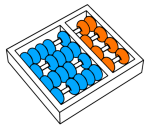
Universidade Estadual de Campinas
Instituto de Computação



Deyvison Nogueira Rodrigues

Problema de Roteamento de Veículos Capacitado com Janelas de Tempo e Clientes Estocásticos

CAMPINAS
2020

Deyvison Nogueira Rodrigues

Problema de Roteamento de Veículos Capacitado com Janelas de Tempo e Clientes Estocásticos

Dissertação apresentada ao Instituto de Computação da Universidade Estadual de Campinas como parte dos requisitos para a obtenção do título de Mestre em Ciência da Computação.

Orientador: Prof. Dr. Flávio Keidi Miyazawa
Coorientador: Prof. Dr. Eduardo Candido Xavier

Este exemplar corresponde à versão final da Dissertação defendida por Deyvison Nogueira Rodrigues e orientada pelo Prof. Dr. Flávio Keidi Miyazawa.

CAMPINAS
2020

Ficha catalográfica
Universidade Estadual de Campinas
Biblioteca do Instituto de Matemática, Estatística e Computação Científica
Ana Regina Machado - CRB 8/5467

R618p

Rodrigues, Deyvison Nogueira, 1995-
Problema de roteamento de veículos capacitado com janelas de tempo e clientes estocásticos / Deyvison Nogueira Rodrigues. – Campinas, SP : [s.n.], 2020.

Orientador: Flávio Keidi Miyazawa.
Coorientador: Eduardo Candido Xavier.
Dissertação (mestrado) – Universidade Estadual de Campinas, Instituto de Computação.

1. Problema de roteamento de veículos. 2. Janela de tempo. 3. Otimização estocástica. I. Miyazawa, Flávio Keidi, 1970-. II. Xavier, Eduardo Candido, 1979-. III. Universidade Estadual de Campinas. Instituto de Computação. IV. Título.

Informações para Biblioteca Digital

Título em outro idioma: Capacitated vehicle rounting problem with time windows and stochastic customers
Palavras-chave em inglês:
Vehicle routing problem
Time window
Stochastic optimization
Área de concentração: Ciência da Computação
Titulação: Mestre em Ciência da Computação
Banca examinadora:
Flávio Keidi Miyazawa [Orientador]
Reinaldo Morabito Neto
Fábio Luiz Usberti
Data de defesa: 16-12-2020
Programa de Pós-Graduação: Ciência da Computação

Identificação e informações acadêmicas do(a) aluno(a)
- ORCID do autor: https://orcid.org/0000-0002-0688-6549
- Currículo Lattes do autor: http://lattes.cnpq.br/9494604006711212

Universidade Estadual de Campinas
Instituto de Computação

Deyvison Nogueira Rodrigues

Problema de Roteamento de Veículos Capacitado com Janelas de Tempo e Clientes Estocásticos

Banca Examinadora:

- Prof. Dr. Eduardo Candido Xavier
  Universidade Estadual de Campinas
- Prof. Dr. Reinaldo Morabito Neto
  Universidade Federal de São Carlos
- Prof. Dr. Fábio Luiz Usberti
  Universidade Estadual de Campinas

A ata da defesa, assinada pelos membros da Comissão Examinadora, consta no SIGA/Sistema de Fluxo de Dissertação/Tese e na Secretaria do Programa da Unidade.

Campinas, 16 de dezembro de 2020

Agradecimentos

Agradeço em primeiro lugar a Deus, pela a força e por ter iluminado o meu caminho durante esta jornada.

Agradeço também a todos os professores que me acompanharam durante todo processo, da graduação ao mestrado, em especial a Professora Tatiane Fernandes Figueiredo e aos professores Flávio Keidi Miyazawa e Eduardo Candido Xavier, que foram importante na minha vida acadêmica e pessoal. A toda minha família que, com muito carinho e apoio, não mediram esforços para que eu chegasse até esta etapa da minha vida. Ao Laboratório de Otimização e Combinatória (LOCo) e ao grupo de pesquisa Núcleo de Estudo em aprendizado de Máquina e Otimização (NEMO) e aos meus amigos pelo apoio e incetivos constantes, em especial ao Matheus Jun Ota e Carlos Victor Dantas Araújo.

O presente trabalho foi realizado com apoio do Conselho Nacional de Desenvolvimento Científico e Tecnológico - Brasil (CNPq) com o processo 134673/2018-2.

Resumo

Este trabalho introduz o estudo do Problema de Roteamento de Veículos Capacitados com Janelas de Tempo e Clientes Estocásticos (CVRPTWSC), uma variação do Problema de Roteamento de Veículos Capacitados com Janelas de Tempo (CVRPTW) onde um subconjunto dos clientes são incertos. O CVRPTWSC é formulado como um problema de programação linear inteira estocástico em dois estágios, que é resolvido através do *Integer L-Shaped Method*. Experimentos computacionais realizados em instâncias adaptadas do CVRPTW demonstram a eficiência e os limites do método proposto. Foram implementados 3 métodos de inserção dos cortes de otimalidade, e foram resolvidas instâncias com 25 e 50 clientes.

Abstract

This work introduces the study of the Capacitated Vehicle Routing Problem with Time Windows and Stochastic Customers (CVRPTWSC), a variation of the Capacitated Vehicle Routing Problem with Time Windows (CVRPTW) where a subset of customers are uncertain. CVRPTWSC is formulated as a two-stage stochastic integer linear programming problem, which is solved using the *Integer L-Shaped Method*. Computational experiments carried out in adapted instances for CVRPTW demonstrate the efficiency and limits of the proposed method. Three methods to insert the optimality cuts were implemented and instances with 25 and 50 customers were resolved.

Lista de Figuras

Lista de Tabelas

Sumário

Capítulo 1

Introdução

O Problema de Roteamento de Veículos (*Vehicle Routing Problem* - VRP) tem o objetivo de determinar um conjunto de rotas ótimas a serem realizadas por uma frota de veículos para servir um determinado conjunto de clientes. É um dos mais importantes e estudados problemas de otimização combinatória [30]. Dantzig e Ramser introduziram o problema em 1959 [7], modelando um problema do mundo real que objetiva entregar gasolina às estações de serviços, e propuseram a primeira formulação de programação matemática para o problema.

No Problema de Roteamento de Veículos Capacitados (*Capacitated Vehicle Routing Problem* - CVRP) os clientes correspondem a receptores e as demandas são determinísticas, ou seja, são dados definidos na entrada do problema. Uma instância do CVRP pode ser descrita da seguinte forma. Temos um grafo direcionado completo $G = (V, A)$, onde $V = \{0, 1, \ldots, n\}$ é o conjunto de vértices e A é o conjunto de arcos. Os vértices no conjunto $\{1, \ldots, n\}$ correspondem aos *clientes*, enquanto o vértice 0 é associado ao *depósito*. Um custo $c_{ij} > 0$ é associado a cada arco $(i, j) \in A$ e representa o custo para percorrer do vértice i ao vértice j. Cada vértice i é associado a uma demanda não negativa d_i a ser atendida. Um conjunto K de veículos idênticos, cada um com capacidade C, está disponível no depósito e cada veículo pode percorrer no máximo uma rota. O CVRP consiste em encontrar uma coleção de k circuitos simples (cada circuito correspondente a rota de um veículo) tal que: i) todos os circuitos têm o início e o fim no depósito; ii) cada cliente pertence a exatamente um circuito; iii) a soma das demandas dos clientes de um circuito não excede a capacidade C de um veículo; iv) o custo total nos arcos dos circuitos é mínimo. Um exemplo de solução de um VRP pode ser visto na Figura 1.1.

O Problema de Roteamento de Veículos com Janela de Tempo (*Vehicle Routing Problem with Time Windows* - VRPTW) é uma extensão do VRP. Nesta variação, para cada arco (i, j) é associado um tempo de viajem t_{ij} indicando o tempo de deslocamento do vértice i ao vértice j. Além disso, cada cliente i possui um tempo de serviço s_i e um intervalo de tempo $[a_i, b_i]$, chamado de janela de tempo, indicando o intervalo no qual o veículo deve chegar no cliente i, para atendê-lo.

O Problema de Roteamento de Veículos Estocástico (*Stochastic Vehicle Routing Problem* - SVRP) é uma variação do VRP onde alguns de seus elementos são incertos como a demanda, tempo de deslocamento do veículo e conjunto de clientes a serem visitados. Gendreau et al. [12] afirmam que o SVRP se diferencia da sua versão determinística em

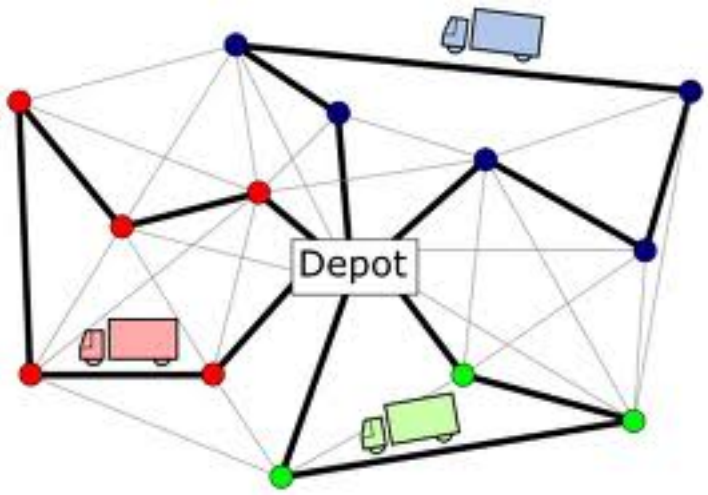


Figura 1.1: Exemplo de saída do VRP

diversos aspectos fundamentais, pois algumas propriedades do VRP não são válidas no caso estocástico, e suas soluções e métodos são mais complicados.

Existem duas abordagens que se destacam para tratar a incerteza em otimização: Otimização Estocástica e Otimização Robusta. Na primeira, a distribuição de probabilidade dos dados incertos é conhecida. A otimização robusta por outro lado, não assume que as probabilidades sejam conhecidas, mas em vez disso, assume que os dados incertos estão em um conjunto de incerteza [13]. O avanço de tecnologias de informação e comunicação, assim como a crescente quantidade de dados, permite reunir informações relevantes para o roteamento de veículos com incerteza. Neste projeto, iremos abordar o Problema de Roteamento de Veículos Capacitado com Janelas de Tempo e Clientes Estocásticos. Inicialmente será utilizada uma abordagem de otimização estocástica do ponto de vista prático, com o desenvolvimento de algoritmos exatos, através da decomposição de Benders utilizada para programação estocástica, também conhecido como *L-shaped Method*.

Esta dissertação, irá desenvolver um algoritmo exato para o Problema de Roteamento de Veículos Capacitados com Janelas de Tempo e Clientes Estocásticos. Nosso algoritmo é desenvolvido através do *Integer L-Shaped Method*, como descrito em Louveaux e Laporte [3], combinado com algoritmos de plano de corte e com algumas variações no método de inserção de cortes de otimalidade. Através de experimentos computacionais realizados, obtivemos sucesso na resolução de instâncias com 25 e 50 clientes, porém foi observado um maior esforço computacional na resolução do primeiro estágio.

1.1 Motivação

Nesta seção apresentamos algumas das motivações para o estudo do problema em otimização estocástica.

Existem diversas aplicações para problemas de roteamento, pois de forma geral, elas abrangem diversas atividades tais como entrega de correspondência, entrega/recolhimento de produtos, coleta de lixo etc. Devido à aplicabilidade do problema, existe um grande número de algoritmos e técnicas para as diversas variações do VRP.

O VRP e as suas variações desempenham um papel crucial na logística, e muitas destas variantes foram amplamente estudadas na literatura durante as últimas décadas [31]. A maior parte das pesquisas foram dedicadas aos problemas determinísticos, onde todos os parâmetros relacionados ao problema são conhecidos na entrada. No entanto, em muitas situações, nem sempre é viável descrever todos os parâmetros do VRP como determinísticos, pois existe alguma incerteza inerente relacionada às demandas do cliente, o tempo de viagem e o conjunto de clientes a serem visitados [5]. Devido à dificuldade de obter previamente todos estes dados, se torna importante o estudo de otimização onde alguns dos dados são incertos.

Apesar de já existirem publicações sobre otimização robusta datadas a partir da década de 1970 [28], o estudo desta área é relativamente novo e ativo. Ritzinger et al. [25] afirmam que nos últimos anos, o interesse em pesquisa em roteamento de veículos tem se concentrado em abordagens dinâmicas e estocásticas. Neste artigo, os autores também alegam que o avanço das tecnologias de informação e comunicação, assim como a crescente quantidade de dados disponíveis, permite reunir informações relevantes para o roteamento de veículos. Além disso, a capacidade de computação atual permite a aplicação de algoritmos de roteamento que fornecem soluções em tempo real, incorporando informações online e considerando possíveis eventos futuros. Esses avanços oferecem a oportunidade de melhorar a qualidade da solução em sistemas de tempo real, mas precisam ser aproveitados por algoritmos de otimização eficientes para resolver problemas de roteamento de veículos dinâmicos e estocásticos. Gorissen et al. [13] demonstram diversos trabalhos em otimização com incerteza, relativamente recentes, em vários campos de aplicação como roteamento de veículos, operações de sistema elétrico, cuidados da saúde, escalonamento etc.

Para exemplificar o problema, considere que uma companhia de entregas deseja servir todos os clientes diariamente. O conjunto de potenciais clientes é fixo e conhecido, mas só se conhece a probabilidade de cada cliente efetuar um pedido. Porém, somente um subconjunto destes clientes precisam ser servidos a cada dia. A companhia não pode reotimizar a rota diariamente, ou simplesmente não o deseja fazer. Então, a rota é planejada a priori para visitar os clientes na mesma ordem, todos os dias, pulando aqueles que não efetuaram o pedido no dia. Esse problema é definido por Schalekamp e Shmoys [26] como *Universal Traveling Salesman Problem* e tem como objetivo minimizar a razão máxima, dentre todos os possíveis subconjuntos de clientes, do tamanho da sub-rota induzida com o tamanho da rota ideal para esse subconjunto. Schalekamp e Shmoys [26] também definem um segundo modelo probabilístico. Nele uma suposição é que existe alguma distribuição de probabilidades nos subconjuntos de clientes, e o objetivo é minimizar o custo esperado da sub-rota induzida que atende os clientes. Este problema é conhecido como *A priori traveling salesman problem.*

1.2 Trabalhos Relacionados

Nesta seção, apresentamos alguns trabalhos encontrados na literatura que abordam os problemas semelhante ao Problema de Roteamento de Veículos Capacitado com Janelas

de Tempo e Clientes Estocásticos e uma breve revisão sobre eles.

O Problema do Caixeiro Viajante Probabilístico (*Probabilistic Traveling Salesman Problem* - PTSP) foi introduzido por Jaillet em [14] como uma variação do Problema do Caixeiro Viajante (*Traveling Salesman Problem* - TSP). Neste problema, temos clientes certos e incertos, onde os últimos possuem uma certa probabilidade de realizarem um pedido ou não. Uma rota, computada *a priori*, deve conter o conjunto total de clientes e define a ordem em que estes serão visitados. O objetivo é encontrar uma rota com o custo esperado mínimo considerando que clientes incertos não realizados serão "pulados" na rota computada. Jaillet também propõe um método para calcular o custo esperado de uma rota, porém assume que todos os clientes incertos possuem uma mesma probabilidade de ocorrência.

Laporte et al. [18] formulam o PTSP como um programa linear inteiro estocástico e a rota é definida em duas fases. Primeiramente é projetado uma rota a priori passando por todos os potenciais clientes e na segunda fase os clientes ausentes são retirados da rota, pulando para o próximo cliente na rota. É definido um modelo de programação linear inteira estocástica que é resolvido utilizando o *Integer L-Shaped Method* que foi inicialmente introduzido por Laporte e Louveaux [17] e é uma extensão do método proposto por Van Slyke e Wets [32]. Laporte et al. [18] conseguem encontrar o valor ótimo para instâncias de tamanho até 50 clientes no total, entre certos e incertos.

O Problema de Roteamento de Veículos com Demandas e Clientes Estocásticos (*Vehicle Routing Problem with Stochastic Demands and Customers* -VRPSDC) foi definido por Gendreau et al. [11] e é uma variação do VRP onde além da incerteza dos clientes, estes também possuem uma demanda estocástica. É utilizado um modelo de otimização estocástica para resolver o problema em dois estágios. No primeiro estágio é realizado o planejamento das rotas. No segundo estágio, quando o conjunto de clientes já é conhecido, as rotas seguem o planejamento do primeiro estágio e os clientes ausentes são removidos da rota. Entretanto, caso a capacidade do veículo seja atingida ou exceda o limite, o veículo retorna ao depósito e depois volta para o cliente onde houve falha, seguindo o planejamento da rota. Os autores propõem uma extensão do *Integer L-Shaped Method*, aplicando uma técnica de *branch and cut*. Gendreau et al. aplicaram o método proposto em instâncias aleatórias e obtiveram soluções ótimas somente para instâncias de pequeno porte. Para entradas com exatamente um cliente certo e $n-1$ clientes incertos, a maior instância resolvida foi com $n = 9$. Para instâncias com apenas um cliente incerto, a solução ótima só foi encontrada com no máximo 46 clientes.

O Problema de Roteamento de Veículos com Rotas Consistentes (*Consistent Vehicle Routing Problems* - conVRP), foi definido inicialmente por Kovacs et al. [15]. Neste problema, um veículo deve sempre percorrer rotas semelhantes periodicamente, ou seja, rotas consistentes, sendo um problema comum na coleta de lixo [29], nas entregas de produtos farmacêuticos e hospitalares [4] e no transporte de estudantes, idosos ou pessoas com necessidades especiais [9], dentre outros. Entretanto, pode existir incerteza associada a demanda dos clientes, o tempo de serviço ou a localização dos clientes. O objetivo do conVRP é definir um conjunto de rotas consistentes levando em consideração as incertezas associadas aos clientes e respeitando restrições de janelas de tempo e capacidade dos veículos.

Laporte et al. abordam em [19] o Problema de Roteamento de Veículos Capacitados com Demandas Estocásticas (*Capacitated Vehicle Routing Problem with Stochastic Demands* - CVRPSD). Nele, os clientes são conhecidos e é considerado que as demandas são estocásticas e a demanda total coletada ao longo da rota, não pode exceder a capacidade do veículo. Em alguns casos, o veículo pode ser incapaz de carregar a demanda do cliente, mesmo que a demanda esperada ao longo da rota não exceda a capacidade do veículo. Tal situação é chamada de falha, e para resolver isto é feita uma viagem de retorno ao depósito para descarregar e, em seguida, o veículo retoma sua viagem conforme planejado originalmente. O CVRPD consiste então em minimizar o custo total das rotas planejadas e das falhas esperadas. Este artigo estuda uma implementação do *Integer L-Shaped Method* para a solução exata deste problema. Os experimentos computacionais indicaram que algumas instâncias com no máximo 100 clientes e entre 2 e 4 veículos são resolvidos na otimalidade quando a demanda segue a distribuição de *Poisson* ou distribuição normal.

O Problema de Roteamento de Veículos com Demanda Estocástica e Janelas de Tempo *(The Capacitated Vehicle Routing Problem with Stochastic Demands and Time Windows* - CVRPSDTW) é estudado por Lei et al. [20]. Este problema é modelado como um programa estocástico com recurso. Os autores propõem uma abordagem heurística através do *adaptive large neighborhood search*. Lei et al. [20] supõem que a demanda nunca é excedida, podendo fazer viagens de volta ao depósito e clientes com janelas de tempo violadas são servidas por veículos separados. São resolvidas instâncias com no máximo 50 clientes.

Em Martinez [8], é estudado o Problema de Roteamento de Veículos com Janelas de Tempo e Múltiplos Entregadores com incerteza. Neste problema, além das decisões de roteamento e sequenciamento, também é decidido a atribuição da alocação dos entregadores a cada uma das rotas, visando reduzir os tempos de serviços e custos. Martinez [8] utiliza diversos parâmetros incertos, sendo eles: a demanda dos clientes, tempos de viagens e tempos de serviços. Neste trabalho o problema é abordado por diferentes abordagens, propondo novos modelos de otimização robusta, que incorporam medidas para lidar efetivamente com o *trade-off* entre o custo e risco, e também novos modelos de programação estocástica de dois estágios com recurso.

O Problema de Roteamento de Veículos com Clientes Probabilísticos (*Vehicle Routing Problem with Probabilistic Customers* - VRPPC) é abordado por Lagos et al. [16]. Nesse trabalho, é proposto um *framework* de geração de colunas para resolver o VRPPC. É apresentado dois novos algoritmos, um que aproxima do custo esperado de uma solução e outro que usa seu custo esperado exato. São resolvidas instâncias com no máximo 40 clientes, onde os algoritmos foram capazes de resolver o problema com o GAP de até 10%.

Capítulo 2

Conceitos preliminares

Neste capítulo, introduzimos os principais conceitos utilizados no texto.

Na Teoria da Computação um algoritmo é chamado de eficiente se resolve um determinado problema em tempo polinomial em relação ao tamanho de sua entrada. Um problema é chamado de *Problema de Decisão* quando a solução para uma instância consiste em responder "sim" ou "não". A classe de todos os problemas de decisão que podem ser resolvidos por um algoritmo em tempo polinomial é chamada de P e chamamos de NP a classe de problemas de decisão que são resolvidos por algoritmos não-determinísticos em tempo polinomial. Uma grande questão em aberto da Teoria da Computação é determinar se P = NP ou não.

Chamamos de NP-difíceis os problemas que são pelo menos tão difíceis quanto qualquer problema em NP, considerando reduções de tempo polinomiais. Um caso particular do VRP é o TSP, que é NP-difícil. Com isso, o VRP também é NP-difícil e não existem algoritmos para resolvê-los de forma eficiente, a menos que P = NP [23].

Neste projeto, desejamos desenvolver novos algoritmos exatos para o Problema de Roteamento de Veículos com Janelas de Tempo e Clientes Estocásticos. A seguir apresentamos as abordagens de pesquisa consideradas.

2.1 Algoritmos Exatos

Nesta seção, relembramos alguns conceitos básicos de algoritmos exatos.

No contexto de algoritmos exatos, procuramos desenvolver algoritmos para encontrar uma solução ótima. Em problemas de otimização combinatória existe, em geral, um grande número de soluções viáveis e enumerá-las pode tomar muito tempo. Ao longo das últimas décadas, técnicas foram desenvolvidas para acelerar o processo de busca por uma solução ótima.

Programação linear busca encontrar valores ótimos de funções lineares de uma série de variáveis sujeitas a restrições lineares nestas variáveis [24]. Esta forma de otimização se torna interessante pois ela é tratável, em um tempo computacional razoável, e pode representar diversas situações próximas a realidade. Matematicamente definimos um programa

linear da seguinte forma:

$$\begin{aligned} &\min\ c^T x \\ &\text{s.t. } Ax = b, \\ &\quad x \geq 0, \end{aligned} \tag{2.1}$$

onde o vetor x representa um conjunto de variáveis que serão otimizadas, de acordo com o seu coeficiente c, respeitando um conjunto de restrições definidas pela igualdade $Ax = b$, onde A é a matriz de recursos e b o vetor limitante de recursos.

Um programa linear é considerado viável se existe alguma atribuição de valores para as variáveis, de tal forma que todas as restrições são satisfeitas simultaneamente. O conjunto de valores no qual todas as restrições são satisfeitas é chamado de região factível.

Uma das principais abordagens para projetar algoritmos exatos é a Programação Linear Inteira (PLI), em que o problema primeiramente é formulado usando Programação Linear com a restrição que todas as variáveis assumam valores inteiros. Quando o problema é formulado com Programação Linear onde existe pelo menos uma variável que assume valores inteiros e também existe pelo menos uma variável que assume valores contínuos temos então um Programa Linear Inteiro Misto [1]. Para a resolução de um PLI podemos usar métodos como *Branch and Bound*, *Branch and Price* e *Branch and Cut* entre outros.

Algoritmos exatos também podem ser baseados em programação dinâmica e enumeração (usando o método *Branch and Bound*, por exemplo), entre outros métodos. O método *Branch and Bound* consiste na enumeração das soluções candidatas, descartando os subconjuntos de soluções infrutíferas, usando cálculos de limitantes superiores e inferiores. Já a programação dinâmica pode ser aplicada quando a solução ótima de um problema pode ser calculada a partir de soluções para problemas menores e possui sobreposição de subproblemas, que sobrepostos compõem o problema original.

2.2 Otimização estocástica

Nesta seção, introduzimos os principais conceitos de otimização estocástica.

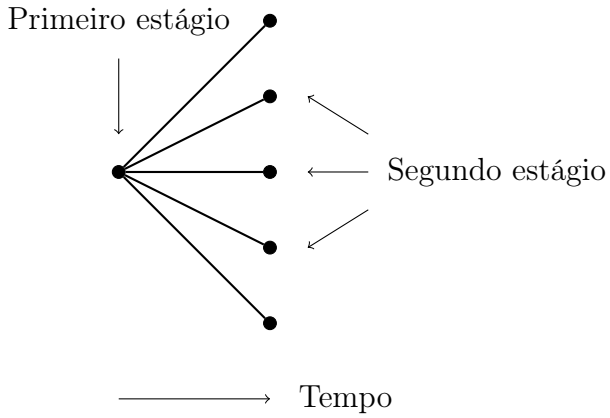
Programação estocástica é uma abordagem para a modelagem de problemas de otimização que envolvem incertezas [27]. Na formulação de problemas determinísticos, usualmente considera-se que todos os dados de entrada são previamente conhecidos. Porém, problemas do mundo real frequentemente incluem parâmetros incertos, e decisões devem ser tomadas antes do conhecimento total dos dados.

Nesta dissertação, nós estamos interessados em resolver problemas de otimização estocástica. Estes são programas lineares multiestágio que lidam com a incerteza em algum nível durante a sua resolução. Por programas lineares multiestágio, queremos dizer problemas no qual uma decisão inicial é tomada, mais informações se tornam conhecidas e em seguida novas decisões são tomadas. Se o programa a cada estágio é um problema de programação linear, então temos um problema multiestágio. Se houver alguma incerteza sobre a forma como a informação pode evoluir de um estágio para o outro e isso for le-

vado em consideração na tomada de decisão, o problema é conhecido como problema de programação estocástica (Murphy [24]).

Considere um problema estocástico em dois estágios. Aqui, uma decisão inicial é feita, o tempo avança levando o universo a um novo estado, neste novo estado, novas informações são reveladas e então uma nova decisão é tomada, conhecida como decisão de correção. Todas as decisões tomadas no segundo estágio, com o conhecimento de novas informações, irão afetar a decisão inicial tomada no primeiro estágio. Um exemplo de problema estocástico em dois estágios pode ser visto na Figura 2.1. Aqui podemos ver que após a decisão tomada no primeiro estágio, todos os possíveis cenários são explorados através do segundo estágio. Cada cenário leva a um diferente problema, com diferentes realizações da incerteza. Todos os problemas do segundo estágio são resolvidos separadamente e suas decisões irão "corrigir" as decisões do primeiro estágio.

Figura 2.1: Exemplo de problema estocástico em dois estágios



Diversos parâmetros de um problema do mundo real podem ser considerados incertos, como custos com combustíveis, demandas dos clientes, etc. Estes eventos podem ser representados através de variáveis aleatórias. Um evento aleatório incerto é denotado por ω e o conjunto que define todos os eventos incertos é representado por Ω. Para a representação de modelos de otimização estocástica, é considerado uma determinada descrição probabilística dos eventos incertos, através das variáveis aleatórias. Uma representação usual dos componentes dos parâmetros incertos é através de um vetor ξ, cujo valor só é conhecido após a realização dos eventos incertos.

Para a resolução de problemas de otimização estocástica em dois estágios, o vetor x é comumente associado à representação das tomadas de decisão do primeiro estágio. Após a tomada de decisão, um evento aleatório ω ocorre afetando as escolhas tomadas. Para o segundo estágio temos um espaço amostral Ω. Cada evento que pode ser realizado neste estágio corresponde a um cenário $\omega \in \Omega$. No segundo estágio, é tomada uma decisão corretiva, representada através do vetor $y(\omega)$, para compensar os efeitos das realizações dos eventos incertos sobre as decisões do primeiro estágio. A sequência de eventos pode ser resumida como:

$$x \to \xi(\omega) \to y(\omega).$$

Baseado no modelo de Dantzig [6], Birge e Louveaux [3] descrevem o programa determinístico equivalente ao modelo de programação estocástica com dois estágios representado como:

$$\begin{aligned} &\min\ c^T x + \mathscr{Q}(x) \\ &\text{s.t. } Ax = b, \\ &\qquad x \geq 0, \end{aligned} \tag{2.2}$$

onde as decisões do primeiro estágio são representadas pelo vetor x, $c^T x$ fornece os termos da função objetivo apenas do primeiro estágio onde os dados são conhecidos, $\mathscr{Q}(x)$ conecta o problema inicial com os problemas do segundo estágio e representa os termos da função objetivo das decisões de correções do segundo estágio. No segundo estágio, temos a oportunidade de tomar decisões, conhecendo as saídas dos eventos incertos para cada cenário. Assim as decisões são feitas de acordo com cada cenário, corrigindo as decisões do primeiro estágio. Com isso, podemos definir $\mathscr{Q}(x)$ pelo valor esperado do segundo estágio:

$$\mathscr{Q}(x) = E_\xi Q(x, \xi(\omega)), \tag{2.3}$$

onde $Q(x, \xi(\omega))$ representa o valor da função objetivo do segundo estágio, dado um determinado x e a realização do cenário ω no vetor de incertezas ξ. Assim o cálculo do valor esperado é realizado através do somatório dos valores da função objetivo para cada cenário possível no segundo estágio, multiplicado pela probabilidade do seu respectivo cenário, onde o valor de um cenário ω, pode ser definido como:

$$Q(x, \xi(\omega)) = \min_y \{q(\omega)^T y(\omega) \mid Wy(\omega) = h(\omega) - T(\omega)x, y \geq 0, \} \tag{2.4}$$

onde y representa o vetor de variáveis do segundo estágio e temos a matriz W relacionada aos recursos fixos do segundo estágio. Para uma determinada realização ω, se tornam conhecidos os vetores com as informações do segundo estágio $q(\omega)^T$ e $h(\omega)$, relacionados aos coeficientes dos custos, limitantes de recursos; e a matriz relacionada aos recursos do primeiro estágio $T(\omega)$ que podem afetar os recursos do segundo estágio.

Assim, Birge e Louveaux [3] descrevem uma formulação clássica de programação estocástica em dois estágios como:

$$\min_x c^T x + E_\xi [\min_{y(\omega)} q(\omega)^T y(\omega)] \tag{2.5}$$

$$\text{s.t. } Ax = b, \tag{2.6}$$

$$T(\omega)x + Wy(\omega) = h(\omega), \qquad \forall \omega \in \Omega, \tag{2.7}$$

$$x \geq 0, y(\omega) \geq 0, \tag{2.8}$$

Seja ξ o vetor formado por todos os componentes de q^T, h^T e T; e E_ξ o valor esperado dos custos no segundo estágio. As decisões do segundo estágio $y(\omega)$ devem ser tomadas afim de corrigir os efeitos das realizações na solução do primeiro estágio. A função objetivo

em (2.5) é composta por um termo determinístico $c^T x$ e o valor esperado da função objetivo do segundo estágio $q(\omega)^T y(\omega)$ sobre todas as realizações dos eventos aleatórios ω. O termo do segundo estágio é o mais difícil porque, para cada realização de ω, o valor de $y(\omega)$ é a solução de um programa linear.

Calcular o valor do segundo estágio para todos os cenários, junto com o *problema mestre*, pode ser muito custoso dependendo da aplicação. Uma abordagem para calcular esse valor é a aproximação do custo do segundo estágio, representado por θ, então Birge e Louveaux [3] descrevem algumas possíveis abordagens para a aproximação dos problemas de multiestágio, tais como:

1. *Aproximação da Função objetivo*: trocando o $\mathscr{Q}(x)$ com alguma representação simplificada, tal como linearização interna e externa;

2. *Relaxação e dualização de restrições*: relaxando restrições em abordagem Lagrangiana ou olhando na forma dual que pode não ser implementável, mas pode dar limitantes;

3. *Restrição de política*: restringindo o conjunto de alternativas de ações a serem tomadas, de forma a permitir computar os estágios de forma eficiente;

4. *Agregação de caminho, tempo e estado ou geração e redução de cenário*: iniciando com um grande conjunto de possibilidade e então combinando (ou selecionando) eles para formar representações mais tratáveis;

5. *Métodos de Monte Carlo*: amostragem para obter representações menores e mais tratáveis.

A resolução de problemas estocásticos onde os recursos do segundo estágio são fixos e possui um número finito de realizações, tem uma ênfase na primeira abordagem, através da utilização de uma técnica chamada *L-Shaped Method* [3]. Para o modelo geral é escolhido alguma decisão inicial que miniminiza o custo atual mais o valor esperado de futuras ações corretivas. Quando existe um número finito de possíveis realizações no segundo estágio e com todas as funções sendo lineares, é sempre possível transformar o modelo de otimização estocástica, na forma determinística equivalente ou forma extensiva. O *L-Shaped method* é baseado na construção de uma linearização externa da função de custo da correção e utiliza esta linearização com uma solução do problema de primeiro estágio.

2.2.1 *L-Shaped Method*

Nesta subseção, explicamos a ideia básica do *L-Shaped Method*, assim como algum dos seus conceitos.

Considerando a formulação (2.2), Birge e Louveaux [3] demonstram que a ideia básica do *L-Shaped method* é aproximar o termo não linear da função objetivo, pois este termo, o custo esperado do segundo estágio, envolve a solução de todas as possíveis realizações de ξ do programa linear do segundo estágio. Portanto é construído um *problema mestre* em x, utilizando somente as informações correspondentes ao primeiro estágio, e só é resolvido

a função de correção para cada subproblema. O *L-Shaped method* também é conhecido como Decomposição de Benders aplicado a problemas de otimização estocástica em dois estágios (Murphy [24]).

Para tornar essa abordagem possível, é suposto que o espaço amostral Ω é finito. Seja $\mathscr{K} = \{1, \ldots, |\Omega|\}$, o conjunto de índices das possíveis realizações $\omega \in \Omega$, p_k a probabilidade de ocorrer o cenário $k \in \mathscr{K}$, q_k^T o coeficiente do custo para corrigir a solução no cenário k, T_k a matriz relacionada aos recursos do primeiro estágio e W a matriz relacionada aos recursos do segundo estágio. A partir dessa suposição, podemos agora escrever o programa determinístico equivalente na forma extensiva como

$$\begin{aligned} & \min c^T x + \sum_{k \in \mathscr{K}} p_k q_k^T y_k \\ \text{s.t.} \quad & Ax = b, && \\ & T_k x + W y_k = h_k, && \forall k \in \mathscr{K}, \\ & x \geq 0, y_k \geq 0, && \forall k \in \mathscr{K}. \end{aligned} \tag{2.9}$$

Da mesma forma que a versão com variáveis contínuas, problemas de Programação Linear Inteira Estocásticos também são modelados em dois estágios. No primeiro estágio a decisão deve ser tomada sem o conhecimento completo da informação de algumas variáveis aleatórias. Quando os valores dos eventos aleatórios são conhecidos, ações corretivas são tomadas no segundo estágio para garantir a viabilidade da solução [17].

O primeiro passo para a resolução do *L-Shaped method* é chegar a uma estimativa inicial para as variáveis x, do primeiro estágio. Esse valor de x é utilizado como valor fixo e resolvemos cada segundo estágio de acordo com o seu respectivo cenário e os valores das variáveis do primeiro estágio. Dado isso, existem duas possibilidades: Ou algum segundo estágio é inviável ou todos são viáveis. No primeiro caso, é necessário adicionar cortes de viabilidade, e no segundo caso é necessário obter cortes de otimalidade [24] para que a solução convirja para a solução ótima.

Se um dos problemas do segundo estágio, denotado por $Q(x, \xi(\omega))$, for inviável, isso significa que o valor de x é inviável para o problema completo. Então queremos utilizar essa informação para adicionar restrições de volta ao primeiro estágio, para evitar essa solução inviável. Tais restrições são conhecidas como cortes de viabilidade. No problema estudado nesta dissertação, pelo modelo de programação estocástica em dois estágios definido na seção 3, quando o segundo estágio é inviável, a solução obtida apresenta um valor muito negativo, e com isso é possível identificar a sua inviabilidade. Neste caso, não foi necessário gerar cortes de viabilidade para o problema. Para o leitor interessado neste tópico, é recomendado o tutorial de Murphy [24].

Após a resolução dos problemas do segundo estágio, precisamos alcançar uma solução ótima. Para isto iremos gerar cortes de otimalidade, para enviar informações de volta ao primeiro estágio. Existem diversas maneiras de encontrar essas desigualdades, que podem variar de acordo com os tipos de variáveis. Vamos abordar a estratégia que foi utilizada neste trabalho para a geração destes cortes, onde as variáveis do primeiro estágio são inteiras e binárias. Este método é conhecido como *Integer L-Shaped Method*.

Birge e Louveaux [3] afirmam que quando as variáveis do primeiro estágio são binárias, é possível derivar um corte de otimalidade específico, a fim de obter um algoritmo convergente baseado no procedimento de *branch and cut* padrão.

Para aplicar o *Integer L-Shaped Method* proposto por Laporte et al. [17] seguindo o modelo de Birge e Louveaux [3], são necessárias as seguintes condições: *(i)* as variáveis de decisão são binárias, *(ii)* para qualquer vetor solução viável x do primeiro estágio o valor $\mathscr{Q}(x)$ pode ser computado, e *(iii)* existe um valor finito $\mathscr{L}$ que é um limitante inferior para qualquer solução válida do segundo estágio, ou seja, $\mathscr{L} \leq \min_x\{\mathscr{Q}(x) \mid Ax = b, x \in \mathbb{B}\}$. Para a condição *(iii)*, não é necessário que o limitante $\mathscr{L}$ seja justo, apesar de ser desejado ter um valor mais justo possível.

Birge e Louveaux [3] propõem adicionar os cortes de otimalidade, gerando restrições válidas para fornecer um limitante inferior para o valor da função $\mathscr{Q}(x)$ e provam a sua validade.

Proposição 1: Seja x^v uma solução viável para o primeiro estágio (satisfaz $Ax^v = b$). Seja $S = \{i|x_i^v = 1\}$ o conjunto que representa a solução viável v para o primeiro estágio. Seja θ^v o valor esperado do segundo estágio para x^v. O seguinte corte de otimalidade

$$\theta \geq (\theta^v - \mathscr{L})\Big(\sum_{i\in S} x_i - \sum_{i\notin S} x_i\Big) - (\theta^v - \mathscr{L})(|S| - 1) + \mathscr{L} \tag{2.10}$$

é valido.

Prova. Para perceber a validade do corte, note que quando $(\sum_{i\in S} x_i - \sum_{i\notin S} x_i) = |S|$ então as variáveis assumem exatamente o valor da solução x^v, e o lado direito da inequação se reduz ao valor θ^v, e a restrição $\theta \geq \theta^v$ é valida pela definição de θ^v. Por outro lado $(\sum_{i\in S} x_i - \sum_{i\notin S} x_i)$ é sempre menor ou igual a $|S|$. Caso seja estritamente menor, isto implica que o lado direito da inequação se reduzirá a um valor menor ou igual a $\mathscr{L}$, que é um limitante inferior válido para qualquer solução x do primeiro estágio. □

A ideia desta desigualdade, é que ou a solução viável representada no conjunto S irá limitar o valor de θ, onde a atual solução deve ser mantida, ou a desigualdade se torna $\theta \leq \mathscr{L}$ que é uma restrição válida.

Foi proposto um procedimento para o *Integer L-Shapped method* por Laporte e Louveaux [17], e será apresentado uma versão simplificada dos passos deste método, onde os cortes de viabilidade podem ser adicionados no passo 3, caso seja necessário.

Passo 0: Inicialize $s = v = 0$, $\overline{z} = \infty$. O valor de θ é definido como $-\infty$ ou um limitante inferior válido $\mathscr{L}$. Crie uma lista que contém um nó pendente com o problema inicial.
Passo 1: Altere $v = v + 1$. Se a lista estiver vazia, pare. Caso contrário, selecione um nó pendente na lista como problema atual.
Passo 2: Resolva o problema atual. Se o problema atual não tem solução viável, remova o nó atual e vá para o passo 1. Caso contrário, seja (x^v, θ^v) uma solução ótima para o problema atual.
Passo 3: Se $cx > \overline{z}$, remova o nó atual e vá para o passo 1.

Passo 4: Verifique as restrições de integralidade. Se a restrição for violada, crie dois nós seguindo o procedimento padrão de *branch and cut*. Inclua esses dois nós na lista de nós pendentes e vá para o passo 1.
Passo 5: Calcule $\mathscr{Q}(x^v)$ e $z^v = c^T x + \theta^v$. Se $z^v \leq \overline{z}$, atualize $\overline{z} = z^v$.
Passo 6: Se $\theta^v \geq \theta$, então remova o nó atual e retorne ao passo 1. Caso contrário, insira o corte de otimalidade de acordo com (2.10), altere $s = s + 1$ e retorne ao passo 2.

Laporte e Louveaux [17], descrevem um método de Otimização Estocástica que pode ser utilizado para abordar diversos problemas de programação linear inteira. Este procedimento será utilizado para resolver o Problema de Roteamento de Veículos Capacitados com Janelas de Tempo e Clientes Estocásticos.

A seguir é apresentado um exemplo com valores fictícios, de como seria a solução de um problema de otimização estocástica em dois estágios através do *Integer L-Shaped Method*. Iremos supor que após a solução do problema mestre pela primeira vez, obtemos o valor de $c^T x^1$ igual a 90. O segundo estágio é resolvido por todos os cenários com o valor de x^1 fixo de acordo com a solução obtida no problema mestre. Após a solução do segundo estágio, é obtido que $\mathscr{Q}(x^1)$ é 15. Então é inserido o corte de otimalidade $\theta \geq 15$, de acordo com (2.10) quando $x = x^1$, assim obtendo como valor total da solução 115. Agora o problema mestre é resolvido novamente, com a nova restrição inserida e é obtido uma nova solução para o problema mestre, tal que $c^T x^2 = 92$. Com essa nova solução, é obtido após a resolução do segundo estágio que $\mathscr{Q}(x^2) = 10$ e é inserido o corte de otimalidade $\theta \geq 10$, totalizando com custo total de 112. Após a inserção do corte, o problema mestre é resolvido novamente com esse corte, e a solução obtida é uma solução já conhecida, que tem $c^T x^3 = 92$ e o valor do segundo estágio $\mathscr{Q}(x^3) = 10$ e obtemos a solução ótima.

- Resolver o problema mestre;
- $c^T x^1 = 90$;
  - resolver o segundo estágio com x^1;
  - $\mathscr{Q}(x^1) = 15$;
  - inserir corte de otimalidade $\theta \geq 15$ quando $x = x^1$.
- Resolver o problema mestre atual;
- $c^T x^2 = 92$;
  - resolver o segundo estágio com x^2;
  - $\mathscr{Q}(x^2) = 10$;
  - inserir corte de otimalidade $\theta \geq 10$ quando $x = x^2$.
- Resolver o problema mestre atual;
- $c^T x^3 = 92$;
  - solução já conhecida;

- $\mathscr{Q}(x^3) = 10$;
- ótimo.

Na Figura 2.2, podemos ver um exemplo visual simples de como funciona o problema estocástico em dois estágios. Nesta figura, temos um grafo com 3 tipos diferentes de vértices, o vértice 0 que representa o depósito, os vértices pretos e brancos que caracterizam os clientes certos e os clientes incertos, respectivamente. Na Figura 2.2.a, podemos ver como seria uma possível solução do primeiro estágio. Nela é realizado uma rota entre todos os clientes certos. Para simplificar o exemplo, foi simulado a resolução de somente um cenário, que ocorre na Figura 2.2.b onde ambos os clientes incertos realizam o pedido neste cenário. Esta figura representa a solução do segundo estágio do problema. Nele é resolvido um novo problema onde os clientes certos estão fixos na rota e também a ordem de atendimento destes clientes, já estão definidas, então são tomadas decisões sobre como atender os clientes incertos, que podem ser alocados nas rotas entre dois clientes quaisquer (certos ou incertos). Após a resolução de todos os problemas de segundo estágio, o corte de otimalidade é inserido no problema mestre, e os custos extras da correção correspondente as soluções do segundo estágio vão ser adicionados à função objetivo do primeiro estágio através da variável θ. Vale ressaltar que é resolvido um problema novo para cada possível cenário. Após a inserção do corte, o método continua a sua resolução do problema mestre, e uma nova solução com custo menor pode ser gerada, resultando na rota representada pela Figura 2.2.c. Este processo será repetido diversas vezes até a solução ótima ser encontrada.

Figura 2.2: Exemplo de problema estocástico em dois estágios

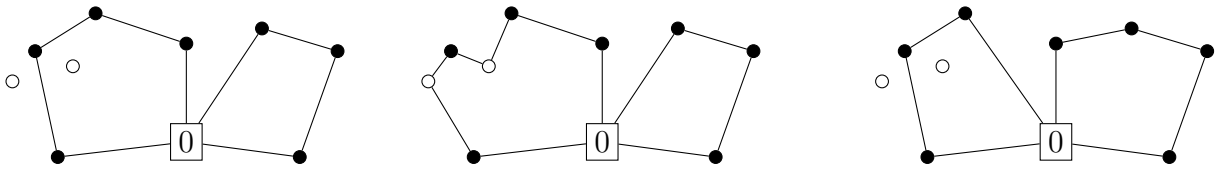


Figura a Figura b Figura c

Capítulo 3

Problema de Roteamento de Veículos Capacitados com Janelas de Tempo e Clientes Estocásticos

Neste capítulo, definimos formalmente o problema estudado neste trabalho e seu modelo em programação linear inteira.

3.1 Definição do Problema

Nesta seção, definimos formalmente o problema a ser investigado neste trabalho.

Para o Problema de Roteamento de Veículos Capacitados com Janela de Tempo e com Clientes Incertos (*Capacitated Vehicle Routing Problem with Time Windows and Stochastic Customers* - CVRPTWSC) está sendo proposto um modelo de Otimização Estocástica que será resolvido inicialmente através do *Integer L-Shaped method*. O modelo tem dois estágios, onde o primeiro estágio é responsável por fazer uma rota para os clientes certos, respeitando a capacidade do veículo e janelas de tempo. No segundo estágio será utilizado inclusão de custos relativos a inserção de novos clientes incertos.

Para definir esse problema, seja $G = (V, A)$ um digrafo simétrico completo, onde cada vértice $i \in V$ representa uma localidade e cada arco $(i, j) \in A$ representa um caminho entre as duas localidades. O conjunto V é particionado em três subconjuntos disjuntos S, D, e I, onde $S \cup D \cup I = V$. O conjunto S possui um único vértice s que representa o depósito, *i.e.*, a localização inicial e final das rotas de todos os veículos. Além disto, $D \subset V$ representa o conjunto dos clientes certos e $I \subset V$ representa o conjunto de clientes incertos e denota-se por V' o conjunto de clientes do problema, *i.e.*, $V' = D \cup I$. Cada cliente $i \in V'$ possui uma demanda $d_i \in \mathbb{R}_{>0}$ e uma janela de tempo $[a_i, b_i]$, indicando o período de atendimento do cliente i, sendo que $0 \leq a_i < b_i$, para todo $i \in V'$. O conjunto que define todos os eventos incertos é representado por Ω e um determinado evento (ou cenário) deste conjunto é denotado por ω. O conjunto $I(\omega)$ representa todos os clientes incertos do cenário ω. Assume-se que cada cliente incerto $i \in I(\omega)$ tem uma probabilidade $p_i > 0$ de realizar um pedido. Cada arco $(i, j) \in A$ é associado a um valor c_{ij} que representa o custo de deslocamento para ir da localidade i para a localidade

j, onde $i, j \in V$. Além disto, cada arco (i, j) também é associado a um valor t_{ij} que representa o tempo necessário para realizar o deslocamento entre as duas localidades $i, j \in V$. Assumimos que os arcos têm custos e tempos simétricos, ou seja, $c_{ij} = c_{ji}$ e $t_{ij} = t_{ji}$ para quaisquer par de localidades $i, j \in V$. Seja A_D os arcos do grafo G entre clientes com pedidos certos, ou seja, $A_D = D \times D$. Um conjunto K de veículos pode ser utilizado, sendo que todo veículo $k \in K$ possui uma mesma capacidade máxima C. Os veículos têm permissão de sair do depósito após um determinado instante de tempo $E = 0$ e devem retornar até um segundo instante de tempo $L > 0$. Ressalta-se que $E < a_i, \forall\, i \in V'$, que $L > b_i, \forall\, i \in V'$ e que existe pelo menos uma partição de V' em $|K|$ partes, cada uma com demanda total no máximo C. Note que um cenário corresponde a um subconjunto de clientes incertos que devem ser atendidos. O conjunto D^k é definido como todos os clientes certos contidos na rota k. Definimos ainda $S^k = D^k \cup I(\omega)$, ou seja, é a união de todos os clientes incertos de um determinado cenário com todos os clientes certos da rota k. Também é definido $A^k = S^k \times S^k$, que são todos os arcos entre os clientes em S^k. Uma solução para o CVRPTWSC consiste em um conjunto de $|K|$ rotas de tal forma que cada cliente $i \in D$ é atendido por exatamente um veículo $k \in K$ respeitando sua janela de tempo. Além disto, cada rota deve respeitar a capacidade máxima C do veículo. O objetivo é encontrar o conjunto de rotas com o menor custo esperado possível, considerando os custos do primeiro e do segundo estágios.

3.2 Modelo PLI para o CVRPTWSC

Nesta seção é detalhado o modelo PLI para o CVRPTWSC.

Baseado em uma formulação do CVRPTW de Toth e Vigo [30], definimos uma formulação PLI para o CVRPTWSC pelas equações (3.1)–(3.24). Esta formulação representa um problema de otimização estocástica em dois estágios. A formulação utiliza um vértice artificial s' que também representa o depósito. Desta forma o conjunto de arcos A, também contém um arco de cada vértice para este nó artificial s', tal que $c_{is'} = c_{si}$ e $t_{is'} = t_{si}$ para todo vértice $i \in V'$. O primeiro estágio utiliza variáveis de decisão $x_{ij}^k \in \mathbb{B} : (i, j) \in A_D, k \in K$ tal que $x_{ij}^k = 1$ se a rota do veículo k passa pelo arco (i, j) e $x_{ij}^k = 0$ caso contrário. Além disso, variáveis auxiliares $w_i^k \in \mathbb{R}_{\geq 0} : i \in V \cup \{s'\} \setminus I, k \in K$ representam o instante de tempo que o veículo k atende um cliente i ou o tempo em que ele chega no depósito. O modelo do primeiro estágio é definido pela função objetivo descrita em (3.1) e pelas restrições em (3.2)–(3.11), onde $\delta^+(i)$ representa o conjunto de arcos que saem do vértice i e $\delta^-(i)$ é o conjunto dos arcos que chegam ao vértice i.

$$\min \sum_{k \in K} \sum_{(i,j) \in A_D} c_{ij} x_{ij}^k + \mathscr{Q}(x), \tag{3.1}$$

$$\text{s.t.} \sum_{k \in K} \sum_{j \in \delta^+(i)} x_{ij}^k = 1, \qquad \forall i \in D, \tag{3.2}$$

$$\sum_{j \in \delta^+(s)} x_{sj}^k = 1, \qquad \forall k \in K, \tag{3.3}$$

$$\sum_{i \in \delta^-(s')} x_{is'}^k = 1, \qquad \forall k \in K, \tag{3.4}$$

$$\sum_{i \in \delta^-(j)} x_{ij}^k - \sum_{i \in \delta^+(j)} x_{ji}^k = 0, \qquad \forall j \in D, \forall k \in K, \tag{3.5}$$

$$w_i^k + s_i + t_{ij} - w_j^k \leq (1 - x_{ij}^k)M, \qquad \forall k \in K, (i,j) \in A_d, \tag{3.6}$$

$$a_i \sum_{j \in \delta^-(i)} x_{ji}^k \leq w_i^k \leq b_i \sum_{j \in \delta^+(i)} x_{ij}^k, \qquad \forall k \in K, \forall i \in D, \tag{3.7}$$

$$E \leq w_i^k \leq L \qquad \forall i \in D \cup \{s'\}, k \in K, \tag{3.8}$$

$$\sum_{i \in D} d_i \sum_{j \in \delta^+(i)} x_{ij}^k \leq C, \qquad \forall k \in K, \tag{3.9}$$

$$x_{ij}^k \in \{0,1\}, \qquad \forall (i,j) \in A_d, \forall k \in K, \tag{3.10}$$

$$w_i^k \geq 0, \qquad \forall i \in D \cup \{s'\} \setminus I, k \in K. \tag{3.11}$$

A função objetivo descrita em (3.1) minimiza o custo esperado das rotas. Temos duas parcelas, uma se refere ao custo do primeiro estágio, que corresponde ao custo das rotas para atender os clientes certos e a segunda parcela $\mathscr{Q}(x)$, representa o valor esperado a ser adicionado aos custos destas rotas considerando o segundo estágio. As restrições (3.2) garantem que cada cliente certo seja atendido por exatamente um veículo. As restrições (3.3), (3.4) e (3.5) são as clássicas restrições de conservação de multi-fluxo e garantem a construção da rota para cada veículo saindo de s e terminando em s'. As restrições (3.6) garantem que, caso um arco (i,j) seja utilizado na rota k, então o tempo de início de atendimento do cliente j pelo veículo k seja igual ou superior ao tempo de início do atendimento do cliente i somado ao tempo de serviço do cliente i e o tempo de deslocamento do arco (i,j). As restrições descritas em (3.7) garantem que todo cliente seja atendido dentro de sua janela de tempo. As restrições apresentadas em (3.8) certificam que todos os veículos saiam após o tempo inicial E e retornem antes do tempo máximo L. A capacidade máxima dos veículos é garantida pelas restrições (3.9). Os domínios das variáveis x_{ij}^k e w_i^k são respectivamente definidos pelas equações (3.10) e (3.11).

Dada uma solução x do primeiro estágio e uma realização ω de clientes incertos, construímos uma nova solução no segundo estágio onde garantimos que os clientes certos permanecem na mesma rota em que estavam no primeiro estágio, e que são visitados na mesma ordem em que estavam nesta rota. Clientes incertos realizados em ω (denotado por $I(\omega)$) devem ser inseridos nestas rotas, mas mantendo-se a viabilidade de cada rota.

O modelo PLI do segundo estágio é bastante semelhante ao do primeiro estágio exceto

pela inclusão de restrições que garantem que os clientes certos continuem nas mesmas rotas e são visitados na mesma ordem original. Seja D^k o conjunto de clientes certos que no primeiro estágio estão na rota k e seja A^k o conjunto de arcos entre quaisquer vértices do conjunto S^k. Temos variáveis de decisão $y_{ij}^k \in \mathbb{B} : (i,j) \in A^k, k \in K$ tal que $y_{ij}^k = 1$ se o cliente $j \in S^k$ é visitado após o cliente $i \in S^k$ na rota do veículo k e $y_{ij}^k = 0$ caso contrário. Além disso, variáveis $\tilde{w}_i^k \in \mathbb{R}_{\geq 0} : i \in D^k, k \in K$ indicam instante de tempo que o veículo k atende o cliente $i \in S^k$. O modelo do segundo estágio é definido pela função objetivo descrita em (3.12) e pelas restrições em (3.13)–(3.24).

$$\min \sum_{k \in K} \sum_{(i,j) \in A} c_{ij} y_{ij}^k - \mathscr{F}(x), \tag{3.12}$$

$$\text{s.t.} \sum_{k \in K} \sum_{j \in \delta^+(i)} y_{ij}^k = 1, \qquad \forall i \in D \cup I(\omega), \tag{3.13}$$

$$\sum_{j \in \delta^+(s)} y_{sj}^k = 1, \qquad \forall k \in K, \tag{3.14}$$

$$\sum_{i \in \delta^-(s')} y_{is'}^k = 1, \qquad \forall k \in K, \tag{3.15}$$

$$\sum_{i \in \delta^-(j)} y_{ij}^k - \sum_{i \in \delta^+(j)} y_{ji}^k = 0, \qquad \forall k \in K, \forall j \in S^k \tag{3.16}$$

$$\tilde{w}_i^k + s_i + t_{ij} - \tilde{w}_j^k \leq (1 - y_{ij}^k)M, \qquad \forall k \in K, (i,j) \in A^k, \tag{3.17}$$

$$a_i \sum_{j \in \delta^-(i)} y_{ji}^k \leq \tilde{w}_i^k \leq b_i \sum_{j \in \delta^+(i)} y_{ij}^k, \qquad \forall k \in K, \forall i \in D^k \cup I(\omega), \tag{3.18}$$

$$E \leq \tilde{w}_i^k \leq L \qquad k \in K, \forall i \in D^k \cup \{s'\} \cup I(\omega), \tag{3.19}$$

$$\sum_{i \in D} d_i \sum_{j \in \delta^+(i)} y_{ij}^k \leq C, \qquad \forall k \in K, \tag{3.20}$$

$$\sum_{j \in \delta^+(i)} y_{ij}^k = 1, \qquad \forall i \in D^k, \tag{3.21}$$

$$\tilde{w}_i^k \leq \tilde{w}_j^k \qquad \forall i,j \in D^k \text{ t.q. } w_i^k < w_j^k, \forall k \in K, \tag{3.22}$$

$$y_{ij}^k \in \{0,1\}, \qquad \forall (i,j) \in A, \forall k \in K, \tag{3.23}$$

$$\tilde{w}_i^k \geq 0, \qquad \forall i \in V \cup \{s'\} \setminus I, k \in K. \tag{3.24}$$

A função objetivo descrita por (3.12) minimiza o custo das rotas do segundo estágio menos o custo das rotas estabelecidas no primeiro estágio, dado por $\mathscr{F}(x)$. Desta forma, esta diferença corresponde exatamente ao incremento do custo das rotas do primeiro estágio ao se levar em consideração os clientes incertos do segundo estágio. As restrições (3.13)–(3.20) são essencialmente as restrições do modelo do primeiro estágio, exceto que agora incluímos também os clientes incertos $I(\omega)$. As restrições (3.21) garantem que se um cliente i estava na rota k no primeiro estágio então ele continuará na rota k no segundo estágio, enquanto as restrições (3.22) garantem que a ordem entre vértices certos no primeiro estágio será mantida nas rotas do segundo estágio.

Capítulo 4

Algoritmos para o CVRPTWSC

Neste capítulo, damos detalhes de como o *Integer L-Shaped Method* foi implementado, assim como as variações de inserção de corte e uso de algoritmos de plano de corte.

4.1 Variações de inserção de corte

Nesta seção, vamos detalhar como os cortes de otimalidade do *Integer L-Shaped Method*, apresentados em 2.10 na subseção 2.2.1, são gerados a partir do segundo estágio e foram inseridos no *problema mestre*.

4.1.1 Inserção 1

O procedimento de inserção 1, como explicado em [3], é indicado na seção 2.2.1. O problema mestre pode ser resolvido através de um programa linear inteiro. Em cada nó do *Branch and Bound* do *problema mestre* no qual é encontrado uma solução viável, é resolvido o segundo estágio, que consiste em um programa linear inteiro misto para cada um dos possíveis cenários. Através da resolução do segundo estágio, é obtido o corte de otimalidade que é inserido no *problema mestre* e o processo de *Branch and Bound* continua. Note que esse processo tem um potencial de gerar vários cortes de otimalidade a medida que o primeiro estágio é resolvido, podendo deixar o *problema mestre* com muitas restrições. Por outro lado, nesta abordagem, só é resolvido apenas um processo de *Branch and Bound* para o *problema mestre*.

4.1.2 Inserção 2

Outra abordagem para a inserção dos cortes, é semelhante ao detalhado na subseção 4.1.1. Porém, nesta variação, como é possível ver no Algoritmo 1, após a inicialização da resolução do *problema mestre*, este é resolvido até a otimalidade sem resolver o segundo estágio. Então é utilizada a solução ótima deste problema para resolver o segundo estágio e gerar os cortes de otimalidade. Após isto, uma única restrição é gerada e inserida no problema mestre que será resolvido novamente. Este processo é repetido até que não existam mais cortes para serem gerados. A ideia desta abordagem, é evitar a inserção de

muitos cortes no *problema mestre*.

Algoritmo 1: Inserção 2

```
1  r ← lowerbound;
2  r̄ ← −∞;
3  inicia problema mestre;
4  Enquanto |r − r̄| > 0 Faça
5  |  r ← resultado do problema mestre;
6  |  resolver segundo estágio;
7  |  gerar e inserir cortes de otimalidade;
8  |  resolver problema mestre;
9  |  r̄ ← resultado do problema mestre;
10 Fim
```

4.1.3 Inserção 3

A cada iteração do *Integer L-Shaped method*, resolver o *problema mestre* se torna cada vez mais difícil, pois durante as iterações são inseridos diversos cortes de otimalidade. No entanto, o processo de solução do *problema mestre* pode produzir mais informações que podem ser úteis. Uma modificação chamada de *L-Shaped* estendido, utilizando uma ideia de Fischetti el al. [10] consiste em: sempre que uma solução incumbente do *problema mestre* é encontrada, nós obtemos o corte de otimalidade correspondente, guardamos este corte em uma lista sem adicionar este corte ao *problema mestre*, e continuamos o processo. Todos cortes gerados neste processo são guardados e adicionados somente quando é encontrado a solução ótima para o *problema mestre*. Após encontrar a solução ótima do *problema mestre*, todos os cortes que estão guardados na lista, são gerados e inseridos ao problema mestre, então o problema mestre é resolvido novamente, agora com os novos cortes, até chegar a otimalidade. Este processo é repetido até que a condição de otimalidade seja atendida. Repare que a cada iteração, são inseridos diversos cortes. Esta abordagem visa inserir os cortes somente no final, para que não dificulte o processo de *Branch and Bound* do *problema mestre*.

Algoritmo 2: Inserção 3

```
1  r ← lowerbound;
2  r̄ ← −∞;
3  inicia problema mestre;
4  Enquanto |r − r̄| > 0 Faça
5  |  inicia a resolução do problema mestre;
6  |  c ← gerar cortes de otimalidade;
7  |  r ← resultado ótima do problema mestre atual;
8  |  inserir os cortes de otimalidade c;
9  |  resolver problema mestre;
10 |  r̄ ← resultado do novo problema mestre;
11 Fim
```

4.2 Cortes de Lysgaard

Nesta seção, descrevemos como as rotinas de geração de plano de cortes disponibilizada por Lysgaard et al. [21] foram utilizados no problema.

Durante a resolução do problema mestre, foi encontrado uma dificuldade em encontrar soluções viáveis. Uma alternativa que foi utilizada para acelerar este processo foi através dos algoritmos de plano de corte. Para isto, foi utilizado o pacote CVPSEP [21] disponibilizado por Lysgaard et al [21]. Esse pacote disponibiliza algumas rotinas de separação que geram cortes válidos para o CVRP. No nosso problema, estas rotinas foram utilizadas somente no primeiro estágio. Foram utilizadas 3 classes de cortes, sendo elas a *Capacity inequalities*, *Framed capacity inequalities* e *Strengthened comb inequalities* que estão descritas em Lysgaard et al. [22]. Estas classes de cortes são aplicadas em um problema de CVRP. Seja $E_d = D \times D$, o conjunto de arestas do grafo entre os clientes certos onde cada aresta e é formada por um par de nós e_1 e e_2. Para ser utilizado no *problema mestre*, que inicialmente corresponde a um CVRPTW, foi criado um conjunto de variáveis x_e para todo $e \in E_d$, onde $x_e = 1$ se a aresta é utilizada em uma rota e 0 caso contrário. Para assegurar a corretude desta variáveis, foi adicionado a restrição 4.1 para o problema mestre, esta restrição significa que aresta e tem o seu valor definido pelos os arcos (i, j) e (j, i). Assim, com estas alterações feitas no modelo, foi possível utilizar a biblioteca de Lysgaard et al. [21].

$$x_e = \sum_{k \in K} \sum_{i \in \delta^+(e_1)} \sum_{j \in \delta^+(e_2)} (x_{ij}^k + x_{ji}^k), \quad \forall e \in E_d. \tag{4.1}$$

A classe de corte *Capacity inequalities* é uma desigualdade bastante conhecida para o CVRP. Ela visa descobrir a quantidade mínima de veículos necessário para atender a demanda de um determinado conjunto de clientes. Separar as *Capacity inequalities* é conhecido por ser um problema NP-difícil [2]. Para evitar o alto custo computacional o pacote CVRPSEP utiliza de heurísticas para separar essa família de cortes. Lysgaard et al. [21] utilizaram um algoritmo de *Backtracking* para resolver a classe *Framed capacity inequalities*, que consiste em resolver o Problema de empacotamento para um determinado conjunto de clientes, pois, sabemos que alguns conjuntos de clientes não podem ficar na mesma rota, senão excederia a capacidade do veículo. Para evitar um grande custo na geração destes cortes, a árvore de decisão explorava no máximo 20000 nós. A classe *Strengthened comb inequalities* é um conjunto de desigualdade do TSP que provaram ser muito úteis em algoritmos de plano de corte [21].

Lysgaard et al. apresenta em [21] como as desigualdades válidas devem ser geradas. Dada uma solução fracionária das variáveis x_e do problema mestre, esta solução é passada para as rotinas de separação disponibilizada na biblioteca. Após a execução destas rotinas, a biblioteca de Lysgaard et al. retornava todas as desigualdades encontradas. O próximo passo era adicionar estas restrições ao problema mestre. Vale ressaltar após a realização de alguns testes preliminares, as rotinas de separação só foram chamadas nos 50 primeiros nós da árvore de *Branch and Bound* do problema mestre.

4.3 O *Integer L-Shaped Method* para o CVRPTWSC

Nesta seção, apresentamos como o *Integer L-Shaped Method* foi implementando para o problema proposto.

Neste trabalho, utilizamos o *Integer L-Shaped Method* proposto por [17] para resolver o CVRPTWSC. O objetivo do L-Shaped é fazer a aproximação do valor esperado das rotas no segundo estágio [3]. Este valor, descrito como $\mathscr{Q}(x)$ na Equação (3.1), é não linear, pois envolve o cômputo de um custo esperado que corresponde a encontrar a solução de todos os problemas de PLI do segundo estágio para cada realização ω possível. O *Integer L-Shaped Method* é construído como um *problema mestre*, definido pelas Equações (3.1)–(3.11), e um conjunto de *subproblemas auxiliares*, definidos pelas equações (3.12)–(3.24).

No corte de otimalidade (2.10) definido por Louveaux et al. [3], é utilizado um limitante inferior (*lower bound*) para a resolução do problema. Note que este limitante tem que ser válido para qualquer solução do problema e para todos os possíveis cenários. Devido a falta de conhecimento prévio de quais clientes irão realizar um pedido, um limitante simples foi gerado. O seu valor consiste em calcular o valor esperado dos cenários, no qual, para cada cliente incerto $v_i \in I$, são selecionados os dois menores arcos (i, j_1) e (i, j_2), tal que $j_1, j_2 \in V$ e $j_1 \neq j_2 \neq i$, e os valores dos arcos são somados e depois é dividido por 2. Note que não existe a restrição de que todos os arcos selecionados formem ciclos, ou rotas. Este limitante é válido pois o valor de qualquer solução válida, terá o custo maior ou igual a seleção das duas arestas de menor custo para cada cliente incerto v_i e tem o seu valor dividido por 2 para os casos de repetição de uma mesma aresta. Os outros termos da expressão 2.10, são formados por θ^v e $\sum_{i \in S} x_i - \sum_{i \notin S} x_i$ que são respectivamente os valores da v-esíma aproximação do termo não linear da função objetivo do segundo estágio e a diferença dos somatórios entre as variáveis que pertencem e as que não pertencem à solução como explicado na subseção 2.2.1.

Com isso, o *Integer L-Shaped Method* foi implementado neste trabalho, para a resolução do problema estudado. De acordo como descrito nas seções 4.1 e 4.2, primeiramente as instâncias foram alteradas para o CVRPTWSC. Depois foi iniciado o método de resolução, de acordo com a estratégia de inserção escolhida, e podendo inserir no *problema mestre*, as desigualdades geradas pela a biblioteca disponibiliza por Lysgaard et al. [21]. Todas as execuções do método utilizaram das mesmas instâncias e algoritmos de plano de corte, variando somente as estratégias de inserção de corte.

O segundo estágio consiste em resolver um programa linear inteiro misto para cada um dos cenários. Para a geração dos cenários, foram criadas todas as combinações possíveis envolvendo os clientes incertos. Assim, a medida que a quantidade de clientes incertos cresce, a quantidade de cenários cresce de forma exponencial. Vale ressaltar que cada programa linear inteiro misto correspondente a um cenário, e foi resolvido de forma independente, sem o compartilhamento de informações entre eles. Para o cálculo do valor esperado do segundo estágio, é calculada a probabilidade de cada cenário ocorrer, que é atribuído de acordo com a probabilidades dos clientes.

Capítulo 5

Experimentos Computacionais

Este capítulo apresenta como as instâncias de testes foram geradas e as condições que os experimentos foram feitos seguido pelas tabelas de resultado.

5.1 Entrada

Nesta seção, iremos detalhar como as instâncias foram obtidas e modificadas para serem usadas no CVRPTWSC.

Neste trabalho, foram utilizadas versões adaptadas das instâncias de CVRPTW de Solomon [1]. As instâncias são separadas em três classes: $R1$, $C1$ e $RC1$. Nas instâncias $R1$, o posicionamento dos clientes foi definido de forma aleatória com distribuição uniforme, nas instâncias $C1$ os clientes são organizados em pequenos *clusters* e as instâncias $RC1$ são formadas por uma mistura entre *clusters* de clientes com clientes posicionados de forma aleatória. Todos os três grupos de instâncias possuem um curto espaço de tempo e, desta forma, um mesmo veículo dificilmente consegue atender a mais de 10 clientes.

Foram selecionadas, de forma aleatória, 10 instâncias de cada classe para gerar versões adaptadas para CVRPTWSC, sendo 5 instâncias com 25 clientes e 5 instâncias com 50 clientes. Além disto, para cada instância, dentre os n clientes, foram selecionados alguns clientes para serem estocásticos. A quantidade de clientes estocásticos variou entre 2 e 5, sendo estes definidos aleatoriamente com probabilidade uniforme com valores entre 0.2 e 0.9. Para cada instância original, foram geradas 3 novas instâncias, mudando-se quais clientes eram sorteados para serem clientes estocásticos. Desta forma, este trabalho avaliou um total de 180 ocorrências. Vale a pena ressaltar que o número k de veículos utilizados em cada instância é fixo, sendo definidas através da instância original.

5.2 Resultados

Nesta seção, apresentamos as condições em que os experimentos foram executados, junto com os resultados e uma discussão sobre estes.

Os experimentos computacionais foram realizados em um computador com processador Xeon CPU E3-1230 V2 com 2.4 GHz de clock e 32 Gb de RAM, com sistema operacional

[1] http://www.bernabe.dorronsoro.es/vrp/index.html?/Problem_Instances/CVRPTWInstances.html

Linux Ubuntu 18.04. O algoritmo foi implementado utilizando `C++14` e compilados com o GNU `g++ 7.5`. Os modelo de PLI de ambos os estágios foram resolvidos utilizando o IBM/ILOG CPLEX[2] versão 12.10 com os parâmetros padrões. Um tempo de 3600 segundos foi utilizado como tempo máximo de resolução de cada instância.

Foi realizado um teste prévio para descobrir qual estratégia de inserção seria utilizada. Foram selecionadas 50 novas instâncias com 25 clientes, para descobrir qual dos três tipos de inserção explicados na seção 4.1, seria aplicado no conjunto geral de instâncias. Como podemos ver na Tabela 5.1, apesar da estratégia de inserção 1 e 2 ter um número de soluções ótimas encontradas semelhantes, o método de inserção 2 teve um tempo computacional menor ou próximo, na maioria das instâncias. Um fator que pode explicar esses resultados é que os cortes de otimalidade gerados foram de maior qualidade, fazendo com que fosse necessário inserir menos cortes. Outro ponto positivo da abordagem de inserção 2 foi o baixo tempo médio para a resolução do segundo estágio. Isso ocorre pois este método resolveu uma quantidade menor de problemas do segundo estágio. A estratégia de inserção 3 teve o maior custo computacional em grande maioria das instâncias e um número menor de soluções inteiras e ótimas encontradas. Uma explicação para este resultado é que este método insere uma grande quantidade de cortes de otimalidade no problema mestre, sendo alguns de baixa qualidade, o que pode ocasionar em uma lentidão para a resolução. Assim foi decidido aplicar o método de inserção 2, para o conjunto completo de entradas geradas.

Tabela 5.1: Comparação inserção

$\|I\|$	Método	Inteiro	Ótimo	gap (%)	cortes	t_1 (s)	t_2 (s)
2	1	46	33	8.3 ± 27.3	6.7 ± 7.7	835.0 ± 1614.2	3.5 ± 5.3
	2	39	38	7.2 ± 26.1	0.9 ± 0.3	959.3 ± 1798.5	0.4 ± 0.2
	3	39	25	7.6 ± 26.0	8.1 ± 7.6	2096.3 ± 3295.3	3.7 ± 4.2
5	1	41	33	6.8 ± 25.5	4.7 ± 6.6	504.8 ± 1066.9	46.6 ± 103.0
	2	41	41	4.7 ± 21.3	1.0 ± 0.2	202.9 ± 761.9	7.0 ± 4.6
	3	41	29	2.4 ± 15.4	6.5 ± 5.8	762.2 ± 1586.2	60.5 ± 92.0

Os resultados dos experimentos computacionais são reportados nas Tabelas 5.2, 5.3 e 5.4, representando respectivamente os resultados para as instâncias da classe $R1$, $C1$ e $RC1$. Em todas as tabelas, a primeira coluna apresenta o número total de clientes, enquanto a segunda coluna exibe o número de clientes incertos. A terceira e quarta colunas denotam, respectivamente, o número de instâncias para as quais o *L-Shaped method* conseguiu encontrar pelo menos uma solução inteira dentro do tempo limite de 3600 segundos e o número de soluções ótimas encontradas dentre deste mesmo limite de tempo. A quinta coluna representa o valor médio e o desvio padrão do *gap* de otimalidade encontrado pelo algoritmo, enquanto a sexta coluna exibe o número médio e o desvio padrão do número de cortes de otimalidade inseridos pelas execuções do segundo estágio. A sétima e oitava colunas representam a média e o desvio padrão do tempo computacional gastos no primeiro e no segundo estágio, respectivamente.

[2]https://www.ibm.com/products/ilog-cplex-optimization-studio

Tabela 5.2: Resultados do L-Shaped Method para as instâncias C1

$\|V'\|$	$\|I\|$	inteira	ótima	gap (%)	cortes	t_1 (s)	t_2 (s)
25	2	15	15	0.0 ± 0.0	1.0 ± 0.0	2.3 ± 2.1	0.3 ± 0.0
	5	14	14	6.7 ± 25.8	0.9 ± 0.3	241.1 ± 927.7	4.2 ± 1.6
50	2	12	12	20.0 ± 41.4	0.8 ± 0.4	88.1 ± 152.0	6.6 ± 3.4
	5	11	11	26.7 ± 45.8	0.7 ± 0.5	990.0 ± 1629.6	92.7 ± 57.9

Tabela 5.3: Resultados do L-Shaped Method para as instâncias R1

$\|V'\|$	$\|I\|$	inteira	ótima	gap (%)	cortes	t_1 (s)	t_2 (s)
25	2	10	10	9.1 ± 30.2	0.9 ± 0.3	1182.7 ± 1302.4	0.5 ± 0.3
	5	11	11	0.0 ± 0.0	1.0 ± 0.0	44.0 ± 66.5	9.1 ± 4.2
50	2	1	0	93.7 ± 23.5	0.1 ± 0.3	3962.9 ± 1350.6	1.4 ± 5.4
	5	1	0	93.7 ± 23.7	0.1 ± 0.3	3954.0 ± 1314.3	21.5 ± 80.3

Tabela 5.4: Resultados do L-Shaped Method para as instâncias RC1

$\|V'\|$	$\|I\|$	inteira	ótima	gap (%)	cortes	t_1 (s)	t_2 (s)
25	2	12	12	20.0 ± 41.4	0.8 ± 0.4	1336.2 ± 2054.2	0.2 ± 0.2
	5	13	13	13.3 ± 35.2	0.9 ± 0.4	490.1 ± 1260.5	5.3 ± 4.3
50	2	2	1	89.0 ± 32.1	0.1 ± 0.3	3874.9 ± 978.6	0.7 ± 2.1
	5	5	5	72.2 ± 46.1	0.3 ± 0.5	3520.2 ± 864.5	51.6 ± 93.1

Pode-se observar destas tabelas que o *Integer L-Shaped method* gastou um tempo maior executando o primeiro estágio (representado pela função objetivo (3.1) e pelas restrições em (3.2)–(3.11)) do que o segundo estágio (descrito pela função objetivo (3.12) e pelas restrições em (3.13)–(3.24)), sendo que o tempo médio gasto no segundo estágio nunca excedeu os 100 segundos. Este resultado pode ser explicado por dois fatos: (*i*) o número de clientes incertos do conjunto I é pequeno, o que faz com que poucas decisões devam ser tomadas no segundo estágio; e (*ii*) poucas execuções do segundo estágio são necessárias, como pode ser observado pelo pequeno número de cortes inseridos. Outro fato que podemos observar, foi a dificuldade de encontrar soluções inteiras. Em todas as classes, houve ao menos uma instância no qual não foi encontrado uma solução inteira. Somente na classe $C1$ com 2 clientes incertos, foram encontradas soluções inteiras para todas as instâncias. Isso reforça a dificuldade em resolver o primeiro estágio. Outra maneira para contornar esse problema, seria a utilização de algoritmos heurísticos apropriados para a geração de uma solução viável para o *problema mestre*.

Também pode-se notar que um número maior de soluções ótimas foi encontrado para instâncias do grupo $C1$ em relação aos grupos $R1$ e $RC1$. Foram encontradas 52 soluções ótimas, dentre as 60 na classe $C1$, enquanto foram encontradas 21 e 31 soluções ótimas nos grupos $R1$ e $RC1$, respectivamente. Estes resultados podem ser explicados pela topologia das instâncias. Nas instâncias $C1$, os clientes são organizados em pequenos *clusters*, o

que facilita a decisão das rotas. Já nas instâncias dos grupos $R1$ e $RC1$, os clientes são localizados respectivamente de forma aleatória e com mistura entre *clusters* de clientes com clientes posicionados de forma aleatória, sendo mais difícil computar um conjunto de rotas que atenda a todos os clientes. Devido alguns clientes do grupo $RC1$ estarem próximos, isso pode ter resultado na facilidade maior de resolução, em relação ao grupo $R1$.

O número de clientes estocásticos teve um leve impacto no número de soluções ótimas encontradas pelo *Integer L-Shaped method*. Apesar dos resultados serem próximos, foi possível observar um tempo maior gasto no segundo estágio, e uma quantidade maior de soluções ótimas encontradas com 5 clientes incertos. Uma hipótese que pode explicar este fato é que, quanto maior o número de clientes incertos, menor é o número de variáveis de decisão do primeiro estágio. Deste modo, o esforço computacional necessário para a resolução do primeiro estágio é exponencialmente menor quando existe um menor número de clientes, e o aumento de clientes incertos no segundo estágio não tem um grande impacto no esforço computacional necessário para resolução do segundo estágio, apesar do aumento na quantidade de problemas.

Um resultado obtido já esperado, é de que quanto maior o número de clientes, mais difícil foi encontrar soluções inteiras e soluções ótimas. Todos os grupos de instâncias obtiveram um número menor de soluções inteiras e ótimas quando tinham 50 clientes, onde somente o grupo $C1$ teve valores próximos em comparação a quantidade total de clientes. No grupo $R1$, não foi encontrado nenhuma solução ótima, o que reforça a dificuldade deste grupo e no grupo $RC1$ com 50 clientes, só foi encontrado soluções ótimas com 5 clientes incertos.

Capítulo 6

Conclusão e trabalhos futuros

Nessa seção, apresentamos as conclusões deste trabalho, assim como trabalhos a serem desenvolvidos futuramente.

Neste trabalho propomos o Problema de Roteamento de Veículos Capacitado com Janelas de Tempo e Clientes Estocásticos (*Capacitated Vehicle Routing Problem with Time Windows and Stochastic Clients* - CVRPTWSC). Este é uma variação do clássico Problema de Roteamento de Veículos Capacitado com Janelas de Tempo onde um ou mais dos clientes são incertos, mas possuem uma determinada probabilidade de realizarem um pedido ou não. O CVRPTWSC foi modelado como um problema de otimização estocástica em dois estágios com recurso e resolvido utilizando o *Integer L-Shaped Method*.

Experimentos computacionais realizados em extensões de instâncias da literatura demonstraram que o *Integer L-Shaped Method* foi capaz de resolver a maioria das instâncias com 25 clientes, mas não foi efetivo em resolver instâncias com 50 clientes. Além disto, observou-se que o maior esforço computacional foi utilizado no primeiro estágio e que produzir cortes de otimalidade de alta qualidade, pode reduzir o tempo de execução do algoritmo.

Trabalhos futuros sobre este problema podem ter diversos focos. O primeiro é melhorar a formulação de Programação Linear Inteira de ambos os estágios do CVRPTWSC. Além disto, também pode-se desenvolver heurísticas primais e duais para computar limitantes mais fortes e, assim, auxiliar na convergência do *Integer L-Shaped Method*. Também é possível utilizar heurísticas primais para gerar soluções inteiras iniciais para o problema mestre com o objetivo de acelerar a resolução do problema.

Referências Bibliográficas

[1] Rui Alves and Catarina Delgado. Programação linear inteira. *In Proceedings of Porto, Faculdade de Economia–Universidade do Porto. Apostila*, 1997.

[2] Ph Augerat, Jose Manuel Belenguer, Enrique Benavent, A. Corberán, D. Naddef, and G. Rinaldi. *Computational results with a branch and cut code for the capacitated vehicle routing problem*, volume 34. IMAG, 1995.

[3] John R. Birge and Francois Louveaux. *Introduction to stochastic programming*. Springer Science & Business Media, 2011.

[4] Pedro Campelo, Fábio Neves-Moreira, Pedro Amorim, and Bernardo Almada-Lobo. Consistent vehicle routing problem with service level agreements: A case study in the pharmaceutical distribution sector. *European Journal of Operational Research*, 273(1):131–145, 2019.

[5] Erbao Cao, Mingyong Lai, and Hongming Yang. Open vehicle routing problem with demand uncertainty and its robust strategies. *Expert Systems with Applications*, 41(7):3569 – 3575, 2014.

[6] George B. Dantzig. Linear programming under uncertainty. *Management science*, 1(3-4):197–206, 1955.

[7] George B. Dantzig and John H. Ramser. The truck dispatching problem. *Management science*, 6(1):80–91, 1959.

[8] J. J. De La Vega Martinez et al. Otimização robusta e programação estocástica para o problema de roteamento de veículos com múltiplos entregadores: formulações e métodos exatos. Universidade Federal de São Carlos, 2019.

[9] Dominique Feillet, Thierry Garaix, Fabien Lehuédé, Olivier Péton, and Dominique Quadri. A new consistent vehicle routing problem for the transportation of people with disabilities. *Networks*, 63(3):211–224, 2014.

[10] Matteo Fischetti, Domenico Salvagnin, and Arrigo Zanette. A note on the selection of benders' cuts. *Mathematical Programming*, 124(1-2):175–182, 2010.

[11] Michel Gendreau, Gilbert Laporte, and René Séguin. An exact algorithm for the vehicle routing problem with stochastic demands and customers. *Transportation science*, 29(2):143–155, 1995.

[12] Michel Gendreau, Gilbert Laporte, and René Séguin. Stochastic vehicle routing. *European Journal of Operational Research*, 88(1):3–12, 1996.

[13] Bram L. Gorissen, İhsan Yanıkoğlu, and Dick den Hertog. A practical guide to robust optimization. *Omega*, 53:124–137, 2015.

[14] Patrick Jaillet. *Probabilistic traveling salesman problems*. PhD thesis, Massachusetts Institute of Technology, 1985.

[15] Attila A. Kovacs, Bruce L. Golden, Richard F. Hartl, and Sophie N Parragh. Vehicle routing problems in which consistency considerations are important: A survey. *Networks*, 64(3):192–213, 2014.

[16] Felipe Lagos, Mathias A Klapp, and Alejandro Toriello. Branch-and-price for routing with probabilistic customers. 2019.

[17] Gilbert Laporte and François V. Louveaux. The integer l-shaped method for stochastic integer programs with complete recourse. *Operations research letters*, 13(3):133–142, 1993.

[18] Gilbert Laporte, Francois V. Louveaux, and Hélene Mercure. A priori optimization of the probabilistic traveling salesman problem. *Operations research*, 42(3):543–549, 1994.

[19] Gilbert Laporte, François V. Louveaux, and Luc Van Hamme. An integer l-shaped algorithm for the capacitated vehicle routing problem with stochastic demands. *Operations Research*, 50(3):415–423, 2002.

[20] Hongtao Lei, Gilbert Laporte, and Bo Guo. The capacitated vehicle routing problem with stochastic demands and time windows. *Computers & Operations Research*, 38(12):1775–1783, 2011.

[21] Jens Lysgaard. k: A package of separation routines for the capacitated vehicle routing problem. 2003.

[22] Jens Lysgaard, Adam N. Letchford, and Richard W. Eglese. A new branch-and-cut algorithm for the capacitated vehicle routing problem. *Mathematical Programming*, 100(2):423–445, 2004.

[23] Gerard Meurant. *Algorithms and complexity*. Elsevier, 2014.

[24] James Murphy. Benders, nested benders and stochastic programming: An intuitive introduction. *arXiv preprint arXiv:1312.3158*, 2013.

[25] Ulrike Ritzinger, Jakob Puchinger, and Richard F. Hartl. A survey on dynamic and stochastic vehicle routing problems. *International Journal of Production Research*, 54(1):215–231, 2016.

[26] Frans Schalekamp and David B. Shmoys. Algorithms for the universal and a priori tsp. *Operations Research Letters*, 36(1):1–3, 2008.

[27] Alexander Shapiro and Andy Philpott. A tutorial on stochastic programming. *Manuscript.*, 17, 2007.

[28] Allen L Soyster. Convex programming with set-inclusive constraints and applications to inexact linear programming. *Operations research*, 21(5):1154–1157, 1973.

[29] João Teixeira, António Pais Antunes, and Jorge Pinho de Sousa. Recyclable waste collection planning a case study. *European Journal of Operational Research*, 158(3):543–554, 2004.

[30] Paolo Toth and Daniele Vigo. *The vehicle routing problem*. SIAM, 2002.

[31] Paolo Toth and Daniele Vigo. The granular tabu search and its application to the vehicle-routing problem. *Journal on computing*, 15(4):333–346, 2003.

[32] Richard M. Van Slyke and Roger Wets. L-shaped linear programs with applications to optimal control and stochastic programming. *SIAM Journal on Applied Mathematics*, 17(4):638–663, 1969.