ALGORITMOS NON DELAYED RELAX-AND-CUT PARA O PROBLEMA DO CAIXEIRO VIAJANTE ASSIMÉTRICO

Hildebrando Barros de Azevedo Junior

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: Ábílio Pereira de Lucena Filho

Rio de Janeiro
Setembro de 2019

ALGORITMOS NON DELAYED RELAX-AND-CUT PARA O PROBLEMA DO CAIXEIRO VIAJANTE ASSIMÉTRICO

Hildebrando Barros de Azevedo Junior

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. Abílio Pereira de Lucena Filho, D.Sc.

Prof. Luiz Satoru Ochi, D.Sc.

Prof. Nelson Maculan Filho, D.Sc.

RIO DE JANEIRO, RJ – BRASIL
SETEMBRO DE 2019

Barros de Azevedo Junior, Hildebrando

Algoritmos Non Delayed Relax-and-Cut para o problema do Caixeiro Viajante Assimétrico/Hildebrando Barros de Azevedo Junior. – Rio de Janeiro: UFRJ/COPPE, 2019.

XI, 46 p.: il.; 29, 7cm.

Orientador: Ábílio Pereira de Lucena Filho

Dissertação (mestrado) – UFRJ/COPPE/Programa de Engenharia de Sistemas e Computação, 2019.

Referências Bibliográficas: p. 41 – 45.

1. Problema do Caixeiro Viajante Assimétrico. 2. Non-Delayed-Relax-And-Cut. 3. Heurística Lagrangeana. I. Pereira de Lucena Filho, Ábílio. II. Universidade Federal do Rio de Janeiro, COPPE, Programa de Engenharia de Sistemas e Computação. III. Título.

Dedico este trabalho à memória
de minha mãe Maria de Lourdes
de Sousa Azevedo

Agradecimentos

Antes de todos, gostaria de agradecer ao meu pai, minha falecida mãe e a minha irmã por todo apoio e ensinamentos. Vocês representam o combustível necessário para que eu consiga enfrentar cada batalha em minha vida.

A Michelle, minha companheira, sem você eu não conseguiria ter chego até aqui. Obrigado por suportar ao meu lado cada momento de incerteza, aguentar meu mau humor em momentos de frustração e por abdicar de si mesma para estar segurando minha mão.

Ao meu orientador Prof. Abílio. Pelo amor que demonstra em sua função como Mestre. Pela paciência em me fazer acreditar que eu poderia concluir este trabalho. Por cada ensinamento valioso que levarei comigo para o resto de minha vida. Meu mais sincero obrigado!

A todos os colegas da COPPE por toda ajuda e companheirismo durante esta jornada.

Por fim, a todos os colegas e amigos do Centro de Pesquisas da GE, que compartilharam comigo seus conhecimentos e fazem parte de cada pedacinho do profissional que me tornei.

A todos vocês,

Meu muito obrigado.

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

ALGORITMOS NON DELAYED RELAX-AND-CUT PARA O PROBLEMA DO CAIXEIRO VIAJANTE ASSIMÉTRICO

Hildebrando Barros de Azevedo Junior

Setembro/2019

Orientador: Ábílio Pereira de Lucena Filho

Programa: Engenharia de Sistemas e Computação

O Problema do Caixeiro Viajante(PCV) é um dos problemas mais estudados na literatura, possuindo aplicações em diferentes áreas tais como logística, robótica e transporte de materiais e pessoas.

Neste trabalho, propomos um algoritmo Non Delayed-Relax-and-Cut(NDRC) para o Problema do Caixeiro Viajante Assimétrico (PCVA), onde ao contrário de algoritmos tradicionais de Relaxação Lagraneana, é possível a dualização exponencial de muitas desigualdades.

Como contribuições adicionais, são propostas duas heurísticas para a obtenção de soluções primais viáveis. Bem como, procedimentos de separação de desigualdades validas para o PCVA, com o intuito de se avaliar o impacto causado por limitantes Lagrangeanos mais fortes.

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

NON DELAYED RELAX-AND-CUT ALGORITHM FOR THE ASYMMETRIC TRAVELING SALESMAN PROBLEM

Hildebrando Barros de Azevedo Junior

September/2019

Advisor: Ábílio Pereira de Lucena Filho

Department: Systems Engineering and Computer Science

The Traveling Salesman Problem (PCV) is one of the most studied problems in the literature, having applications in different areas such as logistics, robotics and transportation of materials and people.

In this Work, we propose a Non Delayed-Relax-and-Cut (NDRC) algorithm for the Asymmetric Traveling Salesman Problem (PCVA), where unlike traditional Lagranian Relaxation algorithms, the exponential dualization of many inequalities is possible.

As additional contributions, two heuristics are proposed for obtaining viable primal solutions. As well as valid inequality separation procedures for PCVA, in order to evaluate the impact caused by stronger Lagrangean limiters.

Sumário

Lista de Figuras

Lista de Tabelas

Capítulo 1

Introdução

Problemas de Otimização Matemática(OM) tem despertado o interesse de pesquisadores de diversas áreas ao longo do tempo, dada sua relação com problemas de nosso cotidiano e os desafios associados a encontrar suas respectivas soluções ótimas. Estes problemas se caracterizam basicamente pela presença de uma função objetivo a ser otimizada, sujeita a um conjunto de restrições. A maior parte do esforço desprendido no estudo destes problemas, concentra-se no desenvolvimento de algoritmos eficientes que nos levem a soluções ótimas, ou de boa qualidade, ao menor custo computacional possível.

Um problema de Otimização Combinatória (OC) é um caso particular de um problema de OM. Por sua vez, problemas de OC, quase que invariavelmente, são formulados como problemas de Programação Linear Inteira (PLI) ou de Programação Linear Inteira Mista (PLIM).

Para fins de ilustração, considere o seguinte problema de Programação Linear:

$$max\{cx : Ax \leq b, x \geq 0\} \tag{1.1}$$

Onde A é uma matriz m x n, c um vetor n-dimensional de linhas, b um vetor m-dimensional de colunas e x um vetor n-dimensional de variáveis.

Suponha agora que adicionemos a 1.1 uma restrição que imponha a condição de que apenas algumas variáveis poderão assumir um valor inteiro na solução, ao passo que outras não. Neste caso, temos um problema de PLIM que pode ser escrito da seguinte forma:

$$\max cx + hy \tag{1.2}$$

$$Ax + Gy \leq b \tag{1.3}$$

$$x \geq 0, y \geq 0 \text{ e inteiro} \tag{1.4}$$

Onde novamente A é uma matriz m x n, G uma matriz m x p, h é um vetor de p linhas e y um vetor p de colunas de variáveis inteiras. Neste caso, excluindo-se da formulação (1.2)-(1.4) as variáveis x, teremos a formulação de um PLI. Adicionalmente, se além da exclusão das variáveis x impormos a condição de que as variáveis y devem assumir valor $\{0, 1\}$, teremos a formulação de um problema de Programação Binária (PB). Neste trabalho, o problema que iremos investigar será considerado sob uma formulação de PB.

Dentre os diversos problemas de OC associados a problemas do mundo real, considere um conjunto de cidades e os custos associados a viagem entre cada par destas. Considere ainda que o objetivo é encontrar um caminho que nos leve a visitar todas as cidades, apenas uma vez, retornando à cidade de origem ao menor custo possível. Este é dos problemas mais estudados na literatura e tem sido utilizado como base para diversos experimentos de métodos de otimização: o Problema do Caixeiro Viajante(PCV) [1]. O PCV destaca-se basicamente pelo fato de ser um problema de fácil compreensão e difícil solução (NP-difícil) [2]. Dentre as mais variadas aplicações relacionadas ao problema do caixeiro viajante, podemos ressaltar a perfuração de placas de um circuito impresso[3], sequenciamento de DNA [4], cristalografia de raio-X [5], mapeamendo de genomas[6], ordem de coleta em armazéns[7], dentre outras relacionadas a transito de veículos, pessoas e materiais.

1.1 Formulação do Problema

Seja $G = (V, A)$ um grafo direcionado, onde $V = \{1, ..., n\}$ representa o conjunto de vértices e $A = \{(i, j) : i, j \in V\}$ o conjunto de arcos. Seja ainda, c_{ij} o custo associado ao arco $(i, j) \in A$ com $(c_{ii} = +\infty)$ para $i \in V$, um circuito Hamiltoniano de G é um circuito que visita cada vértice de V exatamente uma vez. Dito isto, o problema do Caixeiro Viajante(PCV) se resume a encontrar um circuito Hamiltoniano de G onde o custo é mínimo. Para cada arco, existe uma variável x_{ij} binária que associa a existência do arco (i, j) no circuito Hamiltoniano de custo mínimo. Assumindo o valor 1 caso faça parte da solução, e valor 0, caso o contrário.

No PCV, as propriedades da matriz de custos C são utilizadas para a classificação dos problemas. O problema do Caixeiro Viajante Simétrico(PCVS) é um caso particular do problema do Caixeiro Viajante Assimétrico(PCVA), onde $c_{ij} = c_{ji}$, tendo sua representação em um grafo não direcionado $G = (V, E)$. Nesta pesquisa, apesar de utilizarmos como referência parte do material de estudo relacionado ao problema simétrico, abordaremos o problema assimétrico como ponto central desta

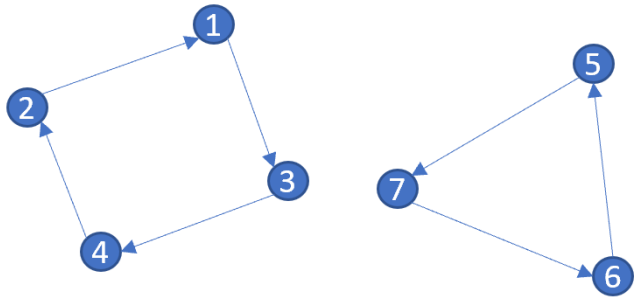

Figura 1.1: Solução com subrotas no digrafo K_7.

dissertação. Uma das primeiras formulações para o problema do PVCA foi sugerida em [8], a formulação de Dantzig-Fulkerson-Johnson. Considerando que todas as instâncias utilizadas neste trabalho correspondem a um grafo completo, descrevemos esta formulação a seguir:

$$\min \sum_{i \in V} \sum_{j \in V} c_{ij} x_{ij} \tag{1.5}$$

Sujeito a:

$$\sum_{j \in V} x_{ij} = 1; \quad i \in V \tag{1.6}$$

$$\sum_{i \in V} x_{ij} = 1; \quad j \in V \tag{1.7}$$

$$\sum_{i \in S, j \in S} x_{ij} \leq |S| - 1; \quad S \subset V, |S| \geq 2 \tag{1.8}$$

$$x_{ij} \in \{0, 1\}; \quad i \in V, j \in V. \tag{1.9}$$

As restrições (1.6) e (1.7) impõem que cada vértice possui exatamente um arco chegando (1.6) e exatamente um arco saindo (1.7). Consequentemente, visam garantir que cada vértice é visitado apenas uma vez. Para analisarmos a restrição (1.8), considere que $n = 7$ e que as seguintes variáveis assumam valor 1 como mostrado na Figura 1.1:

$$x_{12} = x_{23} = x_{34} = x_{41} = x_{65} = x_{57} = x_{76} = 1$$

Apesar desta solução acima satisfazer as restrições (1.6) e (1.7), ela não forma um ciclo hamiltoniano. Sendo assim, não representa uma solução viável para o PCVA. Note que embora tenhamos todos os vértices presentes na solução, esta solução forma dois circuitos distintos(*subrotas*): $\{1, 2, 3, 4\}$ e $\{5, 6, 7\}$. Tomando

como exemplo a solução acima, para que não tenhamos uma subrota no subconjunto $S = \{1, 2, 3, 4\} \subset V$, precisamos garantir que no máximo três arestas de um circuito formado por estes vértices estejam presentes na solução. Para garantir que a solução não contenha subrotas, Dantzig, Fulkerson e Johnson[8] propuseram as restrições de eliminação de subrotas(DES)(1.4). Considerando ainda a solução representada na Figura 1.1 e o subconjunto $S = \{1, 2, 3, 4\}$, em (1.8) temos a DES referente a S. Sendo $x_{12} = x_{23} = x_{34} = x_{41} = 1$,é fácil notar que o lado esquerdo da desigualdade apresentará soma igual a 4, representando assim uma violação da DES.

$$x_{12} + x_{13} + x_{14} + x_{21} + x_{23} + x_{24} + x_{31} + x_{32} + x_{34} + x_{42} + x_{41} + x_{43} \leq 3 \quad (1.10)$$

A formulação DFJ possui $2n$ restrições de designação sendo n associado a cada uma das restrições (1.6) e (1.7). Adicionalmente, apresenta $2^n - 2$ restrições para eliminação de subrotas (1.8).

1.2 Revisão da Literatura

O Problema do Caixeiro Viajante tem sido extensivamente estudado ao longo das últimas décadas. Entretanto, boa parte dos esforços empregados na pesquisa de heurísticas para a solução do problema foram concentrados em sua versão simétrica. Fato este que nos leva a um dos fatores motivadores para este estudo e foco em sua versão assimétrica(PCVA).

Ao longo dos últimos cinquenta anos, foram propostos diferentes algoritmos para a solução PCVA. Estes algorítimos podem ser classificados basicamente em duas grandes categorias: *Métodos Exatos* e *Métodos Heurísticos*.

Apesar de algoritmos exatos garantirem a otimalidade da solução encontrada, estes possuem um fator limitante relacionado ao tamanho do problema e o esforço computacional desprendido para a obtenção da solução. Dentre a classe de algoritmos exatos, muitos procedimentos *branch-and-bound* foram propostos com o intuito de se encontrar a solução ótima do PCVA. Após o paper escrito por Little *et.al* [9], onde o termo *branch-and-bound* foi introduzido pela primeira vez, outros algoritmos foram propostos em [10] e [11]. Dentre os trabalhos mais recentes podemos citar [12] e [13], onde dois diferentes procedimentos *branch-and-cut* são descritos utilizando o Problema de Atribuição como relaxação.

De uma forma geral, um algoritmo *branch-and-cut* é um algoritmo *branch-and-bound* no qual planos de cortes são gerados ao longo da árvore de busca. Em um algoritmo *branch-and-cut*, as formulações dos subproblemas correspondentes aos nós da árvore de enumeração implícita são reforçadas através do uso de desigualdades

válidas para o problema. Isto é feito através do uso de algoritmos de plano de corte elaborados especificamente para cada família de desigualdades válidas.

Os algoritmos heurísticos se apresentam como uma alternativa aos algoritmos exatos. É importante ressaltar que apesar dos algoritmos heurísticos obterem soluções de boa qualidade, não fornecem a garantia da solução ótima ser encontrada. De uma forma geral, estes algoritmos são geralmente bem simples e de baixo custo computacional se comparados a algoritmos exatos. As heurísticas para o PCVA podem ser dividas basicamente em três classes distintas: *Heurísticas construtivas*, *Heurísticas de busca local* e *Heurísticas compostas* [12].

Heurísticas construtivas constroem gradualmente uma solução até que tenhamos um circuito hamiltoniano. De forma geral, inicia-se com uma subrota (geralmente aleatória) e tenta-se expandi-la até uma obtenção de uma rota aproximadamente ótima. Como exemplo destas heurísticas podemos citar a Heurística do Vizinho mais próximo, inserção mais barata e as heurísticas construtivas apresentadas por Glover *et.al* [14].

As Heurísticas de busca local representam um conjunto de heurísticas com o maior índice de sucesso na busca de soluções de boa qualidade. Estas heurísticas visam a modificação de uma solução inicial S_0 com o intuito de encontrar soluções de melhor qualidade. Heurísticas de busca local trabalham geralmente com estruturas de vizinhanças para encontrar melhores soluções partindo-se da solução inicial. Sendo S o conjunto de soluções viáveis para o problema e $S_0(S_0 \subset C)$ a solução inicial, uma solução vizinha $S^{'}$ é obtida a partir de um movimento m qualquer que transforme a solução S_0 em $S^{'}$. Na literatura é possível encontrar diferentes implementações de heurísticas que utilizam como base a busca local, como pode ser visto em Cirasella *et.al* [15].

Um outro conjunto de heurísticas que merece menção são as *Meta-heurísticas*. Meta-heurísticas são estruturas algorítmicas gerais adaptáveis a diversos problemas de otimização. Para o PCVA, podemos citar na literatura aplicações utilizando Algorítmos Genéticos(AG) em [16–19] e VSN (Variable Neighbourhood Search) em [20].

Apesar de existirem na literatura trabalhos relacionados ao PCVA onde se utilizam heurísticas de busca local partindo-se de uma solução inicial viável, a aplicação destas heurísticas é muito mais restrita que em sua versão simétrica. O principal motivo se deve ao fato de que a estrutura de vizinhança definida pela busca local pode implicar em reversão de arcos,limitando as estratégias de busca por soluções viáveis de melhor qualidade e impactando a performance do algoritmo.

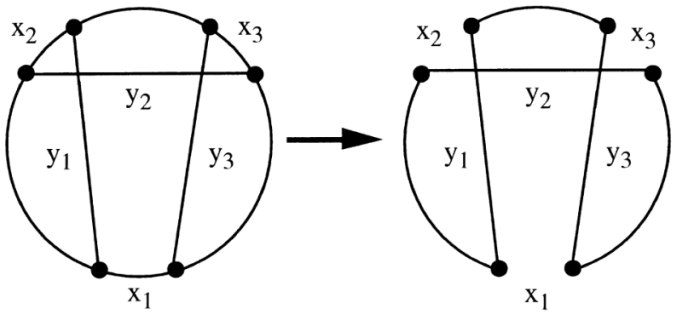

Figura 1.2: Movimento 3-Opt.

1.3 Motivação da Pesquisa

Conforme descrito anteriormente, apesar de ser um caso particular do PCVA,o Problema do Caixeiro Viajante Simétrico(PCVS) tem sido investigado na literatura muito mais intensamente que o PCVA. Isto se aplica tanto em termos de algoritmos exatos quanto em termos de algoritmos de aproximação.

LIN e KERNIGHAN [21] apresentaram uma das heurísticas mais efetivas para obter uma solução ótima ou próxima da ótima para o PCVS. A heurística LK baseia-se em movimentos k-opt. O algoritmo proposto tenta encontrar uma solução de melhor qualidade explorando uma k-vizinhança de uma solução viável conhecida. Ou seja, considera as soluções obtidas através da re-conexão dos caminhos simples resultantes da eliminação de k arestas daquela solução, por k arestas que levem a uma nova solução viável para o problema. Caso a solução tenha um menor custo, toma-se a mesma como a nova solução disponível. A Figura 1.2 ilustra um movimento 3-opt($k = 3$) performado pela heurística de Lin Kernighan para o PCVS. Note que este movimento envolve a substituição de 3 arestas do ciclo Hamiltoniano.

Apesar da heurística LK ser até hoje uma das mais efetivas, Helsgaun [12] salienta que a regra utilizada para a inclusão dos arcos no conjunto Y, restringe-se aos cinco vizinhos mais próximos. Regra esta, que havia sido adicionada com o intuito de diminuir o espaço de busca reduzindo consequentemente o esforço computacional desprendido pelo algoritmo. No entanto, esta restrição implica em um risco maior do algoritmo ficar preso em um ótimo local uma vez que a solução ótima pode conter um arco não conectado aos cinco vizinhos mais próximos. Com o intuito de corrigir essa deficiência, Helsgaun[12] implementou uma heurística com uma regra de proximidade mais eficiente.Esta regra de proximidade utiliza como informação a solução de uma árvore geradora mínima. Com esta atualização, o algoritmo Lin-Kernighan-Helsgaun(LKH) [12] se tornou a heurística mais efetiva para o Problema do Caixeiro Viajante Simétrico(PCVS) conhecida atualmente na literatura.

Estudos empíricos apresentados na literatura sugerem que entre 70% e 80% das arestas de uma solução ótima para o PCVS fazem parte de uma solução de boa

qualidade para a relaxação Lagrangeana da reformulação 1-tree do problema [12]. Com base nisso, o algoritmo de Helsgaun se utiliza da informação dual Lagrangeana, através de uma reformulação do PCVS como uma 1-tree, para a construção e posteriormente melhora de soluções viáveis para o problema. O fator motivador desta pesquisa, toma como estratégia a aplicação da mesma abordagem utilizada no algoritmo de Helsgaun, porém com um foco direcionado a versão assimétrica do problema(PCVA). Em outras palavras, utilizando-se da informação dual lagrangeana para construção e melhoria de soluções viáveis para o problema.

A aplicação da relaxação Lagrangeana aplicada ao problema do Caixeiro Viajante foi iniciada por Held e Karp [22] para a versão simétrica. Em geral, aplica-se a relaxação Lagrangeana quando ao dualizar um conjunto de restrições o problema remanescente pode ser resolvido por métodos eficientes em tempo polinomial. Mesmo sendo uma poderosa ferramenta na solução de problemas complexos, os algoritmos de relaxação Lagrangeana tradicionais encontram limitações em problemas nos quais o número de desigualdades a serem dualizadas é exponencial.

Neste trabalho, propomos um algoritmo Non Delayed-Relax-and-Cut(NDRC) LUCENA(2005) [23] para o PVCA. Ao contrário dos algorítmos de relaxação Lagrangeana tradicionais, algoritmos NDRC conseguem dualizar exponencialmente muitas desigualdades. Isto é possível através de um procedimento Lagrangeano análogo à geração de planos de cortes em algoritmos de programação Inteira. A grande vantagem dos algoritmos NDRC é que , em tese, conseguem ir além dos limites duais de uma relaxação Lagrangeana Tradicional. O objetivo principal deste trabalho é investigar o impacto causado por limitantes duais Lagrangeanos mais fortes, na obtenção de soluções heurísticas para o problema.

1.4 Organização da Dissertação

Além deste capítulo de introdução, esta dissertação contém 4 capítulos adicionais. No Capítulo 2, descrevemos os procedimentos de relaxação Lagrangeana tradicional e apresentamos uma descrição genérica de algorítmos do tipo NDRC. Ainda no Capítulo 2, apresentamos duas versões do algoritmo NDRC para o PCVA, onde o problema é reformulado como um problema de uma 1-Aborescência e como um problema de atribuição. Adicionalmente, são apresentados os procedimentos para dualização dinâmica das desigualdades do tipo clique e do tipo comb. No Capítulo 3, descrevemos as heurísticas para a construção da solução viável para o problema PCVA e posteriormente os procedimentos de busca local adotados para a melhoria da solução. No Capítulo 4, apresentamos os resultados obtidos com os algoritmos. No Capítulo 5, fazemos as considerações finais sobre nosso trabalho e apresentamos aspectos a serem investigados em estudos futuros.

Capítulo 2

Algoritmos Non Delayed Relax-and-Cut

Algoritmos Lagrangeanos, análogos a algoritmos poliedrais de planos de cortes, foram introduzidos originalmente em [24, 25]. Por sua vez, o termo Relax-and-Cut, hoje associado a qualquer algoritmo Lagrangeano com aquelas características, foi introduzido em [26]. O termo *Relax-and-Cut*, inicialmente restrito apenas ao algoritmo proposto em [25], foi posteriormente estendido, em [23], para incluir também o algorítmo sugerido em [24]. Nesse caso, a denominação *Delayed Relax-and-Cut* (DRC) é atribuida ao algoritmo em [25], enquanto *Non Delayed Relax-and-Cut* (NDRC) é associada ao algoritmo em [24]. Ou seja, de acordo com [23], o termo Relax-and-Cut passa a denominar toda a classe de algoritmos Lagrangeanos que são análogos a algoritmos poliedrais de planos de corte. As diferentes variantes dos mesmos, por sua vez, são identificadas através de termos adicionais, como "delayed" ou "non delayed". Resultados computacionais limitados, apresentados em [23], indicam que algoritmos NDRC, quando comparados a algoritmos DRC (para uma mesma formulação de um dado problema), apresentam melhor desempenho.

Neste capítulo, introduzimos diversos algoritmos NDRC para o PVCA. Estes se baseiam numa adaptação do Método do Subgradiente (MS) (Vide [27] para maiores detalhes) proposta em [23]. Tal adaptação, como veremos mais adiante, permite uma *dualização dinâmica* de exponencialmente muitas desigualdades ou seja, estas são dualizadas apenas quando violam a solução de um Subproblema Lagrangeano.

2.1 Relaxação Lagrangeana

Relaxação Lagrangeana (RL) [28, 29] é um método de decomposição popularizado nos anos 1970, com o trabalho de Held and Karp [22] para o PVCS. Continua sendo bastante utilizada, embora seu pico de popularidade tenha ocorrido nos anos 1980.

Para obter mais detalhes sobre a aplicação de Relaxação Lagrangeana à resolução de problemas de Otimização Combinatória, sugerimos a leitura de [27]. Para uma breve descrição do método, considere uma formulação de Programação Inteira (PI), genérica, para um problema de OC:

$$z = \min cx \tag{2.1}$$

sujeito a:

$$Ax \geq b \tag{2.2}$$

$$Bx \geq d \tag{2.3}$$

$$x \in \mathbb{Z}_+^n. \tag{2.4}$$

Sendo B uma matrix $m \times n$, assumimos que (2.3) corresponde a m restrições que *complicam* a resolução de (2.1)-(2.4). Ou seja, se as restrições (2.3) fossem eliminadas daquela formulação, a formulação resultante, $\{\min cx : Ax \geq b, x \in \mathbb{Z}_+^n\}$, corresponderia a um problema *fácil* de resolver. Mais especificamente, um problema passível de resolução por um algoritmo de tempo polinomial.

Uma RL consiste em associar multiplicadores $\mu \in \mathbb{R}_+^m$ às restrições (2.3) e levá-las a função objetivo, como *penalidades*, obtendo assim o seguinte *Subproblema Lagrangeano* (SL):

$$z(\mu) = \min cx + \mu(d - Bx) \tag{2.5}$$

sujeito a:

$$Ax \geq b \tag{2.6}$$

$$x \in \mathbb{Z}_+^n. \tag{2.7}$$

Note que a região de viabilidade de (2.5)-(2.7), $\{x : Ax \geq b, x \in \mathbb{Z}_+^n\}$, está contida na região de viabilidade de (2.1)-(2.4), $\{x : Ax \geq b, Bx \geq D, x \in \mathbb{Z}_+^n\}$ (a segunda região corresponde à primeira, sujeita a restrições adicionais). Dessa forma, a primeira condição necessária para que (2.5)-(2.7) defina uma relaxação de (2.2)-(2.4) é assim satisfeita. A segunda condição requer, para qualquer solução viável de (2.2)-(2.4), que o valor da função objetivo objetivo de (2.1) seja menor ou igual ao valor da função objetivo de (2.5). Assumindo ser $x^* \in \mathbb{Z}_+^n$ uma tal solução, note que $z(\mu) = cx^* + \mu(d - Bx^*) \geq cx^* = z$ se aplica, pois $\mu(d - Bx^*) \geq 0$ corresponde ao produto de dois termos não negativos. Ou seja, a segunda condição necessária é também satisfeita e (2.5)-(2.7) corresponde, de fato, a uma relaxação de (2.2)-(2.4)

Em função do exposto acima, $z(\mu) \geq z$ se aplica a qualquer $\mu \in \mathbb{R}_+^m$. Assim

sendo, determinar o menor valor possível para o limite superior definido por (2.5)-(2.7) consiste em resolver o seguinte *Problema Dual Lagrangeano* (PDL):

$$z_{PDL} = \max\{z(\mu) : \mu \geq \mathbf{0}\}. \tag{2.8}$$

O PDL é geralmente resolvido de forma iterativa, atualizando os multiplicadores μ à cada passo, de forma a obter o valor z_{PDL}. Um procedimento exato para resolvê-lo é o chamado Métodos de Feixos [30] (Bundle Methods, em inglês). Outro método exato é o Método do Subgradiente (MS), que possui diversas variantes heurísticas que se mostraram computacionalmente eficientes e que levam a boas aproximações de z_{PDL} (veja [27], para maiores detalhes). Deste ponto em diante, denominaremos simplesmente por *Método do Subgradiente* ao procedimento heurístico que vamos utilizar para resolver (2.8).

2.2 O Método do Subgradiente

Um procedimento de RL se inicia com a definição das restrições a dualizar. Feito isso, a tarefa que se impõe é a de resolver o PDL, ou seja, identificar valores para os multiplicadores (das restrições dualizadas) de tal forma a obter o melhor limite dual possível para z. A resolução de um PDL pelo MS é feita de forma iterativa, resolvendo-se uma sequência de SLs. Neste processo, os multiplicadores são ajustados à cada iteração, de tal forma a convergir para um SL que forneçe um limite dual próximo a z_{PDL}. Por simplicidade, na descrição que vamos apresentamos para o MS, tomamos por base uma formulação do próprio PCVA. Tal formulação, como veremos, se basiea numa reformulação do problema através do uso de r-arborescências geradoras (mais detalhes, a seguir).

Considere uma r-arborescência geradora (r-AG) de $D = (V, A)$ (vide [31] para mais informações sobre r-AGs). Esta pode ser entendida como uma estrutura análoga a de uma árvore geradora (AG), para um grafo não-direcionado $G = (V, E)$ (vide [32], para mais informações sobre AGs). Uma AG, deve-se notar, define um subgrafo acíclico de G contendo exatamente $|V| - 1$ arestas. Uma r-AG, por sua vez, corresponde a uma árvore geradora de D, com caminhos simples orientados que levam de um vértice raiz, $r \in V$, à cada vértice em $V \setminus \{r\}$. As figuras 2.1 e 2.2 ilustram, respectivamente, uma AG de G e uma 1-AG de D, ou seja, uma arborescência geradora tendo o vértice 1 como raiz.

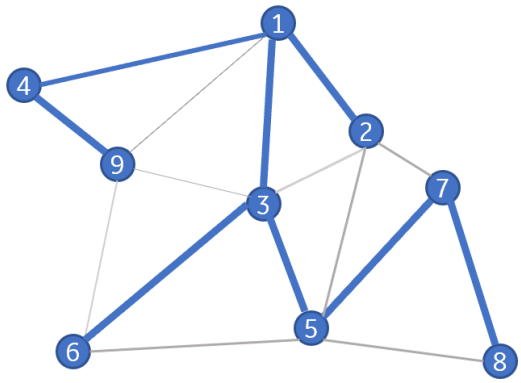

Figura 2.1: Árvore geradora(AG) de G

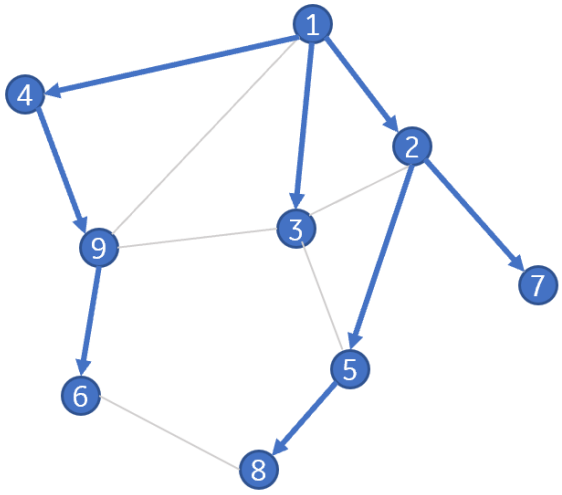

Figura 2.2: 1-Arborescência geradora(1-AG) de D

Considere agora a decomposição de um circuito Hamiltoniano em um caminho Hamiltoniano iniciado em $r \in V$, um r-CH na notação que vamos utilizar, complementado por um arco que aponta para r. Uma vez descrita essa reformulação, vamos definir suas restrições a dualizar e apresentar o MS, como utilizado em nossa investigação.

A variáveis envolvidas na reformulação, $x \in \mathbb{R}^{|E|}$, são as mesmas introduzidas anteriormente para a formulação clássica do PCVA, discutida no Capítulo 1. A reformulação é então definida como:

$$z = \min \sum_{i \in V} \sum_{j \in V \setminus \{i\}} c_{ij} x_{ij} \tag{2.9}$$

sujeito a:

$$\sum_{i \in V \setminus \{j\}} x_{ij} = 1, j \in V \tag{2.10}$$

$$\sum_{j \in V \setminus \{i\}} x_{ij} = 1, i \in V \tag{2.11}$$

$$\sum_{i \in S} \sum_{j \in V \setminus S} x_{ij} \geq 1, r \in S, S \subset V \tag{2.12}$$

$$x_{ij} \in \{0, 1\}, i \in V, j \in V \setminus \{i\}. \tag{2.13}$$

Note que as desigualdades (2.12), denominadas Desigualdades de Cutset (DC), garantem a existência de um caminho que leva de r a qualquer vértice $j \in V \setminus \{r\}$. Ou seja, garantem que a solução é conexa e orietada de r para os demais vértices de D. Por sua vez, as desigualdades (2.10) garantem a existência de um único arco apontando para qualquer $j \in V$. De forma análoga, as desigualdades (2.11) impõem a existência de um único arco apontando para fora de todo $i \in V$. Em conjunto, (2.10)-(2.12) garantem que a solução obtida é um circuito Hamiltoniano.

Uma RL de (2.9)-(2.13) é obtida dualizando-se as desigualdes (2.11). Isso é feito através do uso de multiplicadores $\{\mu_i \geq 0, i \in V\}$, o que nos leva ao seguinte SL, aqui denotado por $SL(\mu)$:

$$z(\mu) = \min \sum_{i \in V} \sum_{j \in V \setminus \{i\}} (c_{ij} - \mu_i) x_{ij} + \sum_{i \in V} \mu_i \tag{2.14}$$

sujeito a:

$$\sum_{i \in V \setminus \{j\}} x_{ij} = 1, j \in V \tag{2.15}$$

$$\sum_{i \in S} \sum_{j \in V \setminus S} x_{ij} \geq 1, r \in S, S \subset V \tag{2.16}$$

$$x_{ij} \in \{0, 1\}, i \in V, j \in V \setminus \{i, r\} \tag{2.17}$$

O problema (2.14)-(2.17) é constituido por dois subproblemas independentes. O primeiro, é o de encontrar uma r-AG de custo mínimo para D. O segundo, é o de escolher o arco de menor custo apontando para r. Para obter uma r-AG de custo mínimo, utilizamos o algoritmo de Chu-Liu/Edmonds [31], como implementado por Fischetti& Toth(vide [33], para maiores detalhes). O segundo subproblema, por sua vez, é resolvido em tempo linear, por inspeção.

Denotando por

$$g_i = 1 - \sum_{j \in V} x_{ij}, i \in V \tag{2.18}$$

ao subgradiente da i-ésima restrição de (2.11), estamos de posse de todos os ingredientes necessários à descrição de nosso MS, que se baseia em [27].

O MS é inicializado por um vetor de multiplicadores $\mu^0 = \mathbf{0}$ (note que qualquer vetor em $\mathbb{R}^{|E|}$ cumpriria essa função, já que μ é irrestrito em sinal). A cada iteração k do método, um SL(μ^k) é resolvido como descrito acima. Feito isso, condições

de otimalidade são checadas e, caso um certificado de otimalidade não tenha sido obtido, os multiplicadores μ são atualizados, como veremos a seguir.

Numa iteração k do MS, custos Lagrangeanos c^k são calculados através da seguinte expressão:

$$\{c_{ij}^k = c_{ij} - \mu_i^k : i \in V, j \in V \setminus \{i\}\}. \tag{2.19}$$

Feito isso, SL(μ^k) é então resolvido, tendo x^k como sua solução.

A seguir, subgradientes g^k são calculados como:

$$g_i^k = 1 - \sum_{j \in V} x_{ij}^k, i \in V. \tag{2.20}$$

Caso $g_i^k = 0$ resulte para todo $i \in V$, pode-se interromper a execução do algoritmo. Isso porque x^k seria, nesse caso, garantidamente ótimo para o PCVA. Caso contrário, note que o subgradiente g^k define, no ponto $x^k \in \mathbb{R}^{|E|}$, uma direção de melhora, $-g^k$. Para determinar o quanto devemos caminhar naquela direção, utilizamos um *tamanho de passo*

$$\theta^k = \frac{\alpha(\overline{z} - z(\mu^k))}{\sum_{i=1,\ldots,m} \left(g_i^k\right)^2}, \tag{2.21}$$

onde α é um número real assumindo valores em $(0, 2]$ e $\overline{z}$ é um limite superior para o valor de z, obtido através de uma heurística para a resolução do PCVA (vide Capítulo 3, para a heurística utilizada nesta dissertação). Em nosso procedimento, fixamos em 2 o valor inicial de α.

A seguir, atualizamos os multiplicadores para a iteração seguinte do MS, aplicando a fórmula

$$\mu_i^{k+1} = \mu_i^k + \theta^k g_i^k, i \in V. \tag{2.22}$$

Se ao invés de restrições de igualdade do tipo (2.10), como aqui descrito, desigualdades $\{\sum_{j \in V \setminus \{i\}} x_{ij} \leq 1, i \in V\}$ são dualizadas, um pequeno ajuste deve ser feito no procedimento descrito acima. Isso porque multiplicadores irrestritos em sinal se aplicam ao primeiro caso, enquanto multiplicadores não negativos se aplicam ao segundo. No segundo caso, a fórmula (2.22) deve ser substituida por

$$\mu_i^{k+1} = \max\{0, \mu_i^k + \theta^k g_i^k\}, i \in V. \tag{2.23}$$

Finalmente, um número máximo de iterações, n_{it}, deve ser fixado para a aplicação do MS, com duas condições adicionais de parada também se aplicando, como veremos mais adiante. Além disso, o parâmetro α, descrito acima, deve ser ajustado ao longo da aplicação do algoritmo. Em nosso caso, α é reduzido à metade sempre que n_{half} iterações consecutivas do MS ocorrem, sem uma atualização do melhor limite dual obtido pelo algoritmo. Note, nesse caso, que o *contador* de iterações associado a

n_{half} deve ser reinicializado em 0 sempre que uma redução no valor de α ocorre. O mesmo se aplica quando o valor do melhor limite dual obtido pelo algoritmo é atualizado.

Como condições adicionais de parada do algoritmo, além da eventual ocorrência de $g_i^{k+1} = 0$, para todo $i \in V$ (o que nos fornece um certificado de *PCVA-otimalidade* para a solução x^k, do SL L(μ^k)), consideramos ainda duas outras alternativas. A saber, $(\overline{z} - z(\mu^k)) < 1$, o que nos daria, por arredondamento inteiro, um certificado de otimalidade para a solução heurística de valor $\overline{z}$ que temos em mãos, e $\alpha < \epsilon$, onde ϵ é uma valor suficientemente pequeno.

2.3 Ajustando o MS para o NDRC

O que distingue uma RL tradicional (como aquela descrita acima para a reformulação (2.9)-(2.13) do PCVA) de uma RL NDRC, é o fato da segunda permitir uma dualização (implícita) de um número exponencialmente grande de desigualdades. Ou seja, um procedimento NDRC pode simultaneamente dualizar conjuntos de restrições contendo um número polinomial de elementos, como uma RL tradicional, e/ou conjuntos de restrições com exponencialmente muitos elementos. Esse segundo tipo de dualização, como mencionado anteriormente, é feito através de um análogo Lagrangeano de uma algoritmo (poliedral) de planos de corte.

No que vamos agora descrever, tomamos a reformulação (2.9)-(2.13) do PCVA como base. Assumimos também que as restrições (2.11) são dualizadas Lagrangeanamente e tratadas como sugerido na Seção 2.2. Finalmente, vamos assumir que as exponencialmente muitas desigualdades

$$a_j x \leq b_j, j = 1, 2...m \tag{2.24}$$

são válidas para o PCVA e serão dualizadas *dinamicamente* ao longo de nossa aplicação do MS. O significado do termo "dinamicamente", ficará mais claro no decorrer desta seção. Da mesma forma, se tornará evidente a razão pela qual vamos denominar *restrições permanentes* àquelas em (2.11) e *restrições temporárias*, às desigualdades em (2.24).

Associamos multiplicadores não negativos $\{\beta_j : j = 1, \ldots, m\}$ à cada uma das desigualdades em (2.24) e atribuimos a eles um valor inicial 0. Dessa forma, na iteração 0 do MS, à exemplo do que também ocorre com as restrições permanentes, (2.11), nenhuma desigualdade temporária, (2.24), irá modificar os custos originais do PCVA (já que todos os seus multiplicadores têm valor 0). Entretanto, ao contrário do que vamos impor às desigualdades (2.24), subgradientes das restrições (2.11) serão sempre utilizados no cálculo do tamanho de passo (vide expressão correspon-

dente, (2.20)). Subgradientes das restrições temporárias, (2.24), só cumprirão tal função quando estas forem *ativas* ou violadas pela solução corrente do SL (como será visto mais adiante, as duas condições podem ocorrer simultaneamente). Denominamos "desigualdades ativas", àquelas cujos multiplicadores são positivos. Note, na segunda alternativa considerada acima, que o multiplicador em questão se tornará estritamente positivo, ao ser atualizado ao final da iteração. Desigualdades temporárias *inativas* são aquelas que não são ativas.

Note que uma desigualdade temporária pode se tornar ativa numa dada iteração do MS, permanecer ativa por várias iterações subsequentes, e se tornar inativa, novamente, mais adiante. Da mesma forma, pode se tornar ativa, mais uma vez, em iterações futuras do MS. Em experimentos computacionais com a formulação (2.9)-(2.13), observamos que algo em torno de, no máximo, $|V|$ desigualdades (num universo de exponencialmente muitas desigualdades temporárias), estão ativas em qualquer iteração do MS NDRC.

Para concluir nossa descrição do MS NDRC, devemos indicar como identificamos as desigualdades (2.24)) violadas pela solução corrente do SL. Esta identificação, como iremos demonstrar mais adiante, se beneficia do fato de que tais soluções são binárias 0-1, o que reduz significativamente a complexidade da tarefa. Como exemplo, a complexidade da separação de desigualdades do tipo *Comb* é $O(n^3)$ [34], no caso geral, ou seja para soluções fracionárias de uma relaxação linear de (2.9)-(2.13) (como se aplica a algoritmos poliedrais do tipo Branch-and-Cut (B&C) - veja [35] para mais informações sobre algoritmos B&C). Por sua vez, a identificação dessas mesmas desigualdades, no nosso caso, é feita em tempo linear.

Concluimos esta seção com duas observações. A primeira diz respeito à atualização dos multiplicadores associados às restrições temporárias, (2.24). Tal atualização deve ser feita através da fórmula (2.23), já que nossas restrições temporárias definem desigualdades. Ao contrário, nossas restrições permanentes, (2.11), correspondem a igualdades e seus multiplicadores devem, assim, continuar a ser atualizados por (2.22).

A segunda observação é relativa às razões pelas quais não consideramos os subgradientes das desigualdades inativas no cálculo de θ (vide fórmula (2.21)). A primeira razão para isso é que, em qualquer iteração do MS NDRC, o número de desigualdades temporárias inativas, contendo subgradientes não nulos, tende a ser exponencialmente grande. Isso, além de tornar impraticável a tarefa de calcular seus subgradientes, um-a-um, levaria a um tamanho de passo nulo, caso pudéssemos efetua-lo (vide (2.21), onde o denominador da fração tenderia a ∞, naquele caso). Isso, por si só, traria problemas de convergência ao método. Finalmente, além da razão anterior, é fácil observar, vide (2.23), que desigualdades temporárias inativas e não violadas continuariam inativas, ao final da iteração (ou seja, seus multiplicadores

permanecem iguais a zero). Em resumo, despenderíamos muito esforço computacional, sem benefícios práticos em contra partida.

Algoritmos NDRC tem sido utilizado com certa frequência na literatura. Dentre várias outras, podemos citar suas aplicações ao Problema de Participação Retangular[36],ao Problema de Roteamento de Veículos [37], ao Problema da Mochila 0-1 [38], ao Problema da Árvore de Steiner com Recolha de Prêmios [39] e ao Problema da Árvore Geradora com Restrição de Grau[40]. Uma aplicação similar ao NDRC proposto neste trabalho para a versão simétrica(PVCS), pode ser vista em [41], onde desigualdades do tipo *path inequalities* são dualizadas. Ao passo que o NDRC aqui proposto, dualiza desigualdades do tipo *comb*.

2.4 Dualizações dinâmicas NDRC para o PCVA

Nesta seção, descrevemos as diferentes desigualdades válidas fortes para o PCVA que investigamos para efeito de dualização dinâmica. Tais desigualdades, vale ressaltar, cumprem a mesma função das exponencialmente muitas desigualdades genéricas, (2.24), que utilizamos para descrever um procedimento MS NDRC.

Os procedimentos NDRC que investigamos para o PCVA necessariamente têm, como ponto de partida, uma das duas formulações aqui descritas para o problema. Ou seja, a formulação clássica, (1.6)-(1.9), discutida no Capítulo 1, e a reformulação baseada em r-AGs, (2.9)-(2.13), descrita neste capítulo.

Vale ressaltar que os procedimentos que investigamos para a formulação (1.6)-(1.9), são puramente NDRC. Ou seja, não envolvem a dualização Lagrangiana tradicional de algumas restrições contidas naquela formulação. Mais especificamente, se baseiam diretamente na dualização NDRC de suas exponencialmente muitas Desigualdades de Eliminação de Subrotas (DES), (1.9). Isso difere do que ocorre para a formulação (2.9)-(2.13), onde as restrições (2.11) são sempre dualizadas no esquema tradicional.

Desigualdades válidas que induzem facetas do politopo do PCVS (definido pela envoltória convexa dos pontos do espaço Euclidiano correspondentes a ciclos Hamiltonianos) foram investigadas, dentre outros, em [42, 43], [44] e [45]. Tais desigualdades incluem as DES e as desigualdades *Comb* [42]. Ambas são válidas para o PCVA, já que o PCVA é uma generalização do PCVS. Embora menos investigadas na literatura, desigualdades que induzem facetas específicas do politopo do PCVA, são também conhecidas. Dentre estas, podemos destacar [46, 47],[48], [49] e, em particular, as desigualdades D_k^- [49], que serão aqui consideradas.

2.4.1 Desigualdades de Eliminação de Subrotas

Parte integrante da formulação (1.6)-(1.9), as exponencialmente muitas DES,

$$\sum_{i \in S} \sum_{j \in V \setminus \{i\}} x_{ij} \leq |S| - 1, S \subset V, \tag{2.25}$$

demandam dualização NDRC para que possam operar de forma Lagrangeana plena. Ressaltamos que um esquema Lagrangeano envolvendo um ajuste heurístico de multiplicadores já foi previamente investigado, em [11].

Note que se excluirmos as DES da formulação (1.6)-(1.9), obtemos a formulação de um Problema de Atribuição (PA) [50]. Ou seja, obtemos uma relaxação combinatória clássica para o PCVA. Note ainda que uma solução ótima do PA, sob a mesma função objetivo aplicada ao PCVA, nos leva a um limite inferior para o valor da solução ótima do PCVA.

Obviamente, melhor do que simplesmente eliminar (2.25) da formulação (2.9)-(2.13), como sugerido acima, seria dualiza-las dinâmicamente, num esquema NDRC. Ao fazer isso, obtemos SLs definidos por PAs. Soluções viáveis de um PA, deve-se notar, definem um ou mais circuitos simples de $D = (V, A)$, cada vértice de V contido em apenas um desses circuitos. Quando a solução do PA define um único circuito, este é necessariamente Hamiltoniano e ótimo para o PCVA. Caso contrário, cada circuito implica numa DES violada. Como indicado anteriormente, sendo essas DES ativas ou não, seus multiplicadores devem ter então atualizados.

De um valor corrente igual a 0, se não são ativas, e de um valor maior que 0, no caso contrário.

Experimentos computacionais com um algoritmo NDRC, baseado na formulação (2.9)-(2.13) (e na dualização dinâmica das DES) são descritos no Capítulo 4.

2.4.2 Desigualdades Comb

Desigualdades do tipo *comb* induzem facetas do politopo do PCVS, quando contêm 5 ou mais vértices (vide [42, 43], para detalhes). São assim denominadas em analogia a um pente (ou seja, um *comb*, em inglês), com seu empunhador (um *handle*, em inglês) e dentes (*teeth*, em inglês). Tais desigualdades podem ser definidas como indicamos a seguir.

Assuma que $t \geq 3$ é um número ímpar e que $H \subset V$ e $T_j \subset V$, para $j = 1, ..., t$, são respectivamente o *handle* e os *teeth* da *comb* que vamos definir. Dois teeth não se interceptam e as seguintes condições adicionais são também satisfeitas: $T_j \cap H \neq \emptyset$ e $T_j \setminus H \neq \emptyset$, para $j = 1, ..., t$. Vide Figura 2.3, para uma ilustração de uma desigualdade *comb*.

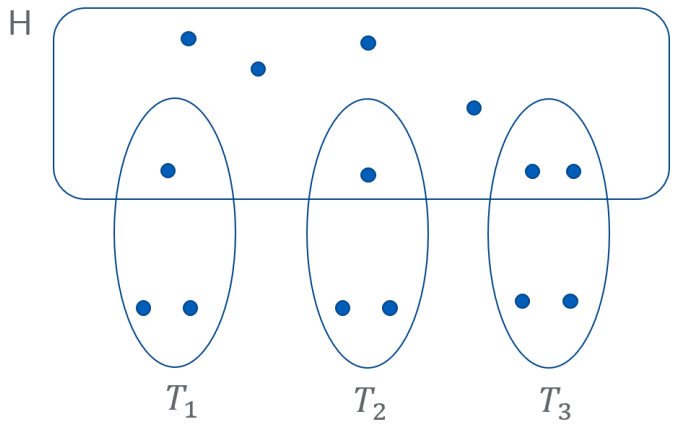

Figura 2.3: Representação de uma *Comb* com três *teeth*.

Desigualdades *comb* são denominadas *Chvátal combs* quando satisfazem, adicionalmente, $T_j \cap H = 1$, para $j = 1, ..., t$[51].

Em sua versão para o PCVA, uma desigualdade *comb* é descrita pela seguinte expressão:

$$\sum_{a \in A(H)} x_a + \sum_{j=1}^{t} \sum_{a \in A(T_j)} x_a \leq |H| + \sum_{j=1}^{t} |T_j| - \lceil \frac{3t}{2} \rceil, \tag{2.26}$$

onde $A(H)$ e $A(T_j)$ definem os conjuntos de arcos com ambas as extremidades em $H \subseteq V$ e $T_j \subseteq V$, respectivamente.

No Capítulo 4, apresentamos resultados computacionais para um algoritmo NDRC baseado na dualização dinâmica de desigualdades *combs*. Tal algoritmo se utiliza da formulação (2.9)-(2.13) e da dualização Lagrangeana tradicional das restrições (2.11). Assim procedendo, soluções viáveis para os SLs são definidas por r-arborescências, acrescidas do arco de menor custo que aponta para r. Por conveniência, utilizamos sempre $r = 1$.

Para ilustrar a identificação dos conjuntos H e T_j, $j = 1, \ldots, t$, considere a solução ótima de um SL indicada na Figura 2.4. Esta tem $(2, 1)$ como o arco apontando para o vértice 1. Dessa maneira, a solução implica num circuito simples e ilegal, formado por $(1, 4), (4, 15), (15, 13), (13, 2)$ e $(2, 1)$. Os vértices desse circuito, ou seja, $\{1, 2, 4, 13, 15\}$, definem o *handle* da nossa desigualdade *comb*. No circuito, todo vértice com grau maior que 2, o que implica em dizer todos eles, nesse caso, apontam para a existência de um *tooth* que os contêm. No caso do vértice 2, por exemplo, o *tooth* a ele associado contém os seguintes vértices $\{2, 10, 11, 17, 18\}$. Tais vértices, vale notar, pertencem a caminhos simples que se iniciam no vértice 2 e não fazem parte do *handle*. Temos então os seguintes *teeth* para a desigualdade *comb* ilustrada na Figura 2.4: $T_1 = \{1, 6\}$, $T_2 = \{4, 7, 8\}$, $T_3 = \{15, 5, 14, 3\}$, $T_4 = \{13, 9, 12, 16\}$ e $T_5 = \{2, 18, 11, 10, 17\}$. Esta, como se pode verificar pela fórmula (2.26), é violada em 4 unidades.

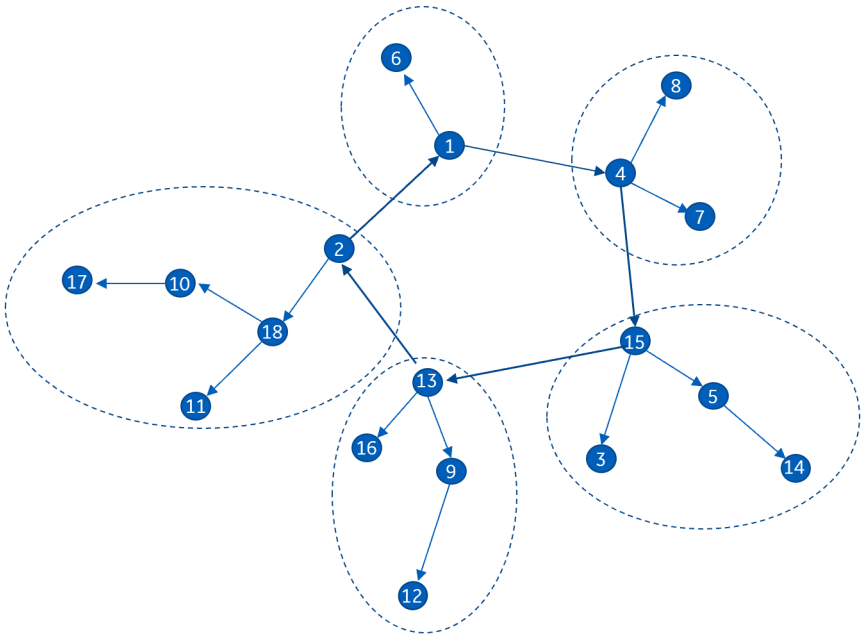

Figura 2.4: Desigualdade *comb* associada a um SL da formulação (2.9)-(2.13).

2.4.3 Desigualdades D_k^-

Assim como desigualdades do tipo *comb*, as desigualdades D_k^+ e D_k^- induzem facetas do politopo para o PCVS(vide[46], para mais detalhes) e foram propostas inicialmente em [43]. Em [46], a família de desigualdades D_k^+ e D_k^- são utilizadas para reforçar a formulação do PCVA.

Na formulação (1.5)-(1.9), apenas as desigualdades D_k^- nos interessam e são dualizadas em conjunto com as DES e *combs* para ajudar a fortalecer os limites duais. Da mesma forma que as DES, as desigualdades D_k^- se aplicam aos subcircuitos gerados pela solução ótima do SL. Cada subcircuito pode levar a tantas desigualdades D_k^-, quanto o número de vértices no subcircuito. No entando, dualizamos apenas uma delas em conjunto com a DES que lhe é correspondente.

A desigualdade D_k^- é descrita pela seguinte expressão:

$$x_{i_k i_1} + \sum_{h=2}^{k} x_{i_{h-1} i h} + 2\sum_{h=2}^{k-1} x_{i_h i_1} + \sum_{h=3}^{k-1} x(i_h, \{i_2, ..., i_{h-1}\}) \leq k - 1 \qquad (2.27)$$

onde $(i_1, ..., i_k)$ é qualquer sequência de $K \in \{3, ..., n-1\}$ nós distintos. D_k^- são desigualdades válidas que podem ser obtidas através do *lifting*(vide [52] para maiores detalhes) de desigualdades de ciclos $\sum_{(i,j)\in C} x_{ij} \leq k - 1$ associadas ao circuito $C := \{(i_1, i_2), ..., (i_{k-1}, i_k), (i_k, i_1)\}$. De uma forma geral, as desigualdades D_k^- podem ser vistas como uma derivação das desigualdades D_k^+ através da troca

dos coeficientes de dois arcos (i, j) e (j, i) para todo $i, j \in V, i<j$(vide [46], para mais detalhes).

Para fins de ilustração, considere o subcircuito gerado pela solução ótima do SL na formulação (1.5)-(1.9), representado na Figura 2.5. Temos então o subcircuito formado pelos arcos (3,5), (5,2), (2,1), (1,8), (8,7) e (7,3). Tome como inicio do subcircuito, ou seja, o vértice i_1 do mesmo, o vértice de menor cardinalidade. Note que o vértice de menor cardinalidade no subcircuito é apresentado pelo vértice 1. Logo, consideramos o subcircuito na seguinte ordenação: (1,8), (8,7) , (7,3), (3,5), (5,2) e (2,1). Ou seja, $k = 6$ e $i_1 = 1$, $i_2 = 8$, $i_3 = 7$, $i_4 = 3$, $i_5 = 5$ e $i_6 = 2$.

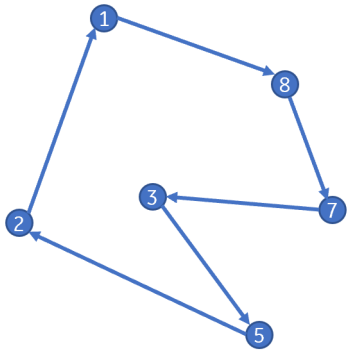

Figura 2.5: Exemplo de um subcircuito gerado pela solução ótima do SL na formulação (1.5)-(1.9).

É importante ressaltar que as desigualdades D_k^- possuem um número menor de arcos que a DES associada. Isso incorre em pesos diferentes, aplicados a alguns dos arcos comuns às duas classes de desigualdades(respectivamente peso 2 e peso 1), contribuindo para o fortalecimento dos limites duais, quando dualizadas em conjunto ao longo da execução do NDRC .

No Capitulo 4, apresentamos os resultados computacionais para o algoritmo NDRC baseado na dualização dinâmica das desigualdades D_k^- combinadas com desigualdades do tipo *comb* e DES.

2.5 Procedimentos de Separação

Como visto anteriormente, o algoritmo NDRC proposto neste estudo é baseado no uso das formulações r-AG (2.9)-(2.13) e clássica (1.5)-(1.9) do PCVA, reforçadas por desigualdades do tipo *comb* e D_k^-. A cada iteração do algoritmo um Subproblema Lagraneano é resolvido e se faz necessário classificar as desigualdades dualizadas em *ativas* e *inativas*. Ou seja, devemos resolver a cada iteração do algoritmo, um problema auxiliar para identificarmos as desigualdades que violam a solução corrente

do SL, ou concluir a inexistência de tais violações. Assim sendo, o problema que devemos resolver é semelhante ao Problema de separação a ser resolvido em um algoritmo B&C, com o grande diferencial de que resolvemos aqui um Problema de Separação sobre uma estrutura inteira. Esta característica, no geral, permite que Problemas de Separação em algoritmos NDRC sejam resolvidos de maneira exata a uma complexidade menor que seus correspondentes algoritmos B&C.

Nesta seção trataremos de três procedimentos de separação de desigualdades associados as reformulações propostas para o PCVA:

1. Separação de Desigualdades do tipo *combs* para formulação r-AG.
2. Separação de Desigualdades do tipo *combs* para formulação do PCVA como um Problema de Atribuição.
3. Separação de Desigualdades D_k^- para formulação do PCVA como um Problema de Atribuição.

Iremos descrever com mais detalhes cada uma das heurísticas propostas a seguir.

2.5.1 Separação de Desigualdades *combs* para a formulação r-AG

Com o intuito identificarmos as desigualdades do tipo *combs* violadas, construímos uma heurística simples que se divide basicamente em duas diferentes etapas:

1. Encontrar o ciclo na solução do SL para formar o conjunto *handle* H.
2. Determinar os vértices que irão compor conjunto dos *teeth* T.

A cada iteração do MS, construímos um grafo suporte correspondente a $\overline{\mathbf{x}}$, onde denotamos por $\overline{D} = (\overline{V}, \overline{A})$ o subgrafo de D induzido pelos vértices $i \in V$ para os quais $\overline{\mathbf{x}}_i = 1$. A primeira etapa do algoritmo(identificação do *handle*) é facilmente concluída com a identificação de um circuito simples e ilegal. Para fins de ilustração, tomemos como exemplo uma solução ótima do SL na formulação (2.9-2.17) representada Figura 2.5, onde o circuito violado é dado pelos vértices $\{1, 4, 15, 13, 2\}$. Consequentemente,é fácil notar que este conjunto define os vértices que fazem parte do *handle*.

Uma vez identificados os vértices que compõem o *handle*, entramos na etapa de identificação dos *teeth*. Para tal, guardamos para todo $i \in \overline{V} \backslash H$ a informação de seus predecessores no subgrafo induzido $\overline{D}$. É importante ressaltar que esta informação já é fornecida pelo algoritmo de Chu-Liu/Edmonds , sem aumentar a

complexidade assintótica do algoritmo, e então utilizada para identificar a que *tooth* T_j o vértice i pertence com uma complexidade O(n).

Para melhor ilustrar esta etapa do procedimento, considere novamente a solução ótima do SL representada na figura 2.5. Considere como T_1, o *tooth* o qual desejamos identificar. A etapa se inicia com a escolha aleatória de um vértice $i \in \overline{V}$, dentre aqueles que não fazem parte do conjunto *handle* . Tomando o vértice 11 como exemplo, temos que seu predecessor em $\overline{D}$ é o vértice 18. Este, por sua vez, tem como seu predecessor o vértice 2. Note aqui, que o vértice 2 faz parte do *handle*, representando assim a intersecção de H com T_1. Assim sendo, os vértices 11,18 e 2 são adicionados a T_1. Neste ponto, o algoritmo prossegue com a escolha aleatória de um outro vértice ainda não atribuído a um *tooth* T_j e continua até que não existam mais vértices a serem escolhidos.

Se durante a execução do MS, o algoritmo encontra um cenário no qual o número de *teeth* é par , tal que $t > 3$, um procedimento adicional é executado para que dois *teeth* consecutivos sejam aglutinados em um único. Neste trabalho, definimos dois *teeth* consecutivos como sendo aqueles ligados por um arco $(k, l) \in H$, tal que $k \in T_j$ e $l \in T_{j+1}$. Para concluirmos, vale a observação de que no caso citado acima, o arco (k, l) apresentará coeficiente com peso 2 na desigualdade dualizada.

2.5.2 Separação de Desigualdades do tipo *combs* para formulação do PCVA como um Problema de Atribuição

O procedimento que descrito nesta seção, é aplicado apenas a formulação do Problema de Atribuição, ou seja , a formulação clássica (1.5-1.9) excluindo-se as DES (1.8), uma vez que a solução ótima do SL define um ou mais circuitos simples.

Diferentemente do procedimento de separação descrito em 2.5.1, a etapa inicial deste procedimento consiste na identificação dos *teeth*. A cada iteração do MS o número de desigualdades de subrotas violadas é identificado. Estas desigualdades, por sua vez, são definidas pelos t circuitos simples formados pela solução ótima do SL. Sendo $t \geq 3$ um número ímpar, denotamos por T_j, para $j = 1, ..., t$ os conjuntos de vértices que compõem os *teeth*. Consequentemente, representados por cada um dos circuitos simples presentes na solução do SL.

A segunda etapa da heurística consiste na identificação dos vértices que compõem o *handle*. Para tal, ordenamos o conjunto dos *teeth*, previamente identificados, associando o índice $j = 1$ àquele de maior cardinalidade. Como estratégia adotada para o passo seguinte, é feita uma busca local pelos arcos (k, l) com menor custo reduzido que ligam dois *teeth* consecutivos, tal que $k \in |T_j|$ e $l \in |T_{j+1}|$. O procedimento descrito acima, determinará um novo circuito ligando o conjunto dos *teeth*.

Este ciclo por sua vez, contem o conjunto do vértices que fazem parte do *handle*.

Assim como no procedimento descrito em 2.5.1, ao decorrer da execução do MS, podemos nos deparar com um caso no qual $t > 3$, porém par. Neste caso, unimos dois circuitos em um único, como um passo anterior a busca local que nos levará a identificação do *handle*. Este procedimento se divide em duas etapas. A primeira etapa consiste na escolha dos dois circuitos a serem unidos em um único. Para tal, tomamos os circuitos de menor e maior cardinalidade dentre os circuitos presentes na solução do SL. A segunda etapa consiste em uma busca local para a identificação das arestas que ligam os dois circuitos a um menor custo possível.

2.5.3 Separação de Desigualdades D_k^- para formulação do PCVA como um Problema de Atribuição

Assim como descrito em 2.5.2, o procedimento descrito a seguir é aplicado somente a reformulação do PCVA como um PA. O algoritmos de separação de desigualdades do tipo D_k^- é um dos mais simples dentre os descritos anteriormente. Isto se deve ao fato de que o algoritmo baseia-se unicamente na identificação circuitos simples gerados pela solução do SL.

Podemos dividir o Problema de Separação de desigualdades do tipo D_k^- em duas etapas:

1. Identificação dos subcircuitos gerados pela Solução do SL.
2. Ordenação dos vértices dos subcircuitos.

De uma forma geral, podemos dizer que a solução Problema de Separação de DES é uma das etapas do algoritmo de separação das desigualdades do tipo D_k^-. Como descrito em 2.4.3, esta afirmação se basea no fato de que as desigualdades D_k^- são dualizadas em conjunto com as DES.

O procedimento de Separação das DES, consiste apenas na identificação dos vértices que estão contidos em cada circuito simples presente na solução do SL. Basicamente, cada circuito simples possui uma DES associada e consequentemente violada de 1 unidade. A etapa final requerida para a separação de uma desigualdade do tipo D_k^-, consiste apenas na ordenação dos vértices presentes no subcircuito associado a uma DES a ser dualizada.

2.6 Fixação de Variáveis

Dada a dificuldade de obtermos a solução ótima para um problema de programação inteira 0-1, é de boa prática que tentemos eliminar variáveis que irão assumir valor 0 ou 1 em qualquer solução ótima. Uma das alternativas para a eliminação destas variáveis é a aplicação de um prodecimento de fixação de variáveis baseada na análise dos custos reduzidos.

Visando a economia do tempo de CPU, o procedimento de fixação de variáveis que será descrito aqui somente é executado no momento em que ocorre uma melhora do limite inferior até então conhecido. As variáveis utilizadas para a fixação são o limite inferior Z_{li} na iteração em que a melhora ocorreu, o melhor limite superior Z_{ls} obtido até então e o custo reduzido $\zeta(i,j)$ associado ao arco $(i,j) \in A$. Os custos reduzidos são fornecidos pelo algoritmo proposto em [53] para a solução do PA.

Considere o problema de encontrar um circuito Hamiltoniano viável para o PCVA que contenha o arco (i,j). Considere ainda, que tal circuito tenha o menor custo possível. Podemos notar que, a solução de tal problema representa o mesmo nível de dificuldade que o problema original. Entretanto, temos que pela definição de custos reduzidos em Programação Linear, $Z_{li} + \zeta(i,j)$ representa um limite inferior válido para o novo problema. Dito isto, podemos afirmar que qualquer solução viável para o PCVA , que contenha (i,j), tem um custo maior que o de uma solução viável Z_{ls} se:

$$Z_{li} + \zeta(i,j) \geq Z_{ls} \tag{2.28}$$

Com base no exposto acima, concluímos que o arco (i,j) não faz parte da solução ótima do problema e consequentemente tem seu valor fixado em 0.

Capítulo 3

Heurísticas Primais

Neste capítulo, apresentamos as heurísticas primais utilizadas para a obtenção de soluções viáveis para o PCVA. Basicamente, são propostas duas heurísticas distintas correspondentes às reformulações apresentadas no Capítulo 2. Ou seja, o PCVA reformulado como uma r-arborescência (2.9)-(2.13) e a reformulação do PCVA como um Problema de Atribuição (PA). Esta última, representada na formulação clássica (1.5)-(1.9) exluindo-se as DES.

De uma forma geral, ambas heurísticas se beneficiam da informação dual Lagraneana na tentativa de produzir soluções viáveis de boa qualidade a cada iteração do MS. Como tentativa de escaparmos de ótimos locais, estas soluções são posteriormente submetidas a um método de busca local. Este método é aplicado de forma idêntica às duas heurísticas e será descrito posteriormente ao final deste capítulo.

Uma vez que o principal objetivo das heurísticas propostas é se beneficiar da informação dual Lagraneana, se faz necessário olhar para a estrutura das soluções produzidas pelo SL. Esta estrutura, deve ser então considerada como um critério para a escolha da estratégia adotada por cada heurística, na construção de soluções viáveis e de boa qualidade para o problema.

Para caso da formulação do PCVA como uma r-arborescência, temos a cada iteração do MS, uma r-AG de custo mínimo com a adição do arco de menor custo apontando para r. Invariavelmente, podemos enxergar a solução do SL como um conjunto de duas estruturas distintas conexas. a) Um circuito simples C e b) um conjunto de k subárvores T_i, tal que $T_i \subset V$ e $T_i \cap C = 1$, para $i = 1, ..., k$.

Dada estrutura descrita acima, optou-se para formulação em questão, pela estratégia de expansão do circuito C através do uso de uma heurística gulosa baseada em inserção mais barata (*cheapest insertion*, em inglês) [54].

Para o caso do PCVA reformulado como um Problema de Atribuição, a solução do SL produz naturalmente um conjunto C de circuitos simples. A estratégia adotada neste caso foi de unir estes circuitos, dois a dois, em um procedimento de *patching* [55] até que um circuito hamiltoniano seja obtido ao final do procedimento.

A seguir, daremos mais detalhes de como estas heurísticas foram implementadas, assim como, o procedimento de busca local.

3.1 *Cheapest Insertion*

Considere $\overline{\mathbf{x}}$ como a solução ótima do SL para formulação r-AG do PCVA. Considere ainda, o grafo suporte $\overline{D} = (\overline{V}, \overline{A})$, ou seja, subgrafo de D induzido pelos vértices $i \in V$ para os quais $\overline{\mathbf{x}}_i = 1$. Sendo C uma *subrota* em $\overline{D}$, formada por um subconjunto V' de vértices, tal que $V' \subset \overline{V}$, a proposta da heurística implementada é a de se expandir C progressivamente, até que se obtenha um circuito hamiltoniano composto por todos os vértices de $\overline{D}$.

Para tal, tomemos como k o vértice a ser inserido em C, tal que $k \in S$ e $S \subset \overline{V} \backslash V'$. Definimos como $\mathbf{TOUR}(C, k)$, uma subrota a ser obtida através da inserção do vértice k em C. Adicionalmente, temos que o custo de inserção do vértice k na subrota C é dado por:

$$\mathbf{COST}(C, k) = c_{ik} + c_{kj} - c_{ij} \tag{3.1}$$

Logo, o procedimento de expansão da subrota C através da inserção de um vértice $k \in S$, segue à cada iteração as duas etapas descritas a seguir :

1. Encontrar o arco (i, j) em C que minimiza o custo da inserção de k:

$$\mathbf{TOUR}(C, k) = \min\{\mathbf{COST}(C, k) \ ; \ k \in S\} \tag{3.2}$$

2. Deletar o arco (i, j) e adicionar os arcos (i, k) e (k, j) em C:

O procedimento é então finalizado, quando não existem mais vértices do subconjunto S a serem adicionados em C. Obviamente, este é um procedimento guloso que tende a nos levar a ótimo local. Entretanto, a medida que avançamos na execução do MS, é esperado que o número de vértices presentes no subconjunto S seja menor do que na iterações iniciais, nos levando assim a um cenário mais próximo da solução ótima.

Com o objetivo de melhorarmos a solução e escapar de um ótimo local, submete-se o circuito obtido como resultado da aplicação da heurística a um procedimento de busca local que será descrito ao final deste capítulo. O pseudocodigo referente a implementação da rotina *cheapest Insertion* pode ser encontrado no Algoritmo 1.

Algorithm 1: Heurística *Cheapest Insertion*

Require: $\overline{D} = (\overline{V}, \overline{A})$, C and S
Ensure: C is a Hamiltonian cycle
  while $S \neq \emptyset$ **do**
    for $k \in S$ **do**
      $BestInsCost \leftarrow \inf$
      for $(i, j) \in C$ **do**
        $currentCost = COST(C, k)$
        if $currentCost < BestInsCost$ **then**
          $BestInsCost \leftarrow currentCost$
          $BestTour \leftarrow TOUR(C, k)$
        end if
      end for
      $C \leftarrow BestTour$
      $S \leftarrow S \backslash \{k\}$
    end for
  end while
  return C

3.2 *Patching Algorithm*

Para produzirmos soluções viáveis para a formulação do PCVA como um Problema de Atribuição, implementamos o algoritmo de *patching* descrito em [55], com uma pequena adaptação.

Seja $D = (V, A)$ um grafo direcionado, V o conjunto de seus vértices e A o conjunto de seus arcos. Denotamos por $F \subset A$ uma cobertura de ciclos mínima determinada pela solução do Problema de Atribuição.

O algoritmo de *patching* consiste em um conjunto de operações que implica em remover um par de arcos de F e adicionar um par de arcos de $A \backslash F$, tal que ao final obtenhamos um circuito Hamiltoniano $H \subset A$. Assim sendo, o custo de um procedimento P de *patching* é denotado por c_p e definido por:

$$c_p(F) = \sum_{a \in H \backslash F} c(a) - \sum_{b \in F \backslash H} c(b) \tag{3.3}$$

onde $c(a)$ é a soma dos custos dos arcos introduzidos por P e $c(b)$ indica a soma dos custos arcos removidos.

A Figura 3.1 ilustra o procedimento para dois ciclos C_1 e C_2, onde o par de arcos (i_1, j_1) e (i_2, j_2) são removidos e o par de arcos (i_1, j_2) e (i_2, j_1) são adicionados para

formar um novo ciclo.

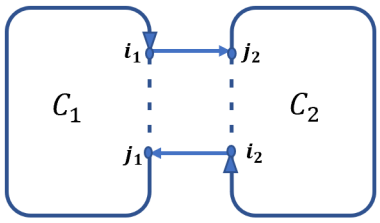

Figura 3.1: Prodecimento de *patching* nos circuitos C_1 e C_2

A heurística se inicia tomando os dois maiores ciclos de F. O passo seguinte, visa encontrar os pares de arcos a e b, que minimizam $c_p(F)$. O novo circuito gerado é então unido , através de uma nova operação de *patching*, ao próximo maior circuito de F. O procedimento é encerrado quando não existem mais circuitos a serem unidos, ou seja, quando um circuito Hamiltoniano é obtido.

Neste trabalho, uma pequena modificação foi feita com relação a proposta original da heurística. Ao invés dos custos originais serem utilizados para a escolha dos arcos que entrariam na solução, consideramos aqueles em que a soma de seus custos reduzidos é a menor possível. O pseudocodigo referente a implementação da heurística de *patching* para o PCVA pode ser encontrado no Algoritmo 2.

Algorithm 2: Heurística *Patching*

Require: $\overline{D} = (\overline{V}, \overline{A})$ and F
  $H \leftarrow \emptyset$
  while H não é um circuito Hamiltoniano **do**
    Seja C_1 e C_2 os maiores ciclos em F
    Escolher $a \in A$ que minimiza $c_p(F)$
    $C \leftarrow C_1 \cup C_2$
    $F \leftarrow F \backslash C_1$
    $F \leftarrow F \backslash C_2$
    $F \leftarrow F \cup C$
    $H \leftarrow C$
  end while
  return H

3.3 Busca Local

Após a aplicação das heurísticas descritas nas Seções 3.1 e 3.2, se faz necessário submeter as soluções resultantes a um procedimento de busca local, com o intuito de refinarmos a solução, fugindo de eventuais ótimos locais.

De acordo descrito em [21], os métodos de melhoria mais utilizados são do tipo k-opt, nos quais k arcos são removidos de uma solução S_0 conhecida e os segmentos resultantes são reconectados com outros k arcos, com a finalidade de se encontrar uma solução de menor custo. De uma forma geral, quanto maior o valor de k, maior a precisão do método, porém a custo computacional mais elevado [56].

Para o PCV, os métodos mais utilizados são o 2-opt e o 3-opt. Onde respectivamente, dois e três arcos são desconectados na tentativa de serem produzidas soluções de menor custo.

Neste trabalho, considerando a característica assimétrica do problema, optou-se pelo movimento 3-opt. Daremos a seguir, os detalhes que justificam tal escolha.

Considere o movimento 2-opt ilustrado na Figura 3.2. No exemplo em questão, dois arcos (x_1, x_2) e (x_3, x_4) são removidos e substituídos pelos arcos (x_1, x_4) e (x_2, x_3) para formar um novo circuito. Note que o seguimento antes delimitado pelos vértices x_2 e x_3, teve seu sentido invertido na nova solução. Consequentemente, isto implica em um esforço computacional adicional no cálculo do custo deste seguimento.

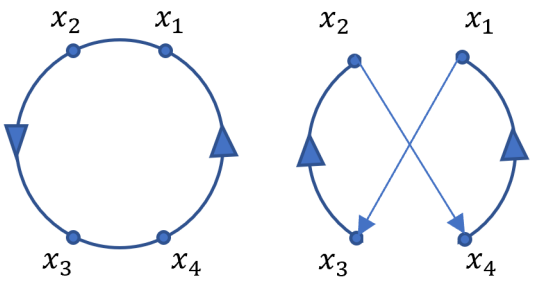

Figura 3.2: Movimento 2-opt

Considere agora o movimento 3-opt ilustrado na Figura 3.3. Note que a remoção de três arcos, no caso em questão (x_2, x_3), (x_4, x_5) e (x_6, x_1), possibilita sete maneiras diferentes de reconexão para a produção de uma novas soluções. Note ainda, que dentre as soluções produzidas, apenas uma não implica em reversão de arcos.

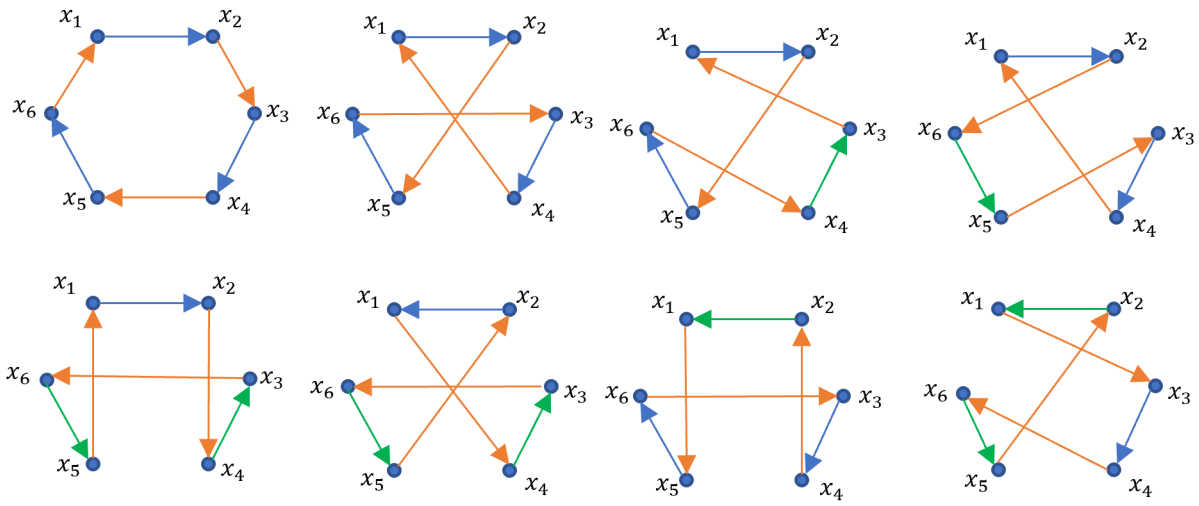

Figura 3.3: Movimento 3-opt

Com base no descrito acima, optou-se neste trabalho pela implementação de uma heurística de busca local baseada em movimentos 3-opt, limitando os movimentos àqueles que não implicam em reversão de segmentos. Ou seja, a cada vez que três arcos são desconectados, apenas a única solução que não implica em reversão de arcos é considerada com o objetivo de se reduzir o esforço computacional.

Adicionalmente, com o intuito de se reduzir as possibilidades de arcos a serem considerados, definimos um conjunto Q de arcos a serem removidos de S_0, tal que $Q \subset S_0$. Para selecionarmos quais arcos farão deste conjunto, tomamos inicialmente a distribuição dos custos de todos os arcos presentes na solução. A seguir, o valor da mediana desta distribuição é utilizado como parte do critério de seleção. Onde os m arcos com custo superior a esse valor são sorteados para fazer parte do conjunto Q. Por fim, os elementos de Q são combinados três a três para que o procedimento de busca por melhores soluções seja executado.

Durante teste computacionais que produziram os resultados que serão apresentados no Capítulo 4 , vimos empiricamente que um valor de $m = \frac{|V|}{4}$ produzia melhores resultados. O Algoritmo 3, nos traz o pseudocódigo referente ao procedimento de busca local implementado.

Algorithm 3: Busca Local 3-opt

Require: S_0

$thrs \leftarrow supQtl(S_0)$

$Q \leftarrow \emptyset$

$S \leftarrow S_0$

for $(i, j) \in S_0$ **do**

  if $c(i, j) \geq thrs$ **then**

    $Q.insert(i, j)$

  end if

end for

$BestCost \leftarrow Cost(S_0)$

for $\{(i_1, j_1), (i_2, j_2), (i_3, j_3)\} \in Q$ **do**

  $S_1 \leftarrow S_0.remove(\{(i_1, j_1), (i_2, j_2), (i_3, j_3)\})$

  $S_1 \leftarrow S_1.insert(\{(i_1, j_2), (i_3, j_1), (i_2, j_3)\})$

  if $Cost(S_1) \leq BestCost$ **then**

    $BestCost \leftarrow Cost(S_1)$

    $S \leftarrow Cost(S_1)$

  end if

end for

return S

Capítulo 4

Resultados e Discussões

Ao longo desta dissertação, apresentamos no Capítulo 2, a descrição do algoritmo NDRC aplicado as formulações (2.9-2.13) e (1.5-1.9), bem como as heurísticas descritas no Capítulo 3, que foram implementadas para a produção de soluções viáveis para o PCVA.

Neste capítulo, apresentamos os resultados dos testes computacionais para ambas formulações propostas neste trabalho, considerando algumas variantes para fins de comparação. Para tal, adotaremos daqui para frente, as seguintes nomeclaturas:

- **NDRC-PA**: Algoritmo NDRC para a formulação do PCVA como um problema de atribuição, considerando apenas a dualização dinâmica de DES.
- **NDRC-PA-CDK**: Algoritmo NDRC para a formulação do PCVA como um problema de atribuição, considerando a dualização dinâmica de DES, desigualdades do tipo *comb* e D_k^-.
- **NDRC-r-AG**: Algoritmo NDRC para a formulação do PCVA como uma 1-AG, considerando a dualização dinâmica de desigualdades do tipo *comb*.

Todos os experimentos computacionais descritos nestes trabalho, foram executados em um computador equipado com um processador Intel I-77500U, com velocidade de 2.7GHz , 16GB de memória RAM em ambiente Linux. Todos os códigos foram escritos na linguagem de programação Python.

4.1 Instâncias para o PCVA

Para os testes computacionais apresentados neste trabalho, consideramos três conjuntos distintos de instâncias.

O primeiro conjunto é representado por 47 instâncias de aplicações reais disponíveis no TSPLIB. Dentre estas, podemos citar como exemplo as duas instâncias

ft que correspondem a um problema de sequenciamento ótimo de tarefas em uma planta de produção de resinas, conforme descrito em [57]. As 17 instâncias **ftv** são referentes a um problema de entregas da industria farmacêutica na cidade de Bolonha [58, 59]. As instâncias **ry48** e **kro124p** são instâncias Euclidianas simétricas, renderizadas como assimétricas, através de perturbações aplicadas às suas matrizes de distâncias através do procedimento descrito em [57]. As quatro instâncias **rbg** são referentes a um problema de movimentação de transelevadores [15]. A instância **p43** é originária de um problema de *scheduling* da industria de engenharia química. As instâncias adicionais remetentes a problemas de sequenciamento e aplicações em robótica, vem de oito diferentes fontes como descrito em [15].

Adicionalmente, um segundo conjunto de 126 instâncias SOLER retiradas do trabalho de David Soler [60], foram utilizadas nos testes e podem ser obtidas em `http://akira.ruc.dk/~keld/research/LKH/`.

4.2 Resultados Algoritmos NDRC

Para os algoritmos implementados neste trabalho, os resultados que serão apresentados nesta seção foram obtidos sob os seguintes parâmetros:

- $MaxIter = 5000$ (Número Máximo de iterações)
- $\alpha = 2$
- $MaxHalf = 300$ (Número de iterações sem melhoria do limite dual no qual o valor de α é reduzido a metade)

As Tabelas 4.1 e 4.2, trazem os resultados comparativos das versões implementadas do algoritmo para o conjunto de instâncias TSPLIB e SOLER, respectivamente. Os resultados apresentados remetem a 10 execuções dos algoritmos para cada instância considerada.

Os resultados são apresentados ao longo de 14 colunas dividias em dois blocos distintos: a) Informações relativas as instâncias, que incluem suas identificações de acordo com suas fontes, o número de vértices(ou cidades) e o valor ótimo conhecido e b) resultados contendo os valores do limite inferior e superior, diferença percentual entre ambos e o tempo de execução para cada reformulação do PCVA considerada, sob os seguintes rótulos para cada coluna:

- $|V|$: Número de vértices do problema.
- **Ótimo**: Valor ótimo conhecido.
- **LI**: Limite Inferior médio obtido na execução dos algoritmos.

- **LS**: Limite Superior médio obtido na execução dos algoritmos.
- **Gap(%)**: Diferença percentual entre o limite inferior e limite superior.
- **t(s)**: Tempo de execução em segundos.

			NDRC-PA				NDRC-PA-CDK				NDRC-r-AG			
Instâncias	$\|V\|$	Ótimo	LI	LS	Gap(%)	t(s)	LI	LS	Gap(%)	t(s)	LI	LS	Gap(%)	t(s)
atex1	16	1812	1150	1819	36.78	11.0	1812	1816	0.22	32.7	1811	**1812**	Opt	15.2
br17	17	39	39	**39**	Opt	0.5	39	**39**	Opt	1.0	39	**39**	Opt	12.3
atex3	32	2952	2952	**2952**	Opt	2.1	2952	**2952**	Opt	4.1	2951	2960	0.30	32.5
ftv33.atsp	33	1286	1286	**1286**	Opt	1.9	1286	**1286**	Opt	7.7	1285	1371	6.27	17.2
ftv35	36	1473	1457	**1475**	1.22	5.2	1457	**1475**	1.22	7.3	1457	1565	6.90	19.0
ftv38	39	1530	1530	**1530**	Opt	23.0	1530	**1530**	Opt	35.1	1514	1676	9.67	27.2
p43	43	5620	5610	5621	0.20	21.3	5612	**5621**	0.16	44.5	5610	5621	0.20	42.3
ftv44	45	1613	1584	**1613**	1.80	31.2	1584	**1613**	1.80	43.2	1584	1727	8.28	36.8
ftv47	48	1776	1776	**1776**	Opt	55.6	1776	**1776**	Opt	73.3	1748	1823	4.11	25.6
ry48p	48	14422	14422	**14422**	Opt	3.9	14422	**14422**	Opt	5.4	14289	14955	4.45	62.7
atex4	48	3218	3019	3296	8.40	12.0	3115	**3222**	3.32	12.3	3115	4412	29.40	65.0
ft53	53	6905	6905	**6905**	Opt	11.2	6905	**6905**	Opt	23.7	6905	**6905**	Opt	40.6
ftv55	56	1608	1583	**1608**	1.55	12.6	1583	**1608**	1.55	23.7	1583	2333	32.15	51.2
ftv64	65	1839	1807	**1839**	1.74	65.3	1807	**1839**	1.74	27.5	1807	2301	21.47	98.7
ft70	70	38673	38652	38677	0.06	35.1	38661	**38675**	0.04	87.3	38652	39453	2.03	89.7
ftv70	71	1950	1908	**1950**	2.15	2.9	1908	**1950**	2.15	123.4	1908	2600	26.62	79.0
atex5	72	5269	5052	5748	12.11	307.0	5056	**5651**	10.53	731.0	4797	6752	28.95	85.0
ftv90	91	1579	1569	1591	1.38	42.3	1571	**1589**	1.13	89.3	1569	1745	10.09	115.1
Td100.1	100	268636	268636	**268636**	Opt	23.1	268636	**268636**	Opt	35.8	244736	276532	11.50	98.0
ftv100	101	1788	1767	1804	2.05	35.0	1767	**1799**	1.78	76.4	1767	2324	23.97	88.0
ftv110	111	1958	1923	**1992**	3.46	38.8	1945	1995	2.51	60.3	1923	2712	29.09	94.0
dc112	112	11109	11091	**11118**	0.24	27.5	11091	**11118**	0.24	55.3	11088	11182	0.84	110.0
ftv120	121	2166	2130	**2206**	3.45	43.2	2130	**2206**	3.45	67.9	2127	3316	35.86	65.7
dc126	126	123235	123195	123383	0.15	35.7	123201	**123273**	0.06	78.5	123045	123258	0.17	54.3
ftv130	131	2307	2286	2345	2.52	51.8	2301	**2338**	1.58	112.9	2282	3822	40.29	177.4
dc134	134	5612	5612	**5612**	Opt	6.8	5612	**5612**	Opt	12.5	5608	5666	1.02	34.5
ftv140	141	2420	2399	**2461**	2.52	71.4	2399	**2461**	2.52	167.3	2397	4076	41.19	212.3
ftv150	150	2611	2590	2633	1.63	84.3	2590	**2630**	1.52	110.7	2324	5816	60.04	128.9
ftv160	160	2683	2646	2696	1.85	95.4	2646	**2690**	1.64	120.3	2601	5876	55.74	134.5
ftv170	170	2755	2715	2768	1.91	106.0	2718	**2761**	1.56	230.5	2750	3001	8.36	341.3
kro124p	171	36230	35992	36674	1.86	42.5	36101	**36624**	1.43	87.6	34504	39755	13.21	101.3
dc176	176	8587	8584	**8587**	0.03	62.0	8584	**8587**	0.03	128.1	8582	8656	0.85	143.1
dc188	188	10225	10221	10238	0.17	71.0	10221	**10227**	0.06	132.3	10220	10230	0.10	123.5
code198	198	4541	4541	**4541**	Opt	13.0	4541	**4541**	Opt	27.6	4460	4451	Opt	45.6
code253	253	106957	106957	106957	Opt	312.0	106957	**106957**	Opt	512.0	106755	107011	0.24	764.2
Td316.10	317	691502	691502	**691502**	Opt	47.3	691502	**691502**	Opt	85.2	679051	692012	1.87	98.6
rbg323	323	1326	1326	1326	Opt	1.2	1326	**1326**	Opt	2.4	1324	1390	4.75	450.3
rbg358	358	1163	1163	1163	Opt	1.7	1163	**1163**	Opt	2.9	1150	1179	2.43	334.2
rbg403	403	2465	2465	**2465**	Opt	2.4	2465	**2465**	Opt	3.7	2463	2501	1.51	574.3
rbg443	443	2720	2720	**2720**	Opt	2.7	2720	**2720**	Opt	4.1	2715	2770	1.99	332.1
dc563	563	25951	25948	25962	0.05	558.0	25950	**25955**	0.02	1213.5	25904	25983	0.30	679.2
atex8	600	39982	36269	46894	22.66	2150.0	39117	**43917**	10.93	4438.0	35587	40021	11.08	12333.0
big702	702	79081	79081	79081	Opt	18.1	79081	**79081**	Opt	21.1	76664	79181	3.18	34.5
dc849	849	37476	37474	37479	0.01	3592.0	37474	**37481**	0.02	5793.0	37211	37545	0.89	6789.3
dc895	895	107699	107652	**107811**	0.15	1130.0	107677	107798	0.11	6234.1	107253	107912	0.61	7689.1
dc932	932	479900	470782	481580	2.24	5312.0	470805	**481217**	2.16	6672.0	470132	481325	2.33	7864.1
Td1000.20	1001	1242183	1242183	**1242183**	Opt	89.3	1242183	**1242183**	Opt	211.3	1241576	1243315	0.14	324.5

Tabela 4.1: Resultados dos testes computacionais nas instâncias TSPLIB

Com base nos resultados apresentados na Tabela 4.1, obtivemos certificados de otimalidade em 18 das 47 instâncias do TSPLIB para os algoritmos NDRC-PA e NDRC-PA-CDK. Já o algortimo NDRC-r-AG, apresentou de uma forma geral resultados inferiores quando comparado aos outros algoritmos, obtendo certificados de otimalidade para apenas 3 das instâncias do TSPLIB consideradas.

Um ponto importante a destacar é o comportamento do algoritmo nas instâncias da família **ftv**, onde o *gap* médio foi de 24.71%. Não obstante, podemos considerar que tanto este grupo de instâncias, quanto o da família de instâncias **atex**, apresentaram o maior nível de dificuldade para todas as versões dos algoritmos NDRC implementados neste trabalho. Para o NDRC-PA, o *gap* médio considerando apenas estas instâncias foi de 6.07%. Já para o NDRC-PA-CDK, o gap médio foi de 2.84%.

A diferença de performance entre as duas formulações propostas, se deve basicamente as diferentes heurísticas primais implementadas e a dualização conjunta de DES, *combs* e desigualdades do tipo D_k^-. No caso da formulação do PCVA como uma r-arborescência, podemos ver que da forma como implementado, as desigualdades do tipo *comb* não contribuíram para a melhoria dos limites duais. Adicionalmente, a heurística de inserção mais barata não se mostrou eficiente em se beneficiar destes mesmos limites.

Considerando todas as instâncias do TSPLIB, o algoritmo NDRC-PA-CDK obteve a melhor performance com um gap médio de 1.91%. Ao passo que o NDRC-PA, obteve como resultado um gap médio de 3.94%. Uma vez que ambos algoritmos representam a reformulação do PCVA como um problema de atribuição, podemos constatar o impacto da dualização dinâmica das desigualdades do tipo *comb* e do tipo D_k^- nos resultados apresentados pelo algoritmo NDRC-PA-CDK. Nas instâncias **atex1** e **atex8** a contribuição das desigualdades válidas dualizadas ficou um pouco mais evidente. Para a instância **atex1**, o NDRC-PA apresentou um *gap* médio de 36%, enquanto NDRC-PA-CDK performou com um *gap* médio de 0.22% atingindo o valor do ótimo pelo limite inferior em todas as execuções. No caso da instância **atex8**, a aplicação do NDRC-PA resultou em um *gap* médio de 22%, ao passo que o NDRC-PA-CDK resultou em um *gap* de 10%.

Em se tratando de tempos de execução, o valor médio para todas as instâncias foi de 5 minutos para o algoritmo NDRC-PA e de 9 minutos para o caso do NDRC-PA-CDK. O tempo adicional se deve basicamente aos procedimentos de separação implementados para a dualização das desigualdades válidas consideradas. Já o NDRC-r-AG se apresentou a performance considerando um tempo médio de execução de 14 minutos.

			NDRC-PA				NDRC-PA-CDK				NDRC-r-AG			
Instâncias	$\|V\|$	Ótimo	LI	LS	Gap(%)	t(s)	LI	LS	Gap(%)	t(s)	LI	LS	Gap(%)	t(s)
D41140	64	417880	417880	**417880**	Opt	2.00	417880	**417880**	Opt	4.71	416290	432322	3.71	32.36
D61440	77	512674	512674	**512674**	Opt	1.69	512674	**512674**	Opt	2.23	511034	529894	3.56	48.04
D4940m	51	336325	336324	**336325**	Opt	1.47	336324	**336325**	Opt	2.10	336219.4	340836	1.35	20.18
D81740m	102	664613	664612	**664613**	Opt	6.34	664612	**664613**	Opt	9.06	662159	700072	5.42	86.26
D4940	67	422508	422507	**422508**	Opt	2.69	422507	**422508**	Opt	6.33	421111	468346	10.09	36.22
D81740	129	809706	809705	**809706**	Opt	18.54	809705	**809706**	Opt	26.51	807383	858887	6.00	136.53
D82040m	103	689971	689971	**689971**	Opt	12.13	689971	**689971**	Opt	16.02	689625.2	700334	1.53	85.54
D102240m	121	803595	803595	**803595**	Opt	14.30	803595	**803595**	Opt	20.44	803129.2	824799	2.63	121.03
D82040	122	790971	790971	**790971**	Opt	25.05	790971	**790971**	Opt	58.88	788563	853666	7.63	122.71
D102240	163	1027751	1027750	**1027751**	Opt	35.84	1027750	**1027751**	Opt	51.25	1027555	1046624	1.82	215.41
D102640m	118	815096	815096	**815096**	Opt	32.69	815096	**815096**	Opt	43.16	814679.5	825073	1.26	115.35
D122640m	130	880335	880335	**880335**	Opt	101.23	880335	**880335**	Opt	144.76	880212.6	893593	1.50	139.89
D122940m	148	997772	997771	**997772**	Opt	30.45	997771	**997772**	Opt	71.56	997386.9	1027420	2.92	187.33
D102640	139	926142	926141	**926142**	Opt	28.20	926141	**926142**	Opt	40.33	926001.5	939957	1.48	163.87
D122640	150	990448	990447	**990448**	Opt	34.50	990447	**990448**	Opt	49.34	990281.2	1018367	2.76	190.25
D122940	209	1316922	1316921	**1316922**	Opt	89.44	1316922	**1316922**	Opt	178.88	1312157	1383525	5.16	398.16
D143040a	161	1072673	1072672	**1072673**	Opt	50.66	1072673	**1072673**	Opt	72.44	1072575	1091589	1.74	220.62
D143340	230	1452339	1452338	**1452339**	Opt	101.75	1452338	**1452339**	Opt	134.31	1446782	1512817	4.37	492.68
D143340m	162	1100177	1100176	**1100177**	Opt	25.31	1100176	**1100177**	Opt	36.19	1099835	1122753	2.04	231.18
D143040	236	1472328	1472327	1472330	0.00	120.30	1472328	**1472328**	Opt	282.71	1468470	1528029	3.90	497.30
D143040b	186	1210175	1210174	**1210175**	Opt	57.95	1210174	**1210175**	Opt	82.86	1206404	1307154	7.71	311.86
D163440a	169	1145914	1145913	**1145914**	Opt	97.63	1145913	**1145914**	Opt	195.25	1144940	1184185	3.31	249.67
D163440	226	1452914	1452914	**1452914**	Opt	77.23	1452914	**1452914**	Opt	110.44	1449122	1522141	4.80	478.63
D163440b	200	1310767	1310766	**1310767**	Opt	44.40	1310766	**1310767**	Opt	104.34	1306977	1426893	8.40	382.32
D183840a	184	1255781	1255781	1255815	0.00	287.14	1255781	**1255781**	Opt	379.02	1255520	1277822	1.75	301.00
D163740	226	1468709	1468709	**1468709**	Opt	282.12	1468709	**1468709**	Opt	564.24	1463533	1585302	7.68	479.63
D183840	258	1652091	1652090	**1652091**	Opt	78.23	1652090	**1652091**	Opt	111.87	1650935	1767737	6.61	231.57
D183840b	220	1449912	1449912	**1449912**	Opt	93.09	1449912	**1449912**	Opt	133.11	1444547	1507908	4.20	271.55
D204240a	213	1438448	1438447	**1438448**	Opt	46.38	1438447	**1438448**	Opt	92.76	1436434	1524755	5.79	191.08
D184040a	196	1331558	1331557	**1331558**	Opt	80.63	1331557	**1331558**	Opt	115.30	1329694	1411451	5.79	237.52
D204240	315	1980807	1980806	**1980807**	Opt	120.11	1980806	**1980807**	Opt	171.76	1975657	2178888	9.33	360.69
D184040	262	1686092	1686091	**1686092**	Opt	123.23	1686091	**1686092**	Opt	176.22	1681540	1820979	7.66	366.54
D184040b	226	1493759	1493758	**1493759**	Opt	30.53	1493758	**1493759**	Opt	40.30	1489875	1643135	9.33	84.62
D204240b	275	1764636	1764635	**1764636**	Opt	74.32	1764635	**1764636**	Opt	106.28	1758813	1817575	3.23	215.74
D41141	68	454381	454292	454397	0.02	45.85	454336	**454392**	0.01	65.57	454108.4	472556.2	3.90	133.76
D4941	70	452101	452086	452167	0.02	78.12	452096	452287	0.01	183.57	451558.5	479227.1	5.77	378.16
D61441	84	561657	561603	561714	0.02	93.61	561601	**561805**	0.04	133.86	561376.2	578506.7	2.96	271.73
D61641	95	633760	633723	633823	0.02	90.56	633759	**633760**	Opt	129.50	631922.1	684460.8	7.68	269.36
D81741	112	730893	730770	730980	0.03	78.39	730792	**730905**	0.02	103.47	729285	752819.8	3.13	210.05
D82041m	122	804695	804586	804751	0.02	110.23	804615	**804703**	0.01	157.63	803246.5	869070.6	7.57	327.87
D102241m	128	854290	854207	854639	0.05	103.65	854205	**854562**	0.04	136.82	851470.8	931176.1	8.56	285.95
D82041	136	880751	880666	880751	0.01	150.21	880663	**880672**	0.00	214.80	879694.1	968826.1	9.20	451.08
D102241	161	1029382	1029304	1029520	0.02	233.45	1029176	**1029433**	0.02	548.61	1026397	1111733	7.68	1141.10
D122641a	152	1011650	1011602	1011719	0.01	221.67	1011640	**1011663**	0.00	292.60	1010031	1102699	8.40	611.54
D102641m	152	1009032	1008898	**1009218**	0.03	278.21	1008830	1009335	0.05	367.24	1006812	1079664	6.75	760.18
D122941m	157	1058436	1058372	1058646	0.03	178.32	1058425	**1058447**	0.00	255.00	1057907	1121942	5.71	525.30
D122641	207	1306865	1306796	1307006	0.02	233.11	1306852	**1306878**	0.00	547.81	1302160	1372208	5.10	1123.01
D102641	188	1199118	1199006	1199270	0.02	378.12	1198998	**1199130**	0.01	499.12	1196720	1223100	2.16	1008.22
D122941	192	1243493	1243445	1243764	0.03	389.89	1243369	**1243530**	0.01	514.65	1242871	1305668	4.81	1055.04
D122641b	179	1156769	1156664	1156972	0.03	451.33	1156653	**1156885**	0.02	1060.63	1152720	1272446	9.41	2227.31
D143041a	164	1103917	1103763	1104280	0.05	366.91	1103807	**1103928**	0.01	484.32	1102040	1159113	4.92	992.86
D143341a	178	1201011	1201011	1201011	Opt	95.68	1201011	**1201011**	Opt	224.85	1196807	1321112	9.41	472.18
D143341b	211	1376011	1376010	**1376011**	Opt	141.57	1376010	**1376011**	Opt	186.87	1375461	1513612	9.13	392.44
D143041	217	1389294	1389178	1389585	0.03	122.21	1389155	**1389308**	0.01	161.32	1388599	1430973	2.96	327.47
D143341	251	1584011	1584010	**1584011**	Opt	206.79	1584010	**1584011**	Opt	272.96	1583061	1726572	8.31	570.49
D143041b	185	1219063	1218949	1219324	0.03	393.56	1218941	**1219075**	0.01	519.50	1217722	1316588	7.51	1080.56
D163441a	198	1313082	1312953	1313338	0.03	231.23	1312951	**1313095**	0.01	305.22	1309537	1352474	3.17	619.60

Tabela 4.2: Resultados dos testes computacionais para as instâncias SOLER (Parte I)

			NDRC-PA				NDRC-PA-CDK				NDRC-r-AG			
Instâncias	$\|V\|$	Ótimo	LI	LS	Gap(%)	t(s)	LI	LS	Gap(%)	t(s)	LI	LS	Gap(%)	t(s)
D163441	282	1761528	1761455	1761728	0.02	371.75	1761352	**1761546**	0.01	490.71	1757476	1902450	7.62	1020.68
D163441b	242	1547424	1547305	**1547777**	0.03	139.32	1547269	1548391	0.07	183.90	1546186	1609321	3.92	375.16
D163741m	200	1343095	1343006	**1343242**	0.02	198.21	1342961	1343315	0.03	261.64	1338797	1477405	9.38	549.44
D163741	235	1530323	1530220	1530403	0.01	211.27	1530170	**1530338**	0.01	278.88	1527874	1576233	3.07	566.12
D183841a	206	1385166	1385077	1385362	0.02	235.17	1385027	**1385180**	0.01	336.29	1381980	1426721	3.14	682.67
D183841	278	1769642	1769505	**1769644**	0.01	405.59	1769465	1769660	0.01	535.38	1763094	1875821	6.01	1102.89
D183841b	237	1552379	1552266	1552456	0.01	491.89	1552224	**1552395**	0.01	649.29	1549585	1692093	8.42	1357.02
D184041b	242	1591521	1591447	1591699	0.02	293.82	1591362	**1591537**	0.01	420.16	1586587	1655182	4.14	857.13
D184041a	224	1491653	1491608	1491781	0.01	466.00	1491504	**1491668**	0.01	615.12	1490460	1506570	1.07	1236.39
D184041	282	1803657	1803608	1803892	0.02	745.00	1803477	**1803675**	0.01	983.40	1799689	1857767	3.13	1996.30
D204241a	224	1510820	1510707	1511083	0.02	456.13	1510669	**1510835**	0.01	652.27	1506288	1616577	6.82	1350.19
D204441a	230	1556414	1556257	1556628	0.02	494.93	1556258	**1556430**	0.01	653.31	1550811	1665363	6.88	1352.35
D204241	291	1872362	1872291	1872691	0.02	351.93	1872175	**1872381**	0.01	464.55	1870864	1965980	4.84	952.32
D204441	315	2007744	2007573	**2008042**	0.02	910.26	2007543	2008819	0.06	1301.67	2001118	2088054	4.16	2655.41
D204241b	258	1693166	1693087	1693374	0.02	340.61	1692997	**1693183**	0.01	487.07	1691134	1862483	9.20	1022.85
D204441b	273	1785585	1785438	1785989	0.03	667.32	1785406	**1785603**	0.01	880.86	1780050	1892720	5.95	1814.58
D4942	80	517768	517622	517939	0.06	66.45	517716	**517780**	0.01	87.72	516473.6	543656.4	5.00	179.83
D61642	138	875072	875002	875115	0.01	264.13	874984	**875111**	0.01	348.65	874634.5	936327	6.59	721.72
D61442	120	768117	768014	768322	0.04	316.24	767848	**768132**	0.04	452.23	765966.3	844928.7	9.35	949.68
D81742	127	828368	828223	828775	0.07	283.42	828119	**828415**	0.04	354.27	825303	836651.7	1.36	712.09
D82042	128	849394	849208	849756	0.06	322.31	849250	**849419**	0.02	460.90	846930.8	883369.8	4.13	940.23
D102242	146	963636	963548	963723	0.02	257.82	963540	**963691**	0.02	340.32	963250.5	973272.4	1.03	684.04
D122642m	162	1077557	1077287	1077932	0.06	559.11	1077234	**1077579**	0.03	799.53	1073786	1174537	8.58	1671.02
D122942m	172	1149670	1149615	**1149693**	0.01	423.47	1149555	1149791	0.01	1037.49	1147256	1241644	7.60	2157.98
D102642	160	1065324	1065135	1065417	0.03	516.18	1065217	**1065345**	0.01	645.23	1064365	1118590	4.85	1322.71
D122642	176	1153625	1153341	1153922	0.05	613.84	1153463	**1153691**	0.02	1503.90	1151548	1257451	8.42	3143.15
D122942	188	1235878	1235815	1235993	0.01	569.59	1235656	**1235974**	0.03	951.21	1234889	1359466	9.16	1997.54
D143042m	184	1224644	1224429	1225341	0.07	926.41	1224522	**1224672**	0.01	2269.70	1220480	1334862	8.57	4743.67
D143342m	200	1329993	1329782	1330780	0.08	303.38	1329860	**1330020**	0.01	379.23	1326269	1343293	1.27	762.24
D143042	211	1369730	1369428	1370329	0.07	322.17	1369593	**1369808**	0.02	538.02	1366991	1397125	2.16	1086.81
D143342	237	1525120	1524940	1525651	0.05	278.93	1524967	**1525151**	0.01	683.38	1523900	1616627	5.74	1407.76
D163442	253	1622974	1622861	**1623067**	0.02	738.40	1622812	1623195	0.02	1233.13	1621513	1655433	2.05	2490.92
D163442a	206	1367607	1367441	1368127	0.05	338.15	1367470	**1367634**	0.01	828.47	1364325	1504368	9.31	1739.78
D163742a	214	1432892	1432688	1433208	0.04	196.17	1432749	**1432974**	0.02	327.60	1428593	1475879	3.20	665.04
D163442b	219	1442855	1442703	1443174	0.03	245.13	1442711	1442884	0.01	600.57	1441268	1457284	1.10	1207.14
D163742b	246	1602916	1602712	**1603294**	0.04	262.57	1602756	1603317	0.04	438.49	1597947	1667033	4.14	894.52
D163742	301	1888149	1887936	1888488	0.03	438.67	1887960	**1888187**	0.01	1074.74	1885506	2058082	8.39	2246.21
D183842m	213	1434387	1434139	1435514	0.10	205.49	1434244	1434469	0.02	369.88	1431805	1577826	9.25	776.75
D183842	233	1542727	1542451	1543342	0.06	265.89	1542573	**1542758**	0.01	651.43	1541493	1573582	2.04	1315.89
D184042m	229	1534827	1534735	**1534892**	0.01	217.26	1534674	1534914	0.02	362.82	1529148	1642265	6.89	751.05
D184042	270	1753907	1753789	1754049	0.01	162.16	1753889	**1753942**	0.00	397.29	1747944	1859141	5.98	818.42
D204242a	236	1587087	1586954	1587303	0.02	210.42	1586943	**1587177**	0.01	231.46	1581850	1682312	5.97	476.81
D204242	289	1872624	1872339	1872957	0.03	524.53	1872459	**1872661**	0.01	1285.10	1866070	2041160	8.58	2685.86
D204242b	265	1742305	1742117	1742491	0.02	299.94	1742072	**1742404**	0.02	329.93	1741782	1864266	6.57	682.96
D41143	108	695196	695043	**695620**	0.08	156.01	695126	695730	0.09	280.82	694778.9	736907.8	5.72	578.49
D61443	128	822217	821917	822420	0.06	168.74	822094	**822264**	0.02	413.41	821641.4	863327.9	4.83	847.50
D61643	134	869353	869032	870069	0.12	110.84	869005	**869370**	0.04	199.51	868657.5	938901.2	7.48	414.98
D81743	147	948708	948507	949044	0.06	115.39	948385	**948762**	0.04	282.71	946336.2	986656.3	4.09	576.72
D82043m	154	1003444	1003209	1004333	0.11	258.22	1003083	**1003715**	0.06	346.01	1001939	1083720	7.55	719.71
D82043	168	1079720	1079445	1080092	0.06	297.41	1079612	**1079782**	0.02	728.65	1077345	1176895	8.46	1522.89
D102643	168	1121130	1120997	1121296	0.03	246.02	1121018	**1121152**	0.01	329.67	1117094	1188398	6.00	679.11
D122643	186	1218825	1218501	**1219120**	0.05	223.31	1218703	1219150	0.04	547.11	1216753	1328519	8.41	1143.46
D122943m	210	1365993	1365734	1366344	0.04	235.62	1365788	**1366020**	0.02	315.73	1365583	1379653	1.02	634.62
D122943	240	1526097	1525830	1526575	0.05	285.44	1525822	**1526184**	0.02	699.33	1520908	1556619	2.29	1412.64
D4944	120	758785	758593	759023	0.06	229.38	758610	**758800**	0.02	511.52	757191.6	819487.8	7.60	1063.95
D61444	152	964730	964498	964863	0.04	308.41	964296	**964785**	0.05	755.60	961642.9	984024.6	2.27	1526.32
D61644	153	984935	984734	985343	0.06	290.66	984837	**984955**	0.01	648.17	982768.1	1034182	4.97	1328.75
D81744	154	995860	995533	**997209**	0.17	220.55	995760	998817	0.31	540.35	992274.9	1075529	7.74	1123.92
D102244m	194	1245433	1245246	1245732	0.04	332.78	1245396	**1245695**	0.02	366.06	1242818	1369976	9.28	768.72
D82044	168	1091866	1091759	**1091918**	0.01	162.56	1091757	1091928	0.02	398.27	1088372	1201053	9.38	836.37
D122644m	198	1295659	1295068	1296940	0.14	319.35	1295529	**1295685**	0.01	542.90	1291902	1321572	2.25	1096.65
D122944m	210	1380447	1380234	1380857	0.05	494.15	1379688	**1380526**	0.06	1210.67	1377824	1435665	4.03	2469.76
D122944	224	1456564	1456325	**1456842**	0.04	562.89	1456472	1458166	0.12	619.18	1452049	1602220	9.37	1300.28
D102644	214	1379555	1379280	1380504	0.09	350.88	1379417	**1379634**	0.02	859.66	1375968	1517511	9.33	1805.28
D143044m	219	1438672	1438368	1439616	0.09	298.47	1438528	**1438701**	0.01	665.59	1435507	1582539	9.29	1397.74
D143044	245	1578725	1578434	1579961	0.10	644.17	1578567	**1578815**	0.02	1578.22	1576673	1720810	8.38	3298.47
D143344	236	1549308	1548853	1550002	0.07	684.89	1549153	**1549339**	0.01	753.38	1546829	1688746	8.40	1574.56
D163744m	254	1671485	1671116	1672083	0.06	731.36	1671318	**1671580**	0.02	1791.83	1668643	1805204	7.56	3727.01
D163744	278	1801619	1801242	1802419	0.07	880.82	1801439	**1801655**	0.01	2158.01	1798016	1963765	8.44	4510.24

Tabela 4.3: Resultados dos testes computacionais para as instâncias SOLER (Parte II)

As Tabelas 4.2 e 4.3, apresentam os resultados dos testes computacionais referentes a aplicação dos algoritmos NDRC nas instancias SOLER. Em 39 das 127 instâncias testadas, foi possível se obter certificados de otimalidade em todas execuções. De uma forma geral, o *gap* médio para os algoritmos NDRC-PA e NDRC-PA-CDK foi muito pequeno. Sendo de 0,02% para o NDRC-PA-CDK e 0,04% para o NDRC-PA. O tempo médio de execução considerando todas as instâncias foi de 4,5 minutos para o NDRC-PA e 7,5 minutos para o algoritmo NDRC-PA-CDK.

No caso do NDRC-r-AG, o gap médio apresentado foi de 4.95%, com um tempo médio de execução de 10 minutos.

Embora o NDRC-r-AG tenha apresentado uma performance ruim quando analisamos a diferença entre os limites primal e dual, é importante observar que quando olhamos unicamente para a diferença média entre os limites inferiores obtidos e as soluções ótimas conhecidas, uma diferença média de apenas 1.8% é observada. Assim sendo, temos um indicativo de que se faz necessário avaliar outras estratégias para obtenção de melhores limites primais.

Capítulo 5

Conclusões e Trabalhos Futuros

Como citamos no Capítulo 1, o Problema do Caixeiro Viajante é um dos problemas mais estudados na literatura, possuindo várias aplicações práticas em áreas como logística, robótica e transporte de pessoas. O problema consiste basicamente em encontrar um circuito Hamiltoniano de custo mínimo. Neste trabalho, tivemos como objetivo principal o desenvolvimento de um algoritmo Non-Delayed-Relax-And-Cut para a versão assimétrica do problema (PCVA).

Iniciamos introduzindo a formulação clássica, seguido da proposta de duas reformulações para o desenvolvimento de uma heurística Lagraneana. Heurística esta, capaz de produzir limites duais e primais de boa qualidade para o problema.

Utilizando as reformulações propostas, investigamos a utilização de desigualdades válidas para o PCVA. Em particular, as desigualdades do tipo *comb* e desigualdades do tipo D_k^-. Para a primeira, investigamos o impacto nos limites duais para ambas as formulações propostas. Ao passo que para as desigualdades do tipo D_k^-, investigamos o impacto mediante sua dualização dinâmica em conjunto com as DES e *combs*, apenas para a reformulação como um Problema de Atribuição.

Com o intuito de obtermos os limites primais, foram propostas duas heurísticas: a) Um algoritmo de *Patching*, onde ciclos são unidos dois a dois para formar uma solução viável para a reformulação do problema como um PA e b) Uma heurística de inserção mais barata para a formulação r-AG. Adicionalmente, um procedimento de busca local baseado em movimentos 3-opt foi desenvolvido e aplicado ao final de cada heurística citada anteriormente.

Com o algoritmo NDRC desenvolvido, foi possível constatar os efeitos da dualização de desigualdades válidas quando comparamos os resultados obtidos aos da formulação clássica do problema, onde apenas desigualdades DES são dualizadas dinamicamente.

No tocante a formulação r-AG, observamos que a aplicação da heurística de inserção mais barata, gerou soluções de baixa qualidade quando comparadas com as soluções produzidas pela heurística de *pacthing*, aplicadas a formulação do pro-

blema como um PA. Como um fator adicional na baixa performance do algoritmo proposto, o procedimento de busca local não obteve os mesmos resultados, uma vez que movimentos do tipo k-opt, dependem da qualidade da solução inicial.

Infelizmente, devido ao curto espaço de tempo disponível, não foi possível trabalhar em estratégias que possibilitassem a melhoria de performance do algoritmo, baseado na reformulação r-AG. Assim sendo, esta é uma investigação que deve ser feita futuramente.

Com base no exposto acima, para cada uma das formulações estudadas, consideramos que os seguintes pontos a serem considerados para estudos futuros:

- **Formulação do problema como um PA**:

  Dualização de desigualdades do tipo *comb*: Embora tenhamos obtido bons resultados na média, é importante ressaltar que, a cada iteração do MS, apenas uma desigualdade é investigada. Esta, considera uma ordenação fixa dos subciclos obtidos como solução do SL. Ou seja, a ordenação do maior para o menor, no procedimento de identificação dos vértices que farão parte do *handle*. Considerar outras estratégias na busca de desigualdades deste tipo violadas, pode contribuir ainda mais na melhoria dos limites duais.

- **Formulação r-AG**:

  Heurística Primal e busca local: Um dos fatores que acarretou na baixa performance do NDRC-r-AG, foi a estratégia adotada para a produção de uma solução viável para o problema. Investigar a aplicação de outros métodos construtivos, meta-heurísticas e adicionalmente outras estratégia de busca local, pode acarretar em uma melhora considerável antes performance do algoritmo proposto neste trabalho.

Referências Bibliográficas

[1] D. Davendra, "Traveling salesman problem, theory and applications," 01 2010.

[2] R. M. Karp, "On the computational complexity of combinatorial problems," *Netw.*, vol. 5, pp. 45–68, Jan. 1975.

[3] M. Jünger, M. Junger, G. Reinelt, G. Reinelt, S. Thienel, and S. Thienel, "Provably good solutions for the traveling salesman problem," 1992.

[4] G. H. Gonnet, C. Korostensky, and S. Benner, "Evaluation measures of multiple sequence alignments," *Journal of Computational Biology*, vol. 7, no. 1-2, pp. 261–276, 2000. PMID: 10890401.

[5] R. G. Bland and D. F. Shallcross, "Large travelling salesman problems arising from experiments in x-ray crystallography: A preliminary report on computation," *Operations Research Letters*, vol. 8, no. 3, pp. 125 – 128, 1989.

[6] R. Guyon, T. D. Lorentzen, C. Hitte, L. Kim, E. Cadieu, H. G. Parker, P. Quignon, J. K. Lowe, C. Renier, B. Gelfenbeyn, F. Vignaux, H. B. DeFrance, S. Gloux, G. G. Mahairas, C. André, F. Galibert, and E. A. Ostrander, "A 1-mb resolution radiation hybrid map of the canine genome," *Proceedings of the National Academy of Sciences*, vol. 100, no. 9, pp. 5296–5301, 2003.

[7] H. D. Ratliff and A. S. Rosenthal, "Order-picking in a rectangular warehouse: A solvable case of the traveling salesman problem," *Operations Research*, vol. 31, no. 3, pp. 507–521, 1983.

[8] G. Dantzig, R. Fulkerson, and S. Johnson, "Solution of a large-scale traveling-salesman problem," *Operations Research*, vol. 2, pp. 393–410, 1954.

[9] K. G. D. W. Little, John D. C., "An algorithm for the traveling salesman problem," *Oper. Res.*, vol. 11, no. 6, pp. 972–989.

[10] G. Carpaneto and P. Toth, "Some new branching and bounding criteria for the asymmetric travelling salesman problem," *Manage. Sci.*, vol. 26, pp. 736–743, July 1980.

[11] E. Balas and N. Christofides, "A restricted lagrangean approach to the traveling salesman problem," *Math. Program.*, vol. 21, pp. 19–46, Dec. 1981.

[12] K. Helsgaun, "An effective implementation of the lin-kernighan traveling salesman heuristic," *European Journal of Operational Research*, vol. 126, pp. 106–130, 2000.

[13] M. Fischetti, A. Lodi, and P. Toth, *Exact Methods for the Asymmetric Traveling Salesman Problem*, pp. 169–205. 02 2006.

[14] F. Glover, G. Gutin, A. Yeo, and A. Zverovich, "Construction heuristics for the asymmetric tsp," *European Journal of Operational Research*, vol. 129, pp. 555–568, 03 2001.

[15] J. Cirasella, D. S. Johnson, L. A. McGeoch, and W. Zhang, "The asymmetric traveling salesman problem: Algorithms, instance generators, and tests," in *Algorithm Engineering and Experimentation* (A. L. Buchsbaum and J. Snoeyink, eds.), (Berlin, Heidelberg), pp. 32–59, Springer Berlin Heidelberg, 2001.

[16] J. Majumdar and A. Bhunia, "Genetic algorithm for asymmetric traveling salesman problem with imprecise travel times," *Journal of Computational and Applied Mathematics*, vol. 235, no. 9, pp. 3063 – 3078, 2011.

[17] I.-C. Choi, S.-I. Kim, and H.-S. Kim, "A genetic algorithm with a mixed region search for the asymmetric traveling salesman problem," *Computers & Operations Research*, vol. 30, no. 5, pp. 773 – 786, 2003.

[18] L. Buriol, P. M. França, and P. Moscato, "A new memetic algorithm for the asymmetric traveling salesman problem," *Journal of Heuristics*, vol. 10, pp. 483–506, Sep 2004.

[19] "A new genetic algorithm for the asymmetric traveling salesman problem," *Expert Systems with Applications*, vol. 39, no. 10, pp. 8947 – 8953, 2012.

[20] E. K. Burke, P. I. Cowling, and R. Keuthen, "Effective local and guided variable neighbourhood search methods for the asymmetric travelling salesman problem," in *Applications of Evolutionary Computing* (E. J. W. Boers, ed.), (Berlin, Heidelberg), pp. 203–212, Springer Berlin Heidelberg, 2001.

[21] S. Lin and B. W. Kernighan, "An effective heuristic algorithm for the traveling-salesman problem," *Oper. Res.*, vol. 21, pp. 498–516, Apr. 1973.

[22] M. Held and R. M. Karp, "The traveling-salesman problem and minimum spanning trees: Part ii," *Math. Program.*, vol. 1, pp. 6–25, Dec. 1971.

[23] A. Lucena, "Non delayed relax-and-cut algorithms," *Annals of Operations Research*, vol. 140, no. 1, pp. 375–410, 2005.

[24] A. Lucena, "Steiner problem in graphs: Lagrangean relaxation and cutting-planes," vol. 21, 1992.

[25] R. Aboudi, Å. Hallefjord, and K. Jörnsten, "A facet generation and relaxation technique applied to an assignment problem with side constraints," *European Journal of Operational Research*, vol. 50, pp. 335–344, 2 1991.

[26] L. F. G. Escudero, "A lagrangian relax-and-cut approach for the sequential ordering problem with precedence relationships," *Annals of Operations Research*, vol. 50, no. 1, pp. 219–237, 1994.

[27] J. E. Beasley, "Modern heuristic techniques for combinatorial problems," ch. Lagrangian Relaxation, pp. 243–303, New York, NY, USA: John Wiley & Sons, Inc., 1993.

[28] M. Guignard and S. Kim, "Lagrangean decomposition for integer programming : theory and applications," *RAIRO - Operations Research - Recherche Opérationnelle*, vol. 21, no. 4, pp. 307–323, 1987.

[29] M. L. Fisher, "The lagrangian relaxation method for solving integer programming problems," *Manage. Sci.*, vol. 50, Dec. 2004.

[30] C. Lemaréchal, "An algorithm for minimizing convex functions.," pp. 552–556, 01 1974.

[31] Y. CHU, "On the shortest arborescence of a directed graph," *Scientia Sinica*, vol. 14, pp. 1396–1400, 1965.

[32] R. L. Graham and P. Hell, "The minimum spanning tree problem," *Annals of the History of Computing*, vol. 7, pp. 43–57, 1985.

[33] M. Fischetti and P. Toth, "An efficient algorithm for the min-sum arborescence problem on complete digraphs," *INFORMS Journal on Computing*, vol. 5, no. 4, pp. 426–434, 1993.

[34] A. Letchford, "Separating a superclass of comb inequalities in planar graphs," *Mathematics of Operations Research*, vol. 25, pp. 443–454, 08 2000.

[35] M. Padberg and G. Rinaldi, "A branch-and-cut algorithm for the resolution of large-scale symmetric traveling salesman problems," *SIAM Rev.*, vol. 33, pp. 60–100, Feb. 1991.

[36] F. C. Calheiros, A. Lucena, and C. C. de Souza, "Optimal rectangular partitions," *Networks*, vol. 41, no. 1, pp. 51–67, 2003.

[37] C. Martinhon, A. Lucena, and N. Maculan, "Stronger k-tree relaxations for the vehicle routing problem," *European Journal of Operational Research*, vol. 158, pp. 56–71, 02 2004.

[38] "A new lagrangian based branch and bound algorithm for the 0-1 knapsack problem," *Electronic Notes in Discrete Mathematics*, vol. 36, pp. 623 – 630, 2010. ISCO 2010 - International Symposium on Combinatorial Optimization.

[39] "A relax-and-cut algorithm for the prize-collecting steiner problem in graphs," *Discrete Applied Mathematics*, vol. 157, no. 6, pp. 1198 – 1217, 2009. Reformulation Techniques and Mathematical Programming.

[40] A. S. da Cunha and A. Lucena, "Lower and upper bounds for the degree-constrained minimum spanning tree problem," *Networks*, vol. 50, no. 1, pp. 55–66, 2007.

[41] A. Belloni, "Improving on the held and karp bound for the tsp via lagrangian relaxation," 2005.

[42] M. Grötschel and M. W. Padberg, "On the symmetric travelling salesman problem i: Inequalities," *Mathematical Programming*, vol. 16, pp. 265–280, 12 1979.

[43] M. P. M. Grötschel, ""polyhedral theory: The traveling salesman problem"," *SIAM J. Discret. Math.*, 1985.

[44] M. Padberg and G. Rinaldi, "Facet identification for the symmetric traveling salesman polytope," *Math. Program.*, vol. 47, pp. 219–257, June 1990.

[45] J.-M. Clochard and D. Naddef, "Using path inequalities in a branch and cut code for the symmetric traveling salesman problem," in *IPCO*, 1993.

[46] M. Fischetti, "Facets of the asymmetric traveling salesman polytope," *Math. Oper. Res.*, vol. 16, pp. 42–56, 1991.

[47] S. Chopra and G. Rinaldi, "The graphical asymmetric traveling salesman polyhedron: Symmetric inequalities," *SIAM Journal on Discrete Mathematics*, vol. 9, pp. 602–624, 11 1996.

[48] M. Queyranne and Y. Wang, "Symmetric inequalities and their composition for asymmetric travelling salesman polytopes," *Math. Oper. Res.*, vol. 20, pp. 838–863, Nov. 1995.

[49] E. Balas and M. Fischetti, "A lifting procedure for the asymmetric traveling salesman polytope and a large new class of facets," *Mathematical Programming*, vol. 58, pp. 325–352, Jan 1993.

[50] H. W. Kuhn, "The hungarian method for the assignment problem," in *50 Years of Integer Programming*, 2010.

[51] V. Chvátal, "Edmonds polytopes and a hierarchy of combinatorial problems," *Discrete Mathematics*, vol. 4, no. 4, pp. 305 – 337, 1973.

[52] E. Balas and M. Fischetti, "Lifted cycle inequalities for the asymmetric traveling salesman problem," *Mathematics of Operations Research*, vol. 24, no. 2, pp. 273–292, 1999.

[53] R. Jonker and T. Volgenant, "A shortest augmenting path algorithm for dense and sparse linear assignment problems," in *DGOR/NSOR* (H. Schellhaas, P. van Beek, H. Isermann, R. Schmidt, and M. Zijlstra, eds.), (Berlin, Heidelberg), pp. 622–622, Springer Berlin Heidelberg, 1988.

[54] D. J. Rosenkrantz, R. E. Stearns, and P. M. Lewis, *An analysis of several heuristics for the traveling salesman problem*, pp. 45–69. Dordrecht: Springer Netherlands, 2009.

[55] R. M. Karp, "A patching algorithm for the nonsymmetric traveling-salesman problem," *SIAM J. Comput.*, vol. 8, pp. 561–573, 1979.

[56] G. Laporte, "The vehicle routing problem: An overview of exact and approximate algorithms," 1992.

[57] M. Fischetti and P. Toth, "An additive bounding procedure for the asymmetric travelling salesman problem," *Math. Program.*, vol. 53, pp. 173–197, Jan. 1992.

[58] M. Fischetti, P. Toth, and D. Vigo, "A branch-and-bound algorithm for the capacitated vehicle routing problem on directed graphs," *Operations Research*, vol. 42, pp. 846–859, 10 1994.

[59] M. Fischetti and P. Toth, "A polyhedral approach to the asymmetric traveling salesman problem," *Manage. Sci.*, vol. 43, pp. 1520–1536, Nov. 1997.

[60] D. Soler, E. Martínez, and J. C. Micó, "A transformation for the mixed general routing problem with turn penalties," *Journal of The Operational Research Society - J OPER RES SOC*, vol. 59, pp. 540–547, 04 2008.