solução ótima (o que permite que se fixem os valores de algumas variáveis) ou
- identificar outras restrições que uma solução ótima deve satisfazer.

Na maioria das aplicações, as instâncias o *Problema de Steiner em Grafos* são relativamente grandes. As reduções aqui descritas são mais efetivas quando aplicadas a grafos com muitas arestas, porém não se aplicam apenas a estes. O *PSG* possui aplicações interessantes onde o grafo em questão é bastante denso ou mesmo completo. Por exemplo, Foulds e Grahman em [17] descrevem uma aplicação prática do problema em um grafo completo. Em nossa abordagem poliédrica, também discutiremos o caso do *PSG* aplicado a grafos completos.

Algumas reduções são derivadas de conclusões óbvias que se tiram ao observar-se apenas propriedades locais do grafo:

1. Se G possui um vértice facultativo de grau 1 podemos remover tal vértice do grafo (lembrando que todas as arestas têm custos positivos);
2. Se G tem um vértice facultativo v de grau 2, podemos remover v do grafo desde que acrescentemos uma nova aresta ligando seus vizinhos. O custo desta aresta deve ser igual à soma dos custos das arestas que incidiam em v ;
3. Se G tem um vértice obrigatório v de grau 1 podemos removê-lo do grafo desde que acrescentemos o vértice adjacente a v ao conjunto de vértices obrigatórios Z ;
4. Se G possui uma aresta xy tal que $c(xy) > \text{dist}(x, y)$, esta aresta pode ser removida do grafo. Se $c(xy) = \text{dist}(x, y)$ e existe um caminho de x a y que não utiliza a aresta xy , também podemos eliminá-la do grafo.

As próximas reduções, apesar de ainda simples, já não são tão imediatas. Justificamos tais reduções com pequenas provas.

A partir de agora, e no decorrer de toda esta seção, vamos supor que quaisquer arestas do grafo têm custos dois a dois distintos. Isso pode ser conseguido através de uma perturbação da função custo. Tal perturbação, desde que suficientemente pequena não atrapalha a busca de uma solução ótima.

1. Se v é um vértice obrigatório de G , u é o vértice mais próximo de v , w é o segundo vértice mais próximo de v e

$$c(uv) + \min_{x \in Z-u} \{dist(u, x)\} \leq c(vw),$$

então a aresta uv está na solução ótima e podemos contraí-la no grafo G .

Dem: Vamos supor por absurdo que a aresta uv não esteja em nenhuma solução ótima. Seja $p \neq u$ o vértice de Z para o qual $dist(u, p)$ em G é mínima e seja P um caminho mínimo em G de u a p .

Tome uma solução ótima T , considere o caminho que liga v a p em T e seja α a aresta deste caminho que incide em v . $H := T + P - \alpha + uv$ é conexo (veja figura 3.20).

Figura 3.20

Além disso,

$$\sum_{e \in H} c(e) = \sum_{e \in T} c(e) - c(\alpha) + \sum_{e \in P} c(e) + c(uv)$$

Mas,

$$c(\alpha) \geq c(vw) \geq c(uv) + dist(u, v),$$

e portanto,

$$\sum_{e \in H} c(e) \leq \sum_{e \in T} c(e).$$

Logo, H é solução ótima e uv está nesta solução, uma contradição.

□

2. De uma forma mais geral (cf [13]) podemos observar que:

Dado um grafo G e uma aresta $e = xy$, se existem $u, v \in Z$ tais que

$$dist(u, v) = dist(u, x) + c(e) + dist(y, v)$$

e todo caminho entre u e v em $G - e$ contém uma aresta f tal que

$$c(f) \geq dist(u, v),$$

então e está na solução ótima e pode ser contraída.

Dem: Seja T a solução ótima do problema. Suponhamos por absurdo que e não está em T . Como u e v são vértices obrigatórios, deve haver em T um caminho que liga u a v e que não passa por e . Pela nossa hipótese, este caminho contém uma aresta f tal que

$$c(f) \geq \text{dist}(u, v).$$

Tome $T' := T + P - f$ onde P é um caminho mínimo que liga u a v e contém e . Observe que T' é uma solução ótima e contém e (contra nossa hipótese).

□

3. Se G contém 2 vértices obrigatórios adjacentes u e v e existe vértice obrigatório w , $u \neq w$ e $v \neq w$ tal que

$$c(uv) \geq \text{dist}(u, w) \quad \text{e} \quad c(uv) \geq \text{dist}(v, w),$$

então uv pode ser removida do grafo.

Dem: Suponhamos por absurdo que toda solução ótima contenha uv .

Seja T uma solução ótima e seja $T' := T - uv$.

Temos, então, dois casos:

- **Caso 1:** u e v estão na mesma componente de T' .

Tome C_1 um caminho mínimo de u a w e $T'' = T' + C_1$.

T'' é conexo e ainda :

$$\begin{aligned} \sum_{e \in T''} c(e) &= \sum_{e \in T} c(e) - c(uv) + \sum_{e \in C_1} c(e) \\ &= \sum_{e \in T} c(e) - c(uv) + \text{dist}(u, w) \\ &\leq \sum_{e \in T} c(e). \end{aligned}$$

Então T'' é solução ótima e $uv \notin T''$ (Contrariando nossa hipótese).

- **Caso 2:** Análogo ao Caso 1 com v e w na mesma componente.

□

As reduções apresentadas até agora dependem ou da forma do grafo de entrada (como as quatro primeiras que vimos) ou da função custo (como as últimas três) para serem realmente eficientes. É fácil obter exemplos em que nenhuma dessas reduções podem ser utilizadas (veja a figura 3.21).

A seguir apresentamos reduções mais efetivas, que para grafos densos, independem da função custo para diminuir o tamanho da instância tratada.

Figura 3.21

Antes, definamos $T_g(G)$ como a (única) floresta geradora mínima de G que é a união das árvores geradoras mínimas de cada componente de G ; e $T_{ST}(G, Z)$ como uma solução ótima de $PSG(G, Z)$. Quando não houver ambigüidades, quanto a que grafo aplicamos os problemas, usaremos as notações T_g e T_{ST} .

Proposição 2 *Sejam $u, v \in VT_{ST}$. Se a aresta uv está em T_g , então esta aresta deve estar em T_{ST} também.*

Dem: Suponha que a aresta uv esteja em T_g e não esteja em T_{ST} . Então, $T_g - uv$ tem pelo menos duas componentes C_1 e C_2 , onde $u \in C_1$ e $v \in C_2$. Seja $X := [C_1, C_2]$. Como u e v estão em T_{ST} , alguma aresta e de T_{ST} está em X . Mas, uv está em T_g , e, portanto, deve ser a aresta de X com o menor custo.

Assim, $T_{ST} - e + uv$ é conexo e tem custo menor que T_{ST} , o que é um absurdo.

□

Observe que podemos utilizar este resultado das seguintes maneiras:

- (a) Pela Proposição 2, toda aresta de T_g que liga dois vértices obrigatórios deve estar em T_{ST} . Estas arestas podem ser contraídas. Aplicando esta redução várias vezes, obtemos uma rede em que todo vértice obrigatório tem como vizinho mais próximo um vértice facultativo.
- (b) Ainda pela Proposição 2, toda aresta de T_g que liga um vértice obrigatório a um facultativo estará em toda solução ótima em que o vértice facultativo estiver. Podemos formular esta restrição através da seguinte inequação:

Se a aresta uv está em T_g , $v \notin Z$, $u \in Z$,

$$x_{uv} \geq x_{vw}, \forall w \in VG.$$

A Proposição 2 trata de arestas que uma solução ótima deve necessariamente conter. Na próxima proposição veremos resultados que identificam arestas que não podem estar em nenhuma solução ótima.

Proposição 3 *Toda aresta em T_{ST} que incide em dois vértices obrigatórios deve estar necessariamente em $T_g(G[Z])$.*

Dem: Seja e uma aresta de T_{ST} com ambas as pontas em Z . Suponhamos, por absurdo, que e não esteja em $T_g(G[Z])$. Considere as duas componentes resultantes ao eliminar e de T_{ST} . Então deve existir alguma aresta $e' \neq e$ com uma ponta em cada componente. Tome

$$T' := T_{ST} - e + e'.$$

Como e não pertence a $T_g(G[Z])$, então $c(e') < c(e)$, e portanto, obtivemos um grafo conexo T' que contém o conjunto Z e

$$\text{custo}(T') < \text{custo}(T_{ST}).$$

□

A Proposição 3 pode ser lida da seguinte forma:

Tome o grafo inicial G e ache $T_g(G[Z])$. Apenas as arestas de $T_g(G[Z])$ devem ser consideradas.

Ou seja, se o grafo for completo, podemos eliminar $\frac{(|Z|-1)(|Z|-2)}{2}$ arestas do grafo.

O mesmo argumento pode ser usado para mostrar as seguintes proposições:

Proposição 4 *Uma aresta e que incide em um vértice obrigatório e um vértice facultativo v está na solução ótima T_{ST} só se está em $T_g(G[Z \cup \{v\}])$.*

Proposição 5 *Uma aresta e que incide em dois vértices facultativos v e w está na solução ótima T_{ST} só se e está em $T_g(G[Z \cup \{v, w\}])$.*

A Proposição 4 nos permite eliminar, para cada vértice facultativo, $|Z| - 1$ arestas, reduzindo, então o tamanho da instância de $(|VG| - |Z|)(|Z| - 1)$ arestas.

Porém, para poder eliminar tais arestas é necessário determinar $T_g(G[Z \cup \{v\}])$ para cada vértice $v \in VG - Z$, ou seja, devemos executar este procedimento $|VG - Z|$ vezes.

A Proposição 5 exige também a determinação de $T_g(G[Z \cup \{v, w\}])$ e nos permite eliminar no máximo 1 aresta. Ou seja, para todos os pares de vértices facultativos devemos

executar o algoritmo para determinação das correspondentes T_g ($|VG - Z|$ ($|VG - Z| - 1$) vezes) para diminuir o tamanho do problema de, no máximo, uma aresta de cada vez.

Observe que os procedimentos de redução podem ser executados várias vezes, e quando se reduz um grafo podem ser geradas oportunidades para se aplicar outras reduções.

Balakrishnan e Patel em [2] obtiveram em testes com grafos relativamente esparsos percentuais médios de redução de até 43,5 % no número de arestas desses grafos. Eles observam que as reduções das Proposições 1 a 4 tendem a ser mais efetivas quanto mais denso for o grafo.

3.4 Casos polinomiais

Como vimos na redução do capítulo anterior, o *Problema de Steiner em Grafos* é $\mathcal{NP}$ -completo até mesmo para grafos planares e bipartidos. Porém, é claro que existem classes de grafos para os quais o problema se torna fácil: Se G é uma árvore, é fácil obter uma solução ótima em tempo linear em $|VG - Z|$. O diagrama da figura 3.22 será completada neste capítulo com os resultados que veremos.

O objetivo dessa seção é apresentar alguns dos resultados da literatura sobre classes de grafos para as quais são conhecidos algoritmos polinomiais.

Nesta seção, ao contrário das outras deste capítulo, nossa atenção estará voltada para as classes de grafos e não para o método.

3.4.1 2-Árvores

Wald e Coulbourn [55] investigaram o *PSG* aplicado a redes IFI (Isolated Failure Immune), que haviam sido introduzidas por Farley [15] em 1981.

Farley introduziu uma noção de segurança de transmissão em redes que leva em conta conexidade de vértice e conexidade de aresta. O objetivo é obter uma maneira

econômica de se projetar redes, de forma que a queda de algumas linhas, ou mesmo alguns pontos de retransmissão não inviabilizasse a comunicação entre quaisquer pontos da rede. Vejamos algumas definições:

- Uma *falha de vértice é isolada* se não há outro vértice adjacente com falha.
- De forma análoga, uma *falha de aresta é isolada* se em nenhuma de suas pontas incide uma aresta com falha e em nenhuma de suas pontas está um vértice com falha.
- Uma rede é *imune a falhas isoladas (IFI)* se mesmo existindo falhas isoladas nesta rede, ainda existe caminho entre quaisquer dois vértices da rede.

Farley mostrou que redes IFI com n vértices têm pelo menos $2n - 3$ arestas. Vamos mostrar a seguir que as menores redes IFI são as 2-árvores.

Uma 2-árvore é definida recursivamente da seguinte forma:

1. um triângulo é uma 2-árvore;
2. se G é uma 2-árvore, e xy é uma aresta de G , então o grafo obtido de G acrescentando-se um novo vértice z e as arestas xz e yz é uma 2-árvore.

É fácil ver que se G é uma 2-árvore, então $|EG| = 2|VG| - 3$. Outra propriedade das 2-árvores é que elas podem ser definidas como grafos que não contêm K_4 como subgrafo homeomorfo.

A seguir demostramos que as 2-árvores são IFI mínimas.

Teorema 7 *Uma rede IFI mínima é uma 2-árvore.*

Dem: Suponhamos que $G = (VG, EG)$ é uma IFI mínima que não é uma 2-árvore. Tomemos G com o menor número possível de vértices. Então, G tem $2|VG| - 3$ arestas e além disso não tem vértices de grau 0 ou 1 (caso contrário não seria IFI).

Assim, G tem pelo menos 1 vértice de grau 2 ou um vértice de grau 3, pois senão

$$|EG| = \frac{1}{2} \sum_{v \in VG} grau(v) \geq 2|VG|.$$

Caso 1 G tem um vértice x de grau 2.

Figura 3.23

Sejam y e z vértices adjacentes a x . (Veja a figura 3.23). Se y e z não forem adjacentes, e suas falhas são isoladas em G , então x ficaria desconectado do resto do grafo. Assim, y e z são adjacentes.

Mas, $G - x$ é uma rede IFI com menos vértices que G . Assim, concluímos que G não pode ter vértices de grau 2.

Caso 2 G tem um vértice x de grau 3.

Inicialmente, observe que podemos tomar, sem perda de generalidade, os vizinhos y, z e w de x de tal forma que y seja adjacente a z , z adjacente a w e y não adjacente a w . Veja a figura 3.24.

Figura 3.24

Nesse caso, $G - z + yw$ é uma rede IFI e tem menos vértices que G . Assim, concluímos que G deve ser uma 2-árvore.

□

Em seu trabalho, Farley estudou algoritmos para transformar uma rede qualquer em uma rede IFI. A idéia do algoritmo [15] é juntar arestas à rede até que a rede resultante seja uma 2-árvore.

Wald e Coulbourn em [55] sugerem um algoritmo linear no número de vértices do grafo para achar uma solução do PSG para o caso de 2-árvores. O algoritmo é baseado no princípio de programação dinâmica e aproveita-se da estrutura recursiva das 2-árvores. A seguir, descrevemos rapidamente o algoritmo. Para isso são necessárias algumas definições:

Dada uma 2-árvore G e um conjunto $Z \subset VG$, a cada aresta $\alpha = xy$ associamos 6 medidas:

- $st(\alpha) = c(\alpha)$

- $dt(\alpha) = 0$
- $yn(\alpha) = \begin{cases} \infty & \text{se } y \in Z \\ 0 & \text{c.c.} \end{cases}$
- $ny(\alpha) = \begin{cases} \infty & \text{se } x \in Z \\ 0 & \text{c.c.} \end{cases}$
- $nn(\alpha) = \begin{cases} \infty & \text{se } y \in Z \text{ ou } x \in Z \\ 0 & \text{c.c.} \end{cases}$
- $none(\alpha) = \begin{cases} \infty & \text{se } y \in Z \text{ ou } x \in Z \\ 0 & \text{c.c.} \end{cases}$

A idéia do algoritmo é aproveitar a característica recursiva das 2-árvores para simular a reconstituição de sua formação. Assim, a cada iteração do algoritmo eliminaremos um vértice v de grau 2 e as duas arestas incidentes a ele, armazenando as informações referentes a estas na aresta que liga os dois vizinhos de v . Assim as medidas acima são recalculadas a cada passo do algoritmo em que um vértice é eliminado. Observe que em cada passo o número de vértices do grafo diminui de 1 e o número de arestas de 2. Observe também que em cada passo o grafo com que estamos lidando é uma 2-árvore.

O algoritmo descrito acima tem, claramente, complexidade linear no número de vértices da 2-árvore. Cabe ainda observar que em cada passo será necessário termos um vértice de grau 2 no grafo. Mas, isso é óbvio devido a construção das 2-árvores.

A seguir damos uma interpretação intuitiva para as medidas que vão resumir as informações sobre o conjunto de vértices S que já foram eliminados e estão sendo representados pela aresta $\alpha = xy$:

- $st(\alpha)$ significa o menor custo para obter uma solução para $PSG(G[S], Z)$ na qual x e y fiquem na mesma árvore;
- $dt(\alpha)$ significa o menor custo em se obter duas árvores disjuntas para $G[S]$ incluindo todos os vértices de Z : uma com x e a outra com y ;
- $yn(\alpha)$ é o mínimo custo para se construir uma solução para $PSG(G[S], Z)$ que inclua x mas não y ;
- $ny(\alpha)$ é o mínimo custo para se construir uma solução para $PSG(G[S], Z)$ que inclua y mas não x ;
- $nn(\alpha)$ é o menor custo de uma solução para $PSG(G[S], Z)$ que não inclua nem x nem y ;
- $none(\alpha)$ é o menor custo de uma solução para $PSG(G[S], Z)$ que não contenha nenhum vértice de S .

Um exemplo de uma das recorrências que são usadas para recalcular estas medidas quando um vértice é eliminado é dado a seguir:

Exemplo: Dado um vértice v de grau 2 com os vizinhos u e w , para eliminarmos v do grafo o novo valor de $st(uw)$ deverá ser :

$$st(uw) = \min\{ st(uw) + \min\{dt(uv) + st(vw), st(uv) + dt(vw), yn(uv) + ny(vw)\}, dt(uw) + st(uv) + st(vw)\}.$$

As recorrências são todas deste mesmo tipo e, portanto, levam tempo constante para serem executadas. Com isso, podemos concluir que o algoritmo realmente tem complexidade linear no número de vértices do grafo.

Este resultado é uma generalização (dos autores) de um resultado sobre grafos “outerplanar” (um grafo é “outerplanar” se tem uma representação (*embedding*) no plano na qual todos os vértices estão na face externa). Observe na figura 3.25 como estão relacionadas as classes de grafos até aqui consideradas. No diagrama vamos considerar a classe das **2-árvores parciais** que são aqueles grafos que podem ser estendidos a 2-árvores acrescentando-se apenas arestas.

Figura 3.25

3.4.2 Redes de Halin

Outra classe importante de grafos para a qual o *PSG* pode ser resolvido em tempo polinomial é a classe das redes de Halin. As redes de Halin foram introduzidas em [29] por R. Halin como exemplos de grafos 3-conexos planares em que a remoção de qualquer aresta acarreta a redução da conexidade da rede. O algoritmo linear para o problema aplicado a redes de Halin foi desenvolvido por Winter em [58].

Para construirmos uma rede de Halin consideramos uma representação no plano de uma árvore sem vértices de grau 2 e depois ligamos as folhas consecutivas na representação tomada, como no exemplo da figura 3.26.

Figura 3.26

É fácil, também, verificar se um grafo dado é ou não uma rede de Halin. Basta tomar uma sua representação no plano (mostrou-se que os grafos 3-conexos tem uma única representação planar) e remover as arestas da face exterior. O grafo inicial é uma rede de Halin se o grafo resultante é uma árvore sem vértices de grau 2. Os vértices da face externa de uma rede de Halin são chamados *folhas*.

No algoritmo que vimos para as 2-árvores os vértices de grau 2 capturavam a essência da estrutura da rede, e, num certo sentido, serviam para resumir sua formação. Nas redes de Halin, a característica recursiva é capturada pelo que chamamos de *leques*.

Um **leque** é um conjunto de vértices formado por um vértice não folha v e as folhas adjacentes a este com a restrição de que deve existir no grafo algum outro vértice não folha adjacente a v . Observe o exemplo da figura 3.27.

Figura 3.27

O vértice não folha de um leque é chamado de **centro**. No algoritmo que veremos a seguir, observaremos que os centros resumirão as informações sobre os leques que eles formam. No caso em que a rede é uma roda, ou seja, existir apenas um vértice não folha na rede (conforme a figura 3.28), o leque será composto por todas as folhas exceto uma arbitrariamente escolhida.

Figura 3.28

A primeira observação interessante é que um leque pode ser contraído a um só vértice e o grafo resultante continua sendo uma rede de Halin. Veja a figura 3.29.

Figura 3.29

O algoritmo é construído baseado na idéia de contração de leques. São introduzidas 15 medidas associadas aos vértices do grafo e estas são recalculadas a cada iteração em que um leque é escolhido e contraído a um vértice. As fórmulas para cada recorrência podem ser calculadas em tempo constante conforme vemos em [58].

As fórmulas das recorrências são complicadas demais para um resumo tão simples como este.

Para incluir esta nova classe na figura 3.25, devemos observar que as redes de Halin não são 2-árvores (nem 2-árvores parciais) (K_4 é uma rede de Halin) e existem 2-árvores que não são redes de Halin (o triângulo não é uma rede de Halin). Assim, obtemos o diagrama da figura 3.30.

3.4.3 Grafos Fortemente Cordais

White, Farber e Pulleyblank apresentam em [56] um algoritmo de complexidade $O(|VG|^3)$ para o caso particular em que o grafo é fortemente cordal e as arestas têm custo unitário.

Figura 3.30

Vale a pena lembrar que o *PSG* continua $\mathcal{NP}$ -completo mesmo quando todas as arestas têm custo unitário.

Um grafo é *cordal* se todo circuito com mais de 4 arestas tem uma corda. Um grafo é *fortemente cordal* se for cordal e todo circuito de tamanho par com mais de 6 arestas tem uma corda ímpar, ou seja, uma corda que separa o circuito em dois circuitos ímpares. Veja um exemplo de um grafo fortemente cordal na figura 3.31.

Figura 3.31

Os autores observam que se os custos das arestas forem inteiros é fácil obter uma rede em que as arestas têm custo unitário. Basta subdividir cada aresta e em $c(e)$ arestas de peso 1.

Inicialmente, definamos uma *vizinhança* N_v de um vértice v como o conjunto de todos os vértices adjacentes a v . Um vértice v é *simples* se para cada par de vértices u e w na vizinhança de v , $N_u \subseteq N_w$ ou $N_w \subseteq N_u$.

Uma enumeração dos vértices do grafo $(v_1, v_2, \dots, v_n)$ é *simples* se para cada $j = 1, 2, \dots, n$ o vértice v_j é simples no subgrafo induzido por $\{v_j, v_{j+1}, \dots, v_n\}$ em G .

Farber mostrou em [14] que um grafo é fortemente cordal se e somente se ele tiver uma enumeração simples dos seus vértices. No mesmo artigo o autor mostra um algoritmo

$O(|VG|^3)$ para construir tal enumeração.

Dada uma tal enumeração, podemos construir uma solução ótima T em tempo linear no número de vértices do grafo da seguinte forma:

1. Inicialmente tome $T = \phi$ e $i = 1$;
2. Examine o vértice v_i da enumeração simples dos vértices do grafo,
 - (a) se v_i é facultativo, então este pode ser eliminado;
 - (b) se v_i é obrigatório, então
 - i. se v_i é adjacente a outro vértice obrigatório, inclua a aresta que o une a tal vértice na solução T ;
 - ii. caso contrário, inclua em T a aresta que liga v_i ao vértice facultativo de maior grau.

Quando analisamos todos os vértices da enumeração, o conjunto T é uma solução ótima para o problema.

Observe que o custo unitário permite que a abordagem de adjacência seja aplicada ao problema. Tal abordagem não é aplicável quando permitimos custos quaisquer. Winter [57] observa em sua resenha que o PSG para grafos fortemente cordais com custos inteiros nas arestas é $\mathcal{NP}$ -completo.

Acrescentando esta nova classe de grafos no nosso diagrama obtemos o diagrama da figura 3.32. Note que toda 2-árvore é um grafo fortemente cordal, porém nem toda rede de Halin é fortemente cordal.

Figura 3.32

Capítulo 4

Abordagem Poliédrica

Na resenha do capítulo anterior mencionamos diversas abordagens que têm sido aplicadas ao *Problema de Steiner em Grafos*: métodos exatos de busca exaustiva, métodos heurísticos e reduções. Neste capítulo apresentaremos uma nova abordagem que faz uso de ferramentas de teoria de poliedros.

A abordagem poliédrica que utilizamos para estudar o $PSG(G, Z)$ é uma técnica que se aplica a muitos problemas de otimização combinatória, em especial aos $\mathcal{NP}$ -completos.

Como já mencionamos na introdução deste trabalho, numa forma bem geral, um *problema de otimização combinatória* pode ser formulado assim:

Dado um conjunto $\mathcal{S}$ (de soluções viáveis) e uma função custo $c : \mathcal{S} \rightarrow \mathbb{R}$ que associa a cada elemento de $\mathcal{S}$ um valor real, encontrar um elemento $S \in \mathcal{S}$ tal que $c(S)$ é mínimo.

Em geral, $\mathcal{S}$ é dado implicitamente. Por exemplo, uma especificação possível seria: “dado um grafo $G = (VG, EG)$, um subconjunto $Z \subseteq VG$ e uma função $c : EG \rightarrow \mathbb{R}$, seja

$$\mathcal{S} := \{A \subseteq EG : G[A] \text{ é conexo e } VG[A] \supseteq Z\}."$$

Note que se definirmos $c(A) := \sum_{e \in A} c(e)$, para os elementos de $\mathcal{S}$ exemplificado, o problema de encontrar $S \in \mathcal{S}$ tal que $c(S)$ é mínimo é exatamente o $PSG(G, Z)$.

O $PSG(G, Z)$ pertence a uma classe especial de problemas de otimização combinatória que são da seguinte forma:

Dado um conjunto básico E , finito, uma função $c : E \rightarrow \mathbb{R}$ e um conjunto $\mathcal{S}$ (de soluções viáveis) onde cada elemento é um subconjunto de E , encontre $S \in \mathcal{S}$ tal que $c(S)$ seja mínimo.

Problemas desse tipo costumam ser chamados de *problemas de otimização combinatória linear*. Outros exemplos de tais problemas são: o problema do caixeiro viajante, o problema do emparelhamento máximo, o problema do corte mínimo, etc.

Para utilizarmos uma abordagem poliédrica para um problema desse tipo, podemos aplicar o seguinte procedimento canônico: Primeiramente associamos ao problema um poliedro P_S cujos vértices estejam em correspondência um a um com as suas soluções viáveis (ie, os elementos de $\mathcal{S}$). Para isso, definimos para cada $S \in \mathcal{S}$ o seu vetor característico $x^S = (x_e^S)_{e \in E} \in \{0, 1\}^E$, onde $x_e = 1$ indica que $e \in S$ e $x_e = 0$ indica que $e \notin S$. Feito isso, definimos

$$P_S := \text{conv}\{x^S \in \{0, 1\}^E : S \in \mathcal{S}\}.$$

Note que P_S é um politopo e seus vértices são precisamente os vetores característicos das soluções viáveis em $\mathcal{S}$. Dessa forma, o problema original fica reduzido ao seguinte problema de programação linear:

$$\begin{aligned} \min \quad & cx \\ \text{s.t.} \quad & x \in P_S. \end{aligned}$$

Mas, essa transformação só é útil do ponto de vista algorítmico se conhecermos um sistema linear que descreve P_S . Como, então achar um tal sistema? Apesar de sua existência estar garantida, não sabemos encontrá-lo eficientemente. Além disso, segundo Papadimitriou [45], uma tal descrição completa e “boa” não existe para poliedros associados a problemas $\mathcal{NP}$ -completos, a menos que $\mathcal{P} = \mathcal{NP}$. Nesses casos, procuramos obter uma melhor descrição possível do poliedro e utilizamos essa informação para desenvolver um algoritmo que combina técnicas de programação linear com planos-de-corte.

O procedimento que descrevemos para associar um poliedro a um dado problema não é único. Pode-se associar poliedros cujos vértices não sejam vetores de entradas em $\{0, 1\}$, ou não estejam necessariamente em correspondência um a um com as soluções viáveis do problema.

Qualquer que seja o poliedro associado, para se obter uma melhor descrição possível, procuramos encontrar inequações que definem facetas do poliedro. Na primeira seção deste capítulo discutiremos alguns poliedros que poderíamos associar ao $PSG(G, Z)$. Nas seções seguintes apresentaremos algumas classes de facetas desses poliedros. Na última seção discutiremos a implementação do algoritmo que desenvolvemos e os resultados computacionais que foram obtidos.

4.1 Escolha do Politopo

Para investigarmos um problema sob uma abordagem poliédrica, precisamos decidir que politopo vamos associar ao problema. No caso de $PSG(G, Z)$, uma possibilidade seria associar um dos politopos que descrevemos a seguir. Antes vejamos algumas notações:

- Se $x \in \{0,1\}^{EG}$ vamos denotar por $G[x]$ o subgrafo $G[A]$ de G onde

$$A := \{e \in EG : x_e = 1\}.$$

- Dado um subgrafo T de G , dizemos que T é uma *solução de Steiner* se T contém um caminho entre cada par de vértices de Z . Se, adicionalmente, T é uma árvore, então dizemos que T é uma *árvore de Steiner*.

Lembremos aqui que estamos assumindo – a menos de menção em contrário – que as instâncias de $PSG(G, Z)$ consistem de um grafo conexo simples $G = (VG, EG)$ com $VG \geq 4$, um subconjunto $Z \subset VG$ e uma função custo $c : EG \rightarrow \mathbb{R}$ positiva.

Todos os politopos que definiremos a seguir são relativos ao par G, Z , podendo G satisfazer ou não outras propriedades adicionais.

Considere os politopos

$$P_1(G, Z) := \text{conv}\{x \in \{0,1\}^{EG} : G[x] \text{ é solução de Steiner}\} \text{ e}$$

$$P_2(G, Z) := \text{conv}\{x \in \{0,1\}^{EG} : G[x] \text{ é árvore de Steiner}\}.$$

Note que para qualquer grafo G , toda árvore de Steiner em G é uma solução de Steiner, mas a recíproca nem sempre é verdade. Logo, $P_2(G, Z) \subseteq P_1(G, Z)$. Considerando que os custos são positivos, toda solução ótima do $PSG(G, Z)$ é uma árvore de Steiner, e portanto o mais natural seria considerar apenas o politopo $P_2(G, Z)$, já que o politopo $P_1(G, Z)$ tem vértices que não são de nosso interesse.

Se escolhermos porém, um desses politopos, esbarramos em uma certa dificuldade técnica. Observe que se G é arbitrário, não é verdade que esses politopos têm dimensão completa. Veja na figura 4.1 que se G é uma árvore e Z o conjunto de suas folhas, os

Figura 4.1

politopos $P_1(G, Z)$ e $P_2(G, Z)$ têm apenas um vértice. Não tendo dimensão completa, o sistema minimal de inequações que as caracteriza não é único. Isto é, as inequações que definem facetas não são expressas de uma única maneira (a menos de um múltiplo escalar). Isto torna tecnicamente mais trabalhosa a determinação de facetas. Assim, por exemplo, resultados como o teorema 1 do capítulo 1 não se aplicam.

Para trabalhar com politopos de dimensão completa, sem com isso restringir a aplicabilidade dos resultados, poderíamos observar o seguinte: certas arestas têm que estar necessariamente em qualquer solução ótima (são de certa forma essenciais para garantir a conexidade da solução). Assim, surge o seguinte conceito:

- uma aresta e em G é uma *Steiner ponte* se e somente se e é uma ponte que separa dois vértices em Z .

Logo, se um grafo G tem Steiner pontes, podemos removê-las de G (já que vão estar na solução ótima) e nos preocuparmos apenas com cada uma das componentes resultantes. Assim, faz sentido considerar o politopo

$$P_3(G, Z) := \text{conv}\{x \in \{0, 1\}^{EG} : G[x] \text{ é solução de Steiner}\},$$

onde G é um grafo conexo sem Steiner pontes.

Este politopo foi mencionado por Grötschel e Monma [23] que estão estudando um problema de conexidade mais geral que o $PSG(G, Z)$ e que será discutido na seção 4.3. Eles mostraram que este politopo tem dimensão completa. Na verdade, esses autores obtiveram um resultado mais geral que especifica a dimensão do politopo $P_1(G, Z)$ em função do número de Steiner pontes de G . É o que veremos a seguir.

Teorema 8 *Seja $G = (VG, EG)$ um grafo, $Z \subseteq VG$ e*

$$St(G, Z) := \{\text{Steiner pontes de } G \text{ com relação a } Z\}.$$

Então,

$$P_1(G, Z) \subseteq \{x \in \{0, 1\}^{EG} : x_e = 1 \forall e \in St(G, Z)\}$$

e

$$\dim(P_1(G, Z)) = |EG| - |St(G, Z)|.$$

Dem: Se uma aresta e é uma Steiner ponte, todo vértice $x \in P_1(G, Z)$ satisfaz $x_e = 1$. Com isso, é claro que $P_1(G, Z)$ está contido em $\{x \in \{0, 1\}^{EG} : x_e = 1, \forall e \in St(G, Z)\}$. Note que $P_1(G, Z)$ está contido no espaço afim de dimensão $|EG| - |St(G, Z)|$.

Para calcularmos a dimensão do politopo, tomemos uma equação $ax = b$ tal que

$$P_1(G, Z) \subseteq \{x \in \{0, 1\}^{EG} : ax = b\} \text{ e}$$

$a_e = 0$ para todo $e \in St(G, Z)$.

É fácil ver que para toda aresta e que não é Steiner ponte, os vetores característicos x^{EG} e x^{EG-e} são pontos de $P_1(G, Z)$ e, portanto, $ax^{EG} = ax^{EG-e} = b$. Assim, $a_e = 0$

para toda aresta e que não é Steiner ponte. Logo, $a = 0$, e assim podemos concluir que $\dim(P_1(G, Z)) = |EG| - |St(G, Z)|$.

□

Na seção 4.3 apresentamos algumas facetas não triviais de $P_3(G, Z)$.

Um outro politopo que poderíamos associar ao $PSG(G, Z)$ surge da seguinte observação. Existem muitos casos em que o grafo G é completo, ou se G não é completo podemos sempre torná-lo completo acrescentando-lhe algumas arestas. Assim, faz sentido associar ao $PSG(G, Z)$ um dos seguintes politopos:

$$\begin{aligned} P_4(G, Z) &:= \text{conv}\{x \in \{0, 1\}^{EG} : G[x] \text{ é solução de Steiner}\} \text{ ou} \\ P_5(G, Z) &:= \text{conv}\{x \in \{0, 1\}^{EG} : G[x] \text{ é árvore de Steiner}\}, \end{aligned}$$

ambos definidos só para grafos **completos**.

Claramente, $P_5(G, Z) \subseteq P_4(G, Z)$. Também é claro que toda solução ótima do $PSG(G, Z)$ é vértice de $P_5(G, Z)$. Assim, resolvemos estudar este último politopo em nossa abordagem.

Teorema 9 $P_5(G, Z)$ tem dimensão completa.

Dem: Para calcularmos a dimensão do politopo, tomemos uma equação $ax = b$ tal que

$$P_5(G, Z) \subseteq \{x \in \{0, 1\}^{EG} : ax = b\}.$$

Considere $v \in VG - Z$ e seja $C := (v = v_1, e_1, v_2, \dots, e_{|VG|-1}, v_{|VG|} = w)$ um caminho que passa por todos os vértices de G (claramente tal caminho existe, desde que G é completo). Note que C é uma árvore de Steiner e portanto, $ax^C = b$.

Mas, para toda aresta $e \in C$, $C_e := C + vw - e$ é árvore de Steiner. Assim, $ax^{C_e} = ax^C$, $\forall e \in C$ e portanto,

$$a_e = a_{vw} \quad \forall e \in C.$$

Além disso, para toda aresta $f \in EG, f \notin C$, $C + f$ tem um circuito. Logo, existe uma aresta $g \in C$ tal que $C + f - g$ é árvore de Steiner. Portanto, existe $\alpha \in \mathbb{R}$ tal que

$$a_e = \alpha \quad \forall e \in EG.$$

Observe agora que $C - e_1$ é árvore de Steiner, e portanto $ax^C = ax^{C-e_1} = b$. Então, $a_{e_1} = 0$ e com isso, $\alpha = 0$. Logo, $a = 0$, o que implica que $\dim(P_5(G, Z)) = |EG|$.

□

É imediato que para toda aresta $e \in EG$ as inequações $x_e \geq 0$ e $x_e \leq 1$ são válidas para os politopos que mencionamos.

Como veremos a seguir, tais inequações quase sempre definem facetas de $P_5(G, Z)$ (se G tem pelo menos 4 vértices).

Teorema 10 *Dada uma aresta $e = uv$ em G , as inequações*

- (a) $x_e \geq 0$ *define uma faceta de $P_5(G, Z)$ e*
- (b) $x_e \leq 1$ *define uma faceta de $P_5(G, Z)$ se $(VG - \{u, v\}) - Z \neq \emptyset$.*

Dem: (a) Dada uma aresta e , tomemos uma inequação $ax \geq b$ que induz uma faceta F tal que

$$F \supseteq \{x \in P_5(G, Z) : x_e = 0\}.$$

Seja w um vértice em Z tal que e não incide em w e construa T da seguinte forma:

$$\begin{aligned} VT &:= VG; \\ ET &:= I(w). \end{aligned}$$

Note que T é árvore de Steiner e $x_e^T = 0$, e portanto $ax^T = b$. Para qualquer aresta $xy \in EG - (ET \cup \{e\})$, $T_i := T + xy - wx$ e $T_j := T + xy - wy$ são árvores de Steiner e

$$x_e^{T_i} = x_e^T = x_e^{T_j} = 0.$$

Com isso, $a_{xy} = a_{wx} = a_{wy}$. Repetindo esse argumento para toda aresta $f \in EG - (ET \cup \{e\})$ podemos concluir que existe $\alpha \in \mathbb{R}$ tal que

$$a_f = \alpha \quad \forall f \in EG - e.$$

Mas, como $Z \neq VG$ e $|VG| \geq 4$, existe $v \in VG - Z$. Assim, $T - wv$ é árvore de Steiner e $ax^{T-wv} = ax^T$. Portanto,

$$a_f = 0 \quad \forall f \in EG - e.$$

Assim, a inequação $ax \geq b$ pode ser reescrita como $a_e x \geq b$. Mas, $x^T \in F$, e portanto $b = 0$. Com isso, a inequação $ax \geq b$ é um múltiplo escalar de $x_e \geq 0$, e portanto, pelo Teorema 1 do primeiro capítulo, $x_e \geq 0$ define uma faceta de $P_5(G, Z)$.

(b) Para mostrar que dada uma aresta $e = uv \in EG$, tal que $(VG - \{u, v\}) - Z \neq \emptyset$, então $x_e \leq 1$ é faceta de $P_5(G, Z)$, vamos tomar uma inequação $ax \leq b$ que induz uma faceta F tal que

$$F \supseteq \{x \in P_5(G, Z) : x_e = 1\}.$$

Considere um vértice w em Z tal que e não incide em w e construa uma árvore T com:

$$\begin{aligned} VT &:= VG; \\ ET &:= I(w) - vw + uv. \end{aligned}$$

Note que T é árvore de Steiner e $x_e^T = 1$, e portanto $ax^T = b$. Para qualquer aresta $xy \in EG - (ET \cup I(v))$, $T_i := T + xy - wx$ e $T_j := T + xy - wy$ são árvores de Steiner e

$$x_e^{T_i} = x_e^T = x_e^{T_j} = 1.$$

Com isso, $a_{xy} = a_{wx} = a_{wy}$. Repetindo esse argumento para toda aresta $f \in EG - (ET \cup I(v))$ podemos concluir que existe $\alpha \in \mathbb{R}$ tal que

$$a_f = \alpha \quad \forall f \in EG - I(v).$$

Por outro lado, o raciocínio é análogo se tomarmos $T' := T + vw - uw$ e portanto, podemos concluir que

$$a_f = \alpha \quad \forall f \in EG - e.$$

Mas, como $(VG - \{u, v\}) - Z \neq \emptyset$, existe $r \in VG - (Z \cup \{u, v\})$. Assim, $T - wr$ é árvore de Steiner e $ax^{T-wr} = ax^T$. Portanto,

$$a_f = 0 \quad \forall f \in EG - e.$$

Assim, a inequação $ax \leq b$ pode ser reescrita como $a_e x \leq b$. Mas, $x^T \in F$, e portanto $b = a_e$. Com isso, a inequação $ax \leq b$ é um múltiplo escalar de $x_e \leq 1$, e portanto, pelo Teorema 1 do primeiro capítulo, $x_e \leq 1$ define faceta de $P_5(G, Z)$.

Note que tanto no caso (a) como no caso (b) assumimos o fato óbvio de que as inequações consideradas definem faces não triviais de $P_5(G, Z)$.

□

Em [23], Grötschel e Monma mostram os seguintes resultados a respeito dessas inequações:

Teorema 11 *Para toda aresta $e \in EG$,*

- (a) $x_e \leq 1$ define uma faceta de $P_3(G, Z)$ e
- (b) $x_e \geq 0$ define uma faceta de $P_3(G, Z)$ sse $G - e$ não tem Steiner pontes.

Dem: a) Para mostrar que $x_e \leq 1$ define uma faceta, basta observar que para qualquer aresta $f \neq e$ os vetores característicos de G e $G - f$ respeitam essa inequação com igualdade, estão em $P_3(G, Z)$, e são linearmente independentes.

b) É fácil ver que se existe uma aresta $f \in EG - e$ que seja Steiner ponte em $G - e$, então $x_e = 0$ implicará $x_f = 1$ e portanto a inequação $x_e \geq 0$ não pode definir faceta de $P_3(G, Z)$. Por outro lado, tome uma inequação $ax \geq b$ que define uma faceta F tal que

$$F \supseteq \{x \in P_3(G, Z) : x_e = 0\}.$$

Como $G - e$ não tem Steiner pontes, os subgrafos $G - e$ e $G - \{e, f\}$, para todo $f \in EG - e$ são todos soluções de Steiner cujos vetores característicos satisfazem $x_e = 0$, e portanto pertencem a F , i.e, satisfazem $ax \geq b$ com igualdade. Logo, $a_f = 0$, para todo $f \in EG - e$. Podemos então reescrever a inequação $ax \geq b$ como $a_e x_e \geq b$. Como $G - e$ é solução de Steiner, $b = 0$. Com isso $x_e \geq 0$ define uma faceta de $P_3(G, Z)$.

□

Na próxima seção apresentamos as demais facetas que obtivemos no estudo do politopo $P_5(G, Z)$.

4.2 Facetas do Politopo $P_5(G, Z)$

Como dissemos na seção anterior, associamos ao *Problema de Steiner em Grafos* o politopo $P_5(G, Z)$ definido para instâncias em que o grafo G é completo. Isso não implica em perda de generalidade já que, para qualquer grafo abordado, podemos sempre adicionar arestas com custo alto até que este se torne completo. Além disso, em muitas aplicações (por exemplo, projeto de redes de telecomunicação) o grafo considerado é completo, ou seja, não se impede *a priori* a realização de qualquer ligação.

Estudando o politopo $P_5(G, Z)$ encontramos algumas facetas não triviais que passaremos a descrever nas próximas subseções.

4.2.1 Facetas de Corte

Inicialmente lembremos que dado $A \subseteq EG$ e um vetor x indexado por EG , adotamos a seguinte notação

$$x(A) := \sum_{e \in A} x_e.$$

A observação que nos levou às facetas definidas nesta seção chamadas **facetas de corte** foi a seguinte: dado um subconjunto $A \subseteq VG$ tal que o corte $[A : \bar{A}]$ no grafo separa um vértice obrigatório de outro (ou seja, existem $u, v \in Z$ tais que $u \in A$ e $v \in \bar{A}$), pelo menos uma das arestas deste corte deve estar na solução. Observe a figura 4.2.

Pelo menos uma aresta de $[A : \bar{A}]$ deve pertencer à solução ótima.

Figura 4.2

No teorema abaixo verificamos que a inequação sugerida pela observação feita acima quase sempre define uma faceta de $P_5(G, Z)$. Antes definamos que dado $Z \subset VG$, um conjunto $A \subset VG$ é **Z-admissível** se $A \cap Z \neq \emptyset$ e se $|A| > 1$ então $A \cap (VG - Z) \neq \emptyset$. Tal definição será bastante útil neste capítulo.

Teorema 12 Para todo $A \subset VG$ tal que $A \cap Z \neq \emptyset$ e $\bar{A} \cap Z \neq \emptyset$, a inequação

$$x(A : \bar{A}) \geq 1$$

é válida para $P_5(G, Z)$ e define uma faceta se e só se A e $\bar{A}$ são Z-admissíveis.

Dem: Inicialmente observamos que a inequação correspondente é válida, ou seja,

$$P_5(G, Z) \subseteq \{x \in \mathbb{R}^{EG} : x(A : \bar{A}) \geq 1\}.$$

Para isso, basta observar que toda árvore de Steiner contém um caminho entre cada par de vértices obrigatórios.

Para mostrar que $x(A : \bar{A}) \geq 1$ define uma faceta quando A e $\bar{A}$ são Z-admissíveis, tomemos uma inequação $ax \geq b$ que define uma faceta F do politopo tal que

$$F \supseteq \{x \in P_5(G, Z) : x(A : \bar{A}) = 1\}.$$

Note que a face definida por $x(A : \bar{A}) \geq 1$ é claramente não trivial. Sejam $u \in A \cap Z$, $v \in \bar{A} \cap Z$ e T uma árvore de G construída da seguinte maneira (veja a figura 4.3):

$$\begin{aligned} VT &:= VG \\ ET &:= \{uw : w \in A\} \cup \{uv\} \cup \{vw : w \in \bar{A}\}. \end{aligned}$$

Figura 4.3

Claramente T contém um caminho entre cada par de vértices de Z e portanto, x^T é vértice de $P_5(G, Z)$. Além disso, T possui apenas uma aresta do corte $[A : \bar{A}]$. Com isso, $x^T \in F$, ou seja, $ax^T = b$.

Seja $xy \in EG - ([A : \bar{A}] \cup ET)$ e tome $T + xy$. Sem perda de generalidade podemos assumir que $x, y \in A$ pois o mesmo argumento vale para as arestas com ambas as pontas em $\bar{A}$. Observe que $T + xy$ contém o circuito formado pelas arestas xu, yu e xy . Assim, $T + xy - xu$ e $T + xy - yu$ são ambas árvores de Steiner e seus vetores característicos respeitam a inequação do corte com igualdade.

Logo, $ax^T = ax^{T+xy-xu} = ax^{T+xy-yu}$ e portanto, $a_{xy} = a_{yu} = a_{xu}$. Observe que repetindo esse argumento para cada aresta em $EG - ([A : \bar{A}] \cup ET)$ podemos concluir que existem $\alpha, \beta \in \mathbb{R}$ tal que

$$a_e = \begin{cases} \alpha & \forall e \in E(A) \\ \beta & \forall e \in E(\bar{A}). \end{cases}$$

Como A é Z -admissível, se $|A| > 1$, existe um vértice facultativo $w \in (VG - Z) \cap A$ e então $T - uw$ é árvore de Steiner (veja a figura 4.3) e seu vetor característico respeita a inequação do corte com igualdade. Assim, $ax^T = ax^{T-uw}$ e, portanto, $\alpha = a_{uw} = 0$. Analogamente, concluímos que $\beta = 0$, de onde segue que

$$a_e = 0 \quad \forall e \in EG - [A : \bar{A}].$$

Por outro lado, se $e \in [A : \bar{A}] - \{uv\}$, então $T + e - uv$ é árvore de Steiner e seu vetor característico respeita a inequação do corte com igualdade. Assim,

$$a_e = a_{uv} \quad \forall e \in [A : \bar{A}].$$

Resumindo os resultados que obtivemos, a inequação $ax \geq b$ pode ser reescrita como

$$a_{uv}x(A : \bar{A}) \geq b.$$

Como $x^T \in F$ e $x^T(A : \bar{A}) = 1$ segue que $b = a_{uv}$. Então a inequação $ax \geq b$ é um múltiplo escalar da inequação do corte. Como vimos no primeiro capítulo, se o politopo

tem dimensão completa cada faceta é induzida por uma única inequação a menos de múltiplo escalar. Portanto, a inequação de corte define uma faceta de $P_5(G, Z)$.

Resta mostrar que se um dos conjuntos A ou $\bar{A}$ não é Z -admissível então $x(A : \bar{A})$ não define uma faceta. Neste caso, basta provar que a inequação é combinação linear não negativa de outras que são válidas para o politopo. Deixamos a prova para depois do próximo teorema (veja a observação 1).

□

A classe de inequações que vimos nesta seção são as que garantirão a conexidade da solução. A seguir, verificaremos que essas facetas são na verdade um caso particular de uma classe mais geral que apresentamos na subseção 4.2.3.

4.2.2 Facetas W -induzidas

Esta classe de facetas foi motivada pela seguinte observação: ao resolver sistemas constituídos apenas das inequações triviais e as inequações de corte, obtivemos em alguns dos testes soluções do tipo indicado na figura 4.4.

Figura 4.4

Observe que a solução x indicada na figura 4.4 respeita todas as inequações de corte vistas na subseção anterior. Observe também que os dois vértices obrigatórios (em W) marcados na figura 4.4 são “atingidos” por arestas cujas entradas em x somam apenas $\frac{3}{2}$ enquanto deveriam somar pelo menos 2. De forma mais geral, observamos que em toda solução x qualquer subconjunto $W \subset Z$ deve ser “atingido” por arestas cujas entradas em x somem pelo menos $|W|$. No teorema a seguir mostramos que tais inequações válidas são facetas de $P_5(G, Z)$.

Teorema 13 *Seja $\phi \subset W \subset Z$ e $U(W) := \bigcup_{v \in W} I(v)$. Então a inequação*

$$x(U(W)) \geq |W|$$

define uma faceta de $P_5(G, Z)$.

Dem: A validade de tal inequação é imediata. Também é óbvio que ela define uma face não trivial. Para provar que ela define uma faceta considere uma inequação $ax \geq b$ que define uma faceta F de $P_5(G, Z)$ tal que

$$F \supseteq \{x \in P_5(G, Z) : x(U(W)) = |W|\}.$$

Tome $v \in Z - W$ e construa um subgrafo T de G (cf. a figura 4.5) tal que

- $VT := VG$
- $ET := \{e \in EG : e \in I(v)\}.$

Figura 4.5

T é árvore de Steiner e $x^T(U(W)) = |W|$. Assim, $ax^T = b$. Observe que para toda aresta $e = xy \in EG - (U(W) \cup I(v))$, $T + xy$ tem um único circuito formado pelas arestas vx, vy e xy . Então, $T + e - vx$ e $T + e - vy$ são árvores de Steiner tais que $ax^{T+e-vx} = ax^{T+e-vy} = ax^T$, e portanto $a_e = a_{vx} = a_{vy}$. Podemos repetir esse argumento para cada aresta $e \in EG - (U(W) \cup I(v))$ e então concluímos que existe $\alpha \in \mathbb{R}$ tal que

$$a_e = \alpha \quad \forall e \in EG - U(W).$$

Como existe um vértice facultativo $w \in VG - Z$, observe que $T - vw$ é árvore de Steiner e $ax^{T-vw} = b$. Portanto, $\alpha = a_{vw} = 0$ e com isso,

$$a_e = 0 \quad \forall e \in EG - U(W).$$

Figura 4.6

Agora, para cada aresta $e \in U(W) - I(v)$ tome uma aresta $f \in I(v) \cap U(W)$ que incide em uma ponta de e . Veja a figura 4.6.

Observe que $T + e - f$ é árvore de Steiner, $x^{T+e-f}(U(W)) = |W|$ e portanto, $a_e = a_f$. Repetindo esse argumento para cada aresta em $U(W) - I(v)$ concluímos que existe $\beta \in \mathbb{R}$ tal que

$$a_e = \beta \quad \forall e \in U(W).$$

Resumindo nossos resultados temos

$$a_e = \begin{cases} \beta, & \text{se } e \in U(W) \\ 0, & \text{c.c.} \end{cases}$$

Podemos então reescrever a inequação $ax \geq b$ como $\beta x(U(W)) \geq b$. Como $x^T \in F$ e $x^T(U(W)) = |W|$, temos que $b = \beta |W|$ e portanto a inequação $ax \geq b$ é um múltiplo escalar da inequação W -induzida.

□

Observação 1: Note que se $A \subset VG$, $A \cap Z \neq \emptyset$, $\bar{A} \cap Z \neq \emptyset$ e A não é Z -admissível temos pelo teorema acima que a inequação $x(U(A)) \geq |A|$ define uma faceta de $P_5(G, Z)$. Mas,

$$x(U(A)) = x(A : \bar{A}) + x(E(A)).$$

Logo, $x(A : \bar{A}) + x(E(A)) \geq |A|$. Como $A \subset Z$, $x(E(A)) \leq |A| - 1$. Multiplicando a última inequação por -1 e somando à anterior, obtemos $x(A : \bar{A}) \geq 1$. Com isso, se A não é Z -admissível então a inequação $x(A : \bar{A}) \geq 1$ não define uma faceta de $P_5(G, Z)$.

4.2.3 Facetas de Partição

O PSG mostrou-se suficientemente complexo a ponto de nos fornecer material para estudo mesmo no caso de exemplos relativamente pequenos (6 vértices). Assim, ao resolvermos

alguns problemas lineares com todas as inequações vistas até agora, encontramos algumas soluções ótimas fracionárias. Este fato nos levou à descoberta de uma classe mais geral de facetas que discutiremos nesta subseção.

Figura 4.7

Na figura 4.7 indicamos uma tal solução fracionária x . Observe que todas as inequações de corte e W -induzidas estão respeitadas. Porém, veja na figura 4.8 que poderíamos particionar o conjunto de vértices do grafo como indicamos e obteríamos que

Figura 4.8

a soma das entradas de x correspondentes às arestas que ligam dois conjuntos distintos resulta $\frac{3}{2}$ e deveria resultar pelo menos 2. De forma mais geral, podemos observar que se tomarmos uma partição $\mathcal{P}$ de VG de modo a deixar pelo menos um vértice obrigatório em cada conjunto da partição, em qualquer solução x a soma das entradas correspondentes às arestas que ligam tais conjuntos deve ser pelo menos $|\mathcal{P}| - 1$.

Como veremos, inequações deste tipo são sugeridas no artigo de Grötschel e Monma [23] que estudaram uma generalização do politopo $P_3(G, Z)$.

A seguir, mostramos que tais inequações válidas em muitos casos definem facetas de $P_5(G, Z)$.

Teorema 14 *Seja $\mathcal{P} = (V_1, V_2, \dots, V_k)$ uma partição qualquer dos vértices de um grafo*

completo G , tal que $V_i \cap Z \neq \emptyset$, $i = 1, \dots, k$, e seja $I(\mathcal{P}) := \{uv \in EG : u \in V_i, v \in V_j, 1 \leq i < j \leq k\}$. Então a inequação

$$x(I(\mathcal{P})) \geq k - 1$$

é válida para $P_5(G, Z)$ e define uma faceta se e somente se cada V_i é Z -admissível.

Dem: A validade da inequação da partição é imediata, bem como o fato de que ela define uma face não trivial. Para provar que ela define uma faceta quando cada V_i é Z -admissível, tomemos uma inequação $ax \geq b$ que induz uma faceta F tal que

$$F \supseteq \{x \in P_5(G, Z) : x(I(\mathcal{P})) = k - 1\}.$$

Considere $v_i \in V_i \cap Z$ para $i = 1, \dots, k$ e construa a seguinte árvore T (conforme a figura 4.9):

- $VT := VG$
- $ET := \bigcup_{i=1}^k \{uv_i : u \in V_i - v_i\} \cup \{v_1v_j : j = 2, \dots, k\}$.

Figura 4.9

É fácil ver que T é árvore de Steiner e $x^T(I(\mathcal{P})) = k - 1$. Portanto, $ax^T = b$.

Seja $xy \in E(V_i) - ET$ para algum $i \in \{1, \dots, k\}$. Lembre que dado $A \subseteq VG$ definimos

$$E(A) := \{xy \in EG : x, y \in A\}.$$

Sabemos que $T + xy$ contém o circuito formado pelas arestas v_ix, v_iy e xy . Assim, $T + xy - v_ix$ e $T + xy - v_iy$ são árvores de Steiner e seus vetores característicos respeitam a inequação da partição com igualdade (veja a figura 4.10). Repetindo esse argumento para as demais arestas de $E(V_i) - ET$ e observando que o mesmo argumento vale para todo $i \in \{1, \dots, k\}$ podemos concluir que se $|V_i| \neq 1$ então existe $\alpha_i \in R$ tal que

$$a_e = \alpha_i \quad \forall e \in E(V_i) - I(\mathcal{P}).$$

Figura 4.10

Como V_i é Z -admissível para todo $i = 1, \dots, k$, existe um vértice facultativo $w_i \in V_i \cap (VG - Z)$ e portanto $T - v_i w_i$ é árvore de Steiner e seu vetor característico respeita a inequação da partição com igualdade. Então $ax^T = ax^{T-v_i w_i} = b$ e portanto $\alpha_i = a_{v_i w_i} = 0$. Como isso vale para todo V_i , temos

$$a_e = 0 \quad \forall e \in EG - I(\mathcal{P}).$$

Por fim, se tomarmos uma aresta $e = u_i u_j$ tal que $u_i \in V_i, u_j \in V_j$ e $1 < i < j \leq k$ observamos que as árvores $T_i := T + e - v_1 v_i$ e $T_j := T + e - v_1 v_j$ são ambas árvores de Steiner e seus vetores característicos respeitam a inequação da partição com igualdade. Portanto,

$$ax^T = ax^{T_i} = ax^{T_j}$$

o que implica que $a_e = a_{v_1 v_i} = a_{v_1 v_j}$. Quando $i = 1$ vale o mesmo argumento para cada aresta e do tipo acima, bastando considerar apenas as árvores T e T_j . Dessa forma, podemos concluir que existe $\beta \in \mathbb{R}$ tal que

$$a_e = \beta \quad \forall e \in I(\mathcal{P})$$

Resumindo nossas conclusões sobre a inequação da partição temos:

$$a_e = \begin{cases} \beta, & \text{se } e \in I(\mathcal{P}) \\ 0, & \text{c.c.} \end{cases}$$

Reescrevendo a inequação $ax \geq b$, temos

$$\beta x(I(\mathcal{P})) \geq b.$$

Como $x^T(I(\mathcal{P})) = k - 1$, então $b = \beta(k - 1)$. Com isso obtivemos que a inequação $ax \geq b$ é um múltiplo escalar da inequação da partição e, portanto, esta define uma faceta de $P_5(G, Z)$.

Resta mostrar que se algum V_i não é Z -admissível, então a inequação da partição não define uma faceta. Sem perda de generalidade, suponhamos que V_1 não seja Z -admissível, $V_1 = \{v_1, \dots, v_r\}$. Vamos provar que $x(I(\mathcal{P})) \geq k - 1$ pode ser obtida fazendo-se combinação linear de inequações válidas para $P_5(G, Z)$.

Suponhamos que $k > 2$ (pois se $k = 2$, $\mathcal{P}$ é uma bipartição e o resultado segue da Observação 1). Considere a partição $\mathcal{P}' := (\{v_1\}, \dots, \{v_r\}, V_2, \dots, V_k)$. A inequação

$$x(I(\mathcal{P})) \geq k + |V_1| - 2$$

é válida para $P_5(G, Z)$. Note que $x(I(\mathcal{P}')) = x(I(\mathcal{P})) + x(E(V_1))$. Como $x(E(V_1)) \leq |V_1| - 1$ é válida para $P_5(G, Z)$, multiplicando esta última inequação por -1 e somando com a primeira obtemos

$$x(I(\mathcal{P})) \geq k - 1.$$

□

Da mesma forma que nos casos anteriores, o número de inequações fornecido por esta classe de facetas é muito grande : existe uma inequação para cada partição dos vértices do grafo que contém um vértice obrigatório em cada conjunto.

Como já mencionamos esta classe é uma generalização das duas classes anteriores. No caso da faceta de corte, basta observar que um corte é uma bipartição do conjunto de vértices do grafo. Já no caso das facetas W -induzidas, dado $W = \{w_1, \dots, w_n\}$ basta tomar a partição $\mathcal{P} = (\{w_1\}, \dots, \{w_n\}, VG - W)$.

4.3 Facetas do Politopo $P_3(G, Z)$

Enquanto estudávamos o *Problema de Steiner em Grafos* soubemos que Grötschel e Monma estavam pesquisando um problema de conexidade de redes que tinha como caso particular o nosso problema. De posse, inicialmente, de um artigo em preparação e depois de um relatório técnico [23] desses autores, pudemos verificar os resultados que eles obtiveram, bem como estudar como se particularizam no caso do PSG .

O modelo estudado pelos autores foi o seguinte:

Seja G um grafo com custos não negativos associados às suas arestas e considere dados para cada par de vértices distintos u e v três números inteiros r_{uv} , d_{uv} e k_{uv} . Deseja-se construir um subgrafo H de custo mínimo que respeita as seguintes condições de conexidade para cada par de vértices $u, v \in VG$:

- H tem pelo menos r_{uv} caminhos disjuntos nas arestas entre u e v ;

- Após a remoção de no máximo k_{uv} vértices de H (distintos de u e v), o subgrafo resultante tem pelo menos d_{uv} caminhos disjuntos nas arestas entre u e v .

Os autores apresentam uma formulação deste problema como um problema de programação linear inteira e observam em seu artigo que o problema é $\mathcal{NP}$ -difícil já que tem como caso particular o *Problema de Steiner em Grafos*.

De fato, no caso do $PSG(G, Z)$, dado $Z \subseteq VG$, basta definir

$$r_{uv} = \begin{cases} 1 & \text{se } u, v \in Z \\ 0 & \text{cc.} \end{cases}$$

$$d_{uv} = k_{uv} = 0 \quad \forall u, v \in VG$$

Com isso, o modelo geral sugerido pelos autores fica reduzido ao seguinte problema:

$$\begin{aligned} & \min \quad cx \\ & \text{s.a.} \quad \begin{cases} x(A : \bar{A}) \geq 1 & \forall A \subseteq VG, A \cap Z \neq \emptyset \text{ e } \bar{A} \cap Z \neq \emptyset \\ 0 \leq x_e \leq 1 & \forall e \in EG \\ x \text{ inteiro} \end{cases} \end{aligned}$$

Lembramos aqui que Grötschel e Monma estudaram um politopo associado ao problema geral que mencionamos, que se particulariza para o PSG ao politopo $P_3(G, Z)$ definido para grafos conexos sem Steiner pontes:

$$P_3(G, Z) := \text{conv}\{x \in \{0, 1\}^{EG} : G[x] \text{ é solução de Steiner} \}.$$

Os autores analisaram os casos em que eliminando-se as restrições de integralidade o sistema de inequações resultante caracteriza completamente o politopo $P_3(G, Z)$. Antes de verificarmos tais resultados lembremos que uma aresta e é uma *Steiner ponte* do grafo se e for uma ponte que separa dois vértices obrigatórios. É interessante notar que existem algoritmos polinomiais para determinar as Steiner pontes de um grafo. No modelo geral dos autores, a determinação das arestas essenciais (generalização das Steiner pontes no modelo geral) é um problema $\mathcal{NP}$ -difícil.

Teorema 15 *Seja G um grafo conexo sem Steiner pontes e $Z \subseteq VG$ tal que $|Z| = 2$. Então o sistema abaixo é uma caracterização completa e não redundante de $P_3(G, Z)$:*

$$\begin{cases} x(A : \bar{A}) \geq 1 & \forall A \subseteq VG, A \cap Z \neq \emptyset \text{ e } \bar{A} \cap Z \neq \emptyset, G[A] \text{ e } G[\bar{A}] \text{ conexos} \\ x_e \leq 1 & \forall e \in EG \\ x_e \geq 0 & \forall e \in EG : G - e \text{ não tem Steiner pontes} \end{cases}$$

A demonstração deste resultado, segundo os autores, segue imediatamente dos resultados sobre a caracterização completa do politopo associado ao problema de caminho mais curto. Eles observam, ainda, que este sistema não caracteriza o politopo em qualquer outro caso, nem mesmo no outro caso extremo onde $Z = VG$. Em [56] White, Farber e Pulleyblank observam que não é suficiente para descrever o politopo associado ao problema em que o grafo G é fortemente cordal (veja a subseção 3.4.3).

A seguir, os autores sugerem uma generalização natural para o sistema descrito acima. Lembremos antes que dada uma partição $\mathcal{P}$ de VG usamos a notação

$$I(\mathcal{P}) = \{e \in EG : e \text{ incide em dois conjuntos diferentes da partição } \mathcal{P}\}.$$

Teorema 16 *Seja G um grafo conexo sem Steiner pontes e $Z \subseteq VG$ um conjunto de vértices obrigatórios. Dada uma partição $\mathcal{P} = (V_1, V_2, \dots, V_k)$ de VG tal que $V_i \cap Z \neq \emptyset$, para $i = 1, \dots, k$, então*

$$x(I(\mathcal{P})) \geq k - 1$$

define uma faceta de $P_3(G, Z)$ se e somente se:

1. $G[V_i]$ é conexo para $i = 1, \dots, k$;
2. $G[V_i]$ não contém Steiner pontes com relação a $Z_i = Z \cap V_i$, para $i = 1, \dots, k$;
3. o grafo $\hat{G}$ obtido de G contraindo-se cada conjunto V_i a um só vértice é 2-conexo.

Dem: Suponhamos, inicialmente, que $G[V_i]$ não é conexo. Sem perda de generalidade podemos considerar $G[V_1]$ desconexo. Seja (V'_1, V'_2) uma bipartição de V_1 tal que $V'_1 \cap Z \neq \emptyset$, $V'_2 \cap Z \neq \emptyset$ e $[V'_1 : V'_2] = \emptyset$. Como G é conexo existe algum conjunto V_j , para $j \in \{2, \dots, k\}$ (digamos V_2) tal que V'_2 e V_2 estão conectados por alguma aresta (veja a figura 4.11).

Figura 4.11

Então, para a nova partição $\mathcal{P}' = (V'_1, V_2 \cup V'_2, V_3, \dots, V_k)$ a inequação

$$x(I(\mathcal{P}')) \geq k - 1$$

é do tipo enunciado no teorema e sua soma com a inequação válida $x(V_2 : V'_2) \geq 0$ resulta na inequação

$$x(I(\mathcal{P})) \geq k - 1$$

Portanto, esta última não define uma faceta. Podemos, então, supor que $G[V_i]$ é conexo para $i = 1, \dots, k$. Suponha que $G[V_i]$ contém alguma Steiner ponte e com respeito a $Z_i = Z \cap V_i$. Vamos mostrar que

para todo vértice $x \in P_3(G, Z)$ tal que $x(I(\mathcal{P})) = k - 1$ temos que $x_e = 1$.

Assim, a inequação não define faceta.

Suponhamos que existe $D \subseteq G - e$ tal que $x^D \in P_3(G, Z)$ e $x^D(I(\mathcal{P})) = k - 1$. Como e é Steiner ponte de $G[V_i]$, D deve conter um caminho ligando dois vértices de Z em componentes diferentes de $G[V_i] - e$. Assim $D' := D + eV_i$ é tal que $x^{D'} \in P_3(G, Z)$ e respeita a inequação da partição com igualdade. Além disso, D' contém um circuito que contém a aresta e e uma aresta $f \in [V_i, \bar{V}_i]$. Mas, neste caso, $D'' := D' - f + e$ é tal que $x^{D''} \in P_3(G, Z)$ e

$$x^{D''}(I(\mathcal{P})) = k - 2,$$

o que é um absurdo. Logo, $G[V_i]$ não tem Steiner pontes com relação a Z_i .

Finalmente suponha que $\hat{G}$ não é 2 conexo. Seja $V\hat{G} = \{v_1, v_2, \dots, v_c\}$ onde v_i é o vértice obtido ao se contrair V_i . Sem perda de generalidade, podemos supor que v_1 é um vértice de corte e que os vértices $v_2, \dots, v_c$ pertencem à mesma componente de $\hat{G} - v_1$.

Tome

$$\begin{aligned} W_1 &:= V_1 \cup (V_{c+1} \cup \dots \cup V_k), \\ W_2 &:= V_1 \cup (V_2 \cup \dots \cup V_c) \end{aligned}$$

e construa as partições

$$\begin{aligned} \mathcal{P}_1 &:= (W_1, V_2, V_3, \dots, V_c), \\ \mathcal{P}_2 &:= (W_2, V_{c+1}, V_{c+2}, \dots, V_k). \end{aligned}$$

Agora, basta notar que a inequação $x(I(\mathcal{P})) \geq k - 1$ é a soma das inequações $x(I(\mathcal{P}_1)) \geq c - 1$ e $x(I(\mathcal{P}_2)) \geq k - c$, portanto não define faceta de $P_3(G, Z)$. Com isso, terminamos uma parte da demonstração.

Resta mostrar que se $\mathcal{P} = (V_1, V_2, \dots, V_k)$ é uma partição como no enunciado do teorema que satisfaz (1), (2) e (3), então a inequação correspondente define uma faceta de $P_3(G, Z)$.

Seja $ax \geq b$ uma inequação que define uma faceta F de $P_3(G, Z)$ tal que

$$\{x \in P_3(G, Z) : x(I(\mathcal{P})) = k - 1\} \subseteq F.$$

Primeiramente observe que a toda árvore geradora T de $\hat{G}$ corresponde uma floresta de G , também denotada por T , tal que

$$x^{T+B}(I(\mathcal{P})) = k - 1 \text{ para todo } B \subseteq R := \bigcup_{i=1}^k EV_i.$$

Seja $e \in EV_i$. Tome dois vértices $u \in Z \cap V_i$ e $v \in VG - V_i$. Como G não contém Steiner pontes, existe um caminho que liga u a v em G que não passa por e . Tome f a aresta de um tal caminho que está no corte $[V_i : \bar{V}_i]$ e construa uma árvore geradora T' de $\hat{G}$ que contém f .

Assim, se tomarmos $D := T' + R$, como $G[V_j]$ é conexo para $j = 1, \dots, k$ e e não é Steiner ponte de $G[V_i]$, $x^D \in P_3(G, Z)$ e $x^{D-e} \in P_3(G, Z)$. Além disso,

$$x^D(I(\mathcal{P})) = x^{D-e}(I(\mathcal{P})) = k - 1$$

e portanto, $a_e = 0$, $\forall e \in EV_i$.

Agora, considere $e, f \in I(\mathcal{P})$. Como $\hat{G}$ é 2-conexo existe um circuito C em $\hat{G}$ que contém e e f . Tome uma árvore geradora T' de $\hat{G}$ que contém $C - f$. Então, $T'' := T' - e + f$ é uma árvore geradora de $\hat{G}$. Seja $D' = T' + R$ e $D'' = T'' + R$. Observe que $x^{D'} \in P_3(G, Z)$, $x^{D''} \in P_3(G, Z)$ e ainda

$$ax^{D'} = ax^{D''}.$$

Logo $a_e = a_f$. Assim, $ax \geq b$ é um múltiplo escalar de $x(I(\mathcal{P})) \geq k - 1$ e portanto, esta define uma faceta de $P_3(G, Z)$.

□

É interessante notar que os resultados são bem mais complexos que os que obtivemos para $P_5(G, Z)$. Os autores explicam em [23] que os resultados obtidos seriam utilizados no desenvolvimento de algoritmos que seriam baseados em heurísticas rápidas para resolver o problema da separação (veja na próxima seção o princípio do algoritmo). Grötschel e Monma observam que para as facetas de corte o problema da separação pode ser resolvido em tempo polinomial usando o procedimento de Gomory e Hu [21]. Para as facetas de partição nenhum algoritmo eficiente para o problema da separação é conhecido. Sabemos que o algoritmo está sendo desenvolvido a fim de projetar-se uma substituição segura e econômica de algumas linhas de transmissão nos EUA, de linhas comuns para fibras óticas.

4.4 Descrição do Algoritmo e Resultados Computacionais

Nesta seção descrevemos o algoritmo para o PSG que desenvolvemos utilizando os resultados teóricos vistos na seção 4.2 e discutimos uma implementação do mesmo. Um dos principais objetivos desta implementação foi ganhar maior familiaridade com o problema e obter experiência no trato com algoritmos baseados em métodos de planos-de-corte e com programas para resolução de problemas de programação linear.

4.4.1 Descrição do algoritmo

O princípio do algoritmo é, em cada iteração resolver um problema de programação linear que é uma relaxação da descrição do politopo associado ao $PSG(G, Z)$. A solução ótima deste problema relaxado (chamaremos **solução parcial**) nos devolve um subgrafo de G que pode ou não ser uma solução viável do $PSG(G, Z)$. Se a solução parcial x' induzir um subgrafo de G que é uma árvore de Steiner e, além disso $x' \in \{0, 1\}^{EG}$, então obtivemos uma solução ótima do $PSG(G, Z)$. Caso contrário, devemos procurar inequações válidas que não sejam satisfeitas pela solução x' para adicionarmos ao sistema que estamos resolvendo e assim obtermos uma nova solução parcial. A seguir, detalhamos um pouco mais estas idéias e discutimos a implementação de um algoritmo.

É dado, então, um grafo $G = (VG, EG)$ com uma função custo $c : EG \rightarrow \mathbb{R}$ e um conjunto $Z \subset VG$ e desejamos resolver o $PSG(G, Z)$. A idéia é começar com um sistema de inequações $Ax \leq b$ constituído de inequações válidas que definam preferencialmente facetas de $P_5(G, Z)$ e então resolver o problema de programação linear

$$\begin{aligned} \min \quad & cx \\ \text{s.a.} \quad & \begin{cases} Ax \leq b \\ 0 \leq x \leq 1 \end{cases} \end{aligned}$$

Observe que temos três possibilidades para a solução ótima x' do problema descrito acima. Lembremos antes que para $x \in \{0, 1\}^{EG}$ chamamos $G[x]$ o subgrafo de G induzido por $\text{supp}(x) := \{e : x_e \neq 0\}$. Quando a solução for fracionária, também denotaremos por $G[x]$ o grafo induzido por $\text{supp}(x)$.

- $x' \in \{0, 1\}^{EG}$ e o grafo induzido $G[x]$ é uma árvore de Steiner. Nesse caso, x' é solução ótima do $PSG(G, Z)$.
- Existe aresta $e \in EG$ tal que $x_e \notin \{0, 1\}$. Nesse caso, devemos encontrar inequações válidas para o politopo a fim de adicionarmos ao sistema e obtermos uma nova solução parcial.

- $x' \in \{0, 1\}^{EG}$ mas o grafo induzido por x' não é árvore de Steiner. Nesse caso, mais inequações devem ser adicionadas ao sistema para obtermos uma nova solução parcial.

Veja a descrição desse algoritmo na figura 4.12

As inequações que adicionaremos ao sistema a cada passo são as que descrevemos na seção 4.2. Na primeira iteração, o sistema inicial consiste das inequações triviais e das inequações $x(I(v)) \geq 1$ para $v \in Z$ (estas impõem a obrigatoriedade de que a todo vértice em Z deve incidir alguma aresta e em $\text{supp}(x)$). Note que estas inequações definem facetas de $P_5(G, Z)$ (cf. seção 4.2.1).

Na descrição geral do algoritmo não especificamos como escolher as inequações que devem ser adicionadas ao sistema em cada passo do algoritmo. Também não discutimos como resolver o seguinte problema: dada uma solução parcial x' , verificar se x' é solução ótima e, se não for, determinar uma inequação válida $ax \leq b$ tal que $ax' > b$. Este problema é conhecido como o *problema da separação* ou *identificação de faceta violada*. Diz-se também que a inequação $ax \leq b$ é **violada** pela solução x' ou que ela é um **plano-de-corte**. Quando a busca de facetas violadas é restrita a uma determinada classe, referimos ao problema da separação para essa classe. Assim, dizemos que o problema da separação para uma certa classe X é fácil ou difícil. Em geral, para problemas $\mathcal{NP}$ -completos para os quais são conhecidas algumas classes de facetas do politopo a ele associado, o problema da separação é fácil para uma ou outra classe e é difícil para muitas delas. Veja por exemplo, o caso do Problema do Caixeiro Viajante [43], do Subgrafo Acíclico [26] e da partição em Cliques [27].

Nesse contexto vale a pena lembrar que, se para um dado problema P conhecemos a descrição completa do politopo a ele associado, e o problema da separação relativo a essa descrição pode ser resolvido em tempo polinomial, então (pelo menos teoricamente) o problema P também pode ser resolvido em tempo polinomial. Isto segue de resultados devidos a Grötschel, Lovász e Schrijver [25]. Conseqüentemente, para problemas $\mathcal{NP}$ -completos (e sob a hipótese de que $\mathcal{P} \neq \mathcal{NP}$) ou não conseguiremos uma descrição completa do politopo a ele associado, ou se encontrarmos uma tal descrição não saberemos resolver eficientemente o problema da separação. Este resultado é de certa forma bastante desencorajador para quem quer desenvolver um algoritmo do tipo que descrevemos para resolver problemas $\mathcal{NP}$ -completos. A favor desta abordagem, podemos mencionar que vários problemas difíceis para os quais algoritmos baseados em planos-de-corte combinados com *branch and bound* têm fornecido resultados altamente satisfatórios na resolução de problemas reais de grande porte. Veja, por exemplo, Padberg e Rinaldi [44] e Padberg e Grötschel [43].

Figura 4.12

4.4.2 Resultados Computacionais

Nesta seção apresentamos detalhes da implementação do algoritmo e também fazemos um histórico de como as classes de facetas do nosso politopo foram sendo descobertas. Apresentamos também um resumo dos resultados que obtivemos nos testes em que aplicamos o algoritmo.

Os nossos testes computacionais seguiram três fases: na primeira utilizamos equipamento compatível com IBM PC/XT (4.77 MHz e 640 Kb de memória) no qual só pudemos executar exemplos com no máximo 10 vértices (devido a limitação de memória). Na segunda fase passamos a utilizar o equipamento PROCEDA 4270-PC (processador Motorola 68020, 20MHz e 4 Mb de memória) onde pudemos testar nosso algoritmo com grafos de até 15 vértices. Finalmente, na terceira e última fase utilizamos o equipamento B7900 da Unysis onde pudemos executar um exemplo de proporções maiores (58 vértices).

O algoritmo que desenvolvemos foi implementado na linguagem FORTRAN 77. A razão de tal escolha foi o fato que dispúnhamos de um programa em FORTRAN 77 para resolver o problema de programação linear em cada etapa – o XMP Linear Programming Library desenvolvido na Universidade do Arizona por R. Marsten [38] – que usa o método simplex. Na última fase da implementação, o programa que desenvolvemos junto com o XMP ficou com mais de 10000 linhas de programa fonte.

Primeira Fase: IBM PC/XT

Utilizamos nessa fase apenas as inequações da classe de facetas de corte (cf. a subseção 4.2.1). Pudemos então fazer alguns testes para grafos com a função custo gerada aleatoriamente a fim de verificarmos empiricamente como se comporta esta classe de facetas e também observar como são as soluções fracionárias.

Relembramos, antes, que para todo corte $[A : \bar{A}]$, tal que A e $\bar{A}$ são Z -admissíveis a inequação $x(A : \bar{A}) \geq 1$ define uma faceta de $P_5(G, Z)$. Com isso, temos um número muito grande de inequações que poderíamos introduzir no sistema. Faz-se necessário, então, definir uma estratégia para determinar quais delas devemos adicionar ao sistema de restrições em cada iteração de nosso algoritmo.

Nós consideramos duas possíveis estratégias para esta classe de facetas, supondo ter uma solução parcial x' :

1. Podemos achar em $G[x']$ um corte cuja inequação correspondente seja válida e esteja violada por x' e então acrescentamos esta inequação ao sistema ;
2. Podemos identificar o tamanho k da menor componente conexa de $G[x']$ e então acrescentar ao sistema todas as inequações correspondentes a cortes $[A : \bar{A}]$ tais que $|A| = k$.

O problema da separação para a classe de facetas de corte pode ser resolvido em tempo polinomial utilizando o algoritmo de Gomory e Hu [21] para determinar cortes mínimos que separam vértices obrigatórios. Tal algoritmo porém, se acoplado a um algoritmo de fluxo máximo de complexidade $O(|VG|^3)$, resultaria em um procedimento $O(|VG|^4)$ para resolver o problema da separação. Decidimos, então, escolher a segunda estratégia que é mais rápida. Na tabela 4.1 resumimos os resultados que obtivemos executando nosso algoritmo para diversos exemplos aleatórios.

$ VG $	$ Z $	no. de casos com sol. ótima inteira	no. de casos com sol. ótima frac
5	3	41	80
6	3	13	51
10	5	7	19

Tabela 4.1: Testes apenas com as inequações de corte.

Segunda Fase : PROCEDA 4270 - PC.

Tendo possibilidade de executar exemplos aleatórios maiores, foi nessa fase que implementamos procedimentos para acrescentar ao sistema as inequações correspondentes às facetas W -induzidas e às facetas de partição.

Como já observamos na seção 4.2, mesmo com todas as inequações de corte respeitadas, ainda obtivemos casos com soluções parciais fracionárias como na figura 4.13.

Figura 4.13

Na subseção 4.2.2 mostramos que para todo $W \subset Z$ a inequação $x(U(W)) \geq |W|$ define uma faceta de $P_5(G, Z)$. Como podemos notar, o número de inequações fornecido por essa classe é muito grande (exponencial em $|Z|$). Nossa estratégia para utilizar as inequações W -induzidas foi ir acrescentando ao sistema inicialmente aquelas com $|W| = 2$, e então $|W| = 3$, e assim sucessivamente até que $|W| = |Z| - 1$. A eficiência dessa classe

de inequações na eliminação de valores fracionários na solução pode ser avaliada através da tabela 4.2.

$ VG $	$ Z $	no. de casos com sol. ótima inteira	no. de casos com sol. ótima frac
5	2	5	0
5	3	41	13
5	4	33	43
6	3	10	15
7	3	3	2
8	4	5	1
9	4	3	1
12	6	2	2
15	7	1	1

Tabela 4.2: Testes apenas com as inequações de corte e W-induzidas.

Como já havíamos observado na subseção 4.2.3 soluções do tipo da figura 4.14 ainda podiam ser soluções parciais. Tal observação, como dissemos, nos levou às facetas da partição que, como vimos, é uma generalização das outras duas classes de facetas. Naquela

Figura 4.14

subseção vimos que para toda partição $\mathcal{P} = (V_1, \dots, V_k)$ de VG , desde que $V_i \cap Z \neq \emptyset$ para $i = 1, \dots, k$ a inequação $x(I(\mathcal{P})) \geq k - 1$ é uma inequação válida para $P_5(G, Z)$ e ainda se cada V_i é Z -admissível a inequação define uma faceta do politopo.

Não se conhece algoritmo eficiente para resolver o problema da separação para esta classe de facetas, ou seja, dada uma solução parcial determinar uma partição de VG cuja inequação correspondente seja violada por essa solução parcial. Utilizamos, então, uma heurística para tentar resolver este problema. A idéia da heurística é que as arestas “mais pesadas” devem ter ambas as pontas num mesmo subconjunto da partição $\mathcal{P}$. Com isso, percorremos duas vezes a solução parcial x' : na primeira determinamos que alguns vértices devem estar juntos, pois estão ligados por arestas “pesadas” e na segunda vez percorremos a solução parcial a fim de determinar a que subconjunto os demais vértices

devem pertencer. Esta heurística, apesar de simples, se mostrou eficaz. Observe os resultados computacionais que resumimos na tabela 4.3.

$ VG $	$ Z $	no. de casos com sol. ótima inteira	no. de casos com sol. ótima frac
5	3	16	0
8	4	15	0
10	5	8	0
12	6	7	0
15	7	4	0

Tabela 4.3: Testes com todas as classes de facetas.

Terceira Fase : um exemplo real – UNYSIS B7900.

Após conseguirmos bons resultados para exemplos pequenos (até 15 vértices) com função custo gerada aleatoriamente, passamos a imaginar qual seria o comportamento do algoritmo quando aplicado a exemplos reais. Assim, decidimos resolver um problema real correspondente a um grafo completo de 58 vértices, representando cidades brasileiras, dos quais 25 são obrigatórios (as capitais) sendo que a função custo é definida pela distância rodoviária entre as cidades que a aresta liga (cf. [61]). Veja a figura 4.15.

Observemos que o número de restrições a serem acrescentadas ao sistema inicial, conforme descrevemos nas subseções anteriores, pode ser muito grande. Basta observar que se o algoritmo fosse, por exemplo, acrescentar as inequações correspondentes aos cortes que separam conjuntos de três vértices deveríamos fornecer mais de 4,5 milhões de elementos não nulos.

Na resolução de problemas reais de grande porte através de estratégias baseadas em métodos de planos-de-corte, tem-se mostrado necessário atacar problemas particulares de forma particular. Utilizando tal princípio, a partir de uma solução parcial já “manipulável”, fomos adicionando – de forma interativa – restrições encontradas ao estudarmos cada solução parcial. Veja nas figuras 4.16, 4.17 e 4.18 algumas das soluções parciais que obtivemos e as inequações violadas que acrescentamos ao sistema a fim de eliminarmos tais soluções.

É interessante notar que a cada vez que obtemos uma solução parcial x' em que o subgrafo $G[x']$ é conexo, temos um limitante inferior e um limitante superior para a solução ótima do problema. O valor objetivo da solução parcial é um limitante inferior para o valor ótimo, pois a solução ótima do problema é uma solução viável do problema relaxado. Por outro lado, como $G[x']$ é conexo, se tomarmos o valor dado por $\sum_{e \in EG[x']} c(e)$, este é um limitante superior para o valor da solução ótima do problema.

Não pudemos (ainda) determinar a solução ótima do problema real que formu-

Figura 4.15

Figura 4.16

Figura 4.17

Figura 4.18

lamos e a melhor solução que obtivemos tem (custo total) 14.030 km e está indicada na figura 4.19. Este valor é um limitante superior para o valor da solução ótima. Pelos limitantes inferiores que obtivemos podemos garantir que o valor da solução ótima está no intervalo [12.849,14.030]. Com isso, o erro máximo que podemos estar cometendo é de 1.181 km (menos de 10 %). O tempo de CPU gasto pelo nosso algoritmo não foi prioritário no desenvolvimento do algoritmo. Particularmente, na execução de algumas iterações deste exemplo real, muitas vezes foi necessário mais de 5 minutos de CPU.

Cabe aqui notar que o algoritmo que desenvolvemos procura determinar uma solução x' que minimiza a função cx sobre um politopo que é uma relaxação de $P_5(G, Z)$ já que temos apenas uma caracterização parcial deste último. Em muitos casos obtém-se assim uma solução que é viável e inteira para o PSG , e que portanto é uma solução ótima. Quando isto não acontece, de posse de limitantes inferiores e superiores, podemos proceder a busca de uma solução ótima utilizando técnicas de *branch and bound*. Não fizemos a implementação de algum método baseado nesta estratégia, mas isto seria certamente interessante pois um algoritmo que implementasse ambas as estratégias sempre forneceria uma solução ótima.

As estratégias que ultimamente têm-se mostrado bastante eficientes para problemas difíceis de grande porte são conhecidas como *branch and cut* e trazem acopladas técnicas de *branch and bound* com planos-de-corte. Estas duas técnicas são combinadas, cada qual se beneficiando com as informações obtidas da outra e explorando estratégias de geração parcial e fixação permanente de variáveis (veja, por exemplo, Padberg e Rinaldi [44] e Padberg e Grötschel [43]).

Figura 4.19

Comentários Finais

A pesquisa científica é uma atividade interessante: apesar de seu objetivo ser acabar com as dúvidas do Homem, na maioria das vezes esta trabalha no sentido inverso. Em nosso caso (ou melhor, *meu caso*), quando iniciamos este trabalho tínhamos apenas uma única dúvida: “*Problema de Steiner !? Que é isso?*” Hoje esta dúvida não mais existe, porém surgiram algumas outras que gostaríamos de compartilhar e contribuir modestamente ao “arsenal de dúvidas da Humanidade”:

- Como se comportará uma estratégia baseada em planos-de-corte para as variantes do problema de Steiner em Grafos?
- Como resolver o problema da separação para a classe de facetas da partição?
- Que propriedades têm os politopos que associamos ao problema? Qual é o seu diâmetro? Quando seus vértices são adjacentes?
- Como tornar nosso algoritmo mais eficiente? Como melhorar as heurísticas para acrescentar inequações ao sistema?
- Certamente seria interessante desenvolver um algoritmo baseado na estratégia de *branch and cut* para o nosso problema. Como seria o comportamento de tal algoritmo?

A respeito do algoritmo que desenvolvemos, vale a pena mencionar que não temos garantia alguma sobre sua eficiência e nem ao menos sobre sua eficácia. Como não caracterizamos completamente o politopo, não podemos afirmar com certeza que nosso algoritmo resolve o problema para qualquer instância. Além disso, não podemos afirmar que nosso algoritmo é melhor que algum outro da literatura, pois qualquer tipo de comentário nesse sentido deveria levar em conta diferenças em máquinas, instâncias, comportamento médio e assim por diante.

Bibliografia

- [1] Y. P. Aneja, "An Integer Linear Programming Approach to the Steiner Problem in Graphs", *Networks* 10 (1980) 167 - 178.
- [2] A. Balakrishnan e N.R. Patel, "Problem Reduction Methods and a Tree Generation Algorithm for the Steiner Network Problem", *Networks* 17 (1987) 65 - 85.
- [3] M.O. Ball, W. Liu e W.R. Pulleyblank, "Two Terminal Steiner Tree Polyhedra", *Technical Report 87021*, Management Science and Statistics, University of Maryland at College Park (1987).
- [4] J.E. Beasley, "An Algorithm for the Steiner Problem in Graphs", *Networks* 14 (1984) 147 - 159.
- [5] J. E. Beasley, "An SST Algorithm for the Steiner Problem in Graphs", *Networks* 19 (1989) 1 - 16.
- [6] B. Bollobás, *Graph Theory: an Introductory Course*, Graduate Texts in Mathematics 63, Springer Verlag, New York (1979).
- [7] J.A. Bondy e U.S.R. Murty, *Graph Theory with Applications*, Macmillan, Londres (1976).
- [8] E. J. Cockayne, "On the Steiner Problem", *Canadian Math. Bulletin* 10, (1967) 431 - 450.
- [9] R. Courant e H. Robbins, *What's Mathematics ?*, Oxford University Press. 16^a Edição (1977).
- [10] E. W. Dijkstra, "A Note on Two Problems in Connexion with Graphs", *Numerical Math.* 1 (1959) 269 - 271.
- [11] S.E. Dreyfus e R.A. Wagner, "The Steiner Problem in Graphs", *Networks* 1 (1971) 195 - 207.
- [12] C.W. Duin e A. Volgenant, "Some Generalizations of the Steiner Problem in Graphs", *Networks* 17 (1987) 353 - 364.
- [13] C.W. Duin e A. Volgenant, "Reduction Tests for the Steiner Problem in Graphs", *Networks* 19 (1989) 549 - 567.

- [14] M. Farber, "Characterization of Strongly Chordal Graphs", *Discrete Math* 43 (1983) 173 - 189.
- [15] A. M. Farley, "Networks Immunes to Isolated Failures", *Networks* 11 (1981) 255 - 268.
- [16] R. W. Floyd, "Algorithm 97, Shortest Path", *Comm. ACM* 5 (1962) 345 .
- [17] L. R. Foulds e R. L. Graham, "The Steiner Problem in Phylogeny is $\mathcal{NP}$ -Complete", *Advances in Applied Mathematics* 3, (1982) 43 - 49.
- [18] H. Frank e I. T. Frisch, "Network Analysis, Large Scale Networks: Theory and Design", eds. F. T. Boesch, *IEEE*, New York (1976).
- [19] M. R. Garey e D. S. Johnson, *Computers and Intractability: A Guide to the Theory of $\mathcal{NP}$ -Completeness*, Freeman, San Francisco (1979).
- [20] M. R. Garey e D. S. Johnson, "The Rectilinear Steiner Problem is $\mathcal{NP}$ -Complete", *SIAM Journal Applied Math* 32 (1977) 824 - 834.
- [21] R. E. Gomory e T. C. Hu, "Multi-terminal Network Flows", *SIAM Journal Applied Math.* 9, (1961) 551 - 570.
- [22] M. Gondram e M. Minoux, *Graphs and Algorithms*, John Wiley New York (1984).
- [23] M. Grötschel e C. L. Monma, "Integer Polyhedra Arising from Certain Network Design Problems with Connectivity Constraints", *Report No. 104*, Anwendungsbezogene Optimierung und Steuerung, Universität Augsburg (1988).
- [24] M. Grötschel, *Approaches to Hard Combinatorial Optimization Problems*, in B. Korte (ed), *Modern Applied Mathematics: Optimization and Operations Research*, North-Holland, Amsterdam, (1982), 438 - 515.
- [25] M. Grötschel, L. Lovász e A. Schrijver, "The Ellipsoid Method and its Consequences in Combinatorial Optimization", *Combinatorica* 1 (1981) 169 - 197.
- [26] M. Grötschel, M. Jünger e G. Reinelt, "A Cutting Plane Algorithm for the Linear Ordering Problem", *Operations Research* Volume 32 No 6, (1984) 1195 - 1220.
- [27] M. Grötschel e Y. Wakabayashi, "A Cutting Plane Algorithm for a Clustering Problem", *Mathematical Programming B*, Volume 45 No 1 (1989) 59 - 96.
- [28] S.L. Hakimi, "Steiner's Problem in Graphs and its Implications", *Networks* 1 (1971) 113 - 133.
- [29] R. Halin, "Studies on Minimally n-Connected Graphs", in *Combinatorial Mathematics and its Applications*, Academic Press, NY (1971) 129 - 136.
- [30] M. Hanan, "On Steiner's Problem with Rectilinear Distance", *SIAM Journal on Applied Mathematics*, Volume 14, No. 2, (1966) 255 - 265.

- [31] R.M. Karp, "Reducibility among Combinatorial Problems", in Miller and Thatcher (Eds.), *Complexity of Computer Computations*, Plenum Press, New York (1972) 85 - 103.
- [32] L. Kou, G. Markowsky e L. Berman, "A Fast Algorithm for Steiner Trees", *Acta Informatica* 15 (1981) 141 - 145.
- [33] J.B. Kruskal, "On the Shortest Spanning Tree of a Graph and the Travelling Salesman Problem", *Proc. Amer. Math. Soc.* 7 (1956) 48 - 50.
- [34] E.L. Lawler, *Combinatorial Optimization: Networks and Matroids*, Holt, Rinehart and Winston, New York (1976).
- [35] D. H. Lee, "Low Cost Drainage Networks", *Networks* 6, (1976) 351 - 371.
- [36] N. Maculan, "The Steiner Problem in Graphs", *Annals of Discrete Math.* 31 (1987).
- [37] N. Maculan, "O Problema de Steiner em Grafos Orientados", *Proceedings of the Latin American Conference on Operations Research*, Buenos Aires (1984).
- [38] R. Marsten, "The Design of the XMP Linear Programming Library", *Transactions on Mathematical Software*, Vol. 7, No. 4 (1981).
- [39] Z. A. Melzak, "On the Problem of Steiner", *Canadian Math. Bull.*, Vol. 4, (1961) 143 - 148.
- [40] M. Minoux, *Mathematical Programming : Theory and Algorithms*, John Wiley and Sons, New York (1986).
- [41] A. Motro, "The Design of FLEX : A Tolerant and Cooperative User Interface to Databases", *Cognitive Engineering in the Design of Human Interaction and Expert Systems*.
- [42] L. Nastansky, S. M. Selkow e N. F. Stewart, "Cost Minimal Trees in Directed Acyclic Graphs", *Zeitschrift für Operations Research* 18 (1974) 59 - 67.
- [43] M. W. Padberg e M. Grötschel, "Polyhedral Aspects of the Travelling Salesman Problem II: Computations", in E. L. Lawler, J. K. Lenstra and A. H. G. Rinnooy Kan (eds.), *The Travelling Salesman Problem*, John Wiley and Sons, London/New York (1985).
- [44] M. W. Padberg e G. Rinaldi, "Optimization of a 532-City Symmetric Travelling Salesman Problem by Branch and Cut", *Operations Research Letters* 6 (1987) 1 - 7.
- [45] C. H. Papadimitriou e K. Steiglitz, *Combinatorial Optimization: Algorithms and Complexity*, Prentice Hall, Englewood Cliffs, New Jersey (1982).
- [46] J. Plesnik, "A Bound for the Steiner Tree Problem in Graphs", *Math. Slovaca* 31, (1981), num. 2, 155 - 163.

- [47] R. C. Prim, "Shortest Connection Networks and Some Generalizations", *Bell System Tech. Journal* 36 (1957) 1389 - 1401.
- [48] D. J. Rosenkrantz, R. E. Stearns e P. M. Lewis II, "An Analysis of Several Heuristics for the Travelling Salesman Problem", *SIAM Journal of Computing* (a ser publicado).
- [49] A. Rothfarb, "Optimal Design of Offshore Natural Gas Pipeline Systems", *ORSA* 18, 6, (1970) 992 - 1010.
- [50] A. Segev, "The Node-Weighted Steiner Tree Problem", *Networks* 17 (1987) 1 - 17.
- [51] A. Schrijver, *Theory of Linear and Integer Programming*, John Wiley, New York (1986).
- [52] M.L. Shore, L.R. Foulds e P.B. Gibbons, "An Algorithm for the Steiner Problem in Graphs", *Networks* 12 (1982) 323 - 333.
- [53] C. C. de Souza, "O Problema de Steiner na Métrica Retilinear: Propriedades, Novas Heurísticas e Estudo Computacional", *Dissertação de Mestrado*, PUC Rio de Janeiro (1989).
- [54] H. Takahashi e A. Matsuyama, "An Approximate Solution for the Steiner Problem in Graphs", *Math. Japonica* 24 (1980) 573 - 577.
- [55] J.A. Wald e C.J. Colbourn, "Steiner Trees, Partial 2-Trees and Minimum IFI Networks", *Networks* 13 (1983) 159 - 167.
- [56] K. White, M. Farber e W. Pulleyblank, "Steiner Trees, Connected Domination and Strongly Chordal Graphs", *Networks* 15 (1985) 109 - 124.
- [57] P. Winter, "Steiner Problem in Networks: a Survey", *Networks* 17 (1987) 129 - 167.
- [58] P. Winter, "Steiner Problem in Halin Networks", *Discrete Applied Math.* 17 (1987) 281 - 294.
- [59] P. Winter, "An Algorithm for the Steiner Problem in the Euclidean Plane", *Networks* 15 (1985) 323 - 345.
- [60] R. T. Wong, "A Dual Ascent Approach for Steiner Tree Problems on a Directed Graph", *Math. Programming* 28 (1984) 271 - 287.
- [61] Guia Quatro Rodas Brasil. Ed. Abril (1986).