IVAN XAVIER ARAÚJO DE LIMA

Algoritmos para problemas de roteamento de veículos com entrega e coleta

NITERÓI

2009

IVAN XAVIER ARAÚJO DE LIMA

Algoritmos para problemas de roteamento de veículos com entrega e coleta

Dissertação de Mestrado submetida ao Programa de Pós-Graduação em Computação da Universidade Federal Fluminense como requisito parcial para a obtenção do título de Mestre. Área de concentração: Otimização Combinatória e Inteligência Artificial.

Orientador:
Prof. Luiz Satoru Ochi, D. Sc.

Co-orientador:
Prof. Eduardo Uchoa Barboza, D. Sc.

Universidade Federal Fluminense

NITERÓI

2009

Algoritmos para problemas de roteamento de veículos com entrega e coleta

Ivan Xavier Araújo de Lima

Dissertação de Mestrado submetida ao Programa de Pós-Graduação em Computação da Universidade Federal Fluminense como requisito parcial para a obtenção do título de Mestre. Área de concentração: Otimização Combinatória e Inteligência Artificial.

Aprovada por:

Prof. Luiz Satoru Ochi, D.Sc. / IC-UFF (Presidente)

Prof. Eduardo Uchoa Barboza, D.Sc. / TEP-UFF

Profa. Adriana C. F. Alvim, D.Sc. / DIA-UNIRIO

Prof. Artur Alves Pessoa, D.Sc. / TEP-UFF

Prof. Marcus V. S. Poggi de Aragão, D.Sc. / DI-PUC-Rio

Niterói, 17 de Abril de 2009.

"...É melhor tentar e falhar, que preocupar-se e ver a vida passar. É melhor tentar, ainda em vão, que sentar-se fazendo nada até o final."

Martin Luther King

Dedico esse trabalho a Deus. A minha esposa, meus familiares, ao Uchoa e demais amigos, por todo apoio, amor e carinho.

Agradecimentos

Acima de tudo agradeço a Deus que conhece o futuro desde o presente, pois toda a sabedoria do mundo pertence a Ele, e também, por ter me enviado as pessoas certas para poder ajudar nessa grande batalha.

Um agradecimento super especial à minha esposa Danielly quem com muito amor me incentiva e concede forças me lembrando que: *"Tudo posso naquEle que me fortalece" (Filipenses 4:13)*. Meus pais e familiares que mesmo distante, nunca estiveram longe de mim.

Ao meu orientador Prof. Dr. Satoru por dar-me o voto de confiança para a realização deste trabalho. Ao meu coorientador Prof. Dr. Uchoa o qual com carinho, atenção e dedicação me encaminhou para o rumo certo.

A meus amigos de longo e curto prazo que no dia-a-dia, através de suas amizades, foram me moldando e auxiliando, a ser quem eu sou e estar onde estou. Não citarei nomes, mas, à todos do IC, principalmente aos que fizeram matéria comigo pois tivemos a oportunidade de nos conhecer melhor e compartilhar experiências, e aos amigos do trabalho (Automatos e Gapso), pois tornaram agradável esta forma que consegui de me manter no rio, um grande abraço.

Agradeço à Maria e à Ângela por serem atenciosas, prestativas e colaboradoras, indo além de suas obrigações para na medida do possível facilita nossas vidas.

Agradeço aos membros da banca (Uchoa, Satoru, Artur Pessoa, Marcus Poggi e Adriana Alvim) que dedicaram um tempo com certeza escasso a colaborar com este trabalho realizando críticas e sugestões que vem a enriquece-lo.

Resumo

Neste trabalho são abordados problemas de roteamento de veículo com coleta e entrega. Dentre eles trabalhou-se, o problema de roteamento de veículo com coleta e entrega, especificamente, o Roteamento de Veículo com Coleta e Entrega e Janelas de Tempo (VRPPDTW - *Vehicle Route Problem Pickup Delivery with Time Windows*) e o Problema do Caixeiro Viajante com Coleta e Entrega (TSPPD - *Traveling Salesman Problem with Pickup and Delivery*). O primeiro foi trabalho com uma abordagem heurística, busca local interativa com movimento randômico de descida (ILS-MRD), e o segundo através por programação inteira, com algoritmo de branch-and-cut.

Palavras-chave: Roteamento de Veículo, Caixeiro Viajante, Coleta e Entrega, Janelas de Tempo, Programação inteira, Geração de cortes, Busca Local Iterativa.

Abstract

In this work are discussed problems with the routing of vehicle pickup and delivery. Among them worked up, the vehicle routing problem with pickup and delivery, specifically, the Vehicle Routing Problem Pickup and Delivery with Time Window (VRPPDTW) and Traveling Salesman Problem with Pickup and Delivery (TSPPD). The the first was working with a heuristic approach, iterated local search, and second through integer program with cuts generation, branch-and-cut.

Keywords: Vehicle Route Problem, Traveling Salesman Problem, Pickup and Delivery, Time Window, Integer Programming, Cut Generation, Iterated Local Search.

Sumário

Lista de Figuras

Lista de Tabelas

Siglas e Abreviações

COPPEAD	:	Instituto de Pesquisa e Pós-Graduação em Administração de Empresas da Universidade Federal do Rio de Janeiro (UFRJ)
DARP	:	do inglês: *Dial-a-Ride Problem* - Problema de Dar uma Volta
EDP	:	do inglês: *Express Delivery Problem* - Problema de Entrega Expressa
GGA	:	do inglês: *Grouping Genetic Algorithm* - Algoritmo Genético Agrupado
ILS	:	do inglês: *Iterated Local Search* - Busca Local Iterada
IPEA	:	Instituto de Pesquisa Econômica Aplicada
FGV	:	Fundação Getúlio Vargas
GPDP	:	do inglês: *General Pickup and Delivery Problem* - Problema Geral de Coleta e Entrega
MDARP	:	do inglês: *M-Dial-a-Ride Problem* - Sem tradução, dispõe de serviços de transporte em domicílio
MRD	:	Método Randômico de Descida
PAC	:	Programa de Aceleração do Crescimento
PIB	:	Produto Interno Bruto
PDP	:	do inglês: *Problem Pickup and Delivery* - Problem Problema de Coleta e Entrega
PDPTW	:	do inglês: *Problem Pickup and Delivery with Time Windows* - Problem Problema de Coleta e Entrega com Janelas de Tempo
RTS	:	do inglês: *Reactive Tabu Search* - Busca Tabu Reativa
SA	:	Simulated Annealing
TS	:	do inglês, *Tabu Search* - Buscas Tabu
TSP	:	do inglês: *Traveling Salesman Problem* - Problema do Caixeiro Viajante
TSPB	:	do inglês: *Traveling Salesman Problem with Backhauls* - Problema do Caixeiro Viajante com Coleta de retorno

TSPPDL	:	do inglês: *The Pickup and Delivery Traveling Salesman Problem with LIFO Loading* - Problema do Caixeiro Viajante com Coleta e Entrega em Forma de Fila
TSPPDF	:	do inglês: *Traveling Salesman Problem First-In-First-Out Loading* - Problema do Caixeiro Viajante com Coleta e Entrega em Forma de Pilha
TSPPD	:	do inglês: *Traveling Salesman Problem with Pickup and Delivery* - Problema do Caixeiro Viajante com Coleta e Entrega
TSPSPD	:	do inglês: *Traveling Salesman Problem With Simultaneous Pickup and Delivery* - Problema do Caixeiro Viajante com Coleta e Entrega Simultâneas
VRP	:	do inglês: *Vehicle Route Problem* - Problema de Roteamento de Veículo
VRPB	:	do inglês: *Vehicle Routing Problem Problema with Backhauls* - Roteamento de Veículos com Coleta de Retorno
VRPPDTW	:	do inglês: *Vehicle Route Problem Pickup Delivery with Time Windows* - Roteamento de Veículo com Coleta e Entrega e Janelas de Tempo
VRPTW	:	do inglês: *Vehicle Routing Problem With Time Windows* - Problema de Roteamento de Veículos com Janelas de Tempo
VNS	:	do inglês: *Variable Neighborhood Search* - Busca por Vizinhança Variável

1 Introdução

1.1 Motivação

A expansão do comércio mundial levou a um crescimento significativo da demanda por transporte. Nas últimas décadas, observou-se pronunciada intensificação dos fluxos comerciais, exigindo a modernização/expansão dos meios de transporte e, conseqüentemente, o aumento dos investimentos no sistema logístico.

O processo de globalização que integrou as economias nacionais trouxe benefícios, mas passou a exigir que a infra-estrutura não apenas atendesse às necessidades básicas da população, mas que também servisse como suporte à competitividade das empresas. Os custos envolvidos no processo de produção, tanto os anteriores à fabricação (como custo de energia) quanto os posteriores (como o de transporte), têm grandes implicações sobre o preço final dos produtos, vinculando fortemente a competitividade das empresas à infra-estrutura nacional.

Entre 2005 e 2007 foram aplicados na recuperação das rodovias cerca de R$ 4,9 bilhões de reais. Percentual baixo se comparado aos R$ 25 bilhões gastos com pagamento de indenizações, perdas de cargas, tempo durante o qual o veículo permanece parado, reparos e/ou substituições causadas pela má conservação das rodovias brasileiras. Em razão desse conjunto de deficiências, o custo logístico no Brasil atinge 12,63% do PIB, contra 8,19% nos EUA. Deve-se considerar que o transporte responde pela maior parcela do custo logístico. "Segundo a COPPEAD, os custos com transporte chegam a 60% dos custos logísticos e a redução de custos nessa área é muito importante, pois correspondem, em média, 20% do custo total das empresas." (DIAS; LIMA, 2008)

Os problemas de roteamento de veículo com coleta e entrega estão diretamente relacionados aos esforços para a redução dos custos de transportes, estes problemas se resumem ao atendimento de uma demanda, que pode se apresentar na forma de coleta e/ou entrega de pessoas ou mercadorias em uma determinada região geográfica ou espacial. A maio-

ria das aplicações para estes problemas são geográficas e representadas por consumidores distribuídos em uma área de atendimento. Desta forma, o objetivo é desenvolver metodologias para atender as demandas de forma otimizada, visando à redução dos gastos com veículos e com o deslocamento dos mesmos.

1.2 Principais Contribuições

Esta dissertação tem o interesse em dois casos de problemas de roteamento de veículos envolvendo coleta e entrega: com janelas de tempo e o problema do caixeiro viajante (roteamento de um único veículo).

No primeiro caso, aplicou-se a heurística de busca local iterada com o método randômico de descida. Esse algoritmo, apesar de simples, mostrou-se capaz de encontrar boas soluções para o problema, comparado a outros algoritmos encontrados na literatura.

No segundo caso, foram propostas novas formulações lineares inteiras e técnicas para geração de cortes, levando ao desenvolvimento de algoritmos do tipo branch-and-cut. As formulações estendidas, apesar de não serem capazes de resolver instâncias de maior porte em tempo razoável, mostraram-se satisfatórias ao obter limites inferiores bastante fortes para o problema.

1.3 Organização

O Capítulo 2 descreve o problema de roteamento de veículos de maneira geral e algumas de suas variantes. A seguir, no Capítulo 3, uma abordagem heurística para o Problema de Roteamento de Veículo com Coleta e Entrega com Janelas de Tempo será apresentada, como também comparações dos resultados com os obtidos na literatura. O Capítulo 4, dedica-se ao problema do Caixeiro Viajante com Coleta e Entrega um caso particular de roteamento onde se propõem algumas formulações e algoritmos para esse problema. Finalmente, no Capítulo 5, tem-se as considerações finais e trabalhos futuros.

2 Problema de Roteamento de Veículos com Coleta e Entrega

Problema de roteamento de veículo (VRP - *Vehicle Routing Problem*) é um nome genérico para uma série de problemas que tem as seguintes características: estabelecer e organizar rotas ou itinerários eficientes para veículos realizarem coletas/entregas de mercadorias, dispondo de uma frota de veículos idênticos ou não, atendendo a um conjunto de clientes, cada um com uma demanda específica. Na literatura, Dantzig e Ramser (1959) foram os primeiros a formular o VRP quando estudaram a aplicação real na distribuição de gasolina para postos de venda de combustível.

Há muitas variantes para esse problema, algumas com janelas de tempo, outras com coleta e entrega, com capacidade, com múltiplos depósitos, periódicas, simultâneas, etc.

Nesta seção são apresentados os conceitos básicos e os principais parâmetros que caracterizam as variantes desse problema com coleta e entrega. A partir do problema básico, são apresentadas suas extensões e nos problemas relacionados a este trabalho haverá uma ênfase em sua explanação.

2.1 Visão Geral

O VRP introduzido por Dantzig e Ramser (1959), é definido como um grafo não orientado, $G = (V, E)$, onde $V = \{0, 1, \ldots, N\}$ é o conjunto de vértices e $E = \{(i, j) : i, j \in V, i < j\}$ o conjunto de arestas. O vértice 0 representa o depósito de onde partem os m veículos e os demais vértices são cidades ou clientes. Um custo não negativo, distância ou tempo de viagem é dado por c_{ij} definido em E. Cada vértice i tem uma demanda q_i e um tempo de serviço s_i. O VRP tem como objetivo definir rotas entre um depósito e um conjunto de pontos de entrega que minimize o número de veículos, a distância percorrida ou o tempo. As restrições básicas do problema consistem em: cada cidade é visitada uma única vez por um único veículo; cada rota é iniciada num depósito e finalizada no mesmo

depósito; todas as demandas de todos os consumidores devem ser satisfeitas, respeitando a capacidade Q do veículo.

Outras restrições que podem ser acrescentadas ao problema são: restrição no número de pontos de entrega em cada rota; restrição de tempo ou distância de uma rota; restrição de janelas de tempo: cada ponto deve ser visitado em um período de tempo específico; restrição de precedência entre cidades: o ponto de entrega i deve ser visitado antes do ponto j; tipo de frota: veículos disponíveis para execução das rotas são homogêneos ou heterogêneos; quantidade de veículos contidos na frota: quantidade limitada ou ilimitada; capacidade dos veículos: veículos com capacidade limitada ou ilimitada; quantidade de depósitos: único depósito ou vários depósitos, etc.

Encontram-se na literatura vários trabalhos publicados que tratam diversos problemas de roteamento de veículos com diferentes abordagens: com um único depósito (BEASLEY, 1983; HO; GENDREAU, 2006; CAMPOS; MOTA, 2000; BRäYSY; GENDREAU; DULLAERT, 2004); ou vários depósitos (SALHI; NAGY, 1999; FAN; WANG; CHEN, 2007); com veículos homogêneos (CAMPOS; MOTA, 2000; CHIN; KIT; LIM, 1999; BRäYSY; GENDREAU; DULLAERT, 2004) ou heterogêneos (BELFIORE; Y.YOSHIZAKI, 2006; CHOI; TCHA, 2007); modelo estático (HO; GENDREAU, 2006; CHIN; KIT; LIM, 1999) ou dinâmico (ALVARENGA et al., 2005; MONTEMANNI et al., 2002); com serviços de coleta e/ou entrega (MIN, 1989) com restrição de janelas de tempo (REIMANN; DOERNER; HARTL, 2002; OMBUKI; ROSS; HANSHAR, 2006; ROUSSEAU; GENDREAU; PESANT, 2002; BRäYSY; GENDREAU; DULLAERT, 2004; DUMITRESCU et al., 2008) entre outros.

Um problema bastante abordado na literatura é o Problema de Roteamento de Veículos com Janelas de Tempo (VRPTW, do inglês Vehicle Routing Problem With Time Windows) que inclui um intervalo de tempo para começar e terminar o atendimento no consumidor e ainda um intervalo para saída e retorno ao depósito (ALVARENGA et al., 2005).

Este trabalho apresenta uma abordagem heurística (Capítulo 3) para um caso particular do Roteamento de Veículos com Coleta e Entrega e Janelas de Tempo (VRPPDTW, do inglês Vehicle Routing Problem Pickup Delivery With Time Windows) descrita na Seção 2.3.

Outro problema bastante abordado na literatura, o Problema do Caixeiro Viajante, consiste em encontrar um trajeto que visite N cidades diferentes, sem repetição, retornando à origem e utilizando a menor rota possível, pode ser considerado um problema de roteamento de veículos quando esse possui apenas um único veículo com capacidade

ilimitada. Este trabalho apresenta um algoritmo de branch-and-cut (Capítulo 4) para o Problema do Caixeiro Viajante com Coleta e Entrega (TSPPD, do inglês Traveling Salesman Problem with Pickup and Delivery) descrito na Seção 2.4.

2.2 Classificação

Vários problemas de roteamento de veículos com coleta e entrega têm sido estudados para tratar as diferentes situações reais. Todos eles se concentram em um único objetivo: uso eficiente de uma frota de veículos que deve satisfazer demandas de entrega e/ou coleta de um conjunto de consumidores ou clientes. As demandas de cada consumidor devem ser satisfeitas por um único veículo (XU et al., 2003).

Tendo em vista essa semelhança, Savelsbergh e Sol (1995) apresentam o Problema Geral de Coleta e Entrega (GPDP, do inglês General Pickup and Delivery Problem) que combina várias características encontradas entre esses problemas práticos de coleta e entrega. O GPDP consiste em definir um conjunto de rotas a fim de satisfazer um conjunto de demandas de transporte. Recentemente Berbeglia et al. (2007), Parragh, Doerner e Hartl (2008a) e Parragh, Doerner e Hartl (2008b), fizeram uma classificação mais detalhada desse tipo de problema.

Cada consumidor possui demandas de coleta e entrega. As demandas de coleta possuem informações sobre a carga total a ser coletada pelo veículo e em quais consumidores essa carga deve ser entregue posteriormente. As demandas de entrega possuem informações sobre a carga total a ser entregue e em quais consumidores elas devem terem sido coletadas anteriormente. Portanto, existe uma regra de precedência entre consumidores, pois para satisfazer uma demanda de entrega o veículo deverá satisfazer anteriormente a respectiva demanda de coleta.

A posição inicial e a posição final dos veículos também são informadas. Assim, o objetivo é definir rotas otimizadas que tenham como ponto de partida e ponto de chegada àqueles definidos para os veículos e que satisfaçam todas as demandas dos consumidores sem extrapolar a capacidade dos veículos.

Parragh, Doerner e Hartl (2008a, 2008b) dividem o Problema Geral de Coleta e Entrega em dois grupos:

- Grupo 1: transporte de cargas do depósito para os consumidores e dos consumidores para o depósito

- Grupo 2: transporte de carga entre consumidores

No primeiro grupo todos os veículos partem do depósito contendo a carga necessária para satisfazer as demandas de entrega dos consumidores. As cargas recolhidas pelo veículo nos consumidores devem ser entregues no depósito. Para o segundo grupo, os veículos partem de um ponto inicial sem qualquer carga e percorrem vários pontos de coleta e/ou entrega de itens e então retornam a um ponto final. Neste caso demandas de entrega são sempre satisfeitas por itens coletados anteriormente pelo veículo. (veja Figura 1)

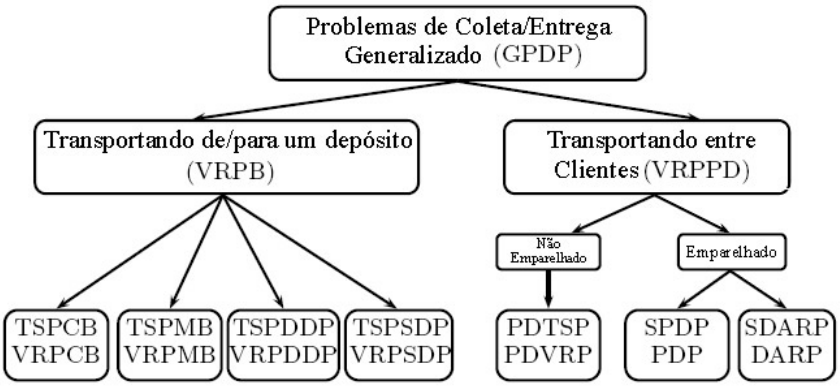


Figura 1: Problemas de Coleta e Entrega (segundo Parragh, Doerner e Hartl (2008a, 2008b))

Outra classificação foi feita por Berbeglia et al. (2007) onde divide o Problema Geral de Coleta e Entrega em três grupos, de acordo com elementos: estrutura, visitas e veículos.

Estrutura refere-se à quantidade de origens e destinos de uma mercadoria, sendo o elemento chave dessa divisão. Este é subdividido em três partes:

- muito-para-muitos (M-M), qualquer vértice pode servir como uma fonte ou como um destino para qualquer mercadoria.
- um-para-muitos-para-um (1-M-1), as mercadorias estão inicialmente disponíveis num armazém e se destinam aos vértices (clientes), além disso muitas mercadorias dos clientes estão destinadas ao depósito. Os conjuntos de pares coleta-entrega dos clientes não são necessariamente disjuntos (ROPKE; PISINGER, 2006b) e, muitas vezes, coincidem como na distribuição de bebidas e a coleta de suas garrafas vazias (PRIVé et al., 2006).
- um-para-um (1-1), cada produto tem uma origem e destino determinado. Problemas desse tipo, surgiram, por exemplo, nas operações dos correios e nos serviços de porta-em-porta (CORDEAU; LAPORTE, 2003a).

Visitas fornece informações sobre a forma como as operações de coletas e entregas são realizadas nos clientes. Usa-se a notação PD para indicar que o cliente é visitado somente uma vez na operação de coleta e entrega (PD, se estas operações são realizadas juntas, e P/D se cada vértice é uma coleta ou entrega independente). Além disso usa-se a letra T quando alguns vértices podem ser utilizados como um ponto intermediário de transbordo.

Veículos indica o número de veículos utilizados, na solução. Sempre que um elemento é indefinido, usa-se a notação "-".

Claramente, está classificação não contempla todos os tipos de PDPs, com todas suas restrições, objetivos alternativos, número de mercadorias, etc., mas consegue-se abranger os casos de forma geral.

2.2.1 Definição

Os Problema de Coleta e Entrega (PDP, do inglês Pickup and Delivery Problem) constituem uma importante classe dos problemas de roteamento de veículos em que as mercadorias ou pessoas têm que ser transportados entre origens e destinos. O PDP é um problema do grupo onde o transporte de cargas é feito entre consumidores (Grupo 2, segundo Parragh, Doerner e Hartl (2008a)). Estes problemas vêm sendo estudados a mais de 30 anos e aplicam-se a diversos contextos, como o da logística, serviços e robótica. No PDP cada demanda de coleta possui informações sobre a carga total a ser coletada pelo veículo e o destino dessa carga. As demandas de entrega possuem informações sobre a carga total a ser entregue e a origem da carga. Todos os veículos, então, partem de um depósito para satisfazer as demandas e retornam novamente ao depósito finalizando o trajeto (SAVELSBERGH; SOL, 1995).

A maioria dos atuais PDPs podem ser definidos dentro do seguinte quadro. Dado $G = (V, A)$ um grafo completo e não orientado com o conjunto de vértices $V = \{0, 1, \ldots, N\}$, onde o vértice 0 representa o depósito e cada um dos vértices restantes $(1, \ldots, N)$, um cliente. Arcos são definidos como $A = \{(i, j) : i, j \in V, i \neq j\}$. Cada arco $(i, j) \in A$ tem um custo c_{ij} não-negativo associado a ele que satisfaz a desigualdade triangular, o custo corresponde à duração da viagem. $H = \{1, \ldots, p\}$ é o conjunto de mercadorias a serem transportadas. Cada vértice, incluindo o depósito, pode necessitar/gerar uma demanda para cada tipo de mercadoria. $D = (d_{hi})$ representa a matriz de tipos de mercadoria. Assim, d_{hi}, positivo, refere-se à quantidade de mercadoria h que sai para o vértice i, e $-d_{hi}$,

negativo, é a quantidade de mercadoria h requerida pelo vértice i. No caso de transporte de passageiros é mais comum falar em pedido em vez de mercadoria, mas do ponto de vista de modelagem matemática não há diferença. Assume-se que $\sum_{i \in V} d_{ih} = 0 : h \in H$, ou seja, deve haver um equilíbrio entre coleta e entrega de cada mercadoria. $K = \{1, \ldots, m\}$ é o conjunto de veículos disponíveis com capacidade Q. Uma rota é um circuito com alguns vértices, começando e terminando no depósito. PDPs consistem em construir rotas com m veículos tais que:

- Todos as coletas e entregas sejam satisfeitas;
- A carga de um veículo não exceda a sua capacidade;
- A soma dos custos das rotas seja minimizada.
- Em uma rota válida um veículo começa e termina seu trajeto no depósito.

2.2.2 Revisão da Literatura

2.2.2.1 Problema com Coleta e Entrega

O Problema de Roteamento de Veículos com Coleta e Entrega (VRPPD, do inglês Vehicle Routing Problem with Pickup and Delivery) é uma extensão do Problema de Coleta e Entrega. Em ambos, as cargas são transportadas entre pontos de coleta e entrega (Grupo 2, segundo (PARRAGH; DOERNER; HARTL, 2008a)), mas no VRPPD uma carga coletada pode ser utilizada para satisfazer qualquer demanda de entrega (PARRAGH; DOERNER; HARTL, 2008b).

Dumas, Desrosiers e Soumis (1991) propuseram um método branch-and-bound que é capaz de resolver problemas com até 55 pedidos. Outro algoritmo exato foi proposto por Ruland e Rodin (1997), usando o método branch-and-cut. Ele mostra formulações para o PDP como um problema de programação inteira. Os testes foram feitos com base em campos aéreos dos Estados Unidos, gerando redes com 7 a 15 nós de demanda e coleta. O tempo de resposta varia de 3 a 1246 segundos.

Outros trabalhos da literatura como Mosheiov (1998), Salhi e Nagy (1999), Montané e Galvão (2006) trabalham com o transporte de cargas do depósito para os consumidores e dos consumidores para o depósito (Grupo 1). Nestes casos os veículos partem do depósito com as cargas a serem entregues nos consumidores e retornam ao depósito com as cargas coletadas pelos consumidores. Um exemplo simples é a entrega de engradados, botijões

de gás, e outros, onde o veículo parte do depósito com vasilhames cheios, os quais são trocados por vazios, e retornam ao depósito.

2.2.2.2 Problema de Entrega Expressa

O Problema de Entrega Expressa (EDP, do inglês Express Delivery Problem) foi proposto por Montané, Ferreira e Galvão (1997), a fim de resolver problemas de entrega aérea expressa. Ele é outro problema com demandas de coleta e entrega em cada nó. Neste caso, o atendimento da demanda é composto por duas fases. A primeira é a de coleta e a segunda de entrega.

Deve-se notar que as fases de coleta e entrega não necessariamente coincidem. Uma rota é definida entre as cidades onde existem demandas de coleta. Todas as demandas de uma mesma cidade são agrupadas em um carregador (demanda de coleta). Depois de efetuadas todas as coletas, o carregamento é encaminhado a um distribuidor (ponto intermediário que pode ser visto como o depósito definido no Problema de Roteamento de Veículos). No distribuidor os carregamentos são desfeitos e as demandas de entrega são organizadas conforme a cidade destino. Uma nova rota é definida partindo do distribuidor até as cidades (pontos de demanda de entrega).

O Problema de Entrega Expressa pode ser considerado um misto entre o Grupo 1 e o Grupo 2, as requisições de transporte especificam uma origem e um destino para cada item o que demonstra uma característica do Grupo 2. No entanto, em uma mesma rota o veículo não efetua coletas e entregas permitindo que todas as demandas a serem entregues e coletadas partam do único ponto, o distribuidor.

Montané, Ferreira e Galvão (1997) mostram algumas heurísticas para resolução do problema e executa testes com 20 cidades comparando-as com resultados exatos e com até 100 cidades comparado-as entre as heurísticas abordadas. Os resultados foram satisfatórios retornando soluções bem próximas da ótima.

2.2.2.3 Problema com Contensão de Via

Caricato et al. (2003) abordam o problema de coleta e entrega com contensão de via. Neste contexto, a rede de transporte é composta por vias, sendo que, cada uma possui uma capacidade unitária, ou seja, em cada via pode trafegar um único veículo. Quando dois ou mais veículos tentam entrar na via ao mesmo tempo, uma contensão é aplicada e o último deles é removido ou roteado novamente. O autor apresenta duas heurísticas

seqüenciais e uma heurística baseada na metaheurística busca tabu paralela.

2.2.2.4 Problema "Peça por uma carona"

O Problema "Peça por uma carona" (DARP, do inglês Dial-a-Ride Problem) é um Problema de Coleta e Entrega onde a carga são pessoas, chamadas de clientes ou consumidores, e o tamanho das cargas são todas iguais a 1. Os clientes são recolhidos em locais pré-definidos e transportados para pontos conhecidos. O objetivo é minimizar o número de veículos, se o problema é composto por múltiplos veículos (m-dial-a-ride ou MDARP), ou distância trafegada, ou ambos. O problema deve considerar o tempo gasto no embarque e desembarque de passageiros.

Este problema foi proposto por Psarafis (1980) que desenvolveu um algoritmo de programação dinâmica para casos com um único veículo. Existem na literatura algumas referências apresentando heurísticas para resolução do trabalho. Ho e Gendreau (2006) implementaram um algoritmo busca tabu e um algoritmo híbrido utilizando GRASP e busca tabu. Ele comprovou que os resultados do primeiro algoritmo foram melhores que o segundo, apesar da maior robustez do segundo. Cordeau e Laporte (2003b) também propuseram uma heurística de busca tabu para o problema com múltiplos veículos. Bergvinsdottir (2004) apresenta um algoritmo genético que utiliza a clássica estratégia "dividir e rotear", assim, a fase de divisão dos nós é feita com um algoritmo genético.

O trabalho apresentado por Cordeau (2006) compara a solução obtida em um algoritmo branch-and-cut utilizando o CPLEX. Os testes foram feitos com redes de, no máximo, 32 consumidores, mostrando um maior desempenho do método branch-and-cut.

O problema dial-a-ride pode ser associado ao Grupo 2, pois considera as requisições de transporte associadas a uma origem e um destino.

2.2.2.5 TSP com Coleta de Retorno e VRP com Coleta de Retorno

O Problema do Caixeiro Viajante com Coleta de Retorno (TSPB, do inglês Traveling Salesman Problem with Backhauls) e o Problema de Roteamento de Veículos com Coleta de Retorno são extensões do TSP e VRP, respectivamente. A diferença é que ambos trabalham com coleta e entrega, sendo que todas as entregas devem ser concluídas antes que as coletas possam ser efetuadas. Ambos os problemas consideram o transporte de carga entre depósito e consumidor e vice-versa, tornando-o um problema do Grupo 1.

Goetschalckx e Jacobs-Blecha (1993) apresentam uma heurística baseada no problema

de assinalamento generalizado. Gelogullari (2004) mostra um algoritmo exato para resolução do VRPB e Reimann, Doerner e Hartl (2002) aplicam a metaheurística "Colônia de Formigas".

Potvin, Duhamel e Guertin (1996) mostram uma heurística construtiva para o VRPB. Ela insere cada consumidor na rota usando uma ordem de prioridade. Posteriormente, os autores propõem um algoritmo genético para melhorar a qualidade das soluções.

Ghaziri e Osman (2003) propõem uma rede neural para resolução do TSPB. Os resultados obtidos mostram que a abordagem proposta é comparável com as heurísticas encontradas na literatura em termos de qualidade da solução e recursos computacionais. Os testes foram aplicados em grandes instâncias, superiores a 1000 consumidores.

2.3 Problema de Roteamento de Veículos com Coleta e Entrega e Janelas de Tempo

Nessa variante, o Problema com Coleta e Entrega com Janelas de Tempo (PDPTW) pretende atender a um número de pedidos e de veículos. O pedido consiste em pegar mercadorias num local e entregá-las noutro. Duas janelas temporais são atribuídas a cada pedido: uma janela de coleta, tempo que especifica quando as mercadorias podem ser apanhadas e um prazo de entrega, janela que avisa quando a mercadoria pode ser entregue. Além disso, um tempo de serviço está associado a cada coleta e entrega. O tempo de serviço indica quanto tempo levará para a coleta ou a entrega ser realizada. É permitido um veículo chegar a um local antes da início da janela de tempo, mas então o veículo deve esperar até o início da janela do tempo antes de iniciar a operação. Um veículo não pode nunca chegar a um local após o término do tempo da janela local. A cada pedido é atribuído um conjunto de veículos viável. Isto pode, por exemplo, ser usado para modelar situações onde alguns veículos não podem entrar num determinado local por causa das dimensões do veículo.

Cada veículo tem capacidade limitada e deve sair e retornar ao depósito sem que no meio do trajeto passe pelo depósito novamente. Dois veículos podem ter depósitos de origens e destinos diferentes. A hora de chegada do veículo ao depósito pode variar, no entanto, o horário de partida do mesmo é sempre no instante inicial, se preciso fica esperando o início da janela de tempo do primeiro cliente.

Devem-se construir rotas válidas para os veículos. A rota é válida se o intervalo das janelas de tempo e a capacidade dos veículos são obedecidos ao longo do percurso, cada

coleta é feita antes da entrega, cada coleta e entrega são realizadas pelo mesmo veículo e deve-se analisar se o cliente pode ser atendido pelo veículo em questão. As rotas devem ser construídas de tal ordem que minimizem seus custos.

Como a quantidade de veículos é limitada, podemos encontrar situações em que alguns pedidos não podem ser atribuídos para nenhum veículo. Estes pedidos são colocados em um banco virtual. Em uma situação real, um operador humano deve decidir o que fazer com esses pedidos. O operador pode decidir, por exemplo, pagar hora extra a fim de servir os pedidos restantes.

O objetivo do problema é minimizar a quantidade de veículos, o custo total da viagem, tempo total de viagem, tempo total de espera. Sendo que na literatura encontramos diferentes formas de calcular o custo. Alguns enfocam somente no custo total da viagens enquanto outros fazem a soma ponderada de cada sub objetivo onde os pesos de cada parcela da soma indicam sua importância.

Este problema é NP-Difícil sendo um caso especial do problema do caixeiro viajante. O objetivo deste trabalho é desenvolver um método exato para encontrar boas soluções. O método desenvolvido deve ser razoavelmente rápido, robusto e capaz de lidar com grandes problemas.

2.3.1 Revisão de Literatura

Na literatura encontram-se vários trabalho para PDPTW. Destes trabalhos tem-se tanto métodos heurísticos como métodos exatos.

Gendreau, Laporte e Vigo (1999), Lu e Dessouky (2006) apresentam uma heurística baseada em inserção aplicada ao Problema de Coleta e Entrega com Janelas de Tempo (PDPTW, do inglês Pickup and Delivery Problem with Time Windows). O procedimento consiste em inserir novos nós em rotas já existentes até que isso não seja mais possível. Quando isso ocorre, uma nova rota é criada e o processo continua. Para decidir qual o próximo nó a ser inserido e onde ele deve ser inserido, a maioria das heurísticas de inserção usam como critério o aumento do custo ou distância da rota com a inserção de um novo nó. Dessa forma, o autor propõe uma alternativa para melhorar os resultados inserindo mais um critério de verificação. Esse novo critério avalia o grau de liberdade para futuras inserções de acordo com as janelas de tempo.

Outro aspecto que difere a heurística proposta por Lu e Dessouky (2006) das demais é o fato de considerar soluções visualmente melhores. O autor desenvolve um método

para avaliar soluções visivelmente mais atrativas chamado Crossing Length Percentage (CLP) para incorporar na heurística de inserção. Os resultados obtidos com a heurística construtiva baseada em inserção proposta pelo autor foram bons, inclusive, melhores que a heurística Simulated Annealing com Busca Tabu proposta por Li e Lim (2001) aplicada ao mesmo problema (PDPTW).

Nanry e Barnes (2000) apresentaram uma Busca Tabu Reativa (RTS, do inglês Reactive Tabu Search) para resolver o problema de coleta e entrega com janelas de tempo. Na abordagem feita os autores consideraram os veículos homogêneos localizados em um único depósito. O transporte requer a coleta da mercadoria num lugar predeterminado durante um tempo especificado (janela de tempo) e esta entregue no seu destino. Esse trabalho marca a primeira aplicação da Busca Tabu Reativa para resolução deste problema apresentando bons resultados. Os testes foram feitos em redes com 25, 50 e 100 clientes. No primeiro caso, os testes foram feitos em 29 instâncias e em todos a solução ótima foi encontrada. Já no segundo caso, dos 15 testes aplicados, 14 retornaram a solução ótima. E no terceiro, 9 testes foram feitos obtendo 8 soluções ótimas. Li e Lim (2001) utilizam uma metaheurística híbrida para resolver o problema. A heurística combina Simulated Annealing e Tabu Search. Lim, Lim e Rodrigues (2002) aplicam otimização de "Squeaky wheel" e busca local para o PDPTW. Sua heurística é testada sobre o conjunto de problemas propostos por Li e Lim (2001). O trabalho de Lau e Liang (2001) também aplica uma Busca Tabu para o PDPTW e eles descrevem várias heurísticas construtivas para o problema. Especial atenção é dada à forma de como testar os problemas podem ser construídos a partir instâncias do VRPTW.

William e Barnes (2001) propuseram, para o problema de roteamento de veículos com múltiplas coletas e entregas e janelas de tempo, uma abordagem de uma busca tabu reativa para minimizar o custo das viagem, utilizando uma penalidade na função objetivo ao tempo de viagem, penalidade por violar as restrições de tempo de carga e das janelas de tempo. O abordagem foi testada em instâncias de 25, 50 e 100 clientes. Estes testes foram construídos a partir de instâncias de Solomon (1987).

Xu et al. (2003) consideram um PDPTW com várias restrições da vida real, incluindo múltiplas janelas temporais, compatibilidade e restrições de tempo máximo de condução. O problema é resolvido usando uma heurística de geração de colunas. Seu artigo consideram instâncias com até 500 pedidos.

Sigurd, Pisinger e Sig (2004) resolveram o PDPTW relacionado ao transporte de gado. Para este problema foram introduzidas algumas restrições extras, tais como a prioridade

entre os pedidos, o que significa que alguns pedidos devem ser servidos antes de outros, a fim de evitar a propagação de doenças. O problema é resolvido à otimalidade usando geração de colunas. Os maiores problemas resolvidos continham mais de 200 pedidos.

Outra heurística encontrada na literatura para o PDPTW é um Algoritmo Genético. Pankratz (2005) propõe um algoritmo em que cada gene representa um grupo de demandas. Os resultados apresentados mostraram que a proposta do GGA (do inglês Grouping Genetic Algorithm) foi boa e bastante competitiva em relação aos demais métodos apresentados na literatura para solucionar o PDPTW.

Outro trabalho encontrado na literatura é o de Bent e Hentenryck (2006), Ropke e Pisinger (2006a) tratam o PDPTW utilizando a heurística "Large Neighborhood Search". Nessa abordagem, Ropke e Pisinger (2006a) limitam a quantidade de veículos. Esse método mostrou bons resultados em um tempo computacional razoável em testes feitos em 350 redes distintas, com um número superior a 500 clientes. O método, em 50% dos problemas, provê melhores soluções em relação aos melhores resultados conhecidos na literatura. A heurística implementada por Bent e Hentenryck (2006) foi testada sobre os problemas propostos por Li e Lim (2001).

2.3.2 Definição e Formulação Matemática

O PDPTW é definido num grafo orientado $G = (N, A)$ com um conjunto de vértices $N = \{0, \ldots, 2n+1\}$ e o conjunto de arcos A. O vértice 0 e $2n+1$ representam o depósito de origem e destino (que podem ser a mesma localidade) enquanto os subconjuntos $P = \{1, \ldots, n\}$ e $D = \{n+1, \ldots, 2n\}$ representam os vértices de coleta e entrega, respectivamente. Cada demanda i está associada com o vértice de coleta i e sua entrega $i+n$. Para cada $i \in N$ está associado uma carga q_i e uma duração de serviço não-negativa d_i que satisfaça: $d_0 = d_{2n+1} = 0$, $q_0 = q_{2n+1} = 0$, e para $i = 1, \ldots, n, q_i \geq 0$ e $q_{i+n} = -q_i$. Uma quantidade ilimitada de veículos com capacidade Q para servir as demandas. Com cada arco $(i, j) \in A$ está associado o custo da rota c_{ij} e um tempo de viajem t_{ij}. Uma janela de tempo $[e_i, l_i]$ é então associada a cada vértice $i \in P \cup D$, onde e_i e l_i representam o tempo mínimo e máximo, respectivamente, para começar o serviço em i. O depósito tem então duas janelas $[e_0, l_0]$ e $[e_{2n+1}, l_{2n+1}]$, correspondendo ao tempo que o veículo pode partir e chagar ao depósito, respectivamente. Assume-se que a desigualdade triangular é válida tanto para custo quanto para o tempo da rota. Finalmente, para restrição de grau e precedência, convencionou-se $\mathcal{S}$ como sendo o conjunto de todos os subconjuntos $S \subseteq N$, tais que, $0 \in S, 2n+1 \notin S$ e se $i \notin S$ então $i+n \in S$.

Para qualquer subconjunto de $S \subseteq N$, define-se complementar como $\overline{S} = N \setminus S$. Assim, $\delta(S) = \delta^+(S) \cup \delta^-(S)$, onde $\delta^+(S) = \{(i,j) \in A | i \in S, j \notin S\}$ e $\delta^-(S) = \{(i,j) \in A | i \notin S, j \in S\}$. Por fim, define-se $x(S) = \sum_{i,j \in S} x_{ij}$ e $x(S:T) = \sum_{i \in S} \sum_{j \in T} x_{ij}$, onde $S, T \subseteq N$.

Para cada arco $(i,j) \in A$ está associado uma variável binária igual a 1 se e apenas se o veículo está partindo de i com destino a j. Para cada $i \in P \cup D$, temos B_i como o instante de tempo a partir do qual o serviço pode ser começado, e Q_i, como sendo a carga atual do veículo.

O Problema do Caixeiro Viajante com Coleta e Entrega e Janelas de Tempo (TSPPDTW) pode ser formulado como programação linear inteira mista, como se segue:

$$\text{Min} \sum_{i \in N} \sum_{j \in N} c_{ij} x_{ij} \tag{2.1}$$

Sujeito a:

$$\sum_{i \in N} x_{ij} = 1 \qquad \forall j \in P \cup D \tag{2.2}$$

$$\sum_{j \in N} x_{ij} = 1 \qquad \forall i \in P \cup D \tag{2.3}$$

$$\sum_{i \in N} x_{ij} \leq |S| - 2 \qquad \forall j \in P \cup D \tag{2.4}$$

$$B_j \geq (B_i + d_i + t_{ij}) x_{ij} \qquad \forall i, j \in N \tag{2.5}$$

$$Q_j \geq (Q_i + q_i) x_{ij} \qquad \forall i, j \in N \tag{2.6}$$

$$e_i \leq B_i \leq l_i \qquad \forall i \in N \tag{2.7}$$

$$max\{0, q_i\} \leq Q_i \leq min\{Q, Q + q_i\} \qquad \forall i, j \in N \tag{2.8}$$

$$x_{ij} \in \{0, 1\} \qquad \forall i, j \in N \tag{2.9}$$

A função objetivo (2.1) minimiza o custo total da rota. As restrições (2.2) e (2.3) requerem que cada vértice seja visitado somente uma vez. As inequações (2.4) representam a precedência, onde cada vértice i precisa ser visitado antes de $i + n$, $|S|$ representa a quantidades de vértices neste conjunto. Estas restrições foram originalmente propostas por Ruland e Rodin (1997) para uma frota homogênea mas podem ser aplicadas diretamente em frotas heterogêneas, as condições são viáveis em ambos os casos. A coerência do

tempo e da carga são asseguradas pelas restrições (2.5) e (2.6). As janelas de tempo e da capacidade do veículo são validadas pelas restrições (2.7) e (2.8).

2.4 Problema do Caixeiro Viajante com Coleta e Entrega

O Problema do Caixeiro Viajante com Coleta e Entrega (TSPPD, do inglês Traveling Salesman Problem with Pickup and Delivery) é uma extensão do TSP, sendo que, em cada ponto ou cada cidade, existe demanda de coleta ou entrega de mercadoria. Se ambas as demandas (entrega e coleta) puderem ocorrer em um mesmo ponto, o problema é denominado Problema do Caixeiro Viajante com Coleta e Entrega Simultâneas (TSPSPD, do inglês Traveling Salesman Problem With Simultaneous Pickup and Delivery). Ambos os problemas consideram o transporte de cargas do depósito para os consumidores e dos consumidores para o depósito. Portanto, eles estão incluídos no Grupo 1.

2.4.1 Revisão de Literatura

O Problema do Caixeiro Viajante com Coleta e Entrega foi proposto por Mosheiov (1994). O autor apresenta uma abordagem heurística para o problema e propõe algumas aplicações.

Balas, Fischetti e Pulleyblank (1995), Ruland e Rodin (1997) trabalham com restrições de precedência para o TSP e TSPPD que servem de base para vários outros trabalhos com algoritmos exatos.

Gendreau, Laporte e Vigo (1999) apresentam heurísticas para TSPPD, a primeira baseada numa solução exata de um grupo especial e outra com base em uma busca tabu. Um algoritmo exato (Branch-and-Cut) é descrito por Hernández-Pérez e González (2004). Este último mostrou bons resultados em problemas com até 75 consumidores.

Sete heurísticas de perturbação são descritas e comparadas por Renaud, Boctor e Laporte (2002) para o TSPPD. Um estudo probabilístico realizado por Beraldi et al. (2005) sobre o TSPPD onde apresenta procedimentos com complexidade computacional de $O(n^3)$ na busca de vizinhanças e algumas aproximações com complexidade $O(n^5)$.

Hernández-Pérez e González (2004) estudaram uma generalização do conhecido problema do caixeiro viajante e apresentam um modelo de programação linear inteira para este problema e utilizam um algoritmo branch-and-cut para encontrar soluções ótimas.

O modelo e os algoritmos são facilmente adaptados para resolver os casos do TSP com Coleta e Entrega.

O objetivo do trabalho de Erdogan, Cordeau e Laporte (2009) é introduzir uma nova variante do TSPPD denominado: TSPPD em Forma de Fila (TSPPDF, do inglês The pickup and delivery traveling salesman problem with first-in-first-out loading) que é semelhante à TSPPD, com a diferença de que as operações de coleta e entrega devem ser executadas em forma de fila, o primeiro a entrar deve ser o primeiro a sair. Ele também descreve cinco métodos para melhorar uma solução viável, e heurísticas que utilizam dois destes: um algoritmo probabilístico de busca tabu e um algoritmo de busca local iterada. As instâncias das heurísticas foram adaptadas a partir das instâncias da TSPLIB. Similarmente Carrabs, Cordeau e Laporte (2007) propõem uma nova variante para o TSPPD chamada TSPPD em Forma de Pilha (TSPPDL, do inglês The pickup and delivery traveling salesman problem with LIFO loading).

2.4.2 Definição Formal do Problema

Seja $G = (V, E)$, um grafo completo não orientado, onde V é o conjunto de vértices e E o conjunto de arestas. O conjunto V é constituído pelos vértices de coleta e entrega, além de dois vértices que correspondem ao depósito, à saída e ahegada a ele. Um par de vértices de coleta e entrega formam uma requisição. A quantidade de requisições é indicada por n. Seja $P = \{1, \ldots, n\}$ o conjunto de vértices de coleta e $D = \{n+1, \ldots, 2n\}$ o conjuntos dos vértices de entrega. O vértice de entrega correspondente ao vértice de coleta $i \in P$ é $i + n \in D$. Desta forma, $V = P \cup D \cup \{0, 2n + 1\}$, onde 0 corresponde a partida do depósito e $2n+1$ a chegada a ele. Para dois vértices i e j, $i < j$, a aresta associada a esse par é denotada como (i, j). Um custo não-negativo c_{ij} é associado com a aresta $(i, j) \in E$. O TSPPD consiste em encontrar um caminho hamiltoniano sobre G que deve começar em (0) e terminar em $(2n + 1)$, passando por uma coleta i antes de sua entrega $i + n$. Juntando-se a esse caminho a aresta $(0, 2n + 1)$, que não possui custo, temos um circuito hamiltoniano sobre G. Maiores detalhes serão apresentados no Capítulo 4.

3 Heurística para o Problema do Roteamento de Veículo com Coleta e Entrega e Janelas de Tempo - VRPPDTW

Este capítulo apresenta detalhes da implementação do método heurístico proposto para o problema de roteamento de veículos com coleta e entrega e janelas de tempo, baseado em uma Busca Local Iterada (Iterated Local Search - ILS) conjugado a um Método Randômico de Descida (MRD). Também são apresentados os resultados computacionais obtidos e comparações com outros métodos da literatura.

3.1 Metaheuristica

Muitas vezes, com uma boa heurística construtiva e busca local, o ótimo local obtido é suficientemente bom. No entanto, em otimização combinatória, pode-se encontrar soluções ainda distantes de um ótimo global. Neste contexto é incentivado o uso de heurísticas que gerem muitos ótimos locais, como é o caso das metaheurísticas. Dentre elas podemos citar a Busca Local Iterada (do inglês, Iterated Local Search - ILS), Busca Tabu (do inglês, Tabu Search - TS), Simulated Annealing, VNS, etc.

3.1.1 Busca Local Iterada (ILS)

Nesse trabalho foi escolhido o método de busca ILS e sua maior vantagem consiste na sua facilidade de implementação. Mas muitas vezes, a sua versão original não consegue soluções suficientemente boas num tempo razoável, o que leva à utilização de variantes mais sofisticados.

O ILS, proposto por Stützle e Hoos (1999), é um método de busca local que procura focar a busca não no espaço completo de soluções, mas num subespaço contendo soluções

que representam ótimos locais. De acordo com Lourenço, Martin e Stützle (2003), o sucesso do ILS é centrado no conjunto de amostragem de ótimos locais, juntamente com a escolha da busca local, das perturbações e do critério de aceitação.

Juntamente com a busca local do ILS será utilizado o MRD (Método Randômico de Descida) que é uma heurística de refinamento que consiste em analisar um vizinho qualquer e o aceitar somente se ele for estritamente melhor que a solução corrente. Caso esse vizinho não seja melhor, a solução corrente permanece inalterada e outro vizinho é gerado. O método é finalizado quando se atinge um número máximo de iterações (maxIter) sem que haja a produção de melhorias na solução corrente. O algoritmo da Figura 2 explica o funcionamento do ILS-MRD.

Saída: Uma solução S

```
1  s_0 ← solucaoInicial
2  s' ← MRD(s_0, maxIter)
3  kp ← 0
4  iter ← 0
5  enquanto kp < kp_max faça
6      iter ← melhorIter
7      enquanto iter < iter_max faça
8          iter ← iter + 1
9          s' ← perturbacao(s', historico)
10         s'' ← MRD(s', maxIter)
11         se (s'' < s') então
12             s' ← s''
13             melhorIter ← iter
14             kp ← 0
15     kp ← kp + delta
16 retorna s'
```

Figura 2: Algoritmo ILS-MRD

3.1.2 Método Randômico de Descida (MRD)

O algoritmo MRD é descrito na Figura 3 onde recebe como entrada a quantidade máxima de iterações e a solução corrente. Na linha 1 inicia-se a contagem das iterações,

enquanto não atingir o máximo de iterações (linha 2), incrementa-se o contador (linha 3) e gera-se uma nova solução através da escolha aleatória de um vizinho (linha 4). Caso encontre melhora então a solução corrente passa a ser a melhor (linha 5), zerando o contador (linha 6) e o loop é retomado.

Input: S,maxIter
Saída: Uma solução S
1 $iter \leftarrow 0$
2 **enquanto** $iter < maxIter$ **faça**
3 $iter \leftarrow iter + 1$
4 Gere aleatoriamente um vizinho $S' \in N(S)$
5 **se** $f(S') < f(S)$ **então**
6 $S \leftarrow S'$
7 $iter \leftarrow 0$
8 **retorna** S

Figura 3: Algoritmo MRD

A Figura 4 representa o objetivo da perturbação que é tentar escapar de um ótimo local, S^*, a seguir obtém-se $S^{'}$ perturbando S^*. A partir de $S^{'}$ efetua-se uma busca local ao seu redor obtendo $S^{*'}$, que pode ser melhor que a solução anterior.

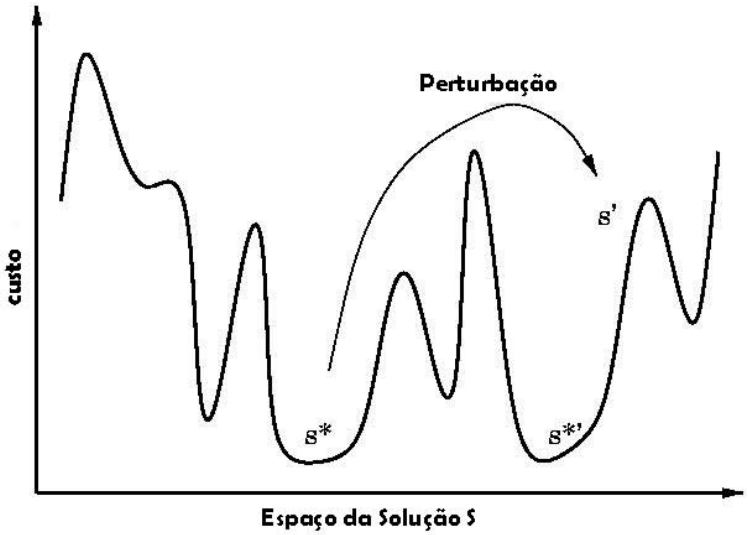


Figura 4: Perturbação - ILS

3.1.3 Objetivo e custos

Na literatura há várias propostas para se calcular a função objetivo no VRPPDTW. Mas basicamente obedecem as seguintes ordens:

1. minimizar o número de veículos, logo o número de rotas, uma vez que há um veículo por rota.
2. minimizar o custo total das viagens.
3. minimizar o tempo total de cada rota.
4. minimizar o tempo de espera.

Li e Lim (2001), Tam e Tseng (2003), Dumitrescu et al. (2008), por exemplo, adaptaram seus algoritmos para obedecerem a essa ordem. No entanto, Pankratz (2005) tinha como único objetivo minimizar o custo da rota, ignorando assim a quantidade de veículos. Neste trabalho, o objetivo do problema VRPPDTW foi usado a seqüência de prioridades descrita inicialmente nesta seção.

3.2 Solução Inicial

Neste trabalho os pares de atividades associados aos cliente (coleta e entrega), serão denotados apenas de "par". Deste modo, o princípio básico dos algoritmos descritos nesta seção é dado por: pega-se aleatoriamente um par, que não esteja em nenhuma rota, o qual irá forma uma rota trivial "depósito-entrega-coleta-depósito" ($\{v_0, v_i, v_j, v_0\} \therefore i \in P, j \in D$) e vai-se inserindo outros pares, ainda não inseridos na rota atual até chegar ao ponto que não possa mais ser colocado nenhum elemento sem que este viole as restrições. Então se cria uma nova rota e tenta-se fazer o mesmo processo acima, até que não haja mais pares, sem rota, para serem colocados.

A inserção de um novo par à rota é feita analisando todas as possíveis posições para um par ser encaixado entre os vértices da rota e escolhendo a melhor das posições a qual obedece a ordem relatada anteriormente.

3.2.1 Vizinhança mais próxima - C1

Partindo do princípio básico (descrino em 3.2) esse algoritmo é um algoritmo não-determinístico de *Vizinhança mais Próxima*. O C1 para o VRPPDTW é descrito no pseudo-algoritmo da Figura 5. O seu objetivo é construir uma solução que consiste de várias rotas onde cada uma dela é construída iterativamente escolhendo-se os vizinhos mais próximos para formá-la.

Primeiramente, pega-se a lista de vértices de coleta ou entrega, nesse caso serão os de coleta (linha 1), pois sempre serão colocados em pares na rota (linha 7). Depois, inicia-se um laço para garantir que todos os vértices serão inseridos. Cria-se uma rota (linha 3) e tenta-se inserir a maior quantidade de elementos que esta ela suportar (linha 7), ou seja, inserir todos os elementos que não violem nenhuma restrição. A escolha do primeiro elemento da rota é feita de forma aleatória (linha 4), o que torna esse algoritmo não-determinístico, no entanto, os demais são escolhidos baseados num critério de distância, onde serão o próximo elemento escolhido será sempre o vizinho mais próximo válido (linha 10). Na linha 6 tem-se um laço para percorrer todos os possíveis candidatos sem rota. Caso a inserção seja bem sucedida então esse o vértice é removido da lista de candidatos (linha 9). A linha 10 refere-se à procura do vizinho mais próximo ao atual inserido e que ainda não foi descartado (linha 8). Finalmente não conseguindo inserir mais nenhum elemento nessa rota, a mesma é adicionada a solução (linha 12) e cria-se outra nova rota vazia para ser preenchida até que não haja mais pares a serem inseridos (linha 2).

```
   Saída: Uma solução S
 1 vect ← pickups
 2 enquanto vect  tiver elementos faça
 3 |   cria-se uma nova rota: route
 4 |   id ←  elemento aleatório de vect
 5 |   i ← 0
 6 |   enquanto i <  quantidades de elementos atuais de vect faça
 7 |   |   sucess ←  insere id e id + n a route
 8 |   |   se sucess então
 9 |   |   |   remove id de vect
10 |   |   id ←  escolhe um novo vizinho ainda não escolhido
11 |   |   i ← i + 1
12 |   Adiciona route a solução parcial S
13 retorna S
```

Figura 5: Vizinhança mais próxima - C1

3.2.2 Flexível - C2

O principal objetivo deste algoritmo não é criar rotas válidas. Como este trabalho propõe um ILS o qual tem uma fase de perturbação. Este algoritmo obriga a ter uma busca local de validação da solução como sendo o primeiro passo da busca local.

Esse algoritmo nem sempre cria rotas válidas, pois depende da folga que é acrescentada a cada janela de tempo, proporcionando assim rotas com maiores quantidade de pares,

porém rotas possivelmente violadas.

Sua estrutura é semelhante ao C1, com uma diferença que na linha 7 deve ser acrescentado uma folga. A folga refere-se ao acréscimo que é dado no final de cada janela de tempo possibilitando momentaneamente que um vértice que antes era violado fique válido. O valor da folga não é fixo, deste modo, de ver pensado um valor que não seja irrelevante e que também não seja exagerado, perturbando demais a solução.

3.3 Estrutura de Vizinhanças

Há várias formas de se fazer a busca de vizinhanças ($N(S)$) a fim de encontrar uma melhor solução. As operações $N(S)$ são: PD-Shift, PD-Exchange, PD-Rearrange (LI; LIM, 2001), PD-Eliminate e Viabiliza Rota. As duas primeiras operações tem o objetivo de tenta fugir de ótimos locais ainda distantes de um ótimo global, a terceira é usada como uma pós-otimização, para melhorar o arranjo dos elementos que compões a rota. A quarta é tentar reduzir a quantidade de rotas da solução e a última tem o objetivo de tentar de forma mais rápida diminuir a quantidade de rotas totais da solução.

As operações PD-Shift, PD-Exchange e PD-Rearrange descrita em Li e Lim (2001) e implementadas neste trabalho tem o seguinte aspecto: sempre haverá duas maneiras para escolher a vizinhança a ser modificada. Ou de forma aleatória ou na escolha de um elemento da lista de vizinhos. Possivelmente em várias iterações a escolha do vizinho seria repetitiva, diminuindo assim a eficiência da busca, e para evitar isso usam-se 2 métodos trabalhados ao mesmo tempo. O primeiro é que a escolha do vizinho é a partir de uma busca aleatória de vizinhos e a outra é criando uma estrutura de memória para evitar movimentos cíclicos.

3.3.1 Operação PD-Shift

Esta operação consiste em pegar um par de uma rota e transferir este para outra e vice-versa. Essa operação é denotada por $N_{PDS}(S)$. Para cada par de rotas selecionadas, no exemplo da Figura 6, temos as rotas: *Rota 1* e *Rota 2*. A operação será a remoção primeiramente de um par da *Rota 1* e a inserção deste na *Rota 2*, e o contrário.

A Figura 6 mostra somente o envio do par P e D retirado da *Rota 1* e sendo colocado na *Rota 2*, ou seja, somente metade do algoritmo. A posição escolhida para ser colocado esse novo par é exatamente como descrito na Seção 3.2.

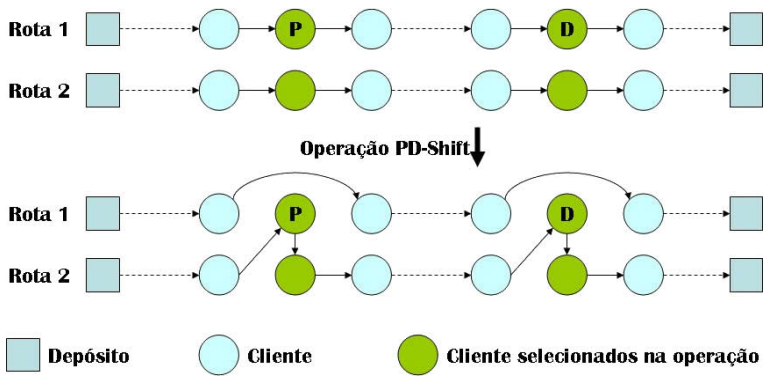


Figura 6: Operação PD-Shift

3.3.2 Operação PD-Exchange

Nesta operação a troca é simultânea de pares entre duas rotas, sujeitos a todas as restrições impostas pelo VRPPDTW. A operação é denotada por $N_{PDE}(S)$. Na Figura 7, os clientes *P1* e *D1* estão inicialmente na *Rota 1*, enquanto os clientes *P2* e *D2* estão na *Rota 2*. A operação PD-Exchange remove dois pares, um de cada rota, depois tenta-se inserir o par *P1-D1* na *Rota 2* na melhor posição viável, enquanto o par *P2-D2* é inserido na rota *Rota 1*.

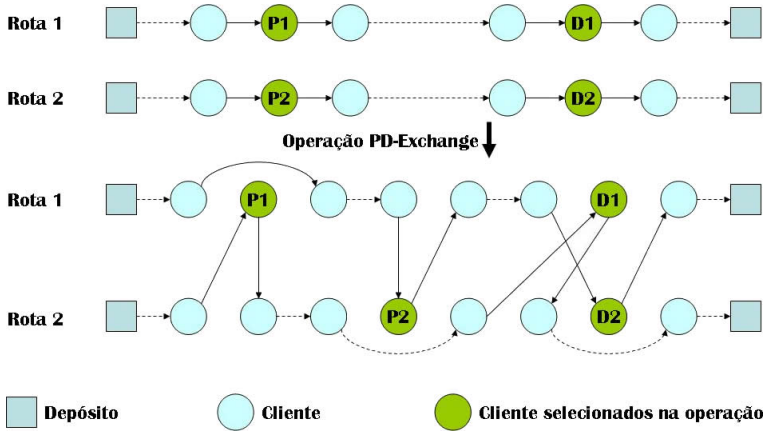


Figura 7: Operação PD-Exchange

3.3.3 Operação PD-Rearrange

A operação PD-Rearrange consiste em rearranjar uma determinada rota. Este processo terá a seguinte nomenclatura: $N_{PDR}(S)$. A Figura 8 ilustra a operação onde um par da rota é removido e depois inserido na mesma rota porém tenta-se a melhor posição para este. É bastante rápido e com bons resultados. Sempre que ocorrer várias perturbações

na solução inicial esta operação deve ser aplicada.

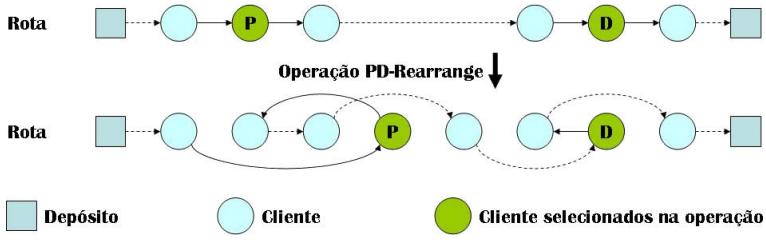


Figura 8: Operação PD-Rearrange

3.3.4 Operação PD-Eliminate

Esta operação é muito semelhante à operação PD-Shift e a chamaremos de $N_{PDX}(S)$. Para isso escolhe-se uma rota e tenta-se retirar os pares de uma delas e colocar nas outras rotas. A rota a qual cada par vai ser realocado é analisado de acordo com as rotas mais próximas deste par. Caso a primeira mais próxima não suporte o par, então tenta-se na segunda mais próxima e assim por diante. No entanto a proximidade é relativa, pois analisando a rota mais próxima do par-coleta teremos uma rota e analisando o par-entrega pode ter outra rota. Assim caso as a rota mais próxima do dois seja a mesma, com certeza está será a escolhida, caso contrário, escolhe-se a rota mais com menor distância do par.

3.3.5 Viabiliza Rota

Como nem sempre são criadas rotas válidas, pois há tanto o algoritmo (Seção 3.2.2) quanto a fase de perturbação (veja 3.4) podem gerar rotas violadas. Fez-se necessário a criação desse processo de viabilizar as rotas. O algoritmo C2 (Seção 3.2.2), por exemplo, na maioria das vezes cria rotas inválidas. Para tornar válida uma rota procede-se de duas formas: a primeira tenta-se retirar elementos desta rota e colocar em outras existentes até que se torne válida, ou seja, remove-se um par da rota violada e tenta-se introduzi-lo em outra rota, o processo pára quando todas as rotas estiverem válidas. Caso na inserção a inicial se torne válida e a rota que ganhou o par se torne violada a inserção é desfeita e procura-se outra rota para esse par. Caso não se ache nenhuma rota em que ele se encaixe, escolhe-se outro par e o ciclo é reiniciado.

A outra forma, corresponde a criar uma nova rota. Esta será realizada somente caso não se consiga solução válida na primeira etapa. Pois acrescentar uma nova rota é um passo contra a função objetivo que tenta diminuir quantidade de veículos utilizados.

3.4 Perturbação

A fim de fugir de ótimos locais, a perturbação tenta encontrar novas soluções que não seriam possíveis somente com a busca local. A perturbação da solução não pode ser muito fraca de forma que volte rapidamente a solução corrente. Nem muito grande o que provavelmente causará uma grande perda em relação a quanlidade de soluções corrente.

Tento todas as rotas já formadas de uma solução S a perturbação ocorre quando deixa-se de validar algumas restrições como por exemplo as janelas de tempo, a capacidade do veículo, etc. Assim, a perturbação utilizada nesse trabalho foi a retirada de alguns pares de uma rota e inseri-los em outra ignorando algumas restrições. Neste trabalho as restrições violadas foram as das janelas de tempo. Tendo em conta o equilíbrio, a remoção e inserção de um par não é feita de forma aleatória, e sim após um varredura na rota e em sua vizinhança para descobrir qual seria o melhor par a ser retirado e também qual a melhor rota que este se encaixaria.

Geralmente a perturbação gera rotas violadas o que torna obrigatória uma busca local para poder validar a rota. Na seção 3.3.5 foi descrito como esse processo de validação é realizado.

3.5 Resultados computacionais

Os testes foram realizados numa máquina com o Sistema Operacional Windows Vista, core 2 due E7300, 2,66GHz e 4Gb de memória RAM. Algoritmos codificados em C++ com Standard Template Library. As instâncias foram baseadas em Solomon (1987).

As instâncias podem ser encontradas em

<http://www.sintef.no/static/am/opti/projects/top/vrp/benchmarks.html>,

onde também estão os melhores resultados encontrados na literatura. Para resolver tais instâncias este trabalho utilizou as seguintes constantes para o ISL-MRD: $maxInter = 100; kpmax = 10; delta = 2;$. Com relação aos métodos de solução inicial foi testado C1 e C2 com uma folga de $100, 500 e 1000$. Foram rodadas 116 instâncias com aproximadamente 100 e 200 clientes.

As Tabelas apresentam os seguintes valores: na primeira coluna tem-se o nome da instância, na segunda a quantidade de veículos (V). T representa o tempo (custo) total de viagem, na seguinte, relativos a literatura. Estes tem siglas que referem-se a: Li & Lim

Instância	Melhor Resultado			ILS-MRD		
	V	T	Pub	V	T	Tempo (s)
lc101	10	**828,94**	Li & Lim	10	**828,94**	0,50
lc102	10	**828,94**	Li & Lim	10	**828,94**	0,44
lc103	**9**	1.035,35	BVH	*10*	**827,87**	0,47
lc104	9	**860,01**	SAM::OPT	9	876,93	2,98
lc105	10	**828,94**	Li & Lim	10	**828,94**	0,48
lc106	10	**828,94**	Li & Lim	10	**828,94**	0,48
lc107	10	**828,94**	Li & Lim	10	**828,94**	0,53
lc108	10	**826,44**	Li & Lim	10	**826,44**	0,62
lc109	**9**	1.000,60	BVH	*10*	**827,82**	1,64
lc201	3	**591,56**	Li & Lim	3	**591,56**	1,29
lc202	3	**591,56**	Li & Lim	3	**591,56**	1,98
lc203	3	**585,56**	Li & Lim	3	591,17	3,58
lc204	3	**590,60**	SAM::OPT	3	596,76	4,49
lc205	3	**588,88**	Li & Lim	3	**588,88**	1,76
lc206	3	**588,49**	Li & Lim	3	**588,49**	2,95
lc207	3	**588,29**	Li & Lim	3	**588,29**	4,74
lc208	3	**588,32**	Li & Lim	3	**588,32**	2,92

Tabela 1: Resultados das Instâncias de 100 pares - LC

Li e Lim (2001); BVH - Bent e Hentenryck (2006); RP - Ropke e Pisinger (2006a); TS - TetraSoft (2003); SAM::OPT - Mathematics (Em processo). Depois o V, T e o Tempo de CPU em segundos encontrados no ILS-MRD. Os resultados apresentados foram realizados a seguir são apenas de C1, pois C2 se mostrou pior que C1.

Os testes mostraram que o ILS-MRD consegue boas soluções em tempos razoáveis. Para as instâncias de 100 clientes, representadas nas Tabelas 1, 2 e 3, das 56 conseguiu-se encontrar o melhor valor da literatura em 35 dos 56 casos. Não foi calculado GAP pois os valores seriam duvidosos uma vez que o objetivo é diminuir a quantidade de rota assim existem instâncias com maior quantidade de rotas e custo menor. O tempo médio dessas instâncias foram de 3,55s. Para as instâncias de 200 clientes, conseguiu-se chegar aos valores da literatura em 26 dos 60 casos. Os tempos foram um pouco maiores comparados aos anteriores com média de 20,82s.

Instância	Melhor Resultado			ILS-MRD		
	V	T	Pub	V	T	Tempo (s)
lr101	19	**1.650,80**	Li & Lim	19	**1.650,80**	0,22
lr102	17	**1.487,57**	Li & Lim	17	**1.487,57**	0,22
lr103	13	**1.292,68**	Li & Lim	13	**1.292,68**	0,20
lr104	9	**1.013,39**	Li & Lim	9	**1.013,39**	0,28
lr105	14	**1.377,11**	Li & Lim	14	**1.377,11**	0,33
lr106	12	**1.252,62**	Li & Lim	12	**1.252,62**	0,36
lr107	10	**1.111,31**	Li & Lim	10	**1.111,31**	3,28
lr108	9	**968,97**	Li & Lim	9	**968,97**	0,50
lr109	11	**1.208,96**	SAM::OPT	11	1.239,96	3,33
lr110	10	**1.159,35**	Li & Lim	11	1.165,83	2,39
lr111	10	**1.108,90**	Li & Lim	10	**1.108,90**	0,42
lr112	9	**1.003,77**	Li & Lim	9	**1.003,77**	0,47
lr201	4	**1.253,23**	SAM::OPT	4	1.256,72	4,69
lr202	3	**1.197,67**	Li & Lim	3	**1.197,67**	2,45
lr203	3	**949,40**	Li & Lim	3	**949,40**	6,24
lr204	2	**849,05**	Li & Lim	2	**849,05**	9,84
lr205	3	**1.054,02**	Li & Lim	3	**1.054,02**	1,79
lr206	3	**931,63**	Li & Lim	3	**931,63**	2,98
lr207	2	**903,06**	Li & Lim	2	**903,06**	13,31
lr208	2	**734,85**	Li & Lim	2	736,00	17,74
lr209	3	**930,59**	SAM::OPT	3	937,05	8,51
lr210	3	**964,22**	Li & Lim	3	**964,22**	6,00
lr211	**2**	911,52	SAM::OPT	*3*	**896,76**	5,94

Tabela 2: Resultados das Instâncias de 100 pares - LR

Instância	Melhor Resultado			ILS-MRD		
	V	T	Pub	V	T	Tempo (s)
lrc101	14	**1.708,80**	Li & Lim	14	**1.708,80**	0,25
lrc102	12	**1.558,07**	SAM::OPT	12	**1.558,07**	4,23
lrc103	11	**1.258,74**	Li & Lim	11	**1.258,74**	0,33
lrc104	10	**1.128,40**	Li & Lim	10	1.129,95	1,47
lrc105	13	**1.637,62**	Li & Lim	13	1.643,88	7,33
lrc106	11	**1.424,73**	SAM::OPT	11	**1.424,73**	4,30
lrc107	11	**1.230,15**	Li & Lim	11	**1.230,14**	5,31
lrc108	10	**1.147,43**	SAM::OPT	10	1.152,87	3,39
lrc201	4	**1.406,94**	SAM::OPT	4	1.439,67	6,00
lrc202	**3**	**1.374,27**	Li & Lim	4	1.385,25	6,45
lrc203	3	**1.089,07**	Li & Lim	3	**1.089,07**	6,89
lrc204	3	**818,66**	SAM::OPT	3	820,66	10,55
lrc205	4	**1.302,20**	Li & Lim	4	1.305,91	0,78
lrc206	3	**1.159,03**	SAM::OPT	3	1.164,35	5,42
lrc207	3	**1.062,05**	SAM::OPT	3	1.064,40	9,58
lrc208	3	**852,76**	Li & Lim	3	861,31	3,00

Tabela 3: Resultados das Instâncias de 100 pares - LRC

Instância	Melhor Resultado			ILS-MRD		
	V	T	Pub	V	T	Tempo (s)
LC1_2_1	20	**2.704,57**	Li & Lim	20	**2.704,57**	0,95
LC1_2_2	19	**2.764,56**	Li & Lim	19	**2.764,56**	0,53
LC1_2_3	**17**	3.128,61	RP	*18*	**2.773,91**	10,76
LC1_2_4	**17**	2.693,41	BVH	*18*	**2.673,57**	10,90
LC1_2_5	20	**2.702,05**	Li & Lim	20	**2.702,50**	0,78
LC1_2_6	20	**2.701,04**	Li & Lim	20	**2.701,04**	0,90
LC1_2_7	20	**2.701,04**	Li & Lim	20	**2.701,04**	0,94
LC1_2_8	20	**2.689,83**	Li & Lim	20	**2.689,83**	0,73
LC1_2_9	18	**2.724,24**	Li & Lim	18	**2.724,24**	0,87
LC1_2_10	**17**	2.943,49	RP	*18*	**2.741,56**	0,69
LC2_2_1	6	**1.931,44**	Li & Lim	6	**1.931,44**	2,42
LC2_2_2	6	**1.881,40**	Li & Lim	6	**1.881,40**	3,96
LC2_2_3	6	**1.844,33**	SAM::OPT	6	**1.844,33**	4,46
LC2_2_4	6	**1.767,12**	Li & Lim	6	1.778,54	10,50
LC2_2_5	6	**1.891,21**	Li & Lim	6	**1.891,21**	1,73
LC2_2_6	6	**1.857,78**	SAM::OPT	6	1.858,25	3,67
LC2_2_7	6	**1.850,13**	SAM::OPT	6	**1.850,13**	3,34
LC2_2_8	6	**1.824,34**	Li & Lim	6	**1.824,34**	2,46
LC2_2_9	6	**1.854,21**	SAM::OPT	6	**1.854,21**	6,15
LC2_2_10	6	**1.817,45**	Li & Lim	6	**1.817,45**	3,73

Tabela 4: Resultados das Instâncias de 200 pares - LC

Instância	Melhor Resultado			ILS-MRD		
	V	T	Pub	V	T	Tempo (s)
LR1_2_1	20	**4.819,12**	Li & Lim	20	**4.819,12**	4,49
LR1_2_2	**17**	4.621,21	RP	*19*	**4.205,50**	5,21
LR1_2_3	15	**3.612,64**	TS	15	3.657,19	29,73
LR1_2_4	10	**3.037,38**	RP	10	3.089,86	11,89
LR1_2_5	16	**4.760,18**	BVH	16	**4.760,18**	5,23
LR1_2_6	14	**4.175,16**	BVH	14	4.201,82	5,96
LR1_2_7	12	**3.550,61**	RP	12	3.851,36	7,78
LR1_2_8	9	**2.784,53**	RP	9	2.828,09	25,85
LR1_2_9	14	**4.354,66**	RP	14	4.411,54	11,45
LR1_2_10	11	**3.714,16**	RP	11	3.744,95	9,16
LR2_2_1	5	**4.073,10**	SAM::OPT	5	**4.073,10**	16,88
LR2_2_2	4	**3.796,00**	SAM::OPT	4	**3.796,00**	7,85
LR2_2_3	4	**3.098,36**	RP	4	**3.098,36**	98,97
LR2_2_4	3	**2.486,14**	RP	3	2.737,22	78,41
LR2_2_5	4	**3.438,39**	SAM::OPT	4	**3.438,39**	80,61
LR2_2_6	4	**3.201,54**	Li & Lim	4	3.208,53	14,74
LR2_2_7	3	**3.135,05**	RP	3	3.337,28	67,11
LR2_2_8	**2**	**2.555,40**	RP	3	2.736,35	107,60
LR2_2_9	**3**	3.930,49	RP	*4*	**3.198,44**	49,66
LR2_2_10	3	**3.323,37**	SAM::OPT	3	3.478,67	26,47

Tabela 5: Resultados das Instâncias de 200 pares - LR

Instância	Melhor Resultado			ILS-MRD		
	V	T	Pub	V	T	Tempo (s)
LRC1_2_1	19	**3.606,06**	SAM::OPT	19	**3.606,06**	4,52
LRC1_2_2	**15**	**3.673,19**	BVH	16	3.681,36	16,44
LRC1_2_3	13	**3.161,75**	BVH	13	3.251,38	11,78
LRC1_2_4	10	**2.631,82**	RP	10	2.655,27	16,60
LRC1_2_5	16	**3.715,81**	BVH	16	**3.715,81**	15,24
LRC1_2_6	17	**3.368,66**	SAM::OPT	17	**3.368,66**	14,96
LRC1_2_7	**14**	3.668,39	RP	*15*	**3.417,16**	25,54
LRC1_2_8	**13**	3.226,72	RP	*14*	**3.087,62**	10,55
LRC1_2_9	13	**3.174,55**	RP	13	3.226,72	36,79
LRC1_2_10	**12**	2.951,29	RP	*13*	**2.833,85**	48,34
LRC2_2_1	6	**3.605,40**	RP	6	3.690,10	12,90
LRC2_2_2	**5**	3.327,18	RP	*6*	**2.666,01**	12,15
LRC2_2_3	4	**2.938,28**	RP	4	3.249,36	53,70
LRC2_2_4	**3**	2.887,97	RP	*4*	**2.795,70**	42,26
LRC2_2_5	5	**2.776,93**	BVH	5	**2.776,93**	32,56
LRC2_2_6	5	**2.707,96**	SAM::OPT	5	**2.707,96**	22,26
LRC2_2_7	**4**	3.050,03	BVH	*5*	**2.816,41**	25,76
LRC2_2_8	4	**2.399,95**	RP	4	3.050,03	46,36
LRC2_2_9	4	**2.208,49**	RP	4	2.750,30	37,61
LRC2_2_10	3	**2.550,56**	RP	3	2.699,55	27,32

Tabela 6: Resultados das Instâncias de 200 pares - LRC

Tabela 7: Resultados das Instâncias de 100 pares - LC

	Li01		Pankratz05		Bent06		Derigs06		Ropke06		ILS-MRD	
	V	T	V	T	V	T	V	T	V	T	V	T
1	**10**	**828,94**	**10**	**828,94**	**10**	**828,94**	**10**	**828,94**	**10**	**828,94**	**10**	**828,94**
2	**10**	**828,94**	**10**	**828,94**	**10**	**828,94**	**10**	**828,94**	**10**	**828,94**	**10**	**828,94**
3	10	827,86	10	827,86	**9**	**1.035,35**	10	827,86	**9**	**1.035,35**	10	827,87
4	9	861,95	10	818,60	**9**	**860,01**	**9**	**860,01**	**9**	**860,01**	9	876,93
5	**10**	**828,94**	**10**	**828,94**	**10**	**828,94**	**10**	**828,94**	**10**	**828,94**	**10**	**828,94**
6	**10**	**828,94**	**10**	**828,94**	**10**	**828,94**	**10**	**828,94**	**10**	**828,94**	**10**	**828,94**
7	**10**	**828,94**	**10**	**828,94**	**10**	**828,94**	**10**	**828,94**	**10**	**828,94**	**10**	**828,94**
8	**10**	**826,44**	**10**	**826,44**	**10**	**826,44**	**10**	**826,44**	**10**	**826,44**	**10**	**826,44**
9	10	827,82	10	827,82	**9**	**1.000,60**	10	827,82	**9**	**1.000,60**	10	827,82
1	**3**	**591,56**	**3**	**591,56**	**3**	**591,56**	**3**	**591,56**	**3**	**591,56**	**3**	**591,56**
2	**3**	**591,56**	**3**	**591,56**	**3**	**591,56**	**3**	**591,56**	**3**	**591,56**	**3**	**591,56**
3	**3**	**585,56**	3	591,17	3	591,17	3	591,17	3	591,17	3	**591,17**
4	3	591,17	**3**	**590,60**	**3**	**590,60**	**3**	**590,60**	**3**	**590,60**	3	596,76
5	**3**	**588,88**	**3**	**588,88**	**3**	**588,88**	**3**	**588,88**	**3**	**588,88**	**3**	**588,88**
6	**3**	**588,49**	**3**	**588,49**	**3**	**588,49**	**3**	**588,49**	**3**	**588,49**	**3**	**588,49**
7	**3**	**588,29**	**3**	**588,29**	**3**	**588,29**	**3**	**588,29**	**3**	**588,29**	**3**	**588,29**
8	**3**	**588,32**	**3**	**588,32**	**3**	**588,32**	**3**	**588,32**	**3**	**588,32**	**3**	**588,32**

Tabela 8: Resultados das Instâncias de 100 pares - LR

	Li01		Pankratz05		Bent06		Derigs06		Ropke06		ILS-MRD	
	V	T	V	T	V	T	V	T	V	T	V	T
1	**19**	**1.650,78**	19	1.650,80	19	1.650,80	19	1.650,80	19	1.650,80	19	1.650,80
2	**17**	**1.487,57**	**17**	**1.487,57**	**17**	**1.487,57**	**17**	**1.487,57**	**17**	**1.487,57**	**17**	**1.487,57**
3	**13**	**1.292,68**	**13**	**1.292,68**	**13**	**1.292,68**	**13**	**1.292,68**	**13**	**1.292,68**	**13**	**1.292,68**
4	**9**	**1.013,39**	**9**	**1.013,99**	**9**	**1.013,39**	**9**	**1.013,99**	**9**	**1.013,39**	**9**	**1.013,39**
5	**14**	**1.377,11**	**14**	**1.377,11**	**14**	**1.377,11**	**14**	**1.377,11**	**14**	**1.377,11**	**14**	**1.377,11**
6	**12**	**1.252,62**	**12**	**1.252,62**	**12**	**1.252,62**	**12**	**1.252,62**	**12**	**1.252,62**	**12**	**1.252,62**
7	**10**	**1.111,31**	**10**	**1.111,31**	**10**	**1.111,31**	**10**	**1.111,31**	**10**	**1.111,31**	**10**	**1.111,31**
8	**9**	**968,97**	**9**	**968,97**	**9**	**968,97**	**9**	**968,97**	**9**	**968,97**	**9**	**968,97**
9	11	1.239,96	**11**	**1.208,96**	**11**	**1.208,96**	**11**	**1.208,96**	**11**	**1.208,96**	11	1.239,96
10	**10**	**1.159,35**	11	1.165,83	**10**	**1.159,35**	**10**	**1.159,35**	**10**	**1.159,35**	11	1.165,83
11	**10**	**1.108,90**	**10**	**1.108,90**	**10**	**1.108,90**	**10**	**1.108,90**	**10**	**1.108,90**	**10**	**1.108,90**
12	**9**	**1.003,77**	**9**	**1.003,77**	**9**	**1.003,77**	**9**	**1.003,77**	**9**	**1.003,77**	**9**	**1.003,77**
1	4	1.263,84	**4**	**1.253,23**	**4**	**1.253,23**	**4**	**1.253,23**	**4**	**1.253,23**	4	1.256,72
2	**3**	**1.197,67**	**3**	**1.197,67**	**3**	**1.197,67**	**3**	**1.197,67**	**3**	**1.197,67**	**3**	**1.197,67**
3	**3**	**949,40**	3	952,29	**3**	**949,40**	**3**	**949,40**	**3**	**949,40**	**3**	**949,40**
4	**2**	**849,05**	**2**	**849,05**	**2**	**849,05**	**2**	**849,05**	**2**	**849,05**	**2**	**849,05**
5	**3**	**1.054,02**	**3**	**1.054,02**	**3**	**1.054,02**	**3**	**1.054,02**	**3**	**1.054,02**	**3**	**1.054,02**
6	**3**	**931,63**	**3**	**931,63**	**3**	**931,63**	**3**	**931,63**	**3**	**931,63**	**3**	**931,63**
7	**2**	**903,06**	**2**	**903,60**	**2**	**903,06**	**2**	**903,06**	**2**	**903,06**	**2**	**903,06**
8	**2**	**734,85**	**2**	**736,00**	**2**	**734,85**	**2**	**734,85**	**2**	**734,85**	2	736,00
9	3	937,05	3	932,43	**3**	**930,59**	**3**	**930,59**	**3**	**930,59**	3	937,05
10	**3**	**964,22**	**3**	**964,22**	**3**	**964,22**	**3**	**964,22**	**3**	**964,22**	**3**	**964,22**
11	2	927,80	3	888,15	2	913,84	3	896,76	**2**	**911,52**	3	896,76

Tabela 9: Resultados das Instâncias de 100 pares - LRC

	Li01		Pankratz05		Bent06		Derigs06		Ropke06		ILS-MRD	
	V	T	V	T	V	T	V	T	V	T	V	T
1	14	1.708,80	15	1.703,21	**14**	**1.708,80**	**14**	**1.708,80**	**14**	**1.708,80**	**14**	**1.708,80**
2	**13**	**1.563,55**	**12**	**1.558,07**	**12**	**1.558,07**	**12**	**1.558,07**	**12**	**1.558,07**	**12**	**1.558,07**
3	**11**	**1.258,74**	**11**	**1.258,74**	**11**	**1.258,74**	**11**	**1.258,74**	**11**	**1.258,74**	**11**	**1.258,74**
4	**10**	**1.128,40**	**10**	**1.128,40**	**10**	**1.128,40**	**10**	**1.128,40**	**10**	**1.128,40**	10	1.129,95
5	**13**	**1.637,62**	**13**	**1.637,62**	**13**	**1.637,62**	**13**	**1.637,62**	**13**	**1.637,62**	13	1.643,88
6	11	1.425,53	**11**	**1.424,73**	**11**	**1.424,73**	**11**	**1.424,73**	**11**	**1.424,73**	**11**	**1.424,73**
7	11	**1.230,14**	**11**	**1.230,14**	**11**	**1.230,14**	**11**	**1.230,14**	**11**	**1.230,14**	**11**	**1.230,14**
8	10	1.147,97	**10**	**1.147,43**	**10**	**1.147,43**	10	1.147,96	**10**	**1.147,43**	10	1.152,87
1	4	1.468,96	4	1.407,21	**4**	**1.406,94**	**4**	**1.406,94**	**4**	**1.406,94**	4	1.439,67
2	**3**	**1.374,27**	4	1.385,25	**3**	**1.374,27**	**3**	**1.374,27**	**3**	**1.374,27**	4	1.385,25
3	**3**	**1.089,07**	**4**	**1.093,89**	**3**	**1.089,07**	**3**	**1.089,07**	**3**	**1.089,07**	**3**	**1.089,07**
4	3	827,78	**3**	**818,66**	**3**	**818,66**	**3**	**818,66**	**3**	**818,66**	3	820,66
5	**4**	**1.302,20**	**4**	**1.302,20**	4	**1.302,20**	**4**	**1.302,20**	**4**	**1.302,20**	4	1.305,91
6	3	1.162,91	**3**	**1.159,03**	**3**	**1.159,03**	**3**	**1.159,03**	**3**	**1.159,03**	3	1.164,35
7	3	1.424,60	**3**	**1.062,05**	**3**	**1.062,05**	**3**	**1.062,05**	**3**	**1.062,05**	3	1.064,40
8	**3**	**852,76**	**3**	**852,76**	**3**	**852,76**	3	865,51	**3**	**852,76**	3	861,31

Tabela 10: Resultados das Instâncias de 200 pares - LC

	Bent06		Ropke06		ILS-MRD	
	V	T	V	T	V	T
1	**20**	**2.704,57**	**20**	**2.704,57**	**20**	**2.704,57**
2	**19**	**2.764,56**	**19**	**2.764,56**	**19**	**2.764,56**
3	17	3.134,08	**17**	**3.128,61**	18	2.773,91
4	**17**	**2.693,41**	**17**	**2.693,41**	18	2.673,57
5	**20**	**2.702,05**	**20**	**2.702,05**	**20**	**2.702,05**
6	**20**	**2.701,04**	**20**	**2.701,04**	**20**	**2.701,04**
7	**20**	**2.701,04**	**20**	**2.701,04**	**20**	**2.701,04**
8	**20**	**2.689,83**	**20**	**2.689,83**	**20**	**2.689,83**
9	**18**	**2.724,24**	**18**	**2.724,24**	**18**	**2.724,24**
10	**17**	**2.741,56**	17	2.943,49	17	2.943,49
1	**6**	**1.931,44**	**6**	**1.931,44**	**6**	**1.931,44**
2	**6**	**1.881,40**	**6**	**1.881,40**	**6**	**1.881,40**
3	**6**	**1.844,33**	**6**	**1.844,33**	**6**	**1.844,33**
4	6	1.778,54	**6**	**1.767,12**	6	1.778,54
5	**6**	**1.891,21**	**6**	**1.891,21**	**6**	**1.891,21**
6	**6**	**1.857,78**	**6**	**1.857,78**	6	1.858,25
7	**6**	**1.850,13**	**6**	**1.850,13**	**6**	**1.850,13**
8	**6**	**1.824,34**	**6**	**1.824,34**	**6**	**1.824,34**
9	**6**	**1.854,21**	**6**	**1.854,21**	**6**	**1.854,21**
10	**6**	**1.817,45**	**6**	**1.817,45**	**6**	**1.817,45**

Tabela 11: Resultados das Instâncias de 200 pares - LR

	Bent06		Ropke06		ILS-MRD	
	V	T	V	T	V	T
1	**20**	**4.819,12**	**20**	**4.819,12**	**20**	**4.819,12**
2	17	4.666,09	**17**	**4.621,21**	19	4.205,50
3	15	3.657,19	**15**	**3.612,64**	15	3.657,19
4	10	3.146,06	**10**	**3.037,38**	10	3.089,89
5	**16**	**4.760,18**	**16**	**4.760,18**	**16**	**4.760,18**
6	**14**	**4.178,24**	**14**	**4.178,24**	14	4.201,82
7	12	3.851,36	**12**	**3.550,61**	12	3.851,36
8	9	2.784,53	**9**	**2.784,53**	9	2.828,09
9	14	4.411,54	**14**	**4.354,66**	14	4.411,54
10	11	3.744,95	**11**	**3.714,16**	11	3.744,95
1	**5**	**4.073,10**	**5**	**4.073,10**	**5**	**4.073,10**
2	**4**	**3.796,00**	**4**	**3.796,00**	**4**	**3.796,00**
3	4	3.100,38	**4**	**3.098,36**	**4**	**3.098,36**
4	3	2.956,15	**3**	**2.486,14**	3	2.737,22
5	**4**	**3.438,39**	**4**	**3.438,39**	**4**	**3.438,39**
6	4	3.208,53	**4**	**3.201,54**	4	3.208,53
7	3	3.337,28	**3**	**3.135,05**	3	3.337,28
8	3	2.407,66	**2**	**2.555,40**	3	2.736,35
9	4	3.198,44	**3**	**3.930,49**	4	3.198,44
10	3	3.478,67	**3**	**3.344,08**	3	3.478,67

Tabela 12: Resultados das Instâncias de 200 pares - LRC

	Bent06		Ropke06		ILS-MRD	
	V	T	V	T	V	T
1	**19**	**3.606,06**	**19**	**3.606,06**	**19**	**3.606,06**
2	16	3.681,36	**15**	**3.674,80**	16	3.681,36
3	**13**	**3.161,75**	13	3.178,17	13	3.251,38
4	10	2.655,27	**10**	**2.631,82**	10	2.655,27
5	**16**	**3.715,81**	**16**	**3.715,81**	**16**	**3.715,81**
6	**17**	**3.368,66**	**17**	**3.368,66**	**17**	**3.368,66**
7	15	3.417,16	**14**	**3.668,39**	15	3.417,16
8	14	3.087,62	**13**	**3.174,55**	14	3.087,62
9	14	3.129,65	**13**	**3.226,72**	**13**	**3.226,72**
10	13	2.833,85	**12**	**2.951,29**	13	2.833,85
1	6	3.690,10	**6**	**3.605,40**	6	3.690,10
2	6	2.666,01	**5**	**3.327,18**	6	2.666,01
3	5	2.523,59	**4**	**2.938,28**	4	3.249,36
4	4	2.795,70	**3**	**2.887,97**	4	2.795,70
5	**5**	**2.776,93**	**5**	**2.776,93**	**5**	**2.776,93**
6	**5**	**2.707,96**	**5**	**2.707,96**	**5**	**2.707,96**
7	**4**	**3.056,09**	**4**	**3.056,09**	5	2.816,41
8	4	3.050,03	**4**	**2.399,95**	4	3.050,03
9	4	2.750,30	**4**	**2.208,49**	4	2.750,30
10	3	2.699,55	**3**	**2.550,56**	3	2.699,55

4 Branch-and-Cut para o Problema do Caixeiro Viajante com Coleta e Entrega

O problema do caixeiro viajante é um dos mais clássicos problemas da área de otimização. Porém esta variante com coleta e entrega vem sendo estudada há menos tempo (veja Seção 2.4). Este capítulo pretende comparar diferentes formulações matemáticas. A formulação não-orientada (veja Seção 4.1), sobre as arestas do grafo, foi proposta inicialmente por Ruland e Rodin (1997) e aprofundada por Dumitrescu et al. (2008). Neste trabalho são propostas uma formulação orientada (veja Seção 4.2), sobre os arcos de um grafo orientado, e também uma formulação estendida, sobre variáveis de arco e posição.

4.1 Formulação Não Orientada

Dumitrescu et al. (2008) fizeram um estudo sobre a estrutura poliedral do TSPPD e aplicaram as desigualdades encontradas em um algoritmo de branch-and-cut. Esse estudo foi feito sobre a formulação não orientada proposta por Ruland e Rodin (1997) e que será apresentada a seguir.

Seja $G = (V, E)$, um grafo completo não orientado, onde V é o conjunto de vértices e E o conjunto de arestas. O conjunto V é constituído pelos vértices de coleta e entrega, além de dois vértices que correspondem ao depósito, a saída e chegada a ele. Um par de vértices de coleta e entrega formam uma requisição. A quantidade de requisições é indicada por n. Seja $P = \{1, \ldots, n\}$ o conjunto de vértices de coleta e $D = \{n+1, \ldots, 2n\}$ o conjunto dos vértices de entrega. O vértice de entrega correspondente ao vértice de coleta $i \in P$ é $i + n \in D$. Desta forma, $V = P \cup D \cup \{0, 2n + 1\}$, onde 0 corresponde a partida do depósito e $2n+1$ a volta a ele. Para dois vértices i e j, $i < j$, a aresta associada a esse par é denotada como (i, j). Um custo não-negativo $c_{i,j}$ é associado com a aresta $(i, j) \in E$. O TSPPD consiste em encontrar um caminho hamiltoniano sobre G que deve começar em $\{0\}$ e terminar em $\{2n + 1\}$, passando por uma coleta i antes de sua entrega $i + n$.

Juntando-se a esse caminho a aresta $(0, 2n+1)$, que não possui custo, temos um circuito hamiltoniano sobre G.

Em adição a esta notação, define-se $\delta(S) = \{(i,j) \in E : i \in S, j \notin S \text{ ou } i \notin S, j \in S\}$ para todo conjunto $S \subseteq V$. Se $S = \{i\}$, escreve-se $\delta(i)$ em vez de $\delta(\{i\})$. O TSPPD foi formulado primeiramente como programação linear inteira por Ruland (1995), associando a variável binária x_{ij} à aresta $(i,j) \in E$. Usamos a notação $x(E')$ para $\sum_{(i,j)\in E'} x_{ij}$, onde $E' \subseteq E$:

$$\text{Minimize: } \sum_{(i,j)\in E} c_{ij}x_{ij} \tag{4.1}$$

sujeito a:

$$x_{0,2n+1} = 1 \tag{4.2}$$

$$x(\delta(i)) = 2 \qquad \forall i \in V \tag{4.3}$$

$$x(\delta(S)) \geq 2 \qquad \forall S \subseteq V, 3 \leq |S| \leq |V|/2 \tag{4.4}$$

$$x(\delta(S)) \geq 4 \qquad \forall S \in \mathcal{U} \tag{4.5}$$

$$x_{ij} \in \{0,1\} \qquad \forall (i,j) \in E \tag{4.6}$$

onde $\mathcal{U}$ é uma coleção de subconjuntos de $S \subset V$ que satisfaz $S \subseteq V, 3 \leq |S| \leq |V|/2$ com $0 \in S, 2n+1 \notin S$ e que exista $i \in P$ tal que, $i \notin S$ e $i+n \in S$. O objetivo é minimizar a soma dos custos das arestas na solução, restrição (4.1). A aresta $(0, 2n+1)$ tem seu valor fixado em 1 como mostra a restrição (4.2). A restrição (4.3) obriga a todo vértice a ter grau 2. A restrição (4.4) é a clássica restrição de eliminação de sub ciclos. Finalmente, a restrição (4.5) obriga um vértice i a ser visitado antes do vértice $i+n$ para todo $i \in P$. Conforme ilustrado na Figura 9, qualquer solução viável para o TSPPD cruza um conjunto $S \in \mathcal{U}$ pelo menos 4 vezes (em particular, a aresta $(0, 2n+1)$ sempre cruza esses conjuntos). As restrições (4.5) podem ser separadas em tempo polinomial usando o algoritmo do corte mínimo.

4.1.1 Desigualdades válidas

Para explicar uma das desigualdades válidas usadas como cortes por Dumitrescu et al. (2008) apresentamos mais algumas notações:

Para o conjunto de vértices $S \subseteq V$, $\pi(S)$ refere-se ao conjunto dos vértices de coleta de S, ou seja, $\pi(S) = \{i \in P : n+i \in S\}$. Similarmente $\sigma(S)$ é o conjunto dos vértices

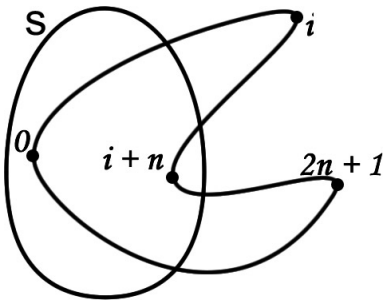


Figura 9: Restrição (4.5)

de entrega de S, $\sigma(S) = \{n + i \in D : i \in S\}$. Então, para $S \subseteq V$, define-se $\overline{S} = V \backslash S$ e $E(S) = \{(i, j) \in E : i \in S, j \in S\}$. Será escrito $x(S)$ em vez de $x(E(S))$.

4.1.1.1 GOC - Generalized Order Constraints

Sejam $S_1, \ldots, S_m \subset P \cup D$ conjuntos mutuamente disjuntos, tal que, $m \geq 2$, $S_i \cap \pi(S_{i+1}) \neq \emptyset, \forall i = 1, \ldots, m$, onde $S_{m+1} = S_1$. Então a inequação

$$\sum_{i=1}^{m} x(S_i) \leq \sum_{i=1}^{m} |S_i| - m - 1 \tag{4.7}$$

é válida.

Exemplo: Considere o subconjunto $S_1 = \{i, n + k\}$, $S_2 = \{j, n + i\}$, $S_3 = \{k, n + j\}$. Claramente $S_i \cap \pi(S_{i+1}) \neq \emptyset, \forall i = 1, 2, 3$. O GOC para esse conjunto é: $x_{i,n+k} + x_{j,n+i} + x_{k,n+j} \leq 2$. Como mostra a Figura 10.

Uma classe semelhante de desigualdades foi proposta por Balas, Fischetti e Pulleyblank (1995) para o *precedence-constrained asymmetric traveling salesman problem* (PCATSP). Foi mostrado em (DUMITRESCU et al., 2008) que em geral as GOCs não definem facetas do poliedro TSPPD. Apesar disso elas são cortes muito efetivos no algoritmo de branch-and-cut. A separação das GOCs é feita de forma exata (através do algoritmo de corte mínimo) para o caso $m = 2$ e de forma heurística para os casos em que $m > 2$. Não existe algoritmo de separação polinomial conhecido para m qualquer.

4.1.1.2 Outras Desigualdades

Várias outras famílias de desigualdade válidas são apresentadas em Dumitrescu et al. (2008) e utilizadas como cortes no algoritmo de branch-and-cut. Elas são as OMC (Order Matching Constraints), DGOMCs (Doubly Generalized Order Matching Constraints), PC

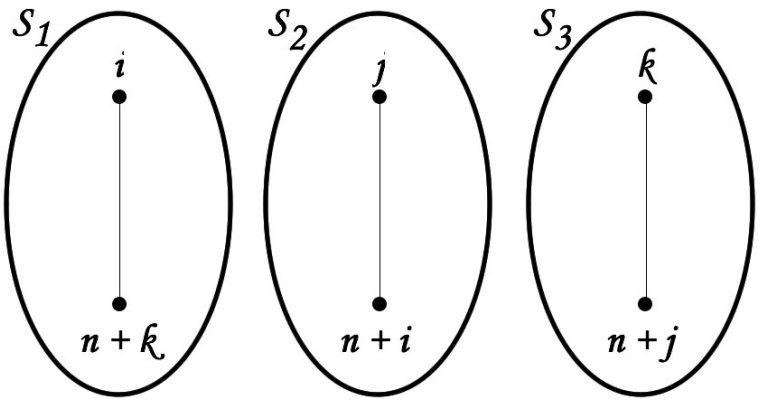


Figura 10: GOC com m = 3

(Precedence Constraints), LSEC (Lifted subtour elimination constraints), GLSEC (Generalized lifted subtour elimination constraints), DC (Depot constraints), StEnC (Start-End Constraints), entre outras.

É interessante notar que, apesar de várias dessas desigualdades definirem facetas do TSPPD, o efeito conjunto de todos esses cortes em termos de redução do gap de integralidade é menor do que apenas as GOCs.

4.2 Formulação Orientada

4.2.1 Modelo Matemático

Nesta sessão apresenta-se uma nova formulação linear inteira para o TSPPD sobre um grafo orientado completo $G = (V, A)$. O depósito será denotado apenas por 0 (não há duplicação do depósito). Temos $V = \{0, 1, \ldots, 2n\}$ como sendo o conjunto de vértices. Novamente as coletas são dadas por $P = \{1, \ldots, n\}$ e as entregas por $D = \{n+1, \ldots, 2n\}$. O arco $a \in A$ com origem em i e destino em j é denotado por (i, j). Seja c_a o custo do arco a, esse custo é simétrico ($c_{ij} = c_{ji}$) e é o mesmo custo da aresta (i, j) na formulação não orientada. Para cada conjunto $S \subset V$, temos $\delta^+(S) = \{(i, j) \in A : i \in S, j \notin S\}$, $\delta^-(S) = \{(i, j) \in A : i \notin S, j \in S\}$, $V^+ = V \setminus \{0\}$, $V^i = V \setminus \{i\}$, $V^{i+n} = V \setminus \{i + n\}$. $A(S_1 : S_2)$ representa o conjunto de arcos $(i, j) \in A$ onde $i \in S_1$ e $j \in S_2$. O conjunto de arcos A não precisa conter os arcos do depósito para a entrega $(0 : D)$ nem os arcos de coleta para o depósito $(P : 0)$, pois nunca uma entrega será o primeiro vértice a ser visitado ao sair do depósito, nem há a possibilidade de partir de uma coleta e ir diretamente ao depósito.

O modelo matemático para TSPPD tem-se:

Variável de decisão:

$$\boldsymbol{x_a} = \begin{cases} 1 & \text{se o arco pertence a rota} \\ 0 & \text{caso contrário} \end{cases}$$

Formulação:

$$\text{Minimize: } \sum_{(i,j)\in A} c_a x_a \tag{4.8}$$

$$\sum_{a\in\delta^-(i)} x_a = 1 \qquad \forall i \in V \tag{4.9}$$

$$\sum_{a\in\delta^+(i)} x_a = 1 \qquad \forall i \in V \tag{4.10}$$

$$\sum_{a\in\delta^+(S)\setminus A(S:\{0\})} x_a \geq 1 \qquad \forall S \subset V^+ | i \in S \text{ e } i+n \notin S \tag{4.11}$$

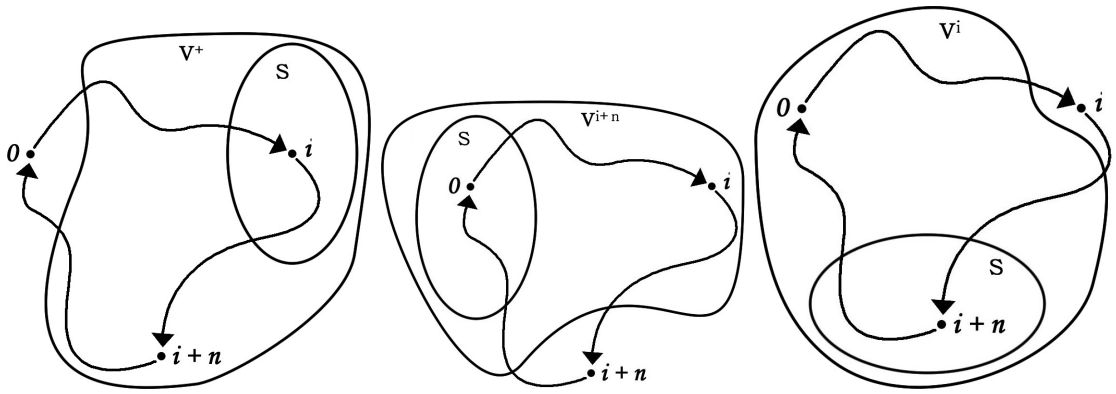
$$\sum_{a\in\delta^+(S)\setminus A(S:\{i+n\})} x_a \geq 1 \qquad \forall S \subset V^{i+n} | 0 \in S \text{ e } i \notin S \tag{4.12}$$

$$\sum_{a\in\delta^+(S)\setminus A(S:\{i\})} x_a \geq 1 \qquad \forall S \subset V^{i} | i+n \in S \text{ e } 0 \notin S \tag{4.13}$$

$$x_a \in \{0,1\} \qquad \forall a \in A \tag{4.14}$$

A função objetivo (4.8) é minimizar a soma dos custos dos arcos. As restrições (4.9) e (4.10) dizem que exatamente um arco entra e um arco sai de cada vértice i. As três próximas restrições servem tanto para eliminar sub ciclos para garantir a seqüência correta de coletas e entregas. A restrição (4.11) indica que deve haver um caminho orientado de i para $i+n$ sem passar pelo depósito (veja Figura 11(a)). A restrição (4.12) indica que deve haver um caminho orientado de 0 até i sem passar por $i+n$ (veja Figura 11(b)). Finalmente, a restrição (4.13) indica que deve existir um caminho orientado de $i+n$ até 0 sem passar por i (veja Figura 11(c)).

Em um algoritmo de branch-and-cut sobre essa formulação, cria-se o problema apenas com as restrições (4.9) e (4.10). As demais restrições vão sendo separadas ao longo do algoritmo. Como essas restrições são essenciais para a viabilidade da solução, elas devem ser separadas em todos os nós da árvore de enumeração. Essa separação é feita através do algoritmo do fluxo máximo - corte mínimo nos 3 casos, conforme mostrado a seguir:



(a) Coleta para entrega (b) Depósito para coleta (c) Entrega para depósito

Figura 11: Desigualdades da formulação orientada

Coleta para entrega : Cria-se um grafo contendo apenas os arcos com valor positivo na solução do problema linear do corrente. A capacidade desses arcos é definida como sendo igual a esse valor. Elimina-se os arcos adjacentes ao depósito. Então aplica-se o algoritmo de fluxo máximo nesse grafo, tendo como origem i e destino $i + n$, para cada $i \in P$. Caso o valor do fluxo seja menor que 1 em algum desses casos, existe um corte violado.

Depósito para coleta : Os grafos são criados de forma similar, porém para cada $i \in P$, elimina-se os arcos adjacentes ao vértice $i+n$. Aplica-se o algoritmo de fluxo máximo em cada um desses grafos, tendo como origem 0 e destino i. Caso o valor do fluxo seja menor que 1, significa que temos um corte violado.

Entrega para o depósito : Procede-se de forma similar ao item anterior, cada grafo elimina os arcos adjacentes ao vértice i. O algoritmo de fluxo máximo é aplicado tendo como origem $i + n$ e como destino 0.

4.2.2 Desigualdade Fortalecida e Separação por Programação Inteira

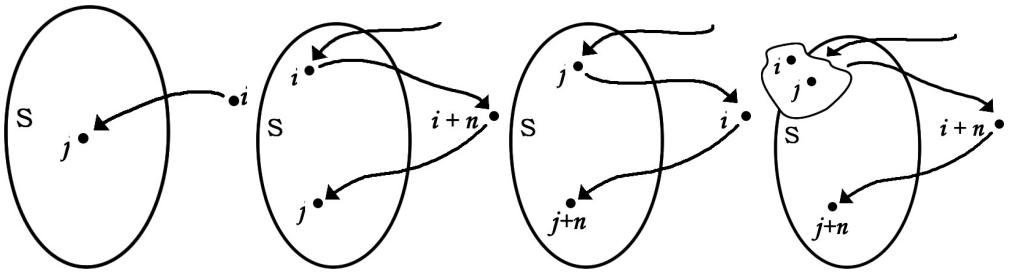
Seja S um subconjunto de V^+. A formulação orientada já implica que em qualquer solução pelo menos um arco deverá entrar em S. Entretanto, analisando-se com mais cuidado quais vértices pertencem a S, é possível obter uma nova desigualdade que domina (4.11):

$$\sum_{a \in \mathfrak{S}} x_a \geq 1 \quad \forall S \subseteq V^+ \tag{4.15}$$

onde, $\mathfrak{S}$ é o subconjunto dos arcos de $\delta^-(S)$ definido da seguinte forma.

1. Se $(i,j) \in \delta^-(S)$, $i,j \in P$, então $(i,j) \in \mathfrak{S}$.
2. Se $(i+n,j) \in \delta^-(S)$, $i+n \in D$, $j \in P$. Caso $i \notin S$ então $(i+n,j) \in \mathfrak{S}$, caso contrário $(i+n,j) \notin \mathfrak{S}$.
3. Se $(i,j+n) \in \delta^-(S)$, $i \in P$, $j+n \in D$. Caso $j \notin S$ então $(i,j+n) \in \mathfrak{S}$, caso contrário $(i,j+n) \notin \mathfrak{S}$.
4. Se $(i+n,j+n) \in \delta^-(S)$, $i+n, j+n \in D$. Caso i ou $j \notin S$ então $(i+n,j+n) \in \mathfrak{S}$, caso contrário $(i+n,j+n) \notin \mathfrak{S}$.

A condição 1 é ilustrada na Figura 12(a), nessa condição (i,j) pode ser o único arco entrando em S, logo ele deve pertencer a $\mathfrak{S}$. Por outro lado, um arco $(i+n,j)$ na condição da Figura 12(b) só pode estar em uma rota, se esta rota já entrou anteriormente em S antes de passar pelo vértice i. Logo esse arco não pertence a $\mathfrak{S}$. Similarmente na condição da Figura 12(c), o arco $(i,j+n)$ entrou em S depois da rota já ter entrado anteriormente em S, passando por j, assim o arco não pertence a $\mathfrak{S}$. Finalmente, na condição da Figura 12(d), um arco $(i+n,j+n)$ só pode estar na rota, se esta rota já entrou em S antes, passando por i e/ou j.



(a) Restrição (4.17) (b) Restrição (4.18) (c) Restrição (4.19) (d) Restrição (4.20)

Figura 12: Desigualdades do Corte - Formulação Inteira

Ao contrário da desigualdades (4.11)-(4.13), que podem ser facilmente separadas pelo algoritmo do corte mínimo, não parece ser possível separar (4.15) em tempo polinomial. Entretanto, é possível separar esse corte utilizando-se a seguinte formulação de programação inteira. Seja $G = (V, A^*)$ o grafo onde $V = \{0, 1, \ldots, 2n\}$ e A^* é o conjunto de arcos com valor positivo na solução do problema linear corrente:

Dados:

$\overline{x_{i,j}}$ valor fracionário da variável associada ao arco (i,j)

Variáveis:

$$y_i = \begin{cases} 1 & \text{se o vértice } i \in S \\ 0 & \text{caso contrário} \end{cases}$$

$$z_{i,j} = \begin{cases} 1 & \text{se o arco } (i,j) \in \mathfrak{S} \\ 0 & \text{caso contrário} \end{cases}$$

Formulação:

$$\text{Minimize: } \sum_{(i,j)\in A} \overline{x_{i,j}} z_{i,j} \tag{4.16}$$

Sujeito a:

$$z_{i,j} \geq y_j - y_i \qquad \forall (i,j) \in A : i \leq n \text{ e } j \leq n \tag{4.17}$$

$$z_{i+n,j} \geq y_j - y_{i+n} - y_i \qquad \forall (i,j) \in A : i \leq n \text{ e } j \leq n \tag{4.18}$$

$$z_{i,j+n} \geq y_{j+n} - y_i - y_j \qquad \forall (i,j) \in A : i \leq n \text{ e } j \leq n \tag{4.19}$$

$$z_{i+n,j+n} \geq y_{j+n} - y_{i+n} - y_i - y_j \qquad \forall (i,j) \in A : i \leq n \text{ e } j \leq n \tag{4.20}$$

$$\sum_{i=1}^{2n} y_i \geq 2 \tag{4.21}$$

$$y_0 = 0 \tag{4.22}$$

O objetivo, (4.16), dessa formulação é minimizar o valor fracionário dos arcos em $\mathfrak{S}$. As restrições (4.17),(4.18),(4.19) e (4.20) correspondem as quatro condições para que um arco pertença a $\mathfrak{S}$. A restrição (4.21) é colocada porque a restrição (4.15) só pode estar violada para conjuntos S de tamanho pelo menos 2, enquanto (4.22) indica que o vértice 0 não deve pertencer a S.

4.3 Formulação Estendida

Seja $N = \{1, \ldots, 2n\}$ e $N_0 = N \cup \{0\}$. Para o conjunto de vértices S, $K(S)$ é o dígrafo completo de S, ou seja, o conjunto de arcos $\{(u,v) : u, v \in S, u \neq v\}$. Existe uma correspondência de um-para-um entre os ciclos hamiltonianos de $K(N_0)$ e os caminhos hamiltonianos de $K(N)$.

O TSPPD no grafo completo $K(N_0)$ pode ser modelado como um problema de programação inteira definido no grafo de camadas $G' = (V, A)$, onde V consiste no vértice 0, uma cópia desse vértice T, e nós intermediários (i,k) para $i, k \in N$. O primeiro índice

corresponde ao identificador do vértice i do grafo $K(N)$ enquanto o segundo refere-se ao nível do caminho hamiltoniano. O conjunto de arcos A é composto por três tipos de arcos. Primeiro os que partem do depósito 0 para os vértices $(i, 1)$, os quais são denotados por $(0, i, 0)$ para $i \in N$. O segundo tipo, $(i, T, 2n)$, são os que chegam ao depósito, partindo dos vértices do último nível $(i, 2n)$ com destino a T. Por fim, para $i, j \in N$ tal que $i \neq j$ e $1 \leq k \leq 2n - 1$, serão denotados como (i, j, k) os arcos que partem do vértice (i, k) até o $(j, k + 1)$. Assim, o grafo G', terá $(2n + 1)$ níveis.

Para melhor explicar a formulação serão definidos os seguintes conjuntos: $N(j) = N \setminus \{j\}$; $V_i = \{(i, 1), (i, 2), \ldots, (i, 2n)\}$; $V^+ = V \setminus \{0, T\}$; $V^i = V \setminus V_i$ é o conjunto de vértices que não contém o vértice i em nenhum dos níveis k e $V^{i+n} = V \setminus V_{i+n}$.

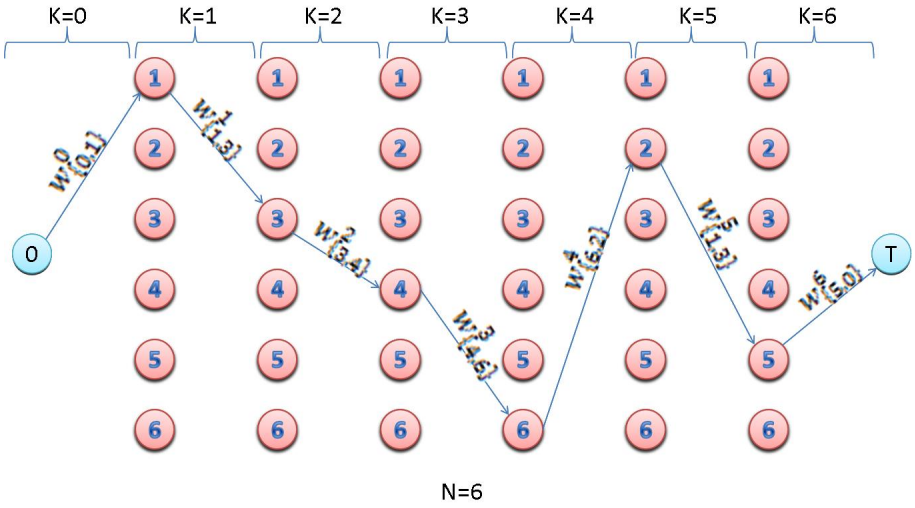


Figura 13: Solução para uma instância com 3 pares de clientes representada como um caminho em G'

Como exemplo, a Figura 13 representa uma solução para uma instância de seis vértices (3 coletas, vértices de 1 a 3, e 3 entregas, vértices de 4 a 6) sobre o grafo G'. A formulação estendida é apresentada a seguir.

Dados:

n é a quantidade de pares (coleta-entrega)

$c_{i,j}$ o custo da aresta (i, j)

Variáveis:

$$w_{i,j}^k \begin{cases} 1 & \text{se o arco } (i, j) \text{ pertence ao caminho no nível k} \\ 0 & \text{caso contrário} \end{cases}$$

- $\begin{cases} 1 & \text{se o arco } (i,j) \text{ pertence ao caminho em qualquer nível} \\ 0 & \text{caso contrário} \end{cases}$

Formulação:

$$\text{Minimize} \sum_{j \in N} c_{0,j} w^0_{0,j} + \sum_{k=1}^{2n-1} \sum_{i \in N} \sum_{j \in N(i)} c_{i,j} \times w^k_{i,j} + \sum_{i \in N} c_{i,T} w^{2n}_{i,T} \tag{4.23}$$

$$\sum_{j \in N} w^0_{0,j} = 1 \tag{4.24}$$

$$w^0_{0,j} = \sum_{i \in N(j)} w^1_{j,i} \qquad \forall j = 1 \ldots 2n \tag{4.25}$$

$$\sum_{i \in N(j)} w^k_{i,j} = \sum_{l \in N(j)} w^{k+1}_{j,l} \qquad \forall j = 1 \ldots 2n \text{ e } \forall k = 1 \ldots 2n-2 \tag{4.26}$$

$$x_{0,i} = w^0_{0,i} \qquad \forall i = 1 \ldots 2n \tag{4.27}$$

$$x_{i,j} = \sum_{k \in N} w^k_{i,j} \qquad \forall i = 1 \ldots 2n, \ \forall j = 1 \ldots 2n \tag{4.28}$$

$$x_{i,0} = w^{2n}_{i,T} \qquad \forall i = 1 \ldots 2n \tag{4.29}$$

$$\sum_{j \in N_0(j)} x_{i,j} = 1 \qquad \forall i = 1 \ldots 2n \tag{4.30}$$

$$\sum_{(i,j,k) \in \delta^+(S) \setminus A(S:\{0\})} w^k_{i,j} \geq 1 \qquad \forall S \subset V^+ | S \supset V_{i1} \text{ e } S \cap V_{i1+n} = \emptyset \tag{4.31}$$

$$\sum_{(i,j,k) \in \delta^+(S) \setminus A(S:\{i1+n\})} w^k_{i,j} \geq 1 \qquad \forall S \subset V^{i1+n} | S \supset \{0\} \text{ e } S \cap V_{i1} = \emptyset \tag{4.32}$$

$$\sum_{(i,j,k) \in \delta^+(S) \setminus A(S:\{i1\})} w^k_{i,j} \geq 1 \qquad \forall S \subset V^{i1} | S \supset V_{i1+n} \text{ e } S \cap V_T = \emptyset \tag{4.33}$$

O objetivo dessa formulação é minimizar a soma dos custos das arestas utilizadas, dada por (4.23). A formulação tem 3 partes com foi explicado anteriormente. A restrição

(4.24) obriga a ter fluxo partindo do depósito com valor 1, como a variável $w^k_{i,j}$ é binária, será escolhida somente um arco. A restrição (4.25) representa a saída do fluxo, a partir do deposito, no nível 0 com valor unitário o este fluxo precisa chegar, novamente ao depósito, no nível $2n$. Analogamente na restrição (4.26) constrói-se o fluxo que passa pelos níveis intermediários e final. As restrições (4.27-4.29) correspondem ao mapeamento das variáveis w nas variáveis x. A restrição (4.30) obriga o fluxo a passar exatamente uma vez em cada vértice (em algum nível). As próximas restrições (4.31-4.33) são semelhantes às explicadas na seção 4.2, aplicadas à variável x (restrições (4.11-4.13), respectivamente) modificadas para a variável w.

Em um algoritmo de branch-and-cut sobre essa formulação, cria-se o problema apenas com as restrições (4.24-4.30). Enquanto as demais serão criadas ao longo do algoritmo. Como essas restrições são essenciais para a viabilidade do problema, estas devem ser separadas em todos os nós da árvore de enumeração. As restrições aplicadas a variável x, descritas na seção anterior, continuam valendo para esta formulação e serão aplicadas da mesma forma e como a variável w tem relação com x também podem-se aplicar as separações com algumas mudanças, que serão descritas a seguir:

Coleta para entrega (em w) Cria-se um grafo G^1 a partir de G', porém sem conter os arcos adjacentes ao depósito 0 e T, para $i \leq n$. Deve-se calcular, usando o algoritmo de fluxo máximo/corte mínimo, o fluxo entre os pontos V_i e V_{i+n}. Para tanto, é necessário adicionar dois super vértices s e t, além disso criar arcos com capacidade infinita $A(s : V_i)$ e $A(V_{i+n} : t)$. Os demais arcos tem como capacidade o seu valor na solução fracionária corrente. Calcula-se o corte mínimo entre s e t. Se o valor desse corte for menor do que 1, uma desigualdade (4.31) está violada. O grafo G^1 é ilustrado na Figura 14.

Depósito para coleta (em w) Cria-se um grafo G^2 a partir de G', que não contenha os vértices adjacentes aos vértices V_{i+n}, para $i \leq n$. Deve-se calcular, usando o algoritmo de fluxo máximo/corte mínimo, o fluxo entre os pontos 0 e V_i. Para tanto, é necessário adicionar um super vértice t, além disso criar arcos $A(V_i : t)$ com capacidade infinita. O depósito será o ponto de partida. Os demais arcos tem como capacidade o seu valor na solução fracionária corrente. Calcula-se o corte mínimo entre 0 e t. Se o valor desse corte for menor que 1, uma desigualdade (4.32) está violada. O grafo G^2 é ilustrado na Figura 15.

Entrega para depósito (em w) Cria-se um grafo G^3 a partir de G', que não contenha os vértices V_i, para $i \leq n$. Deve-se calcular, usando o algoritmo de fluxo má-

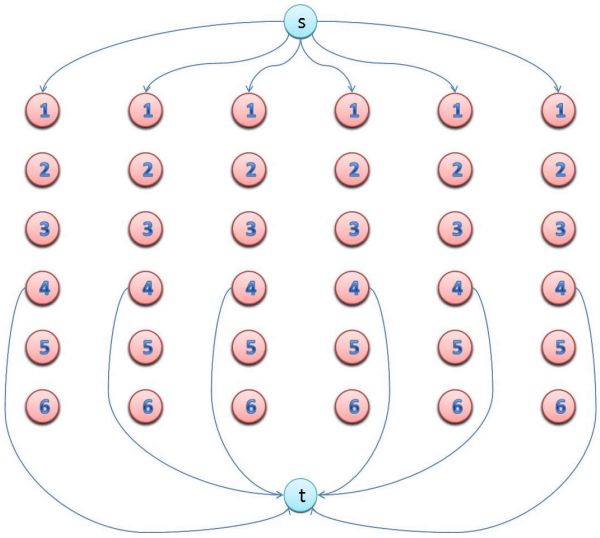


Figura 14: Grafo G^1 - Coleta para entrega (em w)

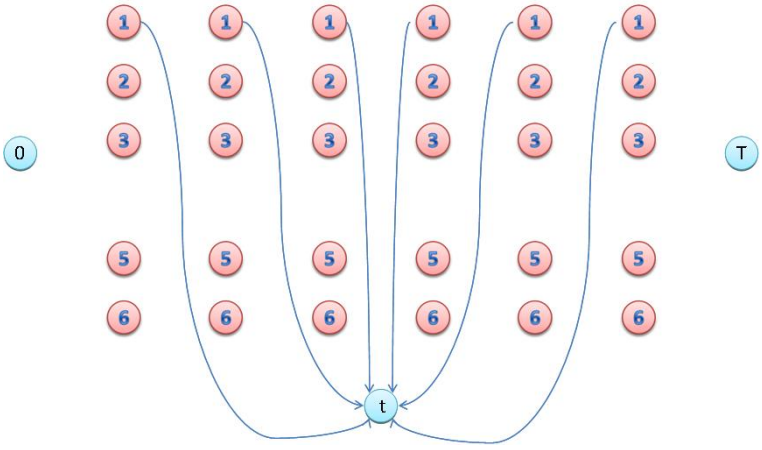


Figura 15: Grafo G^2 - Depósito para coleta (em w)

ximo/corte mínimo, o fluxo deve ser entre os pontos V_{i+n} e T. Para isso, é necessário adicionar um supervértice s, que representará o ponto de partida, além disso criar arcos $A(s : V_{i+n})$ com capacidade infinita. Os demais arcos tem como capacidade o seu valo na solução fracionária corrente. Calcula-se o corte mínimo entre s e T. Se o valor desse corte for menor que 1, uma desigualdade (4.33) está violada. O grafo G^3 é ilustrado na Figura 16

4.3.1 Corte do depósito à duas coletas

Ao analisar uma solução fracionária, após a adição de todos os cortes acima relatados, verificou-se que ainda poderia ser criado mais um corte. Será usado um exemplo para

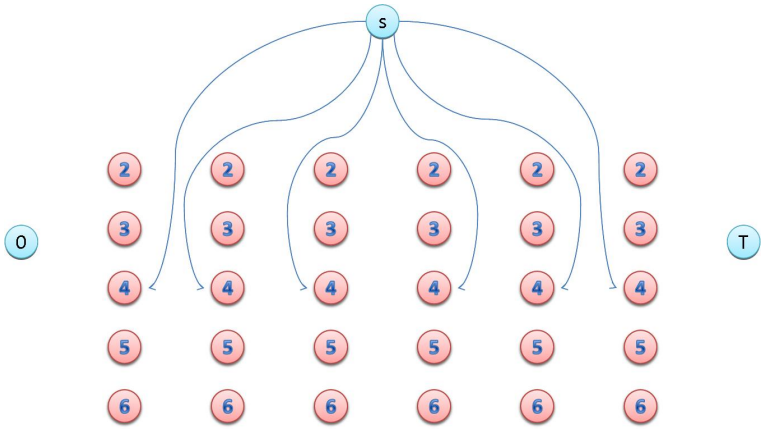


Figura 16: Grafo G^3 - Entrega para depósito (em w)

explicar esse corte. Na Figura 17 mostra-se a solução fracionária da instância prob10a na formulação estendida.

Essa solução fracionária pode ser decomposta em duas rotas fracionárias (com valor 0,5), representadas esquematicamente na Figura 17. Pode-se notar que essa decomposição não é única, outras três decomposições seriam possíveis dependendo de como se escolhe a seqüência das rotas a partir dos pontos de bifurcação nos vértices 4 e 14.

Pode-se notar que esta solução fracionária satisfaz todas as restrições até agora explicadas. Porém ao aplicar a restrição (4.32) a dois pares de coleta ao mesmo tempo, encontra-se mais uma desigualdade (4.32) violada. O Exemplo abaixo removeu os vértices de entrega (15 e 17) dos vértices 5 e 7.

O fluxo máximo (F) encontrado entre 0 e 5 e entre 0 e 7 tem capacidade de $F = 0,5$. Isto é facilmente observado na Figura 19, pois o segundo 7 da $Rota1$ e o primeiro 5 da $Rota2$ não tem mais nenhuma ligação com depósito 0.

O mesmo não ocorre se for eliminado somente um desses arcos, como é mostrado nas Figuras 20 e 21.

4.3.2 Formulação por Fluxos Equivalente

Outra formulação estendida para o problema do TSPPD é a que usa fluxos ao invés de restrições de corte para garantir a existência de certos caminhos no grafo estendido. Esta formulação é equivalente à formulação estendida já apresentada. A diferença é que a formulação por fluxos possui uma quantidade polinomial de variáveis e restrições, enquanto a anterior possui uma quantidade exponencial de restrições. Para substituir as

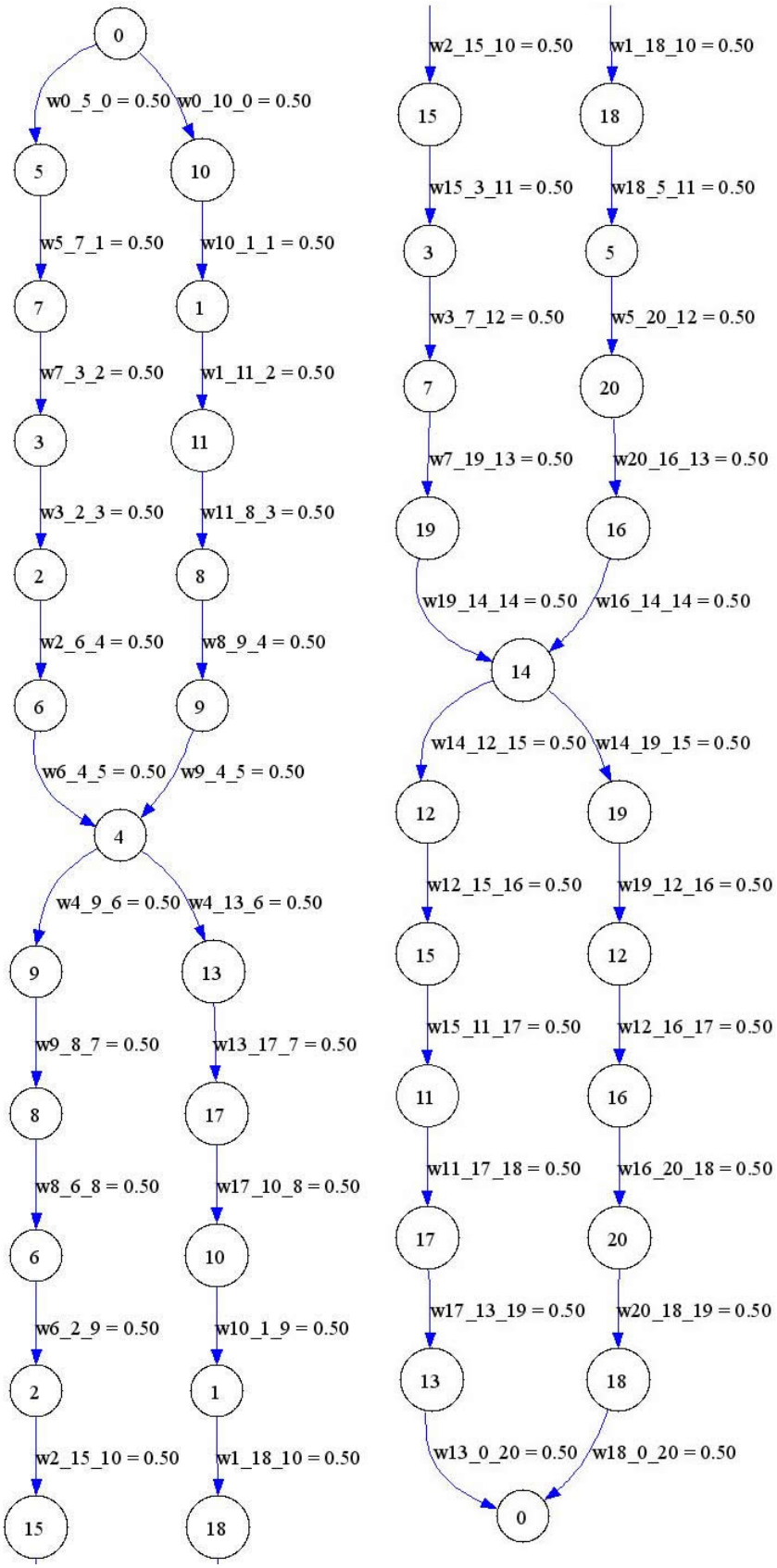


Figura 17: Solução Fracionária da Instância prob10a

```
Rota 1: 0- 5- 7- 3- 2- 6- 4- 9- 8- 6- 2-15- 3- 7-19-14-12-15-11-17-13-T
Rota 2: 0-10- 1-11- 8- 9- 4-13-17-10- 1-18- 5-20-16-14-19-12-16-20-18-T
```

Figura 18: Decomposição em rotas da solução fracionária da instância prob10a

```
Rota 1: 0- 5- 7- 3- 2- 6- 4- 9- 8- 6- 2-  - 3- 7-19-14-12-  -11-  -13-T
Rota 2: 0-10- 1-11- 8- 9- 4-13-  -10- 1-18- 5-20-16-14-19-12-16-20-18-T
```

Figura 19: Retirando as entregas 15 e 17, (restrição (4.32) com $F = 0,5$)

```
Rota 1: 0- 5- 7- 3- 2- 6- 4- 9- 8- 6- 2-  - 3- 7-19-14-12-  -11-17-13-T
Rota 2: 0-10- 1-11- 8- 9- 4-13-17-10- 1-18- 5-20-16-14-19-12-16-20-18-T
```

Figura 20: Retirando a entrega 15 (restrição (4.32) com $F = 1$)

```
Rota 1: 0- 5- 7- 3- 2- 6- 4- 9- 8- 6- 2-15- 3- 7-19-14-12-15-11-  -13-T
Rota 2: 0-10- 1-11- 8- 9- 4-13-  -10- 1-18- 5-20-16-14-19-12-16-20-18-T
```

Figura 21: Retirando a entrega 17 (restrição (4.32) com $F = 1$)

restrições (4.31), para cada d, $1 \leq d \leq n$, definimos que deverá existir no grafo estendido um fluxo entre os vértices V_d e V_{i+d}. De modo similar, para cada d, $1 \leq d \leq n$, definimos no grafo estendido um fluxo entre 0 e V_d que não passe por V_{i+d}. A formulação é descrita a seguir:

Dados:

n é a quantidade de pares (coleta-entrega)

$c_{i,j}$ o custo da aresta (i, j)

Variáveis:

$w_{i,j}^{k}$ (Binária): $\begin{cases} 1 & \text{se a aresta } (i,j) \text{ pertence ao caminho no nível } k \\ 0 & \text{caso contrário} \end{cases}$

$x_{i,j}$ (Contínua): quantidade de fluxo que passa pelos arcos (i, j) em todos os níveis k.

$f_{i,j}^{k,d}$ (Contínua): quantidade de fluxo que passa pelos arcos (i, j) que parte de i até d no nível k, sem passar pelo depósito, 0.

$g_{i,j}^{k,d}$ (Contínua): quantidade de fluxo que passa pelos arcos (i, j) que parte de 0 até d no nível k, sem passar por $i + n$.

Formulação:

$$\begin{aligned}\text{Minimize} \sum_{j \in N} c_{0,j} \times w^0_{0,j} \\ + \sum_{k=1}^{2n-1} \sum_{i \in N} \sum_{j \in N(i)} c_{i,j} \times w^k_{i,j} \\ + \sum_{i \in N} c_{i,0} \times w^{2n}_{i,0} \end{aligned} \tag{4.34}$$

$$\sum_{j \in N} w^0_{0,j} = 1 \tag{4.35}$$

$$\sum_{i \in N(j)} w^k_{i,j} = \sum_{l \in N(j)} w^{k+1}_{j,l} \qquad \forall j = 1 \dots 2n \text{ e}$$

$$\forall k = 1 \dots 2n-2 \tag{4.36}$$

$$x_{0,i} = w^0_{0,i} \qquad \forall i = 1 \dots 2n \tag{4.37}$$

$$x_{i,j} = \sum_{k \in N} w^k_{i,j} \qquad \forall i = 1 \dots 2n,$$

$$\forall j = 1 \dots 2n \tag{4.38}$$

$$x_{i,0} = w^{2n}_{i,T} \qquad \forall i = 1 \dots 2n \tag{4.39}$$

$$\sum_{j \in N_0(j)} x_{i,j} = 1 \qquad \forall i = 1 \dots 2n \tag{4.40}$$

$$f^{k,d}_{i,j} \leq w^k_{j,i} \qquad \forall i = 0 \dots 2n, \forall j = 0 \dots 2n,$$

$$\forall k = 0 \dots 2n, \forall d = 1 \dots n \tag{4.41}$$

$$\sum_{j=1}^{2n} \sum_{k=1}^{2n} f^{k,d}_{d,j} = 1 \qquad \forall d = 1 \dots n \tag{4.42}$$

$$\sum_{i \in N(\{j,k\})} f^{k,d}_{i,j} = \sum_{i \in N(\{j,k+p\})} f^{k-1,d}_{j,i} \qquad \forall j, d, k = 1 \dots 2n \tag{4.43}$$

$$g_{i,j}^{k,d} \leq w_{j,i}^{k} \qquad \forall i = 1 \ldots 2n, \forall j = 1 \ldots 2n, \forall k = 1 \ldots 2n, \forall d = 1 \ldots n \tag{4.44}$$

$$\sum_{j=1}^{n} g_{0,j}^{0,d} = 1 \qquad \forall d = 1 \ldots n \tag{4.45}$$

$$g_{0,j}^{0,d} = \sum_{i=1}^{2n} g_{j,i}^{1,d} \qquad \forall j = 1 \ldots n, \forall d = 1 \ldots n \tag{4.46}$$

$$\sum_{i=1}^{n} g_{i,j}^{k,d} = \sum_{i=1}^{n} g_{i,j}^{k+1,d} \qquad \forall d = 1 \ldots n, \forall j = 1 \ldots n, \forall k = 1 \ldots 2n \tag{4.47}$$

As restrições (4.35) até (4.40) são as mesmas da formulação estendida. As demais são as restrições que diferenciam as duas formulações. As restrições de (4.41) até (4.43), garantem que deve haver um fluxo entre a coleta e entrega sem passar pelo depósito, onde (4.41) faz o mapeamento das variáveis w para as f. A restrição (4.42) obriga existir um fluxo com valor unitário entre cada par de coleta-entrega. Na restrição (4.43) é a conservação do fluxo. As variáveis g, por sua vez, garantem que haja um fluxo entre o depósito, 0, e o vértice de coleta, sem passar pela entrega. Dessa forma deve-se mapear as variáveis w para g, restrição (4.44). Depois precisa sair do depósito um fluxo para cada coleta (4.45), depois deve-se conservar esse fluxo até o seu destino, restrições (4.46) e (4.47).

4.4 Testes e resultados

O algoritmo de branch-and-cut foi implementado usando o CPLEX 10.2. As ramificações na árvore de enumeração foram feitas na variável x_{ij} na formulação orientada e na variável w_{ij}^{k} no caso da formulação estendida.

4.4.1 Descrevendo as Instâncias

O conjunto de instâncias testes utilizado foi criado por Ruland e Rodin (1997) através da geração $2n + 1$ pontos aleatoriamente no quadrado $[0, 1000] \times [0, 1000]$. O primeiro ponto é utilizado como o depósito, enquanto os restantes pontos são pedidos para formar pares (ponto i está emparelhado com ponto $n+i$). O custo de viagem (c_{ij}) foi estabelecido pelo arredondamento das distâncias euclidianas. Instâncias com n em $\{5, 10, \ldots, 30, 35\}$ foram criadas, cinco para cada tamanho. As instâncias são nomeadas probnx, em que

x é uma das letras $\{a, b, c, d, e\}$ usadas para distinguir entre as instâncias com o mesmo tamanho.

4.4.2 Resultados Computacionais

Os algoritmos foram implementados em C++ e rodados numa máquina Core 2 Duo E7300, com 4GB de RAM e CPLEX 10.2. Foram realizados vários testes comparativos entre as formulações implementadas e as da literatura.

As tabelas que serão apresentadas contém os seguintes dados. A primeira coluna indica o nome da instância. Na coluna seguinte o melhor valor da literatura, números em itálico representam valores que não foram provados serem ótimos. As 4 próximas colunas apresentam resultados obtidos neste trabalho através das novas formulações (orientada e estendida), e as outras 4 resultados da formulação não-orientada obtidas por Dumitrescu et al. (2008). São 4 os elementos comparativos: custo - valor do limite inferior obtido; GAP - porcentagem que falta para chegar ao melhor; tempo - tempo total em segundos para se obter o limite; cortes - total de cortes adicionados.

Os resultados descritos na Tabela 13, comparam a formulação orientada somente com as restrições (4.9-4.14) (sem cortes adicionais) com a formulação não orientada com apenas as restrições (4.2-4.6) (também sem cortes adicionais). A formulação orientada consegue limites consistentemente melhores que a não orientada o que sugere que aquela é mais forte que esta. Enquanto o GAP médio da orientada ficou em $9,44\%$ o da não orientada foi $11,38\%$. Não foi indicado em Dumitrescu et al. (2008) a quantidade de cortes (4.2-4.6) usados.

A Tabela 14 mostra resultados da formulação orientada fortalecida com o corte mostrado na Seção 4.2.2, o corte do depósito à duas coletas (veja a Seção 4.3.1) e os cortes do CPLEX. A comparação é com os resultados obtidos em Dumitrescu et al. (2008) usando todos os cortes mencionados na Seção 4.1.1, bem como os cortes do CPLEX. Pode-se ver que os cortes da formulação não-orientada foram bem mais efetivos. Enquanto o GAP médio da orientada caiu de $9,45\%$ para $8,21\%$, a formulação orientada obteve um ganho muito mais significativo,passando de $11,38\%$ para $3,32\%$.

Os cortes da formulação não-orientada mencionados na Seção 4.1.1 também poderiam ser inseridos na formulação orientada, obtendo resultados potencialmente melhores. Entretanto isso dependeria de uma implementação de complexas heurísticas de separação (nenhum desses cortes possui separação conhecida em tempo polinomial). A idéia de se

		Orientada				Não Orientada Dumitrescu et al. (2008)			
Instância	**Melhor**	**Custo**	**GAP**	**Tempo**	**Cortes**	**Custo**	**GAP**	**Tempo**	**Cortes**
prob5a	3.585,00	3.585,00	0,00%	0,0	18	3.495,38	2,50%		-
prob5b	2.565,00	2.565,00	0,00%	0,0	19	2.565,00	0,00%		-
prob5c	3.787,00	3.787,00	0,00%	0,0	25	3.787,00	0,00%	0,0	-
prob5d	3.128,00	3.128,00	0,00%	0,0	19	3.128,00	0,00%		-
prob5e	3.123,00	3.046,57	2,45%	0,0	31	2.966,85	5,00%		-
prob10a	4.896,00	4.473,83	8,62%	0,1	112	4.288,90	12,40%		-
prob10b	4.490,00	3.994,60	11,03%	0,1	260	3.767,11	16,10%		-
prob10c	4.070,00	3.979,80	2,22%	0,0	152	3.903,13	4,10%	0,0	-
prob10d	4.551,00	4.380,00	3,76%	0,1	141	4.323,45	5,00%		-
prob10e	4.874,00	4.711,75	3,33%	0,1	138	4.688,79	3,80%		-
prob15a	5.150,00	4.703,00	8,68%	0,3	329	4.583,50	11,00%		-
prob15b	5.391,00	4.649,28	13,76%	0,5	673	4.323,58	19,80%		-
prob15c	5.008,00	4.798,13	4,19%	0,3	375	4.697,50	6,20%	0,0	-
prob15d	5.566,00	4.987,00	10,40%	0,5	579	4.881,38	12,30%		-
prob15e	5.229,00	5.049,82	3,43%	0,4	395	4.957,09	5,20%		-
prob20a	5.698,00	5.173,74	9,20%	1,0	879	5.054,13	11,30%		-
prob20b	6.213,00	5.502,25	11,44%	0,9	679	5.392,88	13,20%		-
prob20c	6.200,00	5.605,52	9,59%	0,5	461	5.449,80	12,10%	0,1	-
prob20d	6.106,00	5.657,58	7,34%	0,7	621	5.574,78	8,70%		-
prob20e	6.465,00	5.751,60	11,03%	0,8	544	5.676,27	12,20%		-
prob25a	7.332,00	6.177,56	15,75%	1,1	686	6.085,56	17,00%		-
prob25b	6.665,00	5.873,17	11,88%	1,2	657	5.685,25	14,70%		-
prob25c	7.095,00	6.213,86	12,42%	1,2	699	6.087,51	14,20%	0,2	-
prob25d	7.069,00	6.095,42	13,77%	1,1	709	6.015,72	14,90%		-
prob25e	6.754,00	6.069,17	10,14%	1,0	622	5.930,01	12,20%		-
prob30a	7.309,00	6.115,71	16,33%	2,0	900	5.964,14	18,40%		-
prob30b	6.857,00	5.979,53	12,80%	1,8	844	5.732,45	16,40%		-
prob30c	7.723,00	6.668,25	13,66%	1,5	686	6.587,72	14,70%	0,3	-
prob30d	7.310,00	6.457,99	11,66%	2,6	1.159	6.396,25	12,50%		-
prob30e	7.213,00	6.108,50	15,31%	1,4	705	6.022,86	16,50%		-
prob35a	7.746,00	6.739,50	12,99%	3,5	1.221	6.591,85	14,90%		-
prob35b	7.904,00	6.566,80	16,92%	4,2	1.297	6.299,49	20,30%		-
prob35c	7.949,00	6.800,47	14,45%	3,9	1.287	6.629,47	16,60%	0,3	-
prob35d	7.905,00	6.704,63	15,19%	2,3	730	6.600,68	16,50%		-
prob35e	8.530,00	7.115,46	16,58%	2,7	879	7.037,25	17,50%		-
Média	5.927,31	5.291,87	9,44%	1,1	558	5.176,31	11,38%	0,1	-

Tabela 13: Relaxação Linear da Formulação Orientada (4.9-4.14) × Não-orientada (4.2-4.6): sem cortes adicionais

		orientada				Não-orientada Dumitrescu et al. (2008)			
Instância	**Melhor**	**Custo**	**GAP**	**Tempo**	**Cortes**	**Custo**	**GAP**	**Tempo**	**Cortes**
prob5a	3.585,00	3.585,00	0,00%	0,0	41	3.585,00	0,00%		19
prob5b	2.565,00	2.565,00	0,00%	0,0	19	2.565,00	0,00%		6
prob5c	3.787,00	3.787,00	0,00%	0,0	25	3.787,00	0,00%	0,0	6
prob5d	3.128,00	3.128,00	0,00%	0,0	19	3.128,00	0,00%		6
prob5e	3.123,00	3.123,00	0,00%	0,0	63	3.123,00	0,00%		45
prob10a	4.896,00	4.494,00	8,21%	0,3	182	4.857,75	0,78%		134
prob10b	4.490,00	4.075,75	9,23%	0,4	180	4.490,00	0,00%		135
prob10c	4.070,00	4.070,00	0,00%	0,2	136	4.070,00	0,00%	3,1	93
prob10d	4.551,00	4.451,65	2,18%	0,2	124	4.551,00	0,00%		93
prob10e	4.874,00	4.731,77	2,92%	0,3	121	4.874,00	0,00%		97
prob15a	5.150,00	4.743,13	7,90%	1,1	328	5.063,33	1,68%		268
prob15b	5.391,00	4.756,69	11,77%	1,9	374	5.141,22	4,63%		580
prob15c	5.008,00	4.863,96	2,88%	1,4	252	5.008,00	0,00%	12,9	165
prob15d	5.566,00	5.112,37	8,15%	1,4	232	5.435,24	2,35%		390
prob15e	5.229,00	5.123,85	2,01%	1,3	281	5.229,00	0,00%		140
prob20a	5.698,00	5.247,34	7,91%	2,4	383	5.695,42	0,05%		438
prob20b	6.213,00	5.629,02	9,40%	4,6	515	6.144,93	1,10%		473
prob20c	6.200,00	5.658,56	8,73%	4,3	390	6.050,71	2,41%	16,8	444
prob20d	6.106,00	5.671,79	7,11%	1,8	405	6.001,88	1,71%		369
prob20e	6.465,00	5.878,76	9,07%	3,8	388	6.199,20	4,11%		962
prob25a	7.332,00	6.220,85	15,15%	6,8	496	6.670,14	9,03%		3.244
prob25b	6.665,00	5.939,32	10,89%	14,1	498	6.267,22	5,97%		2.266
prob25c	7.095,00	6.300,83	11,19%	9,0	695	6.770,59	4,57%	44,8	1.255
prob25d	7.069,00	6.191,44	12,41%	6,7	620	6.526,56	7,67%		3.873
prob25e	6.754,00	6.220,07	7,91%	11,1	585	6.539,76	3,17%		704
prob30a	7.309,00	6.231,05	14,75%	21,0	698	6.776,44	7,29%		3.238
prob30b	6.857,00	6.053,02	11,72%	13,2	612	6.475,78	5,56%		2.262
prob30c	7.723,00	6.766,18	12,39%	20,9	572	7.281,38	5,72%	73,2	2.019
prob30d	7.310,00	6.486,98	11,26%	10,4	753	7.043,19	3,65%		1.372
prob30e	7.213,00	6.279,94	12,94%	28,2	837	6.716,71	6,88%		3.412
prob35a	7.746,00	6.873,30	11,27%	39,3	1141	7.354,74	5,05%		1.649
prob35b	7.904,00	6.678,19	15,51%	23,5	931	7.104,95	10,11%		2.764
prob35c	7.949,00	6.850,60	13,82%	16,5	672	7.514,87	5,46%	100,3	3.194
prob35d	7.905,00	6.866,00	13,14%	28,2	898	7.324,58	7,34%		2.944
prob35e	8.530,00	7.217,68	15,38%	33,0	760	7.698,54	9,75%		3.562
Média	5.927,31	5.367,77	8,21%	8,8	435	5.687,57	3,32%	35,9	1.218

Tabela 14: Relaxação Linear da Formulação orientada (4.9-4.14) + (4.15) + cortes CPLEX × Não-orientada (4.2-4.6) + todos os cortes da seção 4.1.1 + cortes CPLEX

partir para a formulação estendida foi obter limites comparáveis (e até melhores) usando apenas cortes conceitualmente simples e que podem ser separados em tempo polinomial pelo algoritmo do corte mínimo.

A Tabela 15 apresenta os valores obtidos com formulação estendida, comparando-a com a formulação não-orientada melhorada com todos os cortes mencionados na Seção 4.1.1 (mas não os do CPLEX). Os resultados em termos de limites foram muito bons. Em nove das dez instâncias com $n \leq 10$ a formulação estendida obteve a solução ótima inteira, sendo melhor do que a não-orientada com todos os cortes em 4 casos. No entanto, só foi possível realizar esses testes nessas instâncias menores, pois a geração de cortes na formulação estendida está demorando excessivamente para convergir. Pode-se notar a quantidade de cortes gerados nas instâncias *prob5a* e *prob10a*, que pula da casa das centenas para a dezena de milhar, apenas dobrando a quantidade de clientes.

Para evitar a necessidade da separação de um grande número de restrições na for-

		Estendida				Não-orientada Dumitrescu et al. (2008)			
Instância	**Melhor**	**Custo**	**GAP**	**Tempo**	**Cortes**	**Custo**	**GAP**	**Tempo**	**Cortes**
prob5a	3.585,00	3.585,00	0,00%	0,4	70	3.585,00	0,00%		19
prob5b	2.565,00	2.565,00	0,00%	0,0	29	2.565,00	0,00%		6
prob5c	3.787,00	3.787,00	0,00%	0,0	21	3.787,00	0,00%	0,0	6
prob5d	3.128,00	3.128,00	0,00%	0,0	34	3.128,00	0,00%		6
prob5e	3.123,00	3.123,00	0,00%	0,3	184	3.123,00	0,00%		45
prob10a	4.896,00	**4.842,45**	1,09%	3.240,0	14.966	4.806,15	1,84%		134
prob10b	4.490,00	**4.490,00**	0,00%	3.682,8	13.569	4.458,19	0,71%		135
prob10c	4.070,00	4.070,00	0,00%	20,4	11.458	4.070,00	0,00%	3,1	93
prob10d	4.551,00	**4.551,00**	0,00%	434,1	5.562	4.538,03	0,29%		93
prob10e	4.874,00	**4.874,00**	0,00%	807,7	6.463	4.840,66	0,68%		97
Média	3.906,90	3.901,55	0,11%	818,6	5.236	3.890,10	0,35%	1,6	63

Tabela 15: Relaxação Linear da Formulação Estendida (4.24-4.33) + cortes de 2 coletas × Não-orientada (4.2-4.6) + todos os cortes da seção 4.1.1

		Formulação por Fluxos Equivalente				Não-orientada Dumitrescu et al. (2008)			
Instância	**Melhor**	**Custo**	**GAP**	**Tempo**	**Cortes**	**Custo**	**GAP**	**Tempo**	**Cortes**
prob5a	3.585,00	3.585,00	0,00%	0	-	3.585,00	0,00%		19
prob5b	2.565,00	2.565,00	0,00%	0	-	2.565,00	0,00%		6
prob5c	3.787,00	3.787,00	0,00%	0	-	3.787,00	0,00%	0,0	6
prob5d	3.128,00	3.128,00	0,00%	0	-	3.128,00	0,00%		6
prob5e	3.123,00	3.123,00	0,00%	0	-	3.123,00	0,00%		45
prob10a	4.896,00	**4.827,00**	1,41%	103	-	4.806,15	1,84%		134
prob10b	4.490,00	**4.490,00**	0,00%	110	-	4.458,19	0,71%		135
prob10c	4.070,00	4.070,00	0,00%	107	-	4.070,00	0,00%	3,1	93
prob10d	4.551,00	**4.551,00**	0,00%	105	-	4.538,03	0,29%		93
prob10e	4.874,00	**4.874,00**	0,00%	88	-	4.840,66	0,68%		97
prob15a	5.150,00	**5.087,61**	1,21%	955	-	4.999,31	2,93%		268
prob15b	5.391,00	**5.120,26**	5,02%	1.421	-	5.055,83	6,22%		580
prob15c	5.008,00	5.008,00	0,00%	733	-	5.008,00	0,00%	6,9	165
prob15d	5.566,00	**5.545,66**	0,37%	1.352	-	5.396,90	3,04%		390
prob15e	5.229,00	5.229,00	0,00%	929	-	5.229,00	0,00%		140
Média	4.360,87	4.332,70	0,53%	394	-	4.306,00	1,05%	3,3	145

Tabela 16: Relaxação Linear da Formulação de Fluxo (4.35-4.47) × Não-orientada (4.2-4.6) + todos os cortes da seção 4.1.1

mulação estendida, criou-se a formulação por fluxo equivalente que substitui um número exponencial de restrições por uma quantidade polinomial de variáveis, o que deixa o problema grande, usando muita memória, mesmo assim mais fácil de ser resolvido diretamente pelo CPLEX usando um algoritmo de pontos interiores (método barreira). A Tabela 16 apresenta os resultados com as instâncias de 5 a 15 pares encontrados através da formulação de fluxo equivalente. Nesses testes não foram separados os cortes de duas coletas, o que causa uma pequena redução na qualidade dos limites inferiores em relação a tabela anterior. Esses limites continuam sendo consistentemente melhores dos que os obtidos por (DUMITRESCU et al., 2008) com a formulação não orientada.

Apesar da formulação por fluxo equivalente ter permitido avaliar melhor o potencial do uso das variáveis estendidas no TSPPD, seu uso já se torna inviável para $n = 20$. Espera-se que com o uso de técnicas mais avançadas de programação matemática (técnicas lagrangeanas, algoritmos de dual ascent ou de geração de colunas) seja possível no futuro obter tais limites de forma eficiente.

5 Conclusões e Trabalhos Futuros

5.1 Conclusões

Neste trabalho foram atacados dois problemas relacionados ao roteamento de veículo com coleta e entrega. O primeiro, o VRPPDTW, apresentou um problema com muitas restrições e grande complexidade computacional, assim, decidiu-se que a melhor abordagem a fim de encontrar soluções boas em tempos razoáveis foi uma metaheurística. Para o segundo, como foram propostas formulações inteira utilizando o algoritmo de branch-and-cut e o uma formulação linear resolvida pelo método de barreiras. Escolher um problema com menos restrições não significa que seja mais simples de solução, uma vez que, o TSPPD continua pertencendo a classe NP-difícil e sim que tem menos detalhes para serem observados.

A heurística ILS-MRD, utilizada para resolver o VRPPDTW, apresentou bons resultados comparados à literatura. Apesar de não encontrado novos valores melhores, conseguiu encontrar, em sua grande maioria, os últimos resultados da literatura. Esta se mostrou uma heurística que demanda de pouco tempo computacional para conseguir bons resultados.

Para o problema de caixeiro viajante com coleta e entrega foram propostas 3 formulações para serem comparadas com as da literatura. A formulação orientada se apresentou bastante rápida, no entanto, não conseguiu bons limites inferiores para o problema, pois comparada a formulação não-orientada sem cortes foi superior, mas ao se adicionar cortes a formulação não orientada, a formulação orientada se mostrou mais fraca. Então resolveu-se estender a formulação orientada, esta se mostrou bem mais forte que a anterior com bons limites inferiores, em contrapartida tornou-se lenta, devido a grade quantidade de restrições (um número exponencial). Assim, para amenizar esse problema tentou-se a formulação de fluxo que ficou como intermediária, pois não tem muitas restrições, em contra-partida em um número polinomial de variáveis, que apesar de muitas (utiliza grande quantidade de memória), é ainda menor que a quantidade de restrições

(cortes) necessárias para encontrar uma solução na formulação estendida.

A grande vantagem da formulação de estendida/fluxo do modelo exato, para o TSPPD, mostrou-se bastante satisfatório no que se refere aos valores dos limites inferiores. Em sua totalidade 8 das 15 instâncias foram encontrados o ótimo na raiz, da árvore de branch-and-bound. O que demonstra que a formulação é forte. Conseguindo superar em 7 instâncias os valores anteriormente encontrados na literatura.

5.2 Trabalhos Futuros

Para o VRPPDTW, há dois trabalhos na literatura que apresentaram bons resultados. Bent e Hentenryck (2006) propuseram uma heurística híbrida onde num primeiro estágio tentaram reduzir a quantidade de rotas e num segundo estágio, a custo de cada rota. O outro trabalho foi o de Ropke e Pisinger (2006a) os quais apresentaram uma heurística adaptativa baseada em inserções e remoções, mas para dispondo de uma boa estrutura e algoritmos para tentar escolher os melhores elementos a serem alterados. Com isso percebe-se que para se melhorar uma heurística, tem-se cada vez mais unido várias técnicas, heurísticas híbridas, pois tem-se conseguido bons resultados com elas. Assim, um path relink poderia ser usado no lugar da perturbação em algumas iterações. A fase inicial poderia conter 2 etapas: uma para reduzir a quantidade de rotas e outra para reduzir o custo total da viagem.

Outra proposta para o VRPPDTW seria utilizar a solução heurística como passo inicial para o algoritmo exato de geração de coluna. Foi desenvolvido por Ropke (2007) um branch-and-cut-and-price, e testado com 3 formulações diferentes. O mais importante nesse tipo de algoritmo é conseguir descrever bons critérios de dominância para assim reduzir ao máximo o espaço de busca.

Apenar do TSPPD ter encontrado bons valores, o tempo não foi razoável, assim uma das opções para melhorar o tempo e, conseqüentemente, agilizar a convergência seria a adição dos cortes propostos por (DUMITRESCU et al., 2008) à formulação estendida (Seção 4.3). Outra proposta seria usar a formulação orientada até ela finalizar seus cortes na raiz, depois pegar essa solução e transformá-la em ponto de partida da formulação estendida, assim, esta começaria com bons cortes e um limite inferior bem melhor. Logo, juntando as 2 propostas acima acredita-se que os tempos computacionais reduziriam.

Referências

ALVARENGA, A. V. et al. Classification of breast tumours on ultrasound images using morphometric parameters. In: *Proc. IEEE International Workshop on Intelligent Signal Processing*. [S.l.: s.n.], 2005. p. 206–210.

BALAS, E.; FISCHETTI, M.; PULLEYBLANK, W. R. The precedence-constrained asymmetric traveling salesman polytope. *Mathematical Programming*, v. 68, p. 241–265, 1995.

BEASLEY, J. Route first-cluster second methods for vehicle routing. *Omega*, v. 11, p. 403–408, 1983.

BELFIORE, P. P.; Y.YOSHIZAKI, H. T. *Scatter Search para Problemas de Roteirização de Veículos com Frota Heterogênea, Janelas de Tempo e Entregas Fracionadas*. Tese (Doutorado) — Universidade de São Paulo, 2006.

BENT, R.; HENTENRYCK, P. V. A two-stage hybrid algorithm for pickup and delivery vehicle routing problems with time windows. *Comput. Oper. Res.*, Elsevier Science Ltd., Oxford, UK, UK, v. 33, n. 4, p. 875–893, 2006. ISSN 0305-0548.

BERALDI, P. et al. Efficient neighborhood search for the probabilistic pickup and delivery travelling salesman problem. *Networks*, v. 45, p. 195 – 198, 2005.

BERBEGLIA, G. et al. Static pickup and delivery problems: a classification scheme and survey. *Spanish Society of Statistics and Operations Research*, v. 15, p. 32–44, 2007.

BERGVINSDOTTIR, K. B. *The genetic algorithm for solving the dial-a-ride problem*. Dissertação (Mestrado) — Technical University of Denmark, 2004.

BRäYSY, O.; GENDREAU, M.; DULLAERT, W. Evolutionary algorithms for the vehicle routing problem with time windows. *Journal of Heuristics*, v. 11, p. 587–611, 2004.

CAMPOS, V.; MOTA, E. Heuristic procedures for the capacitated vehicle routing problem. *Comput. Optim. Appl.*, Kluwer Academic Publishers, Norwell, MA, USA, v. 16, n. 3, p. 265–277, 2000. ISSN 0926-6003.

CARICATO, P. et al. Parallel tabu search for a pickup and delivery problem under track contention. *Parallel Comput.*, Elsevier Science Publishers B. V., Amsterdam, The Netherlands, The Netherlands, v. 29, n. 5, p. 631–639, 2003. ISSN 0167-8191.

CARRABS, F.; CORDEAU, J.-F.; LAPORTE, G. Variable neighborhood search for the pickup and delivery traveling salesman problem with LIFO loading. *INFORMS Journal on Computing*, v. 19, n. 4, p. 618–632, 2007. Disponível em: <http://joc.journal.informs.org/cgi/content/short/19/4/618>.

CHIN, A. J.; KIT, H. W.; LIM, A. A new ga approach for the vehicle routing problem. In: *ICTAI '99: Proceedings of the 11th IEEE International Conference on Tools with Artificial Intelligence*. Washington, DC, USA: IEEE Computer Society, 1999. p. 307. ISBN 0-7695-0456-6.

CHOI, E.; TCHA, D.-W. A column generation approach to the heterogeneous fleet vehicle routing problem. *Comput. Oper. Res.*, Elsevier Science Ltd., Oxford, UK, UK, v. 34, n. 7, p. 2080–2095, 2007. ISSN 0305-0548.

CORDEAU, J. F. A branch-and-cut algorithm for the dial-a-ride problem. *Operations Research*, v. 54(6), p. 573–586, 2006.

CORDEAU, J.-F.; LAPORTE, G. The dial-a-ride problem (darp): Variants, modeling issues and algorithms. *4OR*, v. 1, p. 89–101, 2003.

CORDEAU, J. F.; LAPORTE, G. A tabu search heuristic for the static multi-vehicle dial-a-ride problem. *Transportation Research*, Part B: Methodological 37(6), p. 579–594, 2003.

DANTZIG, G. B.; RAMSER, J. H. The truck dispatching problem. *Management Science*, v. 6, n. 1, p. 80–91, Outubro 1959. Disponível em: <http://mansci.journal.informs.org/cgi/content/abstract/6/1/80>.

DIAS, A. d. C.; LIMA, F. R. F. de. A importância dos estudos sobre infra-estrutura e logística. *Análise Conjuntural*, v. 30, p. 15–16, 2008.

DUMAS, Y.; DESROSIERS, J.; SOUMIS, F. The pickup and delivery problem with time windows. *European Journal of Operational Research*, v. 54, n. 1, p. 7–22, September 1991. Disponível em: <http://ideas.repec.org/a/eee/ejores/v54y1991i1p7-22.html>.

DUMITRESCU, I. et al. The travelling salesman problem with pickup and delivery: Polyhedral results and a branch-and-cut algorithm. *Mathematical Programming*, Ser. A, p. 37, 2008.

ERDOGAN, G.; CORDEAU, J.-F.; LAPORTE, G. The pickup and delivery traveling salesman problem with first-in-first-out loading. *Comput. Oper. Res.*, Elsevier Science Ltd., Oxford, UK, UK, v. 36, n. 6, p. 1800–1808, 2009. ISSN 0305-0548.

FAN, J.; WANG, X.; CHEN, Q. A heuristic algorithm for multiple depots vehicle routing problem with backhauls. In: *FSKD '07: Proceedings of the Fourth International Conference on Fuzzy Systems and Knowledge Discovery (FSKD 2007) Vol.3*. Washington, DC, USA: IEEE Computer Society, 2007. p. 421–425. ISBN 0-7695-2874-0.

GELOGULLARI, C. A. An exact algorithm for the vehicle routing problem with backhauls. *Computers and Operations Research*, v. 33, p. 595–619, 2004.

GENDREAU, M.; LAPORTE, G.; VIGO, D. Heuristics for the traveling salesman problem with pickup and delivery. *Comput. Oper. Res.*, Elsevier Science Ltd., Oxford, UK, UK, v. 26, n. 7, p. 699–714, 1999. ISSN 0305-0548.

GHAZIRI, H.; OSMAN, I. H. A neural network algorithm for the traveling salesman problem with backhauls. *Computers and Industrial Engineering*, v. 44(2), p. 267–281, 2003.

GOETSCHALCKX, M.; JACOBS-BLECHA, C. *The vehicle routing problem with backhauls: Properties and solution algorithms*. [S.l.], 1993.

HERNÁNDEZ-PÉREZ, H.; GONZÁLEZ, J. J. S. A branch-and-cut algorithm for a traveling salesman problem with pickup and delivery. *Discrete Applied Mathematics*, v. 145, n. 1, p. 126–139, 2004. Disponível em: <http://dx.doi.org/10.1016/j.dam.2003.09.013>.

HO, S. C.; GENDREAU, M. Path relinking for the vehicle routing problem. *Journal of Heuristics*, v. 12, p. 55–72, 2006. Disponível em: <http://www.springerlink.com/content/5075744n65023367/>.

LAU, H. C.; LIANG, Z. Pickup and delivery with time windows: Algorithms and test case generation. IEEE Computer Society, Washington, DC, USA, p. 333, 2001.

LI, H.; LIM, A. A metaheuristic for the pickup and delivery problem with time windows. p. 160, 2001. Disponível em: <http://computer.org/proceedings/ictai/1417/14170160abs.htm>.

LIM, H.; LIM, A.; RODRIGUES, B. *Solving the Pick up and Delivery Problem with Time Windows using "Squeaky Wheel" Optimization with Local Search*. [S.l.], 2002.

LOURENçO, H. R.; MARTIN, O.; STüTZLE, T. Handbook of metaheuristics. In: _____. [S.l.]: Kluwer Academic Publishers, 2003. cap. Iterated Local Search, p. 251–285.

LU, Q.; DESSOUKY, M. M. A new insertion-based construction heuristic for solving the pickup and delivery problem with hard time windows. *European Journal of Operational Research*, v. 175, p. 672–687, 2006.

MATHEMATICS, S. A. Department of optimisation, technical report in progress. Em processo.

MIN, H. The multiple vehicle routing problem with simultaneous delivery and pickup points. *Transportation Research-A*, v. 23, p. 377–86, 1989.

MONTANé, F. A. T.; FERREIRA, V. J. M. F.; GALVãO, R. D. Determinação de rotas para empresa de entrega expressa. *Pesquisa Operacional*, v. 17, p. 107–135, 1997.

MONTANé, F. A. T.; GALVãO, R. D. A tabu search algorithm for the vehicle routing problem with simultaneous pick-up and delivery service. *Computers and Operations Research*, v. 33(3), p. 595–619., 2006.

MONTEMANNI, R. et al. *A new algorithm for a Dynamic Vehicle Routing Problem based on Ant Colony System*. [S.l.]: Istituto Dalle Molle Di Studi Sull Intelligenza Artificiale, 2002.

MOSHEIOV, G. The travelling salesman problem with pick-up and delivery. *European Journal of Operational Research*, v. 79, n. 2, p. 299–310, December 1994. Disponível em: <http://ideas.repec.org/a/eee/ejores/v79y1994i2p299-310.html>.

MOSHEIOV, G. Vehicle routing with pick-up and delivery: Tour-partition heuristics. *Computer and Industrial Engineering*, v. 34(3), p. 669–684, 1998.

NANRY, W. P.; BARNES, W. J. Solving the pickup and delivery problem with time windows using reactive tabu search. *Transportation Research*, Part B: Methodological 34(2), p. 107–121, 2000.

OMBUKI, B.; ROSS, B. J.; HANSHAR, F. Multi-objective genetic algorithms for vehicle routing problem with time windows. *Applied Intelligence*, Kluwer Academic Publishers, Hingham, MA, USA, v. 24, n. 1, p. 17–30, 2006. ISSN 0924-669X.

PANKRATZ, G. A grouping genetic algorithm for the pickup and delivery problem with time windows. *OR Spectrum*, v. 27(1), p. 21–41, 2005.

PARRAGH, S. N.; DOERNER, K. F.; HARTL, R. F. Part i: Transportation between customers and depot. *Journal für Betriebswirtschaft*, v. 51, p. 21–51, 2008.

PARRAGH, S. N.; DOERNER, K. F.; HARTL, R. F. Part ii: Transportation between pickup and delivery locations. *Journal für Betriebswirtschaft*, v. 51, p. 81–117, 2008.

POTVIN, J.-Y.; DUHAMEL, C.; GUERTIN, F. A genetic algorithm for vehicle routing with backhauling. *Applied Intelligence*, v. 6(4), p. 345–355, 1996.

PRIVé, J. et al. Solving a vehicle routing problem arising in soft drink distribution. *Journal of the Operational Research Society*, v. 57, p. 1045–1052, 2006.

PSARAFIS, H. A dyanmic programming solution to the single vehicle many-to-many immediate request dial-a-ride problem. *Transportation Science*, v. 14, p. 130–154, 1980.

REIMANN, M.; DOERNER, K.; HARTL, R. F. Insertion based ants for vehicle routing problems with backhauls and time windows. In: *ANTS '02: Proceedings of the Third International Workshop on Ant Algorithms*. London, UK: Springer-Verlag, 2002. p. 135–148. ISBN 3-540-44146-8.

RENAUD, J.; BOCTOR, F. F.; LAPORTE, G. Perturbation heuristics for the pickup and delivery traveling salesman problem. *Computers & OR*, v. 29, n. 9, p. 1129–1141, 2002. Disponível em: <http://dx.doi.org/10.1016/S0305-0548(00)00109-X>.

ROPKE, S. *Branch-and-Cut-and-Price for the Pickup and Delivery Problem with Time Windows*. Universitetsparken 1, 2100 Copenhagen, Denmark, Novembro 2007.

ROPKE, S.; PISINGER, D. An adaptive large neighborhood search heuristic for the pickup and delivery problem with time windows. *Transportation Science*, INFORMS, Institute for Operations Research and the Management Sciences (INFORMS), Linthicum, Maryland, USA, v. 40, n. 4, p. 455–472, 2006. ISSN 1526-5447.

ROPKE, S.; PISINGER, D. A unified heuristic for a large class of vehicle routing problems with backhauls. *European Journal of Operational Research*, v. 171, n. 3, p. 750–775, June 2006. Disponível em: <http://ideas.repec.org/a/eee/ejores/v171y2006i3p750-775.html>.

ROUSSEAU, L.-M.; GENDREAU, M.; PESANT, G. Using constraint-based operators to solve the vehicle routing problem with time windows. *Journal of Heuristics*, Kluwer Academic Publishers, Hingham, MA, USA, v. 8, n. 1, p. 43–58, 2002. ISSN 1381-1231.

RULAND, K. *Polyhedral solution to the pickup and delivery problem*. Tese (Doutorado) — Washington University, St. Louis, MO, 1995.

RULAND, K. S.; RODIN, E. Y. The pickup and delivery problem: Faces and branch-and-cut algorithm. *Computers & Mathematics with Applications*, v. 33, p. 1–13, Junho 1997.

SALHI, S.; NAGY, G. A cluster insertion heuristic for single and multiple depot vehicle routing problems with backhauling. *Journal of the Operational Research Society*, v. 50, p. 1034–42, 1999.

SAVELSBERGH, M.; SOL, M. The general pickup and delivery problem. *Transportation Science*, v. 29, p. 17–29, 1995.

SIGURD, M.; PISINGER, D.; SIG, M. The pickup and delivery problem with time windows and precedences,. *Transportation Science*, v. 38, p. 197–209, 2004.

SOLOMON, M. M. Algorithms for the vehicle routing and scheduling problems with time window constrains. Massachusetts Institute of Technology, Operations Research Center, v. 35, p. 254–264, 1987.

STüTZLE, T.; HOOS, H. H. Analysing the run-time behaviour of iterated local search for the travelling salesman problem. In: *Third Metaheuristics International Conference*. [S.l.: s.n.], 1999.

TAM, V.; TSENG, L. C. Y. Effective heuristics to solve pickup and delivery problems with time windows. In: *ICTAI '03: Proceedings of the 15th IEEE International Conference on Tools with Artificial Intelligence*. Washington, DC, USA: IEEE Computer Society, 2003. p. 184. ISBN 0-7695-2038-3.

TETRASOFT, A. Mapbooking algoritm for pickupand delivery solutions with time windows and capacity restraints. Http://www.sintef.no/static/am/opti/projects/top/vrp/benchmarks. 2003.

WILLIAM, P. N.; BARNES, J. W. Solving the pickup and delivery problem with time windows using tabu search. *Transportation Research*, v. 34, p. 107–121, 2001.

XU, H. et al. Solving a practical pickup and delivery problem. *Transportation Science*, v. 37, p. 347–364, 2003.