PROBLEMA DE STEINER EM GRAFOS: UMA EXPERIÊNCIA NUMÉRICA PARA PROBLEMAS DE MÉDIO PORTE UTILIZANDO FORMULAÇÕES COMPACTAS DE MULTI-FLUXO

Arthur Besso

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

Orientador: Nelson Maculan Filho

Rio de Janeiro
Setembro de 2015

PROBLEMA DE STEINER EM GRAFOS: UMA EXPERIÊNCIA NUMÉRICA PARA PROBLEMAS DE MÉDIO PORTE UTILIZANDO FORMULAÇÕES COMPACTAS DE MULTI-FLUXO

Arthur Besso

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

Examinada por:

_____

Prof. Nelson Maculan Filho, D.Habil.

_____

Profa. Laura Silvia Bahiense da Silva Leite, D.Sc.

_____

Prof. Thibaut Victor Gaston Vidal, D.Sc.

RIO DE JANEIRO, RJ - BRASIL

SETEMBRO DE 2015

Besso, Arthur

Problema de Steiner em Grafos: uma Experiência Numérica para Problemas de Médio Porte Utilizando Formulações Compactas de Multi-Fluxo/ Arthur Besso. – Rio de Janeiro: UFRJ/COPPE, 2015.

VI, 36 p.: il.; 29,7 cm.

Orientador: Nelson Maculan Filho

Dissertação (mestrado) – UFRJ/ COPPE/ Programa de Engenharia de Sistemas e Computação, 2015.

Referências Bibliográficas: p. 35-36.

1. Steiner. 2. Multi-Fluxos. 3. Relaxação Linear. I. Maculan, Nelson. II. Universidade Federal do Rio de Janeiro, COPPE, Programa de Engenharia de Sistemas e Computação. III. Título.

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

PROBLEMA DE STEINER EM GRAFOS: UMA EXPERIÊNCIA NUMÉRICA PARA PROBLEMAS DE MÉDIO PORTE UTILIZANDO FORMULAÇÕES COMPACTAS DE MULTI-FLUXO

Arthur Besso

Setembro/2015

Orientador: Nelson Maculan Filho

Programa: Engenharia de Sistemas e Computação

Este trabalho estuda o Problema de Steiner em Grafos, o qual é conhecidamente NP-difícil. No entanto, sua relaxação linear tem complexidade polinomial. Começamos definindo o problema e suas principais características. Apresentamos alguns métodos para resolver o problema já existentes na literatura. Em seguida, mostramos três formulações para o problema usando multi-fluxo, comparando-as antes de selecionar aquela que mais se aproxima do problema original. Realizamos alguns testes em diferentes instâncias para verificar a proximidade entre a relaxação e o problema original.

Abstract of Dissertation presented to COPPE/UFRJ as a partial fulfillment of the requirements for the degree of Master of Science (M.Sc.)

STEINER PROBLEM IN GRAPHS: A NUMERIC EXPERIENCE FOR MEDIUM-SIZE PROBLEMS USING MULTI-FLOW COMPACT FORMULATIONS

Arthur Besso

September/2015

Advisor: Nelson Maculan Filho

Department: Systems and Computer Engineering

This work studies the Steiner Tree Problem, which is known to be NP-hard. However, its linear relaxation has polynomial complexity. We begin by defining the problem and its major characteristics. We introduce some methods to solve the problem that already exist in literature. Then we show three formulations for the problem using multi-flow, comparing them before selecting the one that gets the closest to the original problem. We do some tests in different instances in order to check the closeness between the relaxation and the original problem.

SUMÁRIO

1 Introdução

A programação inteira ou programação linear inteira mista, também chamada de otimização combinatória, é empregada para modelar e resolver grande parte dos problemas de logística envolvendo variáveis discretas. Um dos problemas mais conhecidos e estudados nessa área é o Problema de Steiner em Grafos.

Um grafo é um conjunto de vértices e arestas (que ligam pares de vértices). Dado um grafo conexo (existe caminho entre cada par de vértices), sem laços ou arestas múltiplas e não-direcionado, o Problema de Steiner em Grafos se propõe a encontrar o conjunto de arestas de custo mínimo que conecta um determinado subconjunto de vértices do grafo.

Esse problema, computacionalmente, não é tão simples de se resolver. Sua versão de decisão é NP-difícil. No entanto, a sua relaxação (na versão de decisão) possui complexidade polinomial.

Começamos definindo o problema mais precisamente. Em seguida, apresentamos os principais modelos existentes na literatura. Depois, escolhemos um modelo com três variações para trabalhar. Após compararmos essas variações e as suas respectivas relaxações lineares, optamos por uma delas, cujo desempenho numérico vamos testar antes de, finalmente, avaliarmos os resultados.

Não estamos interessados na eficiência do modelo. O que queremos mostrar é que, para a formulação que vamos utilizar, a relaxação linear é muito próxima ao problema original.

A grande vantagem deste trabalho está na aplicação de uma técnica simples que produz bons resultados para instâncias de médio porte, como será visto mais adiante.

2 Definição do Problema

O Problema de Steiner surgiu a partir do Problema de Fermat, que consiste em, dado um triângulo, encontrar o ponto cuja soma das distâncias aos vértices é mínima. Esse problema foi generalizado para qualquer figura em qualquer dimensão e hoje é conhecido como Problema de Steiner Euclidiano. Posteriormente, formulou-se uma versão em grafos.

O Problema de Steiner em Grafos (em inglês, *Steiner Tree Problem* - STP) consiste em, dado um grafo $G = (V, E)$, onde V é o conjunto de vértices e E é o conjunto de arestas, e um subconjunto $S \subseteq V$ de vértices desse grafo, encontrar um subgrafo conexo $G' = (V', E')$ de custo mínimo que contenha esses vértices.

O problema pode ser mais formalmente definido da seguinte forma:

Dados: Grafo $G = (V, E)$, Custos $C(E)$, $S \subseteq V$;

Objetivo: $C(E')$;

Sujeito a: $G' \subseteq G, S \subseteq V'$.

Os vértices de S são chamados de pontos obrigatórios e os vértices de $V - S$ são opcionais. Os elementos de $V' - S$, ou seja, os vértices opcionais que entram na solução, são conhecidos como Pontos de Steiner. O grafo-solução é denominado Árvore de Steiner, já que a formação de ciclos piora a função objetivo.

Algumas variações do Problema de Steiner em Grafos consideram custos para os vértices (LI et al, 2009), ou então atribuem prêmios ao passar por determinadas arestas (custos negativos), entre outras inúmeras generalizações, muitas delas apresentadas por

HAUPTMANN e KARPINSKI (2015). Neste trabalho, consideramos apenas os custos das arestas, o qual é estritamente positivo.

Esse problema é conhecidamente classificado dentro da otimização combinatória como NP-difícil, como demonstraram GAREY, GRAHAM e JOHNSON (1977), e sua versão de decisão, mostrada abaixo, é NP-completa.

Versão de decisão do Problema de Steiner em Grafos:

Dados: Grafo $G = (V, E)$, Custos $C(E)$, $S \subseteq V$, constante b;

Questão: Existe um subgrafo de G que contém S e possui custo total no máximo b?

Isso significa que, no pior caso, o tempo de resolução é exponencialmente proporcional ao tamanho dos dados de entrada.

2.1 Casos Particulares

Existem algumas instâncias bem simples de verificar que o problema pode ser resolvido em tempo polinomial:

1) $|S| = 1$, solução trivial = S;
2) $|S| = 2$, problema do caminho mínimo (SPP);
3) $|S| = |V|$, problema da árvore geradora mínima (MSTP);
4) $|S| = |V| - k$, onde k é constante, bastando resolver o MSTP para cada uma das 2^k combinações de $G - T$, sendo T um subconjunto dos k vértices.

2.2 Aplicações

O STP tem aplicações em diversas áreas, com destaque para telecomunicações, abastecimento, transporte e localização. Pode-se observar que, para se melhor projetar e utilizar redes, a modelagem de problemas nessa forma é bastante empregada. GUANGHUA e ZHANJIANG (2010) resolveram o problema para determinar a localização de um centro de distribuição, enquanto BURDAKOV, DOHERTY e KVARNSTRÖM (2014) aplicaram o STP para sistemas de monitoramento e vigilância.

Muitas vezes, contudo, o modelo empregado não se baseia no STP original, mas em uma variação do mesmo. Por exemplo, quando se pensa em segurança de redes, são incluídas algumas restrições como:

- cada par de pontos deve ter pelo menos 2 caminhos entre si;
- nenhum vértice do grafo-solução pode ter grau muito alto.

Mesmo assim, a solução do STP original pode servir como ponto de partida para resolver o problema modificado em vez de desenvolver algoritmos ou heurísticas próprias para cada caso. Há situações, ainda, em que se aceita resultados não tão próximos do ótimo ou em que se fixa um custo máximo em função do tamanho do grafo, para as quais se basear no STP também pode ser vantajoso.

3 Algumas Formulações do Problema

Muito já se estudou sobre o Problema de Steiner em Grafos, entre algoritmos e heurísticas. A principal dificuldade na hora de se formular um modelo é a dicotomia entre a precisão dos resultados e o tempo computacional de resolução. Quanto mais próximo do valor ótimo se deseja chegar, maior o tempo gasto para resolver a formulação.

HWANG, RICHARDS e WINTER (1992) compilaram os melhores métodos da literatura para resolver o STP. Estes incluem desde reduções e heurísticas até algoritmos exatos. Para se ter uma visão mais completa do problema, recomenda-se a consulta desse livro, intitulado *The Steiner Tree Problem*.

Em se tratando de instâncias mais complexas do problema, uma alternativa seria aplicar reduções ao grafo dado. Esse processo consiste em identificar subgrafos em que a presença ou ausência de determinados vértices não obrigatórios ou arestas na solução pode ser dada como certa. Um exemplo bem simples seria uma ponte entre dois vértices obrigatórios. Esse tipo de análise, em princípio, reduz a complexidade do problema - diminuindo a quantidade de variáveis e restrições - porém, quando se investe muito nessa parte, o tempo computacional gasto pode superar o tempo de resolução sem aplicar essas reduções.

Em relação às heurísticas, elas geralmente começam com uma árvore contendo todo o conjunto de vértices obrigatórios. Em seguida, busca-se encontrar árvores com custo cada vez menor. ROBINS e ZELIKOVSKY (2005) desenvolveram uma heurística que resolve o problema em tempo polinomial com um fator de aproximação de 1,55. Quando não há vértices não-obrigatórios que sejam adjacentes, esse fator cai para 1,28.

Um algoritmo exato tido como melhor para muitos autores foi formulado por DREYFUS e WAGNER (1971) e por LEVIN (1971), os quais resolveram o STP por

programação dinâmica. Os resultados são tão expressivos que a complexidade do algoritmo é polinomial no tamanho do grafo, sendo exponencial apenas na quantidade de nós obrigatórios.

Aqui vamos apresentar brevemente alguns dos principais modelos. Foram escolhidas formulações mais compactas, mesmo que a resolução seja mais complexa. Para uma relação mais completa de outras formulações e uma análise mais detalhada, o livro citado acima é uma boa sugestão. Outros trabalhos (MACULAN, 1987, e WINTER, 1987) também estudam diferentes formulações do problema.

Algumas notações importantes que aparecem nos modelos são:

$G = (V, E)$ onde G é o grafo dado e V e E são, respectivamente, seus conjuntos de vértices e arestas;

$G_D = (V, A)$ é o grafo direcionado induzido por G;

$A = \{(i,j), (j,i) \mid \{i,j\} \in E\}$ o conjunto de arcos de G_d;

$G' = (V', E')$ é o grafo correspondente à solução ótima do problema, onde V' é o conjunto de vértices e E' é o conjunto de arestas de custo total mínimo

y_{ij}= variável de decisão do problema, binária: $\begin{cases} y_{ij} = 1, se\ (i,j)\ \in E' \\ y_{ij} = 0, se\ (i,j) \end{cases}$;

z_{ij}^{k} =quantidade de fluxo que passa no arco $(i,j) \in A$ tendo o vértice v_1 como origem e v_k como destino;

c_{ij} =valor do custo da aresta $\{i,j\}$;

$C_{ij} = c_{ij} y_{ij}$ =contribuição da aresta $\{i,j\}$ no custo total;

Z = custo total ou valor da função objetivo;

$\Gamma^{+}(i) = \{j \in V | (i,j) \in A\}$;

$\Gamma^{-}(i) = \{j \in V | (j,i) \in A\}$.

3.1 Formulação com Restrição de Grau

Esta formulação foi proposta por BEASLEY (1989).

Dado um grafo $G = (V, E)$, o custo c_{ij} de cada aresta e o conjunto de vértices obrigatórios $S \subseteq V$, existe um algoritmo que resolve o problema da seguinte forma:

- Adicionar um novo vértice (v_0) ao grafo G;
- Para cada vértice não obrigatório, adicionar uma aresta de custo zero ligando esse vértice a v_0;
- Adicionar uma aresta entre v_0 e v_1, sendo que $v_1 \in S$;
- Encontrar a árvore geradora mínima do grafo resultante $G' = (V, E')$ com a condição de que todo vértice não obrigatório que esteja ligado a v_0 tenha grau 1;
- Formular o problema da seguinte forma:

$$Min \sum_{\{i,j\} \in E'} c_{ij}\, y_{ij} \tag{1}$$

Sujeito a:

$$\{\{i,j\} \mid y_{ij} = 1\} \text{ forma uma árvore geradora de G'} \tag{2}$$

$$y_{0i} + y_{pq} \leq 1, (p,q) \in T_i, i \in V - S \tag{3}$$

$$y_{ij} \in \{0,1\}, \{i,j\} \in E' \tag{4}$$

$$\text{onde } T_i = \{(p,q) \mid p = i \; ou \; q = i\} \tag{5}$$

A expressão (1) é a função objetivo. A restrição (2) garante que toda aresta pertencente ao grafo resultante seja um dado de entrada do problema da árvore geradora mínima. (3) e (5) asseguram que todo vértice não obrigatório ligado a v_0 tenha grau 1. (4) é a restrição de binariedade.

Mesmo que a verificação da formação de uma árvore geradora é feita em tempo polinomial, realizar essa operação para cada solução potencialmente viável pode ter uma demora considerável.

3.2 Formulação de Fluxo Acumulado

Proposta por ARPIN (1984), esta formulação é mais compacta que a anterior por envolver operações mais simples nas restrições.

Considerando que um grafo com $|S|$ pontos obrigatórios requer $|S|$ unidades de fluxo, uma para cada elemento de S. Seja z_{ij} a quantidade de fluxo no arco (i,j). O problema pode ser enunciado assim:

$$Min \sum_{i,j \in A} c_{ij}\, y_{ij} \tag{6}$$

Sujeito a:

$$\sum_{j \in \Gamma^{+}(i)} z_{ij} - \sum_{j \in \Gamma^{-}(i)} z_{ij} = 0, i \in V - S \tag{7}$$

$$\sum_{j \in \Gamma^{+}(k)} z_{kj} - \sum_{j \in \Gamma^{-}(k)} z_{jk} = -1\,, k \in S \tag{8}$$

$$0 \leq z_{ij} \leq |S|\, y_{ij}\,, (i,j) \in A \tag{9}$$

$$y_{ij} \in \{0,1\}, (i,j) \in A \tag{10}$$

Ordenando aleatoriamente os vértices, começamos com $|S|$ unidades de fluxo, descontando uma unidade para cada vértice obrigatório que encontramos. Assim, o fluxo resultante em cada vértice obrigatório é -1 (8) e 0 em cada vértice não obrigatório (7). (9) é a restrição de capacidade em cada arco e (10) a restrição de binariedade.

3.3 Formulação de Cobrimento de Cortes

Proposta por ANEJA (1980), esta formulação, aparentemente, tem uma complexidade maior que as anteriores, uma vez que uma das suas restrições tem tamanho exponencial.

Sendo S o conjunto de pontos obrigatórios e $v_1$um elemento de S, qualquer corte $C = (W, \overline{W})$ de G que tiver pelo menos um vértice obrigatório em cada bipartição deve ter pelo menos uma aresta conectando as duas partições que apareça na solução.

$$Min \sum_{(i,j)\in A} c_{ij}\, y_{ij} \tag{11}$$

Sujeito a:

$$\sum_{(i,j)\in C} \; y_{ij} \geq 1, \forall C = \{W, \overline{W}\}, v_1 \in W,\; S \cap \overline{W} \neq \emptyset \tag{12}$$

$$y_{ij} \in \{0,1\}, (i,j) \in A \tag{13}$$

A restrição (12) garante que, se cada bipartição tiver ao menos um vértice obrigatório, pelo menos uma aresta atravesse o corte. Isso é válido para qualquer corte. (13) garante a binariedade.

Está claro que essa formulação possui uma complexidade exponencial, visto que teríamos que calcular todos os cortes possíveis no grafo dado.

4 O Modelo de Fluxos Não Simultâneos

Agora vamos apresentar o modelo que será estudado mais profundamente aqui e sobre o qual vamos testar os resultados. Esse modelo foi proposto por MACULAN, PLATEAU e LISSER (2003), baseado em um trabalho anterior de CLAUS e MACULAN (1983), e tem como principal característica o conceito de fluxos.

As variáveis de decisão do modelo são *y* e *z*, onde *y* (binária) está associada à presença de uma certa aresta no grafo-solução e *z* representa o fluxo real em um dado arco. Esse fluxo está limitado à capacidade de cada aresta (0 ou 1, dependendo se a aresta pertence à solução). Para cada vértice de *S*, exceto o primeiro, existe uma configuração de fluxos para cada arco. Usaremos aqui as mesmas notações apresentadas na seção anterior.

Portanto, temos que:

$$y_{ij} = 1, se\ \{i,j\} \in E' \tag{14}$$

$$y_{ij} = 0, se\ \{i,j\} \notin E' \tag{15}$$

$$0 \leq z_{ij}^{k} \leq y_{ij}, \{i,j\} \in E, k \in S - \{v_1\}, \{v_1\} \in S \tag{16}$$

$$0 \leq z_{ji}^{k} \leq y_{ij}, \{i,j\} \in E, k \in S - \{v_1\}, \{v_1\} \in S \tag{17}$$

O único parâmetro utilizado no modelo é o custo *c* referente a cada aresta do grafo dado: c_{ij} é o custo de incluir a aresta $\{i,j\}$ na solução.

A função objetivo do problema é minimizar o custo do grafo *G'*. Em termos dos parâmetros e variáveis acima:

$$Min \sum c_{ij}\, y_{ij}, \{i,j\} \in E \tag{18}$$

A restrição de conexidade pode ser construída com base nos fluxos. Cada par de vértices de *S* deve ter um caminho que os liga. Para simplificar, pegamos um vértice qualquer (v_1) de

S. Se exigirmos que v_1 esteja ligado a todos os $|S| - 1$ demais vértices obrigatórios, garantimos pelo menos um caminho de v_i a v_j para todo $\{i,j\} \subseteq S,\ i \neq j,\ i,j \neq 1$, o qual passa por v_1.

Para cada um desses caminhos entre v_1 e v_k, ambos os fluxos resultantes (soma dos fluxos de todos os arcos adjacentes) de saída de v_1 (20) e de entrada em v_k (22) deverão ser iguais a 1. Além disso, os fluxos resultantes dentro desses caminhos de vértices intermediários deverão ser iguais a 0 (21), garantindo que o mesmo fluxo que sai de v_1 percorra todos esses vértices até chegar a v_k.

Assim, a modelagem do problema fica da seguinte forma:

(P1):

$$Min \sum c_{ij} y_{ij}\ , \{i,j\} \in E \tag{19}$$

Sujeito a:

$$\sum_{j \in \Gamma^+(1)} z_{1j}^k - \sum_{j \in \Gamma^-(1)} z_{j1}^k = 1 \tag{20}$$

$$\sum_{j \in \Gamma^+(i)} z_{ij}^k - \sum_{j \in \Gamma^-(i)} z_{ji}^k = 0, v_i \in V - \{v_1, v_k\} \tag{21}$$

$$\sum_{j \in \Gamma^+(k)} z_{kj}^k - \sum_{j \in \Gamma^-(k)} z_{k1}^k = -1 \tag{22}$$

$$z_{ij}^k \leq y_{ij} \text{ e } z_{ji}^k \leq y_{ij}, \{i,j\} \in E \tag{23}$$

$$z_{ij}^k \geq 0, (i,j) \in A \tag{24}$$

$$y_{ij} \in \{0,1\}, \{i,j\} \in E \tag{25}$$

$$v_1, v_k \in S, k \neq 1 \tag{26}$$

A restrição (23) limita a capacidade do fluxo em cada arco: só pode haver fluxo positivo se o arco estiver na solução do problema, ou seja, se ele contribuir na função objetivo. (24)

garante que o fluxo seja não-negativo e (25) assegura a binariedade. (26) especifica que, nas restrições de fluxo, os vértices de origem e destino sejam obrigatórios.

Pode-se perceber, nas restrições (20), (21) e (22), que uma é combinação linear das outras duas. Apesar de podermos descartar uma das três, manteremos assim para fins didáticos.

4.1 Variações do Modelo de Fluxos Não Simultâneos

A formulação acima considera um grafo não orientado. Porém, uma vez que introduzimos o conceito de fluxos, devemos atentar para o fato que z_{ij}^ke z_{ji}^k não são sinônimos, logo tivemos que definir para cada aresta $\{i,j\}$ de E um par de arcos opostos tais que z_{ij}^k é o fluxo que passa em (i,j) e z_{ji}^k o fluxo que passa em (j,i). Ainda assim, o valor de y_{ij} era o mesmo para (i,j) e (j,i).

Contudo, se considerarmos que cada arco tem um custo de utilização independente do arco oposto, mesmo que o valor seja o mesmo, podemos obter uma nova formulação para o problema. Substituindo as restrições $z_{ij}^k \leq y_{ij}$ e $z_{ji}^k \leq y_{ij}, \{i,j\} \in E$ por $z_{ij}^k \leq y_{ij}, (i,j) \in A$, o problema passa a ser formulado assim:

(P2):

$$Min \sum c_{ij}(y_{ij} + y_{ji}), \{i,j\} \in E \tag{27}$$

Sujeito a:

$$\sum_{j \in \Gamma^+(1)} z_{1j}^k - \sum_{j \in \Gamma^-(1)} z_{j1}^k = 1 \tag{28}$$

$$\sum_{j \in \Gamma^+(i)} z_{ij}^k - \sum_{j \in \Gamma^-(i)} z_{ji}^k = 0, v_i \in V - \{v_1, v_k\} \tag{29}$$

$$\sum_{j\in\Gamma^{+}(k)} z_{kj}^{k} - \sum_{j\in\Gamma^{-}(k)} z_{k1}^{k} = -1 \tag{30}$$

$$z_{ij}^{k} \leq y_{ij}, (i,j) \in A \tag{31}$$

$$z_{ij}^{k} \geq 0, (i,j) \in A \tag{32}$$

$$y_{ij} \in \{0,1\}, (i,j) \in A \tag{33}$$

$$v_1, v_k \in S, k \neq 1 \tag{34}$$

Essas duas formulações lidam com os fluxos de formas diferentes. Enquanto a primeira considera que o custo de utilizar uma aresta gera a possibilidade de percorrê-la nos dois sentidos devido ao pressuposto da não-simultaneidade, a segunda implica que para utilizar arcos opostos precisamos de um vetor de custo distinto para cada um, como mostra a figura abaixo:

Figura 4.1: Diferença conceitual entre P1 e P2

Uma terceira forma de enxergar a relação entre os fluxos e os arcos seria pensar que os fluxos são cumulativos, ou seja, a utilização simultânea de fluxos opostos deve respeitar a restrição de que a soma dos mesmos não pode exceder a capacidade do arco. Assim, a relação entre o vetor capacidade e os fluxos seria $z_{ij}^{k} + z_{ji}^{k} \leq y_{ij}, \{i,j\} \in E$ (39) e o problema ficaria

(P3):

$$Min \sum c_{ij} y_{ij}, \{i,j\} \in E \tag{35}$$

Sujeito a:

$$\sum_{j \in \Gamma^+(1)} z_{1j}^k - \sum_{j \in \Gamma^-(1)} z_{j1}^k = 1 \tag{36}$$

$$\sum_{j \in \Gamma^+(i)} z_{ij}^k - \sum_{j \in \Gamma^-(i)} z_{ji}^k = 0, v_i \in V - \{v_1, v_k\} \tag{37}$$

$$\sum_{j \in \Gamma^+(k)} z_{kj}^k - \sum_{j \in \Gamma^-(k)} z_{k1}^k = -1 \tag{38}$$

$$z_{ij}^k + z_{ji}^k \leq y_{ij}, \{i,j\} \in E \tag{39}$$

$$z_{ij}^k \geq 0, (i,j) \in A \tag{40}$$

$$y_{ij} \in \{0,1\}, \{i,j\} \in E \tag{41}$$

$$v_1, v_k \in S, k \neq 1 \tag{42}$$

Muitos autores fazem uma distinção entre o Problema de Steiner em Grafos Orientados e Não-Orientados. Este modelo assume, inicialmente, um grafo não-orientado. Nas duas últimas formulações, utilizamos o artifício de orientar o grafo, porém a entrada do problema continua não-orientada. Para um grafo orientado, um bom modelo de um algoritmo dual foi proposto por WONG (1984).

4.2 Comparação entre as formulações

Pelas restrições de fluxo, para cada vértice obrigatório (exceto v_1), o fluxo total que sai de v_1 deve ser igual a 1. Se esse fluxo se dividir em mais de um arco, cada arco terá seu custo computado na função objetivo. Como y_{1j} é binária, esse custo independe da quantidade de fluxo que passa no arco. Estendendo esse raciocínio para todos os vértices de cada caminho

(v_1, v_k), concluímos que a configuração de fluxos ilustrada abaixo (figura 4.1) nunca entrará na solução ótima. Portanto, as variáveis z sempre serão binárias.

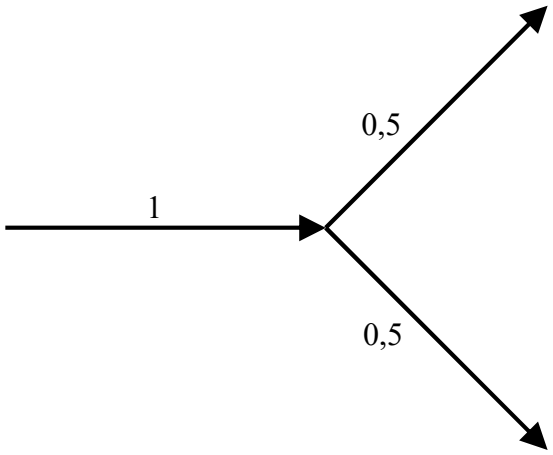

Figura 4.1: Divisão de fluxo impossível em um nó

Verificando a restrição que diferencia P1, P2 e P3, que são, respectivamente, (23), (31) e (39), temos três possibilidades válidas para os valores de z_{ij}^k e z_{ji}^k. Vamos analisar os valores de y_{ij} e y_{ji} em cada caso. Esse raciocínio é válido para cada aresta $\{i, j\} \in E$. Assumimos que o k utilizado é aquele que produz os máximos valores de z.

- $z_{ij}^k = 0$ e $z_{ji}^k = 0$:

  $y_{ij} = 0$ (P1, P2 e P3) e $y_{ji} = 0$ (P2);

- $z_{ij}^k = 1$ e $z_{ji}^k = 0$:

  $y_{ij} = 1$ (P1, P2 e P3) e $y_{ji} = 0$ (P2);

- $z_{ij}^k = 0$ e $z_{ji}^k = 1$:

  $y_{ij} = 1$ (P1 e P3), $y_{ij} = 0$ (P2) e $y_{ji} = 1$ (P2);

- $z_{ij}^k = 1$ e $z_{ji}^k = 1$:

  Este caso não existe (P1, P2 e P3), como prova o teorema abaixo.

Teorema: Sejam v_p e v_q dois vértices obrigatórios de $G, 1 < p \leq |S|, 1 < q \leq |S|$ e $p \neq q$. Então, na solução ótima do problema, não existe uma aresta $\{i, j\}$ tal que $z_{ij}^p > 0$ e $z_{ji}^q > 0$, considerando os caminhos direcionados (v_1, v_p) e (v_1, v_q).

Prova: (Caminho mínimo) Suponha, por contradição, que, na solução ótima, $z_{ij}^p > 0$ e $z_{ji}^q > 0$. Observando o caminho (v_1, v_p), deduzimos que o custo de (v_1, v_i) é menor que o custo de (v_1, v_j) pois cada arco acrescido aumenta o custo acumulado do trajeto. Por outro lado, observando o caminho (v_1, v_q), deduzimos que o custo de (v_1, v_j) é menor que o custo de (v_1, v_i), uma contradição.

A partir da análise acima, concluímos que os valores ótimos de y_{ij} em P1 e P3 são sempre os mesmos. Em P2, esses valores só podem ser diferentes quando $y_{ij} = 1$ em P1 e P3 e $y_{ij} = 0$ e $y_{ji} = 1$ em P2. Em todos esses casos, o valor ótimo da função objetivo é igual para todas as formulações: $Z(P1) = Z(P2) = Z(P3)$. Portanto, podemos dizer que as três formulações são equivalentes:

$$P1 = P2 = P3 \tag{43}$$

Vistas as três formulações, vamos estudar agora as suas relaxações lineares e realizar o mesmo tipo de comparação entre elas.

5 Relaxação Linear

Dado o modelo acima com as suas três variações, podemos explorar uma relaxação a fim de encontrar uma solução viável razoavelmente próxima da solução ótima em um tempo polinomial em relação ao tamanho dos dados de entrada (G, S e $c(E)$). Uma opção seria testar a relaxação lagrangeana, o que produz bons resultados (BEASLEY, 1984). No entanto, a relaxação linear é matematicamente mais simples, apesar de menos preferida de forma geral para esse tipo de problema. Como referência, ÖNCAN, ALTINEL e LAPORTE (2009) já realizaram esse teste, porém para o Problema do Caixeiro Viajante. Afinal, a proposta deste trabalho é justamente fazer a experiência de verificar a qualidade da relaxação linear como uma aproximação para o problema.

No modelo, temos dois tipos de variáveis: y e z, sendo que somente as variáveis do tipo y possuem restrição de integralidade ($y \in \{0,1\}$). Relaxar o problema significa, então, relaxar essa restrição, de modo que y seja tratada como contínua. Contudo, y continua sendo limitada pelos mesmos valores, de modo que $y \in [0,1]$.

Logo, o problema relaxado pode ser formulado assim:

(RL1):

$$Min \sum c_{ij} y_{ij} , \{i,j\} \in E \tag{44}$$

Sujeito a:

$$\sum_{j \in \Gamma^+(1)} z_{1j}^k - \sum_{j \in \Gamma^-(1)} z_{j1}^k = 1 \tag{45}$$

$$\sum_{j \in \Gamma^+(i)} z_{ij}^k - \sum_{j \in \Gamma^-(i)} z_{ji}^k = 0, v_i \in V - \{v_1, v_k\} \tag{44}$$

$$\sum_{j\in\Gamma^+(k)} z_{kj}^k - \sum_{j\in\Gamma^-(k)} z_{k1}^k = -1 \quad (45)$$

$$z_{ji}^k \leq y_{ij}, \{i,j\} \in E \quad (46)$$

$$z_{ij}^k \geq 0, (i,j) \in A \quad (47)$$

$$y_{ij} \in [0,1], \{i,j\} \in E \quad (48)$$

$$v_1, v_k \in S, k \neq 1 \quad (49)$$

A relaxação linear das outras variações do modelo são descritas a seguir:

(RL2):

$$Min \sum c_{ij}\,(y_{ij} + y_{ji}), \{i,j\} \in E \quad (50)$$

Sujeito a:

$$\sum_{j\in\Gamma^+(1)} z_{1j}^k - \sum_{j\in\Gamma^-(1)} z_{j1}^k = 1 \quad (51)$$

$$\sum_{j\in\Gamma^+(i)} z_{ij}^k - \sum_{j\in\Gamma^-(i)} z_{ji}^k = 0, v_i \in V - \{v_1, v_k\} \quad (52)$$

$$\sum_{j\in\Gamma^+(k)} z_{kj}^k - \sum_{j\in\Gamma^-(k)} z_{k1}^k = -1 \quad (53)$$

$$z_{ij}^k \leq y_{ij}, (i,j) \in A \quad (54)$$

$$z_{ij}^k \geq 0, (i,j) \in A \quad (55)$$

$$y_{ij} \in [0,1], (i,j) \in A \quad (56)$$

$$v_1, v_k \in S, k \neq 1 \quad (57)$$

(RL3):

$$Min \sum c_{ij} y_{ij}, \{i,j\} \in E \quad (58)$$

Sujeito a:

$$\sum_{j \in \Gamma^+(1)} z_{1j}^k - \sum_{j \in \Gamma^-(1)} z_{j1}^k = 1 \quad (59)$$

$$\sum_{j \in \Gamma^+(i)} z_{ij}^k - \sum_{j \in \Gamma^-(i)} z_{ji}^k = 0, v_i \in V - \{v_1, v_k\} \quad (60)$$

$$\sum_{j \in \Gamma^+(k)} z_{kj}^k - \sum_{j \in \Gamma^-(k)} z_{k1}^k = -1 \quad (61)$$

$$z_{ij}^k + z_{ji}^k \leq y_{ij}, \{i,j\} \in E \quad (62)$$

$$z_{ij}^k \geq 0, (i,j) \in A \quad (63)$$

$$y_{ij} \in [0,1], \{i,j\} \in E \quad (64)$$

$$v_1, v_k \in S, k \neq 1 \quad (65)$$

Como a relaxação contém o problema original, deduzimos que $Z(RL) \leq Z(P)$, independentemente da formulação utilizada.

A comparação utilizada na versão original do problema não se aplica na versão relaxada. Aqui podemos fazer as seguintes deduções, observando a desigualdade que diferencia as três formulações:

Comparando RL1 (48) e RL3 (64):

RL1: $z_{ij}^k \leq y_{ij}$

$z_{ji}^k \leq y_{ij}$

Somando: $z_{ij}^k + z_{ji}^k \leq 2y_{ij}$

RL3: $z_{ij}^k + z_{ji}^k \leq y_{ij}$

Observamos que RL1 é mais folgada que RL3, logo

$$Z(RL1) \leq Z(RL3) \tag{66}$$

Comparando RL2 (56) e RL3 (64):

RL2: $z_{ij}^k \leq y_{ij}$

$z_{ji}^k \leq y_{ji}$

Na solução ótima, $y_{ij} = \max(z_{ij}^k)$ e $y_{ji} = \max(z_{ji}^k)$, considerando a aresta $\{i, j\}$ fixa e k variando. A relação de igualdade provém do sentido da função objetivo. Nesse caso, $C_{ij} = c_{ij}(y_{ij} + y_{ji}) = c_{ij}\, max_k(z_{ij}^k) + c_{ij}\, max_k(z_{ji}^k)$.

RL3: $z_{ij}^k + z_{ji}^k \leq y_{ij}$

Na solução ótima, $\min(z_{ij}^k, z_{ji}^k) = 0$ para um mesmo k, ou seja, $y_{ij} = max_k(z_{ij}^k, z_{ji}^k)$, considerando a aresta $\{i, j\}$ fixa e k variando. Nesse caso, $C_{ij} = c_{ij} y_{ij} = c_{ij}\, max_k(z_{ij}^k, z_{ji}^k)$. Como $C_{ij}^{RL3} \leq C_{ij}^{RL2}$, conclui-se que

$$Z(RL3) \leq Z(RL2) \tag{67}$$

Logo, temos que

$$Z(RL1) \leq Z(RL3) \leq Z(RL2) \tag{70}$$

Para ilustrar, vamos mostrar um exemplo em que $Z(RL1) < Z(RL2)$:

Considere o grafo abaixo (figura 5.1) com seus respectivos custos relativos às arestas, sendo que todos os vértices são obrigatórios.

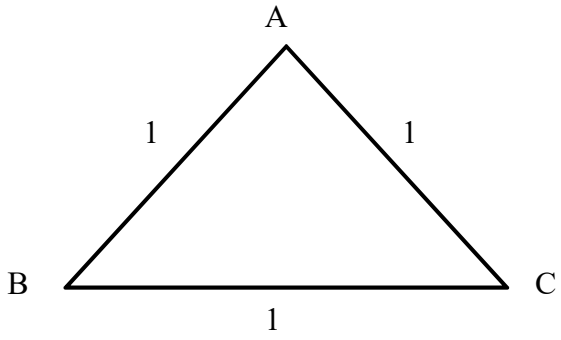

Figura 5.1: Exemplo de grafo simples

A solução ótima da versão original (inteira) do problema é qualquer subconjunto de tamanho 2 das arestas e o valor da função objetivo nesse caso é 2, independentemente da formulação escolhida, conforme ilustrado abaixo nas figuras 5.2, 5.3 e 5.4 (os números representam os valores de y).

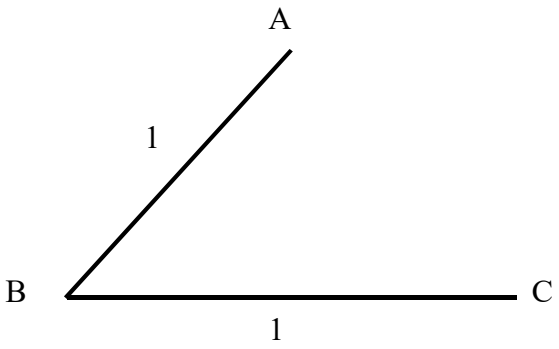

Figura 5.2: Solução ótima 1

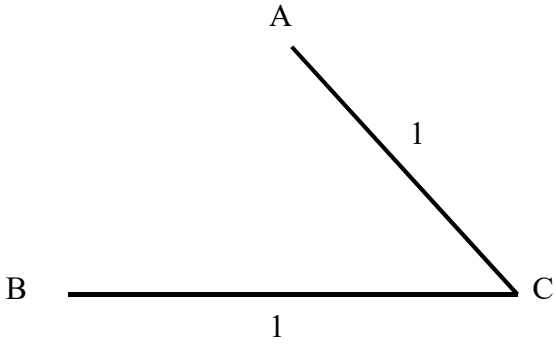

Figura 5.3: Solução ótima 2

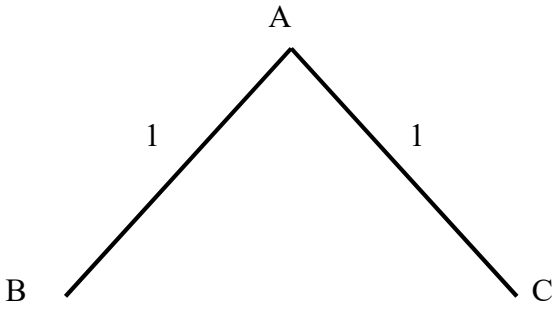

Figura 5.4: Solução ótima 3

Quando consideramos a relaxação $RL1$, o valor ótimo da função objetivo cai para 1,5 com a seguinte solução (figura 5.5):

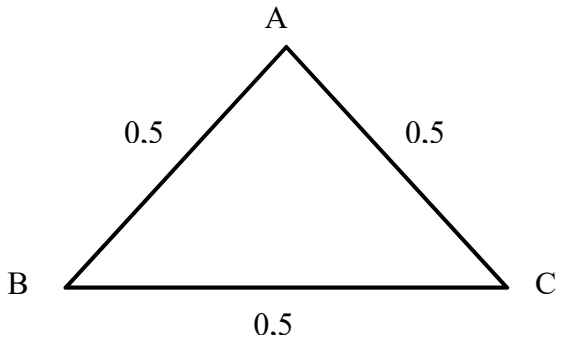

Figura 5.5: Solução ótima de RL1

Considerando o vértice A como origem, essa solução é viável porque o fluxo de A para C pode se dividir entre AC (0,5) e ABC (0,5), e o mesmo se aplica de A para B e de B para C. Por nenhuma aresta passa um fluxo maior que 0,5.

Já em $RL2$ isso não ocorre, pois z_{BC}^{C} e z_{CB}^{B} estão associados a arcos distintos, requerendo uma variável não-nula para cada arco. Dessa forma, a divisão de fluxos nessa formulação tende a ser rara e uma solução ótima é a seguinte (figura 5.6):

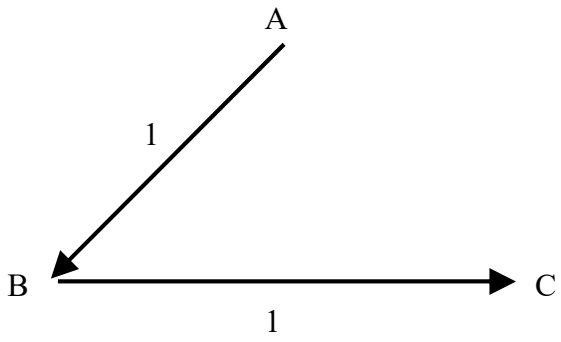

Figura 5.6: Solução ótima de RL2

Como estamos interessados em uma aproximação do problema inteiro, a formulação mais próxima da relaxação linear é $RL2$, pois $Z(RL1) \leq Z(RL3) \leq Z(RL2) \leq Z(P)$. Escolhida a formulação, vamos trabalhar com alguns testes computacionais para verificar a proximidade entre $Z(RL2)$ e $Z(P)$.

6 Resultados Computacionais

6.1 Ambiente Computacional

A fim de comparar a relaxação linear do problema com a sua versão inteira original, testamos algumas instâncias do problema.

Foram utilizadas duas máquinas do Laboratório de Otimização do Programa de Engenharia de Sistemas e Computação da UFRJ: a *conan*, com processador Intel(R) Xeon(R) CPU X5472, capacidade 3GHz, memória 12GB e sistema operacional Ubuntu 14.04 LTS; e a *kappa*, com processador Intel(R) Xeon(R) CPU X5675, capacidade 3,07GHz, memória 48GB e sistema operacional Ubuntu 12.04 LTS.

Os dados foram retirados da *OR-Library* (BEASLEY) e organizados de modo a facilitar a leitura do programa responsável por compilar o modelo. O *software* utilizado foi o AMPL versão 20150721 (Linux x86_64). Este, por sua vez, utilizou o CPLEX(R) Interactive Optimizer 12.6.0.0 como *solver*.

Primeiramente, as instâncias mais simples foram testadas na *conan*. A partir do momento em que a memória dessa máquina passou a ser insuficiente, testamos as demais instâncias na *kappa*. Mesmo assim, parte das instâncias continuou exigindo uma memória maior do que a disponível na máquina.

6.2 Descrição dos Resultados

A tabela a seguir mostra o tamanho das instâncias em termos de quantidade de vértices total ($|V|$), quantidade de arestas ($|E|$) e quantidade de vértices obrigatórios ($|S|$).

Tabela 6.1: Tamanho das Instâncias

Instância	Pontos	Arestas	Obrigatórios
b1	50	63	9
b2	50	63	13
b3	50	63	25
b4	50	100	9
b5	50	100	13
b6	50	100	25
b7	75	94	13
b8	75	94	19
b9	75	94	38
b10	75	150	13
b11	75	150	19
b12	75	150	38
b13	100	125	17
b14	100	125	25
b15	100	125	50
b16	100	200	17
b17	100	200	25
b18	100	200	50
c1	500	625	5
c2	500	625	10
c3	500	625	83
c4	500	625	125
c5	500	625	250
c6	500	1000	5
c7	500	1000	10
c8	500	1000	83

c9	500	1000	125
c10	500	1000	250
c11	500	2500	5
c12	500	2500	10
c13	500	2500	83
c14	500	2500	125
c15	500	2500	250
c16	500	12500	5
c17	500	12500	10
c18	500	12500	83
c19	500	12500	125
c20	500	12500	250
d1	1000	1250	5
d2	1000	1250	10
d3	1000	1250	167
d4	1000	1250	250
d5	1000	1250	500
d6	1000	2000	5
d7	1000	2000	10
d8	1000	2000	167
d9	1000	2000	250
d10	1000	2000	500
d11	1000	5000	5
d12	1000	5000	10
d13	1000	5000	167
d14	1000	5000	250
d15	1000	5000	500
d16	1000	25000	5

d17	1000	25000	10
d18	1000	25000	167
d19	1000	25000	250
d20	1000	25000	500
e1	2500	3125	5
e2	2500	3125	10
e3	2500	3125	417
e4	2500	3125	625
e5	2500	3125	1250
e6	2500	5000	5
e7	2500	5000	10
e8	2500	5000	417
e9	2500	5000	625
e10	2500	5000	1250
e11	2500	12500	5
e12	2500	12500	10
e13	2500	12500	417
e14	2500	12500	625
e15	2500	12500	1250
e16	2500	62500	5
e17	2500	62500	10
e18	2500	62500	417
e19	2500	62500	625
e20	2500	62500	1250

Das 78 instâncias disponíveis nessa biblioteca, 49 foram executadas com sucesso em pelo menos uma das máquinas e as demais requeriam uma quantidade de memória computacional acima da capacidade das máquinas utilizadas. Dessas 49 instâncias, o resultado

da função objetivo da versão original e da relaxação RL2 foi o mesmo em **todos** os casos exceto um (*c18*). Os resultados para o valor ótimo da função objetivo, assim como os tempos de execução para P2 e RL2 (em segundos), são mostrados a seguir.

Tabela 6.2: Resultados dos Testes

Instância	Valor ótimo da FO	Tempo P2 (s)	Tempo RL2 (s)
b1	82	0,029816	0,011383
b2	83	0,017348	0,015156
b3	138	0,030536	0,030129
b4	59	0,025302	0,026025
b5	61	0,026231	0,027734
b6	122	0,123923	0,124025
b7	111	0,044627	0,030505
b8	104	0,034261	0,043912
b9	220	0,088182	0,085621
b10	86	0,087335	0,055728
b11	88	0,101192	0,103585
b12	174	0,48156	0,45402
b13	165	0,44583	0,43806
b14	235	0,111448	0,107215
b15	318	0,145138	0,149599
b16	127	0,128929	0,131864
b17	131	0,202319	0,20312
b18	218	0,605989	0,599645
c1	85	0,143905	0,062865
c2	144	0,485687	0,287503
c3	754	5,66101	6,79748
c4	1079	23,6026	22,9791

c5	1579	60,8093	71,3232
c6	55	0,68561	0,17583
c7	102	1,48653	1,43862
c8	509	149,863	205,771
c9	707	1.004,31	2.218,25
c10	1093	1.865,51	575,52
c11	32	2,8464	1,5249
c12	46	15,9122	7,6168
c13	258	4.856,99	8.440,30
c14	323	4.130,90	10.246,80
c15	556	7.076,49	119.248,00
c16	11	12,743	5,938
c17	18	57,1455	39,4076
c18	(P2) 113 / (RL2) 112,21	48.009,80	31.724,10
c19	146	361.601,00	110.588,00
c20	137602	erro	Erro
d1	106	1,37303	0,28993
d2	220	1,92434	0,34046
d3	memória insuficiente		
d4	memória insuficiente		
d5	memória insuficiente		
d6	67	6,12535	1,26065
d7	103	11,752	2,778
d8	memória insuficiente		
d9	memória insuficiente		
d10	memória insuficiente		
d11	29	8,13107	5,75921
d12	42	49,7962	37,5852
d13	memória insuficiente		

d14	memória insuficiente		
d15	memória insuficiente		
d16	13	30,6025	22,1697
d17	memória insuficiente		
d18	memória insuficiente		
d19	memória insuficiente		
d20	memória insuficiente		
e1	11	1,05735	0,54802
e2	214	3,64182	1,77127
e3	memória insuficiente		
e4	memória insuficiente		
e5	memória insuficiente		
e6	73	10,0968	4,0827
e7	145	84,6429	memória insuficiente
e8	memória insuficiente		
e9	memória insuficiente		
e10	memória insuficiente		
e11	memória insuficiente		
e12	memória insuficiente		
e13	memória insuficiente		
e14	memória insuficiente		
e15	memória insuficiente		
e16	memória insuficiente		
e17	memória insuficiente		
e18	memória insuficiente		
e19	memória insuficiente		
e20	memória insuficiente		

A única instância cujo valor ótimo da função objetivo apresenta discrepância entre P2 e RL2 é *c18*. Mesmo assim, o *bound* inteiro em RL2, nesse caso, é igual ao valor em P2. Isso mostra que, em todos os casos que obtivemos resultados, a relaxação linear se apresentou como uma técnica eficaz para aproximar o problema original.

Observando a tabela acima, vemos que, na maioria das instâncias, a relaxação foi executada mais rapidamente que o problema inteiro. Isso só não ocorreu nos demais casos porque o *solver* CPLEX, ao resolver problemas de programação inteira ou inteira mista, aplica alguns cortes de pré-processamento antes de iniciar a resolução do problema em si, o que, às vezes, pode economizar um tempo significativo.

É importante ressaltar que as formulações que serviram de base para os nossos testes utilizam até variáveis tridimensionais, o que contribui significativamente para aumentar o tempo de processamento das variáveis e das restrições, assim como a memória exigida, explicando, também, o fato de não termos obtido resultado para algumas instâncias. Portanto, esse modelo não é ideal para resolver instâncias de grande porte. Todavia, o objetivo que estipulamos de avaliar se a relaxação linear é forte foi alcançado.

6.3 Comparação com outro modelo

Para situar os modelos P2 e RL2 em relação a uma referência mais robusta, observamos na penúltima coluna da tabela abaixo o tempo de processamento das mesmas instâncias pelo Algoritmo do Volume (LEITE, 2000). As demais colunas contêm os mesmos dados apresentados nas tabelas 6.1 e 6.2.

Tabela 6.3: Comparação com Algoritmo do Volume

Instância	$\|V\|$	$\|E\|$	$\|S\|$	Z	Tempo P2	Tempo RL2	Tempo AV	Melhor modelo
b1	50	63	9	82	0,300	0,011	0,350	RL2
b2	50	63	13	83	0,017	0,015	0,440	RL2
b3	50	63	25	138	0,031	0,030	1,100	RL2
b4	50	100	9	59	0,025	0,026	0,430	P2
b5	50	100	13	61	0,026	0,028	0,890	P2
b6	50	100	25	122	0,124	0,124	2,740	P2 = RL2

b7	75	94	13	111	0,045	0,031	0,650	RL2
b8	75	94	19	104	0,034	0,044	1,750	P2
b9	75	94	38	220	0,088	0,086	4,160	RL2
b10	75	150	13	86	0,087	0,056	0,650	RL2
b11	75	150	19	88	0,101	0,104	1,870	P2
b12	75	150	38	174	0,482	0,454	8,200	RL2
b13	100	125	17	165	0,446	0,438	2,140	RL2
b14	100	125	25	235	0,111	0,107	2,980	RL2
b15	100	125	50	318	0,145	0,150	11,660	P2
b16	100	200	17	127	0,129	0,132	3,460	P2
b17	100	200	25	131	0,202	0,203	4,340	P2
b18	100	200	50	218	0,606	0,600	14,490	RL2
c1	500	625	5	85	0,144	0,063	1,790	RL2
c2	500	625	10	144	0,486	0,288	6,720	RL2
c3	500	625	83	754	5,661	6,797	192,950	P2
c4	500	625	125	1079	23,603	22,979	449,920	RL2
c5	500	625	250	1579	60,809	71,323	5379,220	P2
c6	500	1000	5	55	0,686	0,176	2,630	RL2
c7	500	1000	10	102	1,487	1,439	7,170	RL2
c8	500	1000	83	509	149,863	205,771	216,420	P2
c9	500	1000	125	707	1004,310	2218,250	4058,820	P2
c10	500	1000	250	1093	1865,510	575,516	10501,200	RL2
c11	500	2500	5	32	2,846	1,525	11,320	RL2
c12	500	2500	10	46	15,912	7,617	26,240	RL2
c13	500	2500	83	258	4856,990	8440,300	1700,650	AV
c14	500	2500	125	323	4130,900	10246,800	664,410	AV
c15	500	2500	250	556	7076,490	119248,000	3506,140	AV
c16	500	12500	5	11	12,743	5,938	17,820	RL2

c17	500	12500	10	18	57,146	39,408	100,270	RL2
c18	500	12500	83	(P2) 113 / (RL2) 112,21	48009,800	31724,100	5043,900	AV
c19	500	12500	125	146	361601,000	110588,000	3325,490	AV
d1	1000	1250	5	106	1,373	0,290	8,740	RL2
d2	1000	1250	10	220	1,924	0,340	14,940	RL2
d6	1000	2000	5	67	6,125	1,261	8,120	RL2
d7	1000	2000	10	103	11,752	2,778	19,900	RL2
d11	1000	5000	5	29	8,131	5,759	24,140	RL2
d12	1000	5000	10	42	49,796	37,585	36,220	AV
d16	1000	25000	5	13	30,603	22,170	42,430	RL2
e1	2500	3125	5	11	1,057	0,548	19,310	RL2
e2	2500	3125	10	214	3,642	1,771	53,400	RL2
e6	2500	5000	5	73	10,097	4,083	20,220	RL2
e7	2500	5000	10	145	84,643	memória insuficiente	86,360	P2

Dado que o Algoritmo do Volume obteve os mesmos resultados para o valor da função objetivo, a comparação dos tempos se torna válida. Apesar de este ter sido mais vantajoso em apenas 12,5% das instâncias acima, ele se sobressaiu nos casos mais complexos, mostrando um crescimento menor no tempo de execução em relação ao tamanho dos dados de entrada. Além disso, ele conseguiu resolver outras instâncias não descritas acima, as quais não tivemos sucesso aqui devido à alta exigência de memória computacional.

6.4 Exemplo Ilustrativo

A fim de observarmos um exemplo menos complexo, consideremos o grafo abaixo, na figura 6.1, onde os vértices obrigatórios já estão destacados e os números adjacentes às arestas representam os custos.

O valor ótimo da função objetivo na versão inteira foi igual a 7. Já a versão relaxada retornou 7 considerando RL2 e 6,5 considerando RL1 e RL3, devido ao triângulo formado pelos vértices 4, 5 e 8, idêntico ao da figura 5.5. Isso reforça ainda mais a eficácia da formulação RL2 como uma aproximação para o problema inteiro.

A solução ficou assim: (os demais valores são nulos)

$$P1 = P2 = P3 = RL2: y_{12} = y_{23} = y_{34} = y_{45} = y_{48} = 1 \quad (68)$$

$$RL1 = RL3: y_{12} = y_{23} = y_{34} = 1 \quad (69)$$

$$y_{45} = y_{48} = y_{58} = 0{,}5 \quad (70)$$

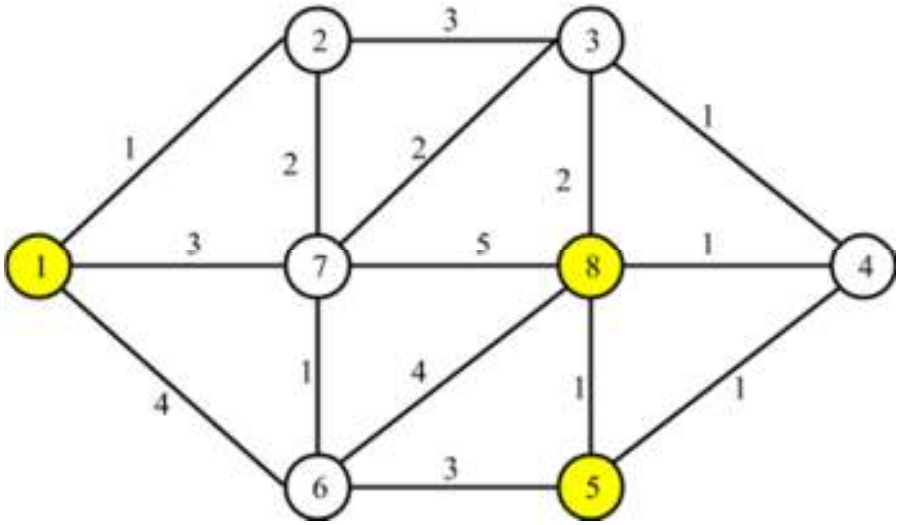

Figura 6.1: Exemplo de grafo com vértices numerados e pesos das arestas especificados

7 Conclusão

A hipótese formulada inicialmente era que a relaxação linear seria uma boa aproximação para o Problema de Steiner em Grafos. Nesse sentido, apresentamos o modelo de fluxos com as suas três formulações e escolhemos aquela cujo valor ótimo da função objetivo do problema relaxado mais se aproximava do problema original. Os testes mostraram que essa formulação, quando relaxada, continua com praticamente os mesmos resultados de antes nas instâncias testadas.

Isso significa que podemos abrir mão de resolver o problema original uma vez que temos uma relaxação linear com resultados fortemente próximos aos obtidos pela versão inteira. Se pensarmos em instâncias muito maiores do que as aqui tratadas, desde que sejam executadas em máquinas com memória e capacidade de processamento à altura, a economia de tempo tende a ser bastante significativa. Além disso, bastaria um *software* que resolvesse problemas de programação linear.

Assim, apesar de o Problema de Steiner em Grafos ser NP-difícil, podemos reduzi-lo a sua relaxação sem perda significativa de precisão nos resultados, diminuindo a complexidade para polinomial.

Uma sugestão para um trabalho posterior é comparar os resultados obtidos aqui com outros modelos além do Algoritmo do Volume, a fim de situar melhor a eficiência do nosso modelo. Outra ideia é pesquisar contraexemplos para a igualdade $Z(RL2) = Z(P2)$, ou seja, exemplos em que $Z(RL2) < Z(P2)$, como na instância *c18* deste trabalho, e estudar o significado dessa diferença segundo a lógica de fluxos.

Referências Bibliográficas

ANEJA, Y. P., "An Integer Linear Programming Approach to the Steiner Problem in Graphs", *Networks*, v. 10, n. 1, pp. 167-178, 1980.

ARPIN, D., 1984, *Algorithme pour la Résolution du Problème de Steiner sur un Graphe Orienté*. Dissertação de M.Sc., Université de Montréal, Montréal, QC, Canada.

BEASLEY, J. E., "An Algorithm for the Steiner Problem in Graphs", *Networks*, v. 14, n. 1, pp. 147-159, 1984.

BEASLEY, J. E., "An SST Based Algorithm for the Steiner Problem in Graphs", *Networks*, v. 16, n. 1, pp. 1-16, 1989.

BEASLEY, J. E. *OR-Library.* Diponível em: <http://people.brunel.ac.uk/~mastjjb/jeb/orlib/steininfo.html> Acesso em: 10 jun. 2015, 11:00.

BURDAKOV, O., DOHERTY, P., KVARNSTRÖM, J., "Local Search for Hop-constrained Directed Steiner Tree Problem with Applications to UAV-based Multi-target Surveillance", Technical Report LiTH-MAT-R–2014/10–SE, Department of Mathematics, Linkoping University, Sweden, Aug. 2014.

CLAUS, A., MACULAN, N., "Une Nouvelle Formulation du Problème de Steiner sur un Graphe", Technical Report 280, *Centre de Recherche sur les Transports*, Université de Montréal, 1983.

DREYFUS, S. E., WAGNER, R. A., "The Steiner Problem in Graphs", *Networks*, v. 1, pp. 195-207, 1971.

GAREY, M. R., GRAHAM, R. L., JOHNSON, D. S., "The Complexity of Computing Steiner Minimal Trees", *SIAM Journal on Applied Mathematics*, v. 32, n. 4, pp. 835-859, Jun. 1977.

GUANGHUA, W., ZHANJIANG, S., "Application of AHP and Steiner Tree Problem in the Location-Selection of Logistics Distribution Center", *2nd International Conference on Networking and Digital Society (ICNDS)*, v. 1, pp. 290-293, 2010.

HAUPTMANN, M., KARPINSKI, M., "A Compendium on Steiner Tree Problems", *Department of Computer Science and Hausdorff Center for Mathematics,* University of Bonn, 2015.

HWANG, F. K., RICHARDS, D. S., WINTER, P., *The Steiner Tree Problem*. 1 ed. Amsterdam, Elsevier, 1992.

LEITE, L. S. B. S., 2000, *O Algoritmo do Volume: Convergência e Resolução do Problema de Steiner em Grafos*. Tese de D.Sc., COPPE/UFRJ, Rio de Janeiro, RJ, Brasil.

LEVIN, A. Y., "Algorithm for the Shortest Connection of a Group of Graph Vertices", *Soviet Math. Dokladi*, v. 12, pp. 1477-1481, 1971.

LI, X., XU, X., ZOU, F., DU, H., WAN, P., WANG, Y., WU, W., "A PTAS for Node-weighted Steiner Tree in Unit Disk Graphs", *3rd International Conference on Combinatorial Optimization and Applications*, pp. 36-48, 2009.

MACULAN, N., "The Steiner Problem in Graphs", *Annals of Discrete Mathematics*, v. 31, pp. 185-212, 1987.

MACULAN, N., PLATEAU, G., LISSER, A., "Integer Linear Models with a Polynomial Number of Variables and Constraints for Some Classical Combinatorial Optimization Problems", *Pesquisa Operacional*, v. 23, n. 1, pp. 161-168, 2003.

ÖNCAN, T., ALTINEL, I. K., LAPORTE, G., "A Comparative Analysis of Several Asymmetric Traveling Salesman Problem Formulations", *Computers and Operations Research*, v. 36, pp. 637-654, 2009.

ROBINS, G., ZELIKOVSKY, A., "Tighter Bounds for Graph Steiner Tree Aproximation", *SIAM Journal of Discrete Mathematics*, v. 19, n. 1, pp. 122-134, 2005.

WINTER, P., "Steiner Problem in Networks: a Survey", *Networks*, v. 17, pp. 129-167, 1987.

WONG, R. T., "A Dual Ascent Approach for Steiner Tree Problems on a Directed Graph", *Mathematical Programming*, v. 28, pp. 271-287, 1984.