FUNDAÇÃO EDSON QUEIROZ
UNIVERSIDADE DE FORTALEZA – UNIFOR
VICE-REITORIA DE ENSINO DE GRADUAÇÃO E PÓS-GRADUAÇÃO – VRE
DIRETORIA DE PÓS-GRADUAÇÃO – DPG
PROGRAMA DE PÓS-GRADUAÇÃO EM INFORMÁTICA APLICADA – PPGIA

BUSCA EM VIZINHANÇA COM PROFUNDIDADE VARIÁVEL BASEADA EM CLIQUES PARA O PROBLEMA DE ROTEAMENTO DE VEÍCULOS VERDE COM COLETA E ENTREGA SIMULTÂNEAS

VICTOR TIEZZI HENRIQUES

FORTALEZA – CE

JUNHO, 2025

Victor Tiezzi Henriques

Busca em Vizinhança com Profundidade Variável Baseada em Cliques para o Problema de Roteamento de Veículos Verde com Coleta e Entrega Simultâneas

Dissertação apresentada ao Programa de Pós-Graduação em Informática Aplicada – PPGIA da Universidade de Fortaleza – UNIFOR, como parte dos requisitos para obtenção do título de Mestre em Informática Aplicada.

Universidade de Fortaleza – UNIFOR

Orientador: Prof. Dr. Napoleão Vieira Nepomuceno

Fortaleza – CE

Junho, 2025

Ficha catalográfica da obra elaborada pelo autor através do programa de geração automática da Biblioteca Central da Universidade de Fortaleza

---

Henriques, Victor Tiezzi.
Busca em Vizinhança com Profundidade Variável Baseada em Cliques para o Problema de Roteamento de Veículos Verde com Coleta e Entrega Simultâneas / Victor Tiezzi Henriques. - 2025
43 f.

Dissertação (Mestrado Acadêmico) - Universidade de Fortaleza. Programa de Mestrado Em Informática Aplicada, Fortaleza, 2025.
Orientação: Prof. Dr. Napoleão Vieira Nepomuceno.

1. Problema de Roteamento de Veículo. 2. Busca em Vizinhança. 3. Programação Linear Inteira. I. Nepomuceno, Prof. Dr. Napoleão Vieira. II. Título.

Victor Tiezzi Henriques

Busca em Vizinhança com Profundidade Variável Baseada em Cliques para o Problema de Roteamento de Veículos Verde com Coleta e Entrega Simultâneas

Dissertação apresentada ao Programa de Pós-Graduação em Informática Aplicada – PPGIA da Universidade de Fortaleza – UNIFOR, como parte dos requisitos para obtenção do título de Mestre em Informática Aplicada.

Aprovada em: 20 de Junho de 2025

BANCA EXAMINADORA

Prof. Dr. Napoleão Vieira Nepomuceno
Orientador – Universidade de Fortaleza – UNIFOR

Prof. Dr. Rommel Dias Saraiva
Universidade de Fortaleza – UNIFOR

Prof. Dr. Leonardo Sampaio Rocha
Universidade Estadual do Ceará – UECE

Agradecimentos

Agradeço primeiramente aos meus pais e ao meu irmão, pelo amor, incentivo constante e apoio incondicional que foram fundamentais para a realização deste trabalho.

Aos meus amigos Thiago Bochensky e Fernanda Caroline, agradeço pelo companheirismo e apoio durante toda a jornada do mestrado.

À Universidade de Fortaleza, sua direção, administração e, em especial, ao corpo docente, expresso minha gratidão pela oportunidade de aprimoramento acadêmico e profissional. A experiência aqui vivida abriu-me um horizonte de possibilidades, fortalecendo minha crença no mérito e na ética.

Aos professores que me proporcionaram um aprendizado valioso, minha sincera gratidão. Em particular, manifesto meus agradecimentos ao Dr. Napoleão Vieira Nepomuceno, pela orientação acadêmica e apoio inestimável durante a elaboração desta dissertação.

Resumo

Este trabalho apresenta uma abordagem baseada em Busca em Vizinhança com Profundidade Variável para a resolução do Problema de Roteamento de Veículos Verde com Coleta e Entrega Simultâneas. A proposta explora subinstâncias reduzidas, modeladas em Programação Linear Inteira e resolvidas com auxílio de um *solver* exato. São incorporadas estratégias que orientam a definição da vizinhança de uma solução, com base em critérios de proximidade entre clientes e no uso de múltiplas soluções previamente encontradas. Essas estratégias promovem um equilíbrio entre intensificação e diversificação da busca. Experimentos computacionais em instâncias de *benchmark* demonstram que a abordagem proposta supera métodos existentes em termos de qualidade e robustez das soluções, especialmente em problemas de maior escala. A avaliação individual das estratégias adotadas confirma seu impacto positivo na eficácia do método.

Palavras-chave: Problema de Roteamento de Veículo. Busca em Vizinhança. Programação Linear Inteira.

Abstract

This work presents a Variable-Depth Neighborhood Search approach for solving the Green Vehicle Routing Problem with Simultaneous Pickup and Delivery. The method explores reduced subinstances modeled as Integer Linear Programs and solved by an exact solver. Strategies are incorporated to guide the neighborhood definition, based on customer proximity and the reuse of multiple previously found solutions. These mechanisms foster a balance between intensification and diversification. Computational experiments on benchmark instances demonstrate that the proposed approach outperforms existing methods in both solution quality and robustness, particularly on large-scale problems. An ablation study confirms the positive impact of the adopted strategies on overall performance.

Keywords: Vehicle Routing Problem, Neighborhood Search, Integer Linear Programming.

Lista de ilustrações

Lista de tabelas

Lista de abreviaturas e siglas

VRP	Problema de Roteamento de Veículos
VRPB	Problema de Roteamento de Veículo com *Backhauls*
VRPSPD	Problema de Roteamento de Veículos com Coleta e Entrega Simultâneas
G-VRP	Problema de Roteamento de Veículos Verde
G-VRPSPD	Problema de Roteamento de Veículos Verde com Coleta e Entrega Simultâneas
PL	Programação Linear
PLI	Programação Linear Inteira
VLSN	Busca em Vizinhança de Larga Escala
VDNS	Estrutura de Vizinhança de Profundidade Variável
SRI	Solucionador de Instância Reduzidas
GRI	Gerador de Instâncias Reduzidas
CMSA	*Construct, Merge, Solve & Adapt*
HH-ILS	Hiper-heurística - Busca Local Iterativa
FCR	Taxa de Consumo de Combustível

Sumário

1 Introdução

A crescente preocupação com os impactos ambientais das atividades econômicas tem impulsionado transformações relevantes na gestão logística. No setor de transportes, um dos principais responsáveis pela emissão de gases de efeito estufa, observa-se a busca por soluções que aliem eficiência operacional à responsabilidade ambiental (MCKINNON, 2018). Nesse contexto, ganham destaque estratégias como a redução no uso de combustíveis fósseis, a adoção de fontes de energia alternativas e a incorporação de práticas de logística reversa nos fluxos de distribuição (ROGERS; TIBBEN-LEMBKE, 2001; DEKKER; BLOEMHOF; MALLIDIS, 2012). A logística reversa, além de reduzir resíduos, viabiliza o reaproveitamento de recursos e impõe novos desafios ao planejamento de rotas, exigindo a integração de coletas e entregas em percursos eficientes e sustentáveis.

1.1 Definição do Problema

O Problema de Roteamento de Veículos (VRP, do inglês *Vehicle Routing Problem*) é um problema clássico de otimização combinatória que visa determinar rotas eficientes para uma frota de veículos atender a um conjunto de clientes geograficamente distribuídos, respeitando restrições operacionais. Dada sua ampla aplicabilidade, o VRP possui diversas extensões que capturam especificidades de diferentes contextos logísticos. Entre essas, destaca-se o Problema de Roteamento de Veículos com Coleta e Entrega Simultâneas (VRPSPD, do inglês *Vehicle Routing Problem with Simultaneous Pickup and Delivery*), originalmente formulado por Min (1989). Nesse problema, cada cliente demanda simultaneamente serviços de coleta e entrega, os quais devem ser realizados por um único veículo em uma única visita. Essa característica torna o VRPSPD especialmente relevante em sistemas de logística reversa (DETHLOFF, 2001).

A incorporação de critérios ambientais ao planejamento logístico motivou o desenvolvimento de variantes do VRP com enfoque sustentável. O Problema de Roteamento de Veículos Verde (G-VRP, do inglês *Green Vehicle Routing Problem*) (ERDOĞAN; MILLER-HOOKS, 2012; ASGHARI; MIRZAPOUR AL-E-HASHEM, 2021) é uma dessas extensões, considerando fatores como o consumo de combustível, as emissões de CO_2 e o uso de veículos movidos a combustíveis alternativos. Dentre as variantes do G-VRP, destaca-se o Problema de Roteamento de Veículos Verde com Coleta e Entrega Simultâneas (G-VRPSPD, do inglês *Green Vehicle Routing Problem with Simultaneous Pickup and Delivery*), que combina as particularidades do VRPSPD com objetivos ambientais, penalizando o consumo proporcional à carga transportada (HUANG et al., 2012; OLGUN; KOÇ; ALTIPARMAK, 2021).

1.2 Objetivos

A seguir, apresentam-se os objetivos que nortearam o desenvolvimento da pesquisa, com destaque para o objetivo geral e suas etapas desdobradas em objetivos específicos.

1.2.1 Objetivo Geral

Este trabalho tem como objetivo geral desenvolver estratégias mais eficazes de intensificação e diversificação para uma implementação do algoritmo de Busca em Vizinhança com Profundidade Variável (VDNS, do inglês *Variable-Depth Neighborhood Search*) (AHUJA et al., 2002; ALTNER et al., 2014), baseada na estrutura de vizinhança por cliques proposta por Barroso e Nepomuceno (2024) para o VRPSPD, adaptando essa abordagem à resolução da variante verde do problema (G-VRPSPD).

1.2.2 Objetivos Específicos

- Revisar a literatura sobre o G-VRPSPD e sobre algoritmos baseados em VDNS aplicados a problemas de roteamento.
- Aprimorar a formulação de Olgun, Koç e Altıparmak (2021) para o G-VRPSPD por meio da modificação de restrições que fortalecem a estrutura do modelo.
- Integrar a implementação do VDNS de Barroso e Nepomuceno (2024) à formulação do G-VRPSPD, estabelecendo a base para os aprimoramentos metodológicos propostos.
- Desenvolver mecanismos para guiar a construção das vizinhanças, explorando aspectos como cobertura de clientes, reutilização de soluções anteriores e proximidade geográfica.
- Avaliar, em experimentos controlados, o impacto individual dos mecanismos desenvolvidos sobre o desempenho do VDNS.
- Realizar testes comparativos com o VDNS original e com a heurística de Olgun, Koç e Altıparmak (2021), utilizando a base de instâncias de *benchmark* de Salhi e Nagy (1999).
- Comparar os resultados obtidos com os de um *solver* exato aplicado à formulação completa, com e sem uso de solução inicial.

1.3 Hipótese

A hipótese deste trabalho é que a combinação de estratégias que orientam a construção das vizinhanças no VDNS — baseadas na proximidade geográfica entre clientes,

na variação de clientes utilizados em iterações anteriores e na exploração conjunta de múltiplas soluções estruturalmente próximas — permite ampliar de forma eficaz o espaço de busca, equilibrando intensificação e diversificação. Espera-se que essa abordagem melhore a qualidade e a consistência das soluções obtidas para o G-VRPSPD, superando os métodos existentes ao facilitar a fuga de ótimos locais e a convergência para soluções de maior desempenho global.

1.3.1 Questões de Pesquisa

- As instâncias do G-VRPSPD disponíveis na literatura são solucionáveis, considerando-se recursos computacionais razoáveis, pelo *solver* CPLEX?
- A utilização de múltiplas soluções previamente obtidas para construir as subinstâncias exploradas em cada iteração contribui para melhorar a diversidade e a profundidade da busca no VDNS?
- De que forma os critérios de seleção de clientes — como proximidade geográfica e participação em iterações anteriores — influenciam a qualidade das vizinhanças geradas e o desempenho do algoritmo?
- A versão aprimorada do VDNS aplicada ao G-VRPSPD apresenta desempenho competitivo em relação a algoritmos heurísticos e o *solver* CPLEX?

1.4 Visão Geral da Metodologia

A metodologia adotada neste trabalho está estruturada em duas etapas. Na primeira, o G-VRPSPD é modelado por meio de uma formulação em Programação Linear Inteira (PLI), baseada no modelo de Olgun, Koç e Altıparmak (2021), que considera o consumo de combustível em função da carga transportada e da distância percorrida. Essa formulação é reforçada com restrições propostas por Rieck e Zimmermann (2013), visando melhorar sua estrutura e resolubilidade.

Na segunda etapa, aplica-se um algoritmo VDNS, originalmente proposto por Barroso e Nepomuceno (2024) para o VRPSPD. O VDNS pertence à classe de métodos de Busca em Vizinhança de Larga Escala (VLSN, do inglês *Very Large-Scale Neighborhood Search*) (AHUJA et al., 2002), nos quais vizinhanças potencialmente exponenciais são exploradas de forma eficiente por meio da resolução de subproblemas restritos.

Neste trabalho, o VDNS é adaptado ao G-VRPSPD e aprimorado com mecanismos que orientam a construção de vizinhanças mais eficazes. A cada iteração, seleciona-se uma clique formada por clientes escolhidos com base em proximidade geográfica e participação anterior, buscando evitar repetições e explorar regiões promissoras. Essa clique é combinada

com componentes de múltiplas soluções previamente obtidas, definindo um espaço de busca restrito. A subinstância resultante é resolvida por um *solver* exato, com tamanho ajustado dinamicamente conforme o progresso da otimização. A abordagem segue o princípio de redução de instância (BLUM; RAIDL, 2016), alinhando-se a estratégias híbridas como *Generate and Solve* (SARAIVA; NEPOMUCENO; PINHEIRO, 2013), *Construct, Merge, Solve and Adapt* (CMSA) (BLUM et al., 2016) e *Relaxation Induced Neighborhood Search* (DANNA; ROTHBERG; PAPE, 2005).

A avaliação empírica foi realizada com as instâncias de *benchmark* CMTX e CMTY de Salhi e Nagy (1999), amplamente utilizadas na literatura, abrangendo casos com 50 a 199 clientes. Os resultados obtidos com o VDNS proposto foram comparados com três abordagens de referência: o VDNS original de Barroso e Nepomuceno (2024), a heurística de Olgun, Koç e Altıparmak (2021) e a resolução direta da formulação completa via *solver* CPLEX, com e sem uso de solução inicial. Essa comparação permitiu avaliar a qualidade, robustez e eficiência da abordagem desenvolvida.

1.5 Escopo e Limitações

Este trabalho concentra-se exclusivamente no estudo do G-VRPSPD. Embora a abordagem proposta possa ser aplicada, com adaptações mínimas, a outras variantes do VRPSPD e do VRP em geral, optou-se por delimitar o escopo ao G-VRPSPD em razão de restrições computacionais. Essa decisão permitiu uma avaliação mais controlada e profunda dos mecanismos desenvolvidos, sem comprometer a viabilidade experimental.

Os resultados heurísticos disponíveis na literatura para o G-VRPSPD são essencialmente os reportados por Olgun, Koç e Altıparmak (2021), que representam o principal ponto de referência atual, embora no artigo original os autores tenham rodado seus experimentos apenas para um subconjunto de instâncias; diante disso, foi realizada uma reimplementação baseada na descrição do artigo, cujos resultados obtidos mostraram-se, em muitos casos, superiores aos relatados originalmente. Também foi incorporado ao estudo o VDNS original de Barroso e Nepomuceno (2024), executado no mesmo ambiente computacional adotado para os testes da proposta.

No que diz respeito à avaliação das estratégias de construção da estrutura de vizinhança, o ideal seria realizar pelo menos 30 repetições por instância e por configuração, a fim de garantir significância estatística, dado o caráter estocástico das abordagens. No entanto, o elevado custo computacional limitou os experimentos a 10 execuções por cenário. Estima-se que a realização completa dos testes, considerando oito configurações, 3 instâncias e 30 repetições, demandaria aproximadamente $8 \times 3 \times 30 = 720$ horas, ou 30 dias de processamento contínuo, desconsiderando eventuais falhas ou interrupções.

1.6 Contribuições

Este trabalho apresenta contribuições em três frentes principais. Primeiramente, são propostas novas estratégias para orientar a construção da vizinhança baseada em cliques no algoritmo VDNS, originalmente desenvolvido por Barroso e Nepomuceno (2024). Essas estratégias envolvem a seleção sistemática de clientes com base em critérios de cobertura, proximidade geográfica e o uso de múltiplas soluções previamente obtidas, permitindo uma exploração mais ampla e eficaz do espaço de busca. Em segundo lugar, a formulação em Programação Linear Inteira proposta por Olgun, Koç e Altıparmak (2021) para o G-VRPSPD é refinada com a incorporação de restrições adicionais inspiradas em Rieck e Zimmermann (2013), visando fortalecer sua estrutura e melhorar sua resolubilidade. Por fim, os experimentos computacionais demonstram que a abordagem proposta supera métodos do estado da arte — incluindo o VDNS original, a heurística de Olgun, Koç e Altıparmak (2021) e a resolução direta via *solver* exato — em termos de qualidade e robustez das soluções obtidas. Os resultados obtidos também indicam que a proposta pode ser aplicada, com adaptações mínimas, a outras variantes do VRPSPD, oferecendo um caminho promissor para pesquisas futuras em otimização de rotas com múltiplos critérios.

1.7 Organização do Trabalho

Este documento está organizado da seguinte forma. O Capítulo 2 apresenta os conceitos teóricos necessários à compreensão do problema e das abordagens empregadas. O Capítulo 3 revisa os principais trabalhos relacionados ao VRPSPD e ao G-VRPSPD, situando a proposta deste estudo no contexto da literatura. O Capítulo 4 formaliza o problema, detalha a formulação matemática adotada e descreve a estrutura de vizinhança utilizada pelo VDNS, incluindo as estratégias propostas. O Capítulo 5 apresenta os experimentos realizados com instâncias de *benchmark*, bem como a análise comparativa dos resultados. Por fim, o Capítulo 6 sintetiza as contribuições do trabalho e sugere possíveis direções para investigações futuras.

2 Fundamentação Teórica

Este capítulo apresenta os fundamentos teóricos necessários à compreensão do trabalho. Inicia-se com os conceitos de teoria dos grafos (BONDY; MURTY, 1976; GOLDBARG; GOLDBARG, 2012), fundamentais para a modelagem do problema de roteamento. Em seguida, são abordadas técnicas clássicas da Pesquisa Operacional, com destaque para Programação Linear (PL), Programação Linear Inteira (PLI) e o método *branch-and-bound* (HILLIER; LIEBERMAN, 2013; GOLDBARG; GOLDBARG; LUNA, 2015), que sustentam a formulação matemática empregada. Na sequência, exploram-se estratégias de busca em vizinhança, com ênfase nos métodos em Larga Escala (VLSN, do inglês *Very Large-Scale Neighborhood Search*), especialmente a Profundidade Variável (VDNS, do inglês *Variable-Depth Neighborhood Search*), e nas abordagens baseadas em redução de instância, que servem de base para o desenvolvimento da proposta.

2.1 Modelagem com Grafos

Grafos são estruturas de abstração amplamente utilizadas para representar relações entre elementos e solucionar diversos tipos de problemas. Neste trabalho, desempenham um papel essencial na modelagem do G-VRPSPD, em que os clientes e o depósito são representados como vértices, e as rotas que os conectam correspondem a arcos do grafo (GOLDBARG; GOLDBARG, 2012).

De acordo com Bondy e Murty (1976), um grafo simples G é definido como uma tripla ordenada $(V(G), E(G), \psi_G)$, onde $V(G)$ é um conjunto não vazio de vértices, $E(G)$ é um conjunto de arestas disjunto de $V(G)$, e ψ_G é uma função de incidência que associa a cada aresta de G um par não ordenado de vértices (não necessariamente distintos). Se e é uma aresta e u, v são vértices tais que $\psi_G(e) = uv$, diz-se que e liga u e v, sendo esses vértices chamados de extremidades de e.

Além dos grafos simples, existem os grafos direcionados, ou dígrafos. Um grafo direcionado D é uma tripla ordenada $(V(D), A(D), \psi_D)$, composta por um conjunto não vazio $V(D)$ de vértices, um conjunto $A(D)$ de arcos (disjunto de $V(D)$) e uma função de incidência ψ_D que associa a cada arco um par ordenado de vértices (também não necessariamente distintos). Se $\psi_D(a) = (u, v)$, então a é um arco que liga u a v; u é a cauda e v, a cabeça do arco.

Um subgrafo H de um grafo G, denotado por $H \subseteq G$, é um grafo cujos vértices e arestas pertencem a G, ou seja, $V(H) \subseteq V(G)$, $E(H) \subseteq E(G)$. A função de incidência ψ_H de H é definida como a restrição da função de incidência ψ_G de G ao conjunto de arestas

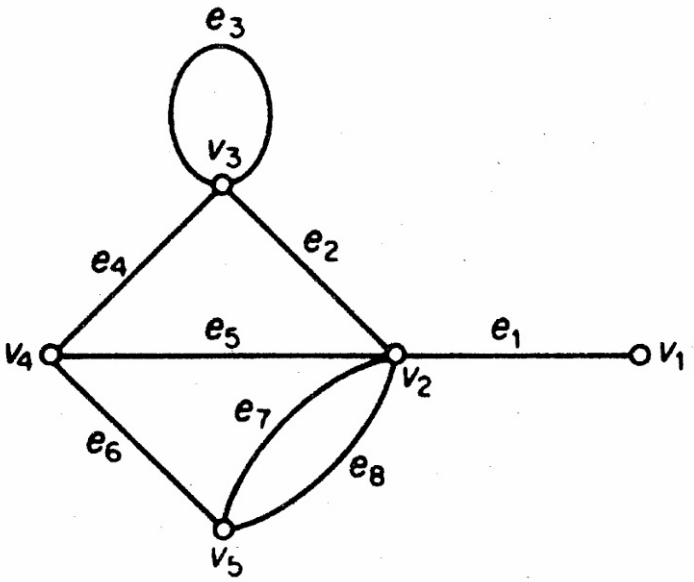
de H, ou seja, $\psi_H(e) = \psi_G(e)$ para todo $e \in E(H)$. Isso significa que as conexões entre os vértices em H são exatamente aquelas já definidas em G.

Um caso particular de subgrafo é a clique, que corresponde a um subconjunto $S \subseteq V(G)$ tal que o subgrafo induzido $G[S]$ é completo, ou seja, todos os pares de vértices de S estão conectados por arestas. Para ilustrar, considere o grafo G representado na Figura 1, com vértices $V(G) = \{v_1, v_2, v_3, v_4, v_5\}$, arestas $E(G) = \{e_1, e_2, e_3, e_4, e_5, e_6, e_7, e_8\}$, e função de incidência ψ_G definida por:

$$\psi_G(e_1) = v_1v_2, \quad \psi_G(e_2) = v_2v_3, \quad \psi_G(e_3) = v_3v_3, \quad \psi_G(e_4) = v_3v_4,$$
$$\psi_G(e_5) = v_2v_4, \quad \psi_G(e_6) = v_4v_5, \quad \psi_G(e_7) = v_2v_5, \quad \psi_G(e_8) = v_2v_5.$$

Um exemplo de subgrafo H poderia conter os vértices v_2, v_3, v_4 e v_5, e as arestas e_2, e_4, e_5, e_6 e e_8. Esse subgrafo, no entanto, não constitui uma clique, pois não há aresta entre v_3 e v_5. Por outro lado, o subgrafo induzido pelos vértices v_2, v_3 e v_4, com arestas e_2, e_4 e e_5, forma uma clique, pois todos os vértices estão diretamente conectados entre si.

Figura 1 – Diagrama do grafo G.



Fonte: Bondy e Murty (1976)

2.2 Pesquisa Operacional

A Pesquisa Operacional pode ser compreendida como um conjunto de técnicas que utilizam o método científico para apoiar a tomada de decisões em contextos complexos (HILLIER; LIEBERMAN, 2013). A área desempenha um papel central na otimização de processos e na resolução de problemas em diversas aplicações da computação. Por meio da construção de modelos matemáticos e da aplicação de algoritmos especializados,

permite analisar sistemas e identificar soluções eficientes para desafios como alocação de recursos, planejamento de atividades e operações logísticas.

2.2.1 Programação Linear

A Programação Linear (PL) é uma técnica clássica da Pesquisa Operacional utilizada para resolver problemas de otimização nos quais tanto a função objetivo quanto as restrições são lineares. De modo geral, um modelo de PL visa determinar os valores de variáveis de decisão que maximizam ou minimizam uma função linear, respeitando um conjunto de restrições também lineares (TAHA, 2011; HILLIER; LIEBERMAN, 2013).

A formulação geral de um problema de PL é dada por:

$$\begin{aligned} \max \quad & z = c^T x && (2.1)\\ \text{sujeito a} \quad & Ax \leq b && (2.2)\\ & x \geq 0 && (2.3) \end{aligned}$$

em que $x \in \mathbb{R}^n$ representa o vetor de variáveis de decisão, $c \in \mathbb{R}^n$ é o vetor de coeficientes da função objetivo, $A \in \mathbb{R}^{m \times n}$ é a matriz de coeficientes das restrições e $b \in \mathbb{R}^m$ representa os recursos ou limites disponíveis.

Para ilustrar, considere o seguinte exemplo adaptado de (HILLIER; LIEBERMAN, 2013). Uma empresa deseja determinar a produção semanal de dois produtos, de modo a maximizar o lucro total. Seja x_1 a quantidade do produto 1 e x_2 a do produto 2. Cada unidade de produto gera um lucro de 1 unidade monetária. Ambos consomem tempo em duas máquinas com capacidade limitada, conforme a Tabela 1.

Tabela 1 – Coeficientes do modelo de PL

Máquina	**Produto 1 (h/unid)**	**Produto 2 (h/unid)**	**Capacidade (h)**
1	1	2	4
2	4	2	12

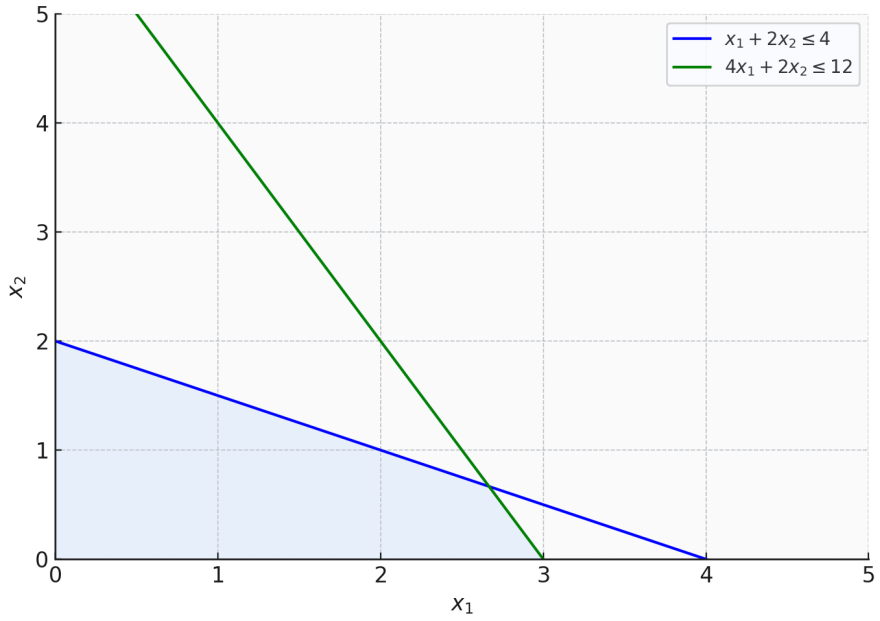
Fonte: Adaptado de Hillier e Lieberman (2013).

O modelo de PL é então:

$$\begin{aligned} \text{Maximizar} \quad & z = x_1 + x_2 && (2.4)\\ \text{sujeito a} \quad & x_1 + 2x_2 \leq 4 && (2.5)\\ & 4x_1 + 2x_2 \leq 12 && (2.6)\\ & x_1, x_2 \geq 0 && (2.7) \end{aligned}$$

A Figura 2 mostra a região de solução viável para esse problema, destacando o conjunto de soluções que satisfazem todas as restrições.

Figura 2 – Região viável do problema de PL.



Fonte: Elaborado pelo autor.

Esse exemplo evidencia como a PL permite modelar problemas reais de forma sistemática. Sua aplicação é ampla, abrangendo áreas como produção industrial, finanças, agricultura, logística e ciência da computação — especialmente em contextos que exigem o uso eficiente de recursos limitados. Para a resolução desses modelos, destaca-se o método Simplex, desenvolvido por Dantzig (2002), reconhecido por sua eficiência mesmo em problemas de grande escala.

2.2.2 Programação Linear Inteira

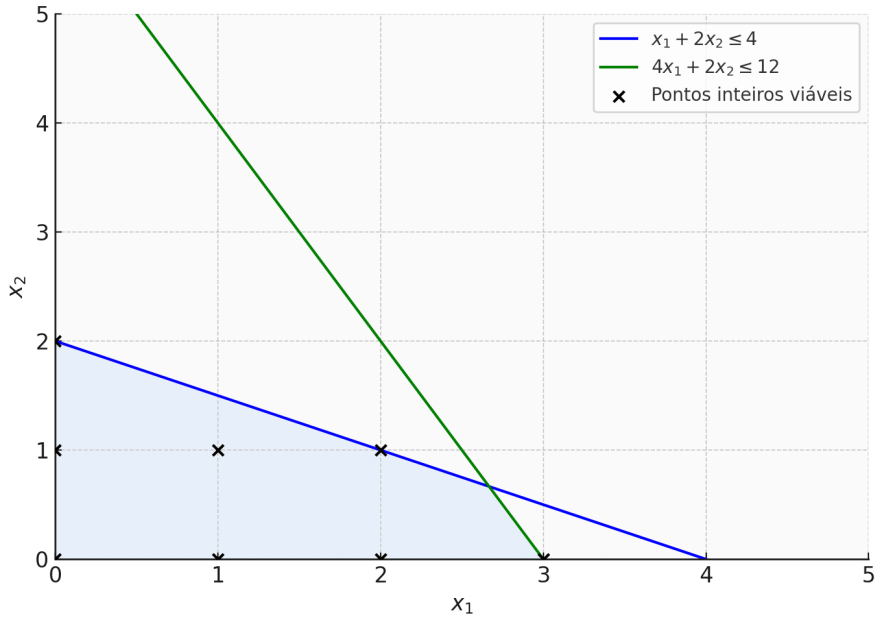
Em muitos problemas reais, as soluções viáveis devem assumir valores inteiros. Nestes casos, utiliza-se a PLI, uma extensão da PL na qual impõe-se que algumas ou todas as variáveis sejam inteiras (WOLSEY, 1998; GOLDBARG; GOLDBARG; LUNA, 2015).

No exemplo anterior, suponha que os produtos sejam vendidos em lotes indivisíveis. Assim, as variáveis x_1 e x_2 devem assumir valores inteiros. O modelo torna-se:

$$\begin{aligned} \text{Maximizar} \quad & z = x_1 + x_2 && (2.8)\\ \text{sujeito a} \quad & x_1 + 2x_2 \leq 4 && (2.9)\\ & 4x_1 + 2x_2 \leq 12 && (2.10)\\ & x_1, x_2 \in \mathbb{Z}^+ && (2.11) \end{aligned}$$

A Figura 3 mostra os pontos inteiros viáveis dentro da região de solução da relaxação linear. Observa-se que a solução ótima da PL não é inteira, evidenciando a necessidade de tratamento específico via PLI.

Figura 3 – Soluções inteiras viáveis para o problema de PLI.



Fonte: Elaborado pelo autor.

A inclusão de restrições de integralidade, em geral, torna o problema mais complexo, pois a região de solução deixa de ser convexa. Para resolver esse tipo de problema, métodos como o *branch-and-bound* são amplamente utilizados.

2.2.3 Branch-and-Bound

O método *branch-and-bound* é um algoritmo exato clássico para resolver problemas de PLI (HILLIER; LIEBERMAN, 2013). A ideia central é particionar o espaço de soluções em subproblemas (*branching*) e, com base em limites superiores e inferiores, eliminar subárvores (*bounding*) que não podem conter a solução ótima.

Aplicando o *branch-and-bound* ao modelo inteiro do exemplo anterior, inicia-se pela relaxação linear (ignorando a integralidade). A solução ótima da relaxação é obtida na interseção das equações $x_1 + 2x_2 = 4$ e $4x_1 + 2x_2 = 12$, resultando em $x_1 = \frac{8}{3}$ e $x_2 = \frac{2}{3}$, com valor objetivo $z = \frac{10}{3}$. Como essa solução é fracionária (o que ocorre em muitos problemas práticos), procede-se com a seleção de uma variável não inteira (por exemplo, x_1) e a geração de dois subproblemas com restrições adicionais: $x_1 \leq 2$ e $x_1 \geq 3$.

Esse processo é repetido recursivamente, particionando o espaço viável em subárvores e eliminando aquelas que não podem conter soluções melhores do que a melhor solução inteira já encontrada. A ramificação prossegue até que todas as folhas da árvore representem soluções inteiras viáveis ou sejam podadas por inviabilidade, quando o subproblema não possui solução viável, ou por limites, quando o melhor resultado do subproblema não é melhor que o resultado já encontrado. O método *branch-and-bound* assegura, assim, a

obtenção da solução ótima inteira, sendo uma ferramenta central na resolução de problemas combinatórios e de natureza discreta.

2.3 Busca em Vizinhança

Os algoritmos de busca em vizinhança, também conhecidos como algoritmos de busca local, constituem uma classe fundamental de heurísticas para problemas de otimização combinatória. A ideia central consiste em partir de uma solução viável e, iterativamente, buscar melhorias em sua vizinhança — um subconjunto do espaço de soluções definido em torno da solução corrente. A escolha da estrutura de vizinhança é um fator crítico: vizinhanças maiores tendem a proporcionar soluções de melhor qualidade, mas aumentam o custo computacional de cada iteração (AHUJA et al., 2002).

Considere uma instância I de um problema de otimização combinatória, com X representando o conjunto finito (geralmente muito grande) de soluções viáveis, e uma função de custo $c : X \to \mathbb{R}$ associando a cada solução seu valor objetivo. Assume-se aqui que o problema é de minimização, ou seja, o objetivo é encontrar uma solução $x^* \in X$ tal que $c(x^*) \leq c(x)$ para todo $x \in X$. A vizinhança de uma solução $x \in X$ é definida como um subconjunto $N(x) \subseteq X$; isto é, N é uma função que associa a cada solução um conjunto de soluções vizinhas. Diz-se que uma solução x é localmente ótima (ou um ótimo local) com respeito a N se $c(x) \leq c(x')$ para todo $x' \in N(x)$.

Com essas definições, pode-se descrever o funcionamento de um algoritmo de busca local. Dada uma solução inicial x, o algoritmo avalia as soluções da vizinhança $N(x)$ e seleciona $x' = \arg\min_{x'' \in N(x)} c(x'')$, ou seja, a melhor solução da vizinhança. Se $c(x') < c(x)$, realiza-se a atualização $x \leftarrow x'$, e o processo é repetido a partir da nova solução. Quando não há melhoria possível, isto é, quando x é um ótimo local, o algoritmo é encerrado. Esse procedimento é conhecido como *steepest descent* (descida mais íngreme), por sempre escolher a melhor solução na vizinhança atual.

2.3.1 Busca em Vizinhança em Larga Escala

O conceito de Busca em Vizinhança de Larga Escala (VLSN, do inglês *Very Large-Scale Neighborhood Search*) refere-se a algoritmos que exploram vizinhanças exponencialmente grandes em relação ao tamanho da instância, mas de forma eficiente, sem a necessidade de enumerar explicitamente todos os vizinhos. O estudo de Ahuja et al. (2002) é uma das principais referências sobre o tema, propondo uma categorização das abordagens VLSN em três classes:

- **Métodos de profundidade variável (VDNS)**: realizam buscas parciais em vizinhanças muito amplas, com mecanismos de controle sobre o tamanho e a composição

da vizinhança em cada iteração.

- **Métodos baseados em fluxos de rede**: exploram vizinhanças exponenciais por meio de algoritmos clássicos de fluxo em redes, aplicados a grafos de melhoria (*improvement graphs*).
- **Vizinhanças com estrutura polinomial**: definem subconjuntos do espaço de soluções por meio de restrições que podem ser resolvidas em tempo polinomial, permitindo otimização eficiente mesmo em espaços exponenciais.

Técnicas VLSN são particularmente eficazes em problemas NP-difíceis, nos quais métodos exatos se tornam impraticáveis para instâncias de grande porte. Aplicações citadas por Ahuja et al. (2002) incluem algoritmos baseados em *column generation* em programação linear, detecção de ciclos de custo negativo em problemas de fluxo de custo mínimo, e estratégias de *ejection chains* aplicadas a problemas de roteamento e escalonamento.

2.3.2 Busca em Vizinhança com Profundidade Variável

Os métodos de Busca em Vizinhança com Profundidade Variável (VDNS, do inglês *Variable-Depth Neighborhood Search*) (AHUJA et al., 2002) formam uma subclasse relevante das técnicas de VLSN e representam uma evolução das estratégias tradicionais de busca local para problemas de otimização combinatória. Enquanto algoritmos clássicos exploram vizinhanças definidas por alterações pontuais — como uma única troca, inserção ou remoção —, o VDNS permite a execução de sequências de operações com profundidade variável, ampliando substancialmente o espaço de busca acessado em cada iteração.

A principal ideia do VDNS é aplicar, a partir de uma solução inicial, cadeias de movimentos de diferentes comprimentos, adaptadas dinamicamente conforme o progresso da otimização. A escolha da profundidade e da composição dessas cadeias é guiada por heurísticas que buscam priorizar regiões promissoras do espaço de soluções. Essa abordagem é especialmente eficaz quando vizinhanças convencionais se mostram insuficientes para escapar de ótimos locais, mas a exploração exaustiva de vizinhanças muito grandes seria computacionalmente inviável. Assim, o VDNS equilibra profundidade de busca e viabilidade computacional, sendo uma alternativa robusta para problemas de grande complexidade.

2.3.3 Redução de Instância do Problema

Os *solvers* gerais de programação matemática, como CPLEX e GUROBI, são frequentemente muito eficazes até um determinado porte da instância. Quando o problema se torna grande demais para ser resolvido diretamente por um *solver*, pode ser vantajoso aplicar técnicas de redução de instância. O objetivo é gerar uma subinstância que contenha soluções de alta qualidade — ou até mesmo ótimas — da instância original, e que possa

ser resolvida de forma eficiente por métodos exatos. Dessa forma, é possível aproveitar a valiosa *expertise* em Pesquisa Operacional incorporada aos *solvers*, mesmo no contexto de problemas de grande escala. Estratégias representativas dessa abordagem incluem o algoritmo *Construct, Merge, Solve & Adapt* (CMSA) (BLUM et al., 2016) e o *framework Generate-and-Solve* (SARAIVA; NEPOMUCENO; PINHEIRO, 2013).

O *framework Generate and Solve* (SARAIVA; NEPOMUCENO; PINHEIRO, 2013) exemplifica essa abordagem ao combinar uma metaheurística, denominada Gerador de Instâncias Reduzidas (GRI, do inglês *Generator of Reduced Instances*), com um resolvedor exato, denominado Solucionador de Instâncias Reduzidas (SRI, do inglês *Solver of Reduced Instances*). A metaheurística gera subinstâncias da instância original, com base em um subconjunto de componentes de soluções. Essas subinstâncias são então resolvidas por métodos exatos, implementados em *solvers*. Os resultados obtidos pelo SRI são utilizados como *feedback* para orientar o GRI na geração de novas subinstâncias promissoras. Com isso, o *framework* explora iterativamente o espaço de busca de forma mais controlada e eficiente, conciliando a capacidade exploratória das heurísticas com a precisão dos métodos exatos.

Outra abordagem baseada na redução de instância é o algoritmo *Construct, Merge, Solve & Adapt* (CMSA), proposto por Blum et al. (2016). O CMSA é um *framework* genérico voltado para problemas de otimização combinatória, que opera por meio de uma geração iterativa e adaptativa de subinstâncias. Em cada iteração, soluções são construídas por métodos probabilísticos ou heurísticos, cujos componentes (por exemplo, arcos ou rotas) são agregados para formar uma subinstância reduzida do problema original. Essa subinstância é então resolvida por um método exato ou um resolvedor especializado. O algoritmo adapta progressivamente o processo de construção e a composição da subinstância com base nos resultados obtidos, buscando concentrar os esforços computacionais nas regiões mais promissoras do espaço de soluções.

3 Trabalhos Relacionados

Este capítulo revisa a literatura relevante sobre problemas de roteamento de veículos que envolvem coletas e entregas, com ênfase nas variantes simultâneas e sustentáveis. Inicia-se com uma visão geral do Problema de Roteamento de Veículo com *Backhauls* (VRPB, do inglês *Vehicle Routing Problem with Backhauls*) e suas principais subclasses, destacando o Problema com Coleta e Entrega Simultânea, foco deste trabalho. Em seguida, são discutidas contribuições voltadas ao VRPSPD, incluindo formulações exatas, heurísticas e metaheurísticas propostas na literatura. Também são abordadas as extensões verdes do problema, que incorporam critérios ambientais ao modelo. Por fim, destaca-se a evolução de estratégias modernas como busca em vizinhança com profundidade variável e métodos híbridos baseados em redução de instância, que têm se mostrado promissores na solução de instâncias de maior porte e complexidade.

3.1 Problema de roteamento de veículo com *backhauls*

O Problema de Roteamento de Veículos (VRP, do inglês *Vehicle Routing Problem*), introduzido por Dantzig e Ramser (1959), constitui a base para diversas variantes relevantes na literatura de logística e transporte. No VRP clássico, uma frota parte de um depósito para atender a clientes com demandas de entrega, buscando minimizar o custo total das rotas. Extensões naturais desse problema consideram a presença de coletas, resultando nos chamados problemas com *backhauls*.

O Problema de Roteamento de Veículo com *Backhauls* (VRPB, do inglês *Vehicle Routing Problem with Backhauls*), formalizado por Goetschalckx e Jacobs-Blecha (1989), introduz dois tipos de clientes: os de entrega (*linehauls*) e os de coleta (*backhauls*), cujos produtos devem ser recolhidos e transportados de volta ao depósito. De acordo com Parragh, Doerner e Hartl (2008), o VRPB pode ser classificado em quatro variantes principais:

- **VRPCB (*Clustered Backhauls*)**: todos os clientes de entrega devem ser atendidos antes que qualquer cliente de coleta.
- **VRPMB (*Mixed Linehauls and Backhauls*)**: permite que entregas e coletas ocorram em qualquer ordem.
- **VRPDDP (*Divisible Delivery and Pickup*)**: clientes podem ser visitados até duas vezes — uma para entrega e outra para coleta.
- **VRPSPD (*Simultaneous Pickup and Delivery*)**: cada cliente possui simultaneamente uma demanda de entrega e coleta, e deve ser visitado uma única vez.

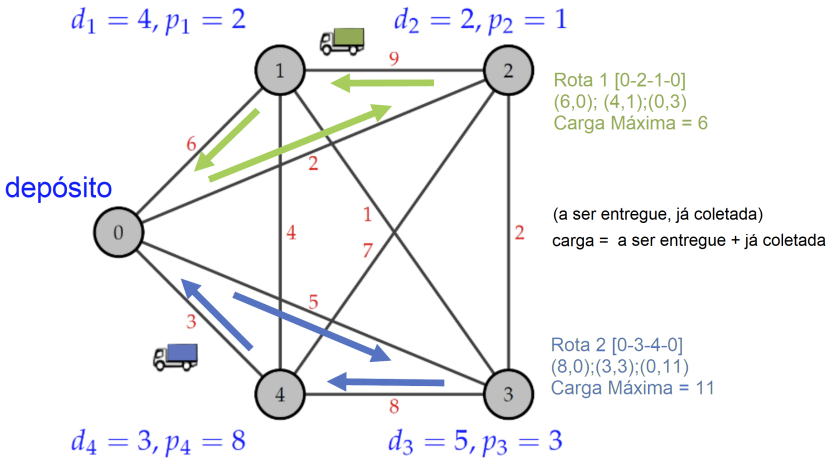
Essas variantes são todas NP-difíceis, pois generalizam o Problema do Caixeiro Viajante (GAREY; JOHNSON, 1979), e têm ampla aplicação prática ao reduzirem deslocamentos vazios e otimizarem a utilização da frota.

3.2 Problema de Roteamento de Veículo com Coleta e Entrega Simultânea

O Problema de Roteamento de Veículos com Coleta e Entrega Simultâneas (VRPSPD, do inglês *Vehicle Routing Problem with Simultaneous Pickup and Delivery*), introduzido por Min (1989), exige que cada cliente seja visitado exatamente uma vez, realizando-se entrega e coleta simultaneamente. Essa característica o diferencia do VRPDDP, onde os serviços podem ser realizados em visitas separadas (PARRAGH; DOERNER; HARTL, 2008). O VRPSPD é particularmente relevante em contextos como logística reversa, distribuição de bebidas, alimentos e coleta de produtos recondicionados (KOÇ; LAPORTE; TÜKENMEZ, 2020; SANTOS et al., 2023).

A Figura 4 ilustra um exemplo do VRPSPD, com clientes que apresentam demandas de entrega (d_i) e de coleta (p_i), atendidos por uma frota na qual cada veículo possui uma capacidade máxima de carga que não pode ser excedida em nenhum ponto da rota. O objetivo é minimizar o custo total das rotas. Neste exemplo, o grafo é composto por 5 nós numerados de 0 a 4. O ponto 0 representa o depósito central, de onde partem e para onde retornam os caminhões, enquanto os nós 1, 2, 3 e 4 correspondem aos clientes.

Figura 4 – Exemplo ilustrativo do VRPSPD



Fonte: Adaptado de Nepomuceno, Saboia e Pinheiro (2019)

No exemplo, são utilizados dois caminhões, cada um com capacidade de 11 unidades. O primeiro caminhão parte do depósito, visita os clientes 2 e 1, e retorna ao depósito. Durante o percurso, sua carga máxima acumulada atinge 6 unidades, valor inferior ao limite permitido. O segundo caminhão atende os clientes 3 e 4, alcançando, em dado momento, a

carga máxima de 11 unidades, exatamente igual à sua capacidade. Em ambos os casos, as rotas foram planejadas de modo a garantir o respeito às restrições de capacidade em todos os pontos do trajeto. Vale destacar que, caso a ordem de atendimento do segundo caminhão fosse invertida — visitando primeiro o cliente 4 e depois o cliente 3 —, a carga acumulada atingiria 13 unidades, ultrapassando o limite permitido e inviabilizando a rota.

Diversas estratégias têm sido desenvolvidas para tratar o VRPSPD. Heurísticas tradicionais incluem métodos de inserção por agrupamento (SALHI; NAGY, 1999), busca tabu (MONTANÉ; GALVÃO, 2006), colônia de formigas (GAJPAL; ABAD, 2009), algoritmos genéticos (TASAN; GEN, 2012) e busca local iterativa (SUBRAMANIAN et al., 2010). Em anos recentes, metaheurísticas híbridas — combinando fases construtivas e mecanismos de intensificação — têm se destacado, como em (GOKSAL; KARAOGLAN; ALTIPARMAK, 2013; AVCI; TOPALOGLU, 2016; NEPOMUCENO; SABOIA; COELHO, 2023; BARROSO; NEPOMUCENO, 2024). Métodos exatos baseados em programação inteira também foram propostos (DELL'AMICO; RIGHINI; SALANI, 2006; SUBRAMANIAN et al., 2013).

3.2.1 Problema de Roteamento de Veículos Verde com Coleta e Entrega Simultâneas

O G-VRPSPD estende o VRPSPD ao incorporar critérios de sustentabilidade. Embora ainda sejam escassos os trabalhos dedicados especificamente a essa variante, há uma literatura mais ampla sobre problemas de roteamento com objetivos ambientais. Nessas outras variantes do VRP, o consumo de combustível e as emissões de CO_2 têm sido modelados como funções da carga transportada, velocidade, topografia, aceleração e frenagem do veículo (DEMIR; BEKTAŞ; LAPORTE, 2011; ALINAGHIAN; NADERIPOUR, 2016; FRANCESCHETTI et al., 2017).

Outras abordagens consideram o uso de frotas compostas por veículos elétricos ou movidos a combustíveis alternativos, sujeitas a restrições de autonomia e recarga (ERDOĞAN; MILLER-HOOKS, 2012; MACRINA et al., 2019). Apesar da relevância ambiental do problema, aspectos como pegada de carbono e emissões de gases de efeito estufa ainda são pouco explorados na literatura dedicada ao VRPSPD.

Huang et al. (2012) apresentaram uma das primeiras formulações matemáticas para o G-VRPSPD. Mais recentemente, Olgun, Koç e Altıparmak (2021) avançaram na modelagem do problema ao propor uma formulação matemática e um algoritmo hiper-heurístico, baseado em busca local Iterativa (HH-ILS, do inglês *Hyper-Heuristic - Iterative Local Search*) e descida por vizinhança variável. Os autores conduziram experimentos para avaliar o impacto da função objetivo ambiental sobre o consumo total de combustível, comparando os modelos G-VRPSPD e VRPSPD. Os resultados indicam que a inclusão

de critérios ambientais na otimização tem efeito significativo na redução dos custos operacionais.

Embora a literatura sobre VRPs ambientais esteja em expansão, o G-VRPSPD ainda é pouco explorado. Essa lacuna justifica investigações adicionais, tanto em termos de modelagem quanto no desenvolvimento de algoritmos eficientes.

4 Metodologia

Neste capítulo, apresenta-se uma formulação em PLI para o G-VRPSPD, adaptada de Olgun, Koç e Altıparmak (2021). Em seguida, revisita-se a estrutura de vizinhança de profundidade variável proposta por Barroso e Nepomuceno (2024) e propõe-se um algoritmo VDNS para resolver o problema, com ênfase no aprimoramento do mecanismo de geração de vizinhanças promissoras.

4.1 Formulação Matemática

A formulação matemática adotada neste trabalho segue a proposta de Olgun, Koç e Altıparmak (2021), que combina uma modelagem baseada em fluxo com uma função de consumo de combustível dependente da carga transportada e da distância percorrida. O problema é definido sobre um grafo dirigido completo $G = (N, A)$, onde $N = \{0, 1, \ldots, n\}$ representa o conjunto de nós (sendo o depósito o nó 0 e os demais os clientes), e A o conjunto de arcos. O conjunto de clientes é denotado por $N_c = N \setminus \{0\}$. Cada cliente $i \in N_c$ possui uma demanda de entrega d_i e uma demanda de coleta p_i. A distância entre os nós i e j é representada por c_{ij}, e c_0 é a constante associada ao custo de combustível por unidade de distância. A capacidade do veículo é dada por Q, sendo $Q_D = \sum_{i \in N_c} d_i$ e $Q_P = \sum_{i \in N_c} p_i$ os totais de entrega e coleta, respectivamente. A taxa de consumo de combustível (FCR, do inglês *Fuel Consumption Rate*) por unidade de distância é modelada como uma função linear da carga transportada. Sejam p^* e p_0 os valores da FCR com o veículo totalmente carregado e vazio, respectivamente. Define-se o coeficiente $\alpha = (p^* - p_0)/Q$, permitindo expressar o consumo no arco (i, j) como $p_0 + \alpha \cdot q_{ij}$, onde q_{ij} é a carga total transportada. Definem-se as variáveis de decisão: $x_{ij} \in \{0, 1\}$: indica se o arco (i, j) é utilizado por algum veículo; $U_{ij} \geq 0$: quantidade de carga de entrega transportada ao longo do arco (i, j); e $V_{ij} \geq 0$: quantidade de carga de coleta transportada ao longo do arco (i, j). O problema pode ser formulação em PLI como:

$$\min \quad \sum_{i \in N} \sum_{j \in N} c_0 \cdot c_{ij} \cdot (p_0 \cdot x_{ij} + \alpha \cdot (U_{ij} + V_{ij})) \tag{4.1}$$

$$\text{s.t.} \quad \sum_{i \in N} x_{ij} = 1 \qquad \forall j \in N_c \tag{4.2}$$

$$\sum_{j \in N} x_{ij} = 1 \qquad \forall i \in N_c \tag{4.3}$$

$$U_{ij} + V_{ij} \leq M_{ij} \cdot x_{ij} \qquad \forall i, j \in N \tag{4.4}$$

$$\sum_{j \in N_c} U_{0j} = Q_D \tag{4.5}$$

$$\sum_{i \in N_c} U_{i0} = 0 \tag{4.6}$$

$$\sum_{i \in N_c} V_{i0} = Q_P \tag{4.7}$$

$$\sum_{j \in N_c} V_{0j} = 0 \tag{4.8}$$

$$U_{ij} \leq (Q - d_i) \cdot x_{ij} \qquad \forall i \in N_c, j \in N \tag{4.9}$$

$$V_{ij} \leq (Q - p_j) \cdot x_{ij} \qquad \forall i \in N, j \in N_c \tag{4.10}$$

$$U_{ij} \geq d_j \cdot x_{ij} \qquad \forall i \in N, j \in N_c \tag{4.11}$$

$$V_{ij} \geq p_i \cdot x_{ij} \qquad \forall i \in N_c, j \in N \tag{4.12}$$

$$\sum_{i \in N} V_{ji} - \sum_{i \in N} V_{ij} = p_j \qquad \forall j \in N_c \tag{4.13}$$

$$\sum_{i \in N} U_{ij} - \sum_{i \in N} U_{ji} = d_j \qquad \forall j \in N_c \tag{4.14}$$

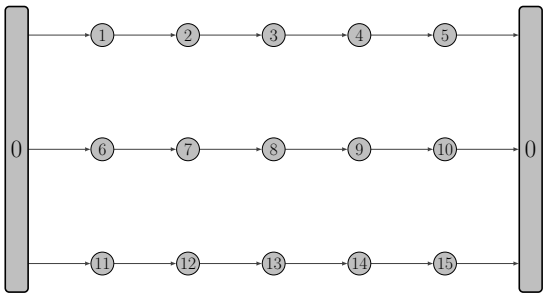
$$x_{ij} \in \{0, 1\} \qquad \forall i, j \in N \tag{4.15}$$

$$0 \leq U_{ij}, V_{ij} \leq Q \qquad \forall i, j \in N \tag{4.16}$$

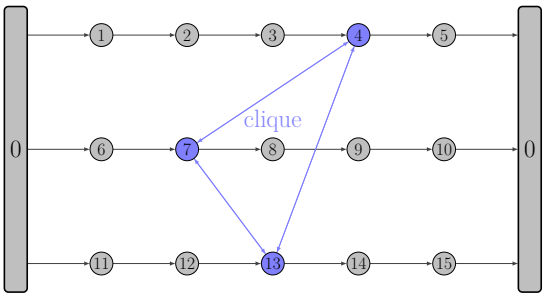
A função objetivo (4.1) minimiza o consumo total de combustível. As restrições (4.2) e (4.3) garantem que cada cliente seja visitado exatamente uma vez. A restrição (4.4) assegura que apenas arcos ativados transportem carga, utilizando o parâmetro $M_{ij} = Q - \max\{0, d_i - p_i, p_j - d_j\}$, conforme proposto por Rieck e Zimmermann (2013). As restrições (4.5)–(4.8) garantem o fluxo correto de entregas e coletas no depósito. As restrições (4.9)–(4.12) limitam a carga transportada com base na capacidade do veículo e nas demandas dos clientes. As restrições (4.13) e (4.14) asseguram o equilíbrio de fluxo em cada cliente. Por fim, as restrições (4.15) e (4.16) definem o domínio das variáveis. Cabe destacar que, embora algumas formulações do problema — como a apresentada por Olgun, Koç e Altıparmak (2021) — incluam uma restrição explícita sobre o número máximo de veículos, essa limitação não foi adotada na prática pelo algoritmo HH-ILS (OLGUN; KOÇ; ALTIPARMAK, 2021). De forma similar, este trabalho assume uma frota ilimitada.

4.2 Estrutura de Vizinhança

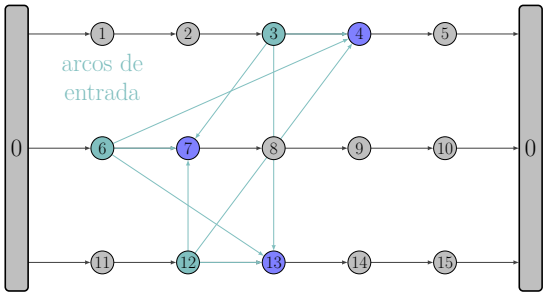
A estrutura de vizinhança proposta neste trabalho segue a proposta de Barroso e Nepomuceno (2024), baseada na construção de instâncias reduzidas (ou subinstâncias) a partir de uma solução viável s. Cada solução é composta por rotas que partem e retornam ao depósito (nó 0), visitando conjuntos disjuntos de clientes (Figura 5a). A construção da vizinhança segue um procedimento sistemático, guiado por um parâmetro de profundidade κ, que define o número de clientes selecionados para compor uma clique.



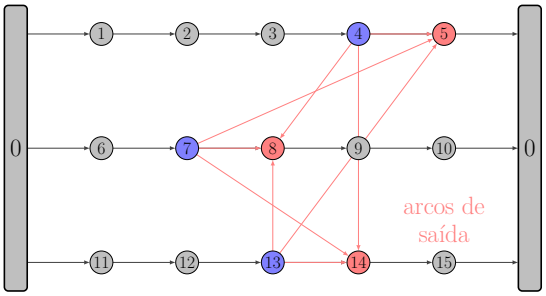
(a) Representação de uma solução.



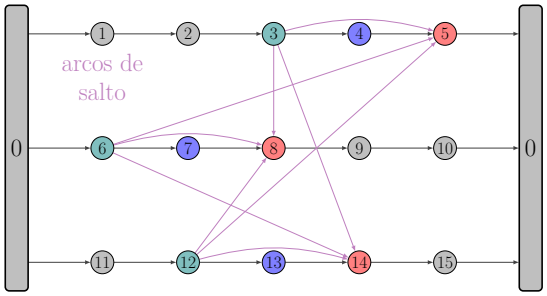
(b) Arcos da clique.



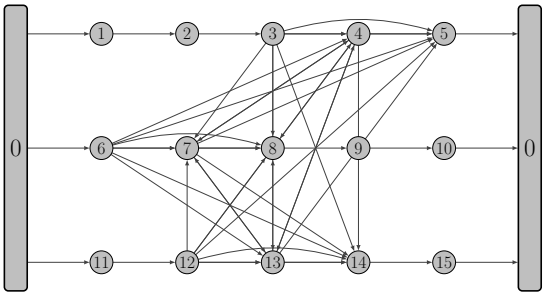
(c) Arcos de entrada para a clique.



(d) Arcos de saída da clique.



(e) Arcos de salto da clique.



(f) Grafo da subinstância.

Figura 5 – Etapas de construção da subinstância a partir da solução original.

Inicialmente, escolhe-se um subconjunto de κ clientes e são adicionados à subinstância todos os arcos entre eles, formando uma clique (Figura 5b). Em seguida, identificam-se os nós de entrada, que são os vértices que precedem imediatamente os nós da clique em suas rotas originais, e os nós de saída, que são os vértices que os sucedem. Três conjuntos adicionais de arcos são então incorporados à subinstância: (i) os arcos entre os nós de entrada e os nós da clique, chamados de arcos de entrada (Figura 5c); (ii) os arcos entre os nós da clique e os nós de saída, chamados de arcos de saída (Figura 5d); e (iii) os arcos diretos entre nós de entrada e nós de saída, denominados arcos de salto (Figura 5e). A inclusão desses arcos visa permitir rearranjos mais amplos e expressivos nas rotas originais, expandindo significativamente o espaço de busca explorado pelo algoritmo.

O subgrafo resultante define a vizinhança $\eta(s)$ da solução atual s, como ilustrado na Figura 5f. Quando apenas os arcos da solução original são considerados, o espaço de busca se restringe à própria estrutura de s. À medida que novos arcos são adicionados, o conjunto de soluções viáveis se expande exponencialmente. Caso todos os arcos do grafo

original sejam incluídos, a vizinhança passa a englobar a instância completa do problema, permitindo sua resolução exata — o que, no entanto, é impraticável para instâncias de grande porte.

4.3 Algoritmo de VDNS

O algoritmo proposto constrói subinstâncias do problema a partir de uma estrutura de vizinhança parametrizada, exploradas por meio de técnicas exatas de otimização e refinamento heurístico. Embora compartilhe a estrutura geral do VDNS apresentado por Barroso e Nepomuceno (2024), este trabalho introduz avanços importantes em duas frentes principais: (i) a estratégia de seleção dos clientes que compõem a clique, visando melhorar a exploração do espaço de busca; e (ii) a construção mais criteriosa das subinstâncias, baseada em múltiplas soluções correlacionadas, ampliando a diversificação sem comprometer a resolubilidade do subproblema.

O Algoritmo 1 apresenta um resumo do procedimento. O processo inicia-se com a geração de uma solução viável s^* por meio de uma heurística gulosa baseada no vizinho mais próximo (NEPOMUCENO; SABOIA; PINHEIRO, 2019). Primeiramente, um cliente é selecionado aleatoriamente da lista, e os clientes seguintes da rota são definidos por uma regra probabilística: ou seleciona-se o cliente mais próximo do último atendido, ou um cliente aleatório, de acordo com uma probabilidade estabelecida. Esse processo continua até que não seja possível acrescentar novos clientes sem exceder a capacidade do veículo. A solução criada é refinada com operadores clássicos de busca local *2-opt*, *insertion*, *interchange* e *sub-path exchange* (GAJPAL; ABAD, 2009), com o objetivo de melhorar sua qualidade antes do início do processo iterativo. A solução refinada é armazenada em um conjunto de soluções S, que será utilizado como base para a construção das vizinhanças nas iterações seguintes.

A cada iteração do VDNS, seleciona-se um subconjunto de κ clientes para compor a clique, a partir da qual será definida a subinstância a ser resolvida. Diferentemente da abordagem de Barroso e Nepomuceno (2024), em que os clientes eram escolhidos aleatoriamente, esta proposta adota uma estratégia sistemática com três objetivos: (i) evitar a repetição de clientes já selecionados, promovendo melhor cobertura do espaço de busca ao longo da execução; (ii) favorecer a seleção de clientes geograficamente próximos, com base em uma roleta ponderada inversamente proporcional à distância em relação a um cliente de referência; e (iii) garantir diversidade, reinicializando a lista de candidatos sempre que todos os clientes tiverem sido utilizados. Essas estratégias buscam compensar o número reduzido de iterações típicas de métodos baseados em subinstâncias, proporcionando melhor exploração do espaço global de soluções. A subinstância é construída a partir da união das vizinhanças $\eta(s)$ de cada solução $s \in S$:

Algoritmo 1: Pseudocódigo do algoritmo VDNS.

Criar um conjunto de soluções $S = \emptyset$;
Gerar uma solução viável s^* e refiná-la com busca local;
Adicionar s^* ao conjunto de soluções S;
while *critério de parada não satisfeito* **do**
  Selecionar κ clientes para a clique com base em histórico e proximidade;
  Definir os arcos da subinstância como $\mathcal{A} = \bigcup_{s \in S} \eta(s)$;
  Gerar o modelo μ induzido por $\mathcal{A}$;
  Resolver o modelo μ com tempo limite τ_{solver};
  Seja γ o *gap* de otimalidade retornado;
  Ajustar o tamanho da clique κ com base em γ;
  Seja L o conjunto das λ melhores soluções encontradas;
  Refinar cada solução $s \in L$ por busca local;
  Atualizar s^* com a melhor solução em $S \cup L$;
  Atualizar o conjunto de soluções $S = \{s^*\} \cup L$;
end
return s^*;

$$\mathcal{A} = \bigcup_{s \in S} \eta(s)$$

A construção da subinstância é um dos elementos centrais do algoritmo. Em Barroso e Nepomuceno (2024), a vizinhança era definida a partir de uma única solução atual, que também era fornecida como ponto de partida ao *solver* (*warm start*), favorecendo a convergência do *gap* de otimalidade da subinstância. Neste trabalho, adota-se uma abordagem distinta: a solução atual não é fornecida ao *solver*, permitindo uma exploração mais livre da subinstância e favorecendo a geração de um conjunto mais diverso de soluções. Ao final da resolução da subinstância, respeitando-se um limite de tempo τ_{solver}, selecionam-se as λ melhores soluções, que são refinadas por busca local e complementadas pela melhor solução da execução, atualizando o conjunto S de soluções a ser utilizado na próxima iteração. Como essas soluções são obtidas a partir da mesma subinstância, tendem a ser estruturalmente próximas entre si, o que permite explorar vizinhanças amplificadas sem provocar explosão combinatória.

A cada iteração, o valor de κ é ajustado dinamicamente com base no *gap* de otimalidade γ retornado pelo *solver*: se $\gamma \leq 1\%$, o tamanho da clique é aumentado em uma unidade; caso contrário, é reduzido. Essa adaptação busca equilibrar a profundidade da vizinhança com o tempo disponível para resolução do subproblema, permitindo ao *solver* explorá-lo suficientemente bem em cada iteração. O processo se repete até que o tempo total de execução τ_{vdns} seja atingido.

5 Experimentos Computacionais

Para avaliar as potencialidades do método proposto, foram realizados experimentos computacionais utilizando instâncias de *benchmark* da literatura, especificamente as instâncias CMTX e CMTY de Salhi e Nagy (1999) com número de clientes variando entre 50 e 199. Os algoritmos foram implementados em Java 23.0.1 e executados em um processador Intel Core i7-14650HX (5.2 GHz) com 64 GB de RAM DDR5. A resolução dos modelos em PLI foi realizada com o *solver* IBM ILOG CPLEX 22.1.1. Os algoritmos avaliados incluem o HH-ILS (OLGUN; KOÇ; ALTIPARMAK, 2021), o VDNS Original (BARROSO; NEPOMUCENO, 2024) e a versão aprimorada do VDNS proposta neste trabalho. Os parâmetros utilizados nos experimentos foram $\tau_{vdns} = 3.600\ s$, $\tau_{solver} = 30\ s$ e $\lambda = 5$. O CPLEX foi configurado com ênfase heurística (MIPEmphasis = 5) para que o *solver* trabalhe exclusivamente em encontrar soluções viáveis de alta qualidade o mais cedo possível, e o melhor gap também é calculado, mas quase nenhum esforço é feito para provar a otimalidade (INTERNATIONAL BUSINESS MACHINES CORPORATION, 2022). Os algoritmos VDNS e HH-ILS estão disponíveis respectivamente em github.com/VictorTiezzi/G-VRPSPD_VDNS e github.com/VictorTiezzi/G-VRPSPD-HH-ILS.

5.1 Avaliação das Estratégias de Construção e Diversificação

Antes de realizar os experimentos, foi conduzido um estudo para avaliar o impacto das diferentes estratégias de construção da clique e formação da subinstância no desempenho do VDNS. A Tabela 2 resume os resultados obtidos com as combinações das três estratégias propostas:

- **Roleta:** favorece a escolha de clientes próximos na construção da clique, com base em uma roleta inversamente proporcional à distância.
- **Cobertura:** impede a repetição de clientes na clique até que todos tenham sido utilizados, promovendo uma melhor exploração do espaço de busca.
- **Multisolução:** constrói a subinstância a partir da união dos arcos de múltiplas soluções correlacionadas, em vez de uma única solução, o que permite um equilíbrio entre diversificação e resolubilidade.

Cada linha da tabela representa uma configuração distinta de estratégias ativadas (✓) ou desativadas (✗). As colunas indicam o melhor valor e a média das soluções encon-

tradas em 10 execuções. Os experimentos foram conduzidos em três instâncias do conjunto de *benchmarks*: CMT11X (120 clientes), CMT4X (150 clientes) e CMT5X (199 clientes).

Tabela 2 – Resultados das estratégias de construção e diversificação do VDNS

				CMT11X		CMT4X		CMT5X	
Config.	Roleta	Cobert.	Multisol.	Melhor	Média	Melhor	Média	Melhor	Média
A	✗	✗	✗	1422,93	1460,63	1417,33	1450,66	1777,61	1807,38
B	✓	✗	✗	1419,41	1445,31	1406,93	1432,36	1755,29	1785,59
C	✗	✓	✗	1421,60	1436,91	1409,11	1444,25	1749,66	1796,35
D	✗	✗	✓	1419,12	1419,13	1392,54	1393,40	1704,23	1706,23
E	✓	✓	✗	1421,44	1450,62	1423,20	1448,82	1742,46	1783,88
F	✓	✗	✓	1419,12	1419,13	1392,54	1394,31	1704,56	1705,65
G	✗	✓	✓	1419,12	1419,13	1392,54	1393,34	1704,38	1705,91
H	✓	✓	✓	1419,12	1419,12	1392,54	1393,78	1704,38	1706,69

Legenda: ✓: Estratégia ativada ✗: Estratégia desativada

Fonte: Produzido pelo autor

Os resultados indicam que a estratégia de multisolução (configuração D) apresenta o maior impacto isolado na melhoria da qualidade das soluções, com reduções substanciais nos custos médios em todas as instâncias testadas. A estratégia de roleta (configuração B) também contribui positivamente, enquanto a estratégia de cobertura (configuração C) apresentou ganhos mais discretos, mas ainda superiores à ausência total de estratégias (A). A combinação entre as estratégias gera resultados progressivamente melhores. A configuração H, que ativa as três estratégias simultaneamente, obteve os melhores resultados médios em duas das três instâncias e manteve desempenho robusto mesmo na maior delas (CMT5X), confirmando o caráter complementar das abordagens. Por essa razão, a configuração H foi adotada como padrão nas avaliações posteriores.

5.2 Avaliação Comparativa

A Tabela 3 apresenta uma comparação entre três métodos: o HH-ILS (conforme descrito em Olgun, Koç e Altıparmak (2021) e reimplementado neste trabalho), o VDNS original proposto por Barroso e Nepomuceno (2024), e a versão aprimorada desenvolvida neste estudo (VDNS proposto). Para cada instância, são reportados o melhor valor obtido e a média das soluções ao longo de 30 execuções.

Os resultados evidenciam que o VDNS proposto apresenta desempenho robusto e consistentemente superior nas instâncias avaliadas. Em nenhuma delas ele foi superado por outro método, seja em termos do melhor valor obtido ou da média das soluções. Em parte das instâncias, houve empate com outros métodos no melhor valor, mas o VDNS proposto manteve superioridade na média, indicando maior robustez. Nas instâncias de menor porte (com até 75 clientes), como CMT01X, CMT06X e CMT02Y, os três métodos alcançaram os mesmos melhores valores. No entanto, a média das soluções favoreceu o

Tabela 3 – Comparação entre HH-ILS, VDNS Original e VDNS Proposto.

Instância	N_c	HH-ILS			VDNS Original		VDNS Proposto	
		Olgun *et al.* 2021	Melhor	Média	Melhor	Média	Melhor	Média
CMT01X	50	745,10	**741,68**	742,26	**741,68**	742,48	**741,68**	**741,68**
CMT01Y	50	742,21	**741,68**	742,41	744,37	744,88	**741,68**	**741,68**
CMT06X	50		**741,68**	742,12	**741,68**	743,56	**741,68**	**741,68**
CMT06Y	50		**741,68**	742,26	**741,68**	743,81	**741,68**	**741,68**
CMT02X	75	1155,69	**1118,68**	1132,62	1121,72	1127,36	**1118,68**	**1118,68**
CMT02Y	75	1124,92	**1118,68**	1127,61	**1118,68**	1126,02	**1118,68**	**1118,68**
CMT07X	75		**1118,68**	1130,33	**1118,68**	1131,21	**1118,68**	**1118,68**
CMT07Y	75		1121,95	1131,28	**1118,68**	1123,74	**1118,68**	**1118,68**
CMT03X	100	1177,70	**1133,06**	1136,56	1133,52	1141,95	**1133,06**	**1133,14**
CMT03Y	100	1224,17	**1133,06**	1135,90	1133,81	1143,31	**1133,06**	**1133,14**
CMT08X	100		**1133,06**	1137,72	**1133,06**	1145,69	**1133,06**	**1133,14**
CMT08Y	100		**1133,06**	1133,66	1133,36	1140,19	**1133,06**	**1133,19**
CMT12X	100	1111,18	**1100,70**	1102,08	**1100,70**	1103,19	**1100,70**	**1100,70**
CMT12Y	100	1150,35	**1100,70**	1102,27	**1100,70**	1108,97	**1100,70**	**1100,70**
CMT14X	100		**1100,70**	1100,76	**1100,70**	1102,68	**1100,70**	**1100,70**
CMT14Y	100		**1100,70**	1100,95	**1100,70**	1105,96	**1100,70**	**1100,70**
CMT11X	120	1529,96	**1419,12**	1422,41	1419,87	1433,51	**1419,12**	**1419,12**
CMT11Y	120	1478,03	**1419,12**	1422,71	1419,20	1438,06	**1419,12**	**1419,12**
CMT13X	120		1419,40	1420,62	1419,32	1427,52	**1419,12**	**1419,12**
CMT13Y	120		**1419,12**	1422,95	1421,50	1430,31	**1419,12**	**1419,12**
CMT04X	150	1446,85	1395,78	1406,88	1412,77	1424,70	**1392,54**	**1392,83**
CMT04Y	150	1462,87	**1392,54**	1403,24	1407,81	1424,08	**1392,54**	**1393,11**
CMT09X	150		**1392,54**	1405,49	1403,08	1419,68	**1392,54**	**1393,25**
CMT09Y	150		1401,26	1405,05	1409,19	1423,43	**1392,54**	**1393,28**
CMT05X	199	1803,86	1718,09	1741,63	1726,28	1757,11	**1703,87**	**1705,46**
CMT05Y	199	1785,54	1717,89	1741,46	1734,00	1772,22	**1703,87**	**1705,05**
CMT10X	199		1709,54	1736,37	1730,36	1752,94	**1703,87**	**1705,49**
CMT10Y	199		1705,90	1739,32	1731,06	1782,04	**1704,23**	**1704,92**

Fonte: Produzido pelo autor.

VDNS proposto em praticamente todos os casos, demonstrando maior estabilidade. Já em instâncias de maior porte, como CMT04X, CMT05X e CMT10Y (com até 199 clientes), o VDNS proposto obteve as melhores soluções tanto em valor mínimo quanto médio, destacando-se significativamente em cenários de maior complexidade.

Apesar de apresentar bons resultados em instâncias pequenas e moderadas, o desempenho do HH-ILS decai em problemas maiores, tanto em qualidade das soluções quanto em variabilidade dos resultados. Já o VDNS original apresenta desempenho competitivo, mas foi superado pelo VDNS proposto, especialmente em termos de estabilidade das soluções. Em síntese, os dados mostram que as melhorias incorporadas ao VDNS proposto não apenas aumentam a qualidade das soluções finais, mas também conferem maior consistência aos resultados.

5.3 Comparação com a Resolução Direta do Modelo

Nesta análise, compara-se o desempenho do VDNS proposto com o CPLEX aplicado à formulação completa do G-VRPSPD, em duas configurações: (i) com *warm start*, utilizando a mesma solução inicial adotada pelo VDNS; e (ii) sem solução inicial. Todas

as abordagens são executadas com limite de 1 hora por instância. A Tabela 4 resume os resultados, apresentando o valor da solução inicial, os valores finais obtidos pelo CPLEX (com e sem *warm start*) e o valor final alcançado pelo VDNS proposto.

Tabela 4 – Comparação entre CPLEX e o VDNS Proposto

Instância	Solução Inicial	CPLEX com SI	GAP (%) com SI	CPLEX sem SI	GAP (%) sem SI	VDNS Proposto
CMT01X	748,83	746,58	12,36	753,23	12,75	**741,68**
CMT01Y	750,62	744,31	11,74	743,01	11,75	**741,68**
CMT06X	745,75	745,06	12,41	753,23	12,75	**741,68**
CMT06Y	748,07	745,13	12,24	743,01	11,75	**741,68**
CMT02X	1144,02	1144,02	13,75	1166,77	15,69	**1118,68**
CMT02Y	1149,47	1149,47	14,27	1138,85	13,87	**1118,68**
CMT07X	1138,82	1134,06	13,28	1168,91	15,83	**1118,68**
CMT07Y	1143,98	1142,63	14,22	1146,16	14,42	**1118,68**
CMT03X	1141,05	1141,05	14,80	1197,60	18,97	**1133,06**
CMT03Y	1149,25	1149,25	15,25	1166,12	16,91	**1133,06**
CMT08X	1140,28	1140,28	14,95	1243,74	21,98	**1133,06**
CMT08Y	1148,24	1146,57	15,55	1164,23	16,78	**1133,06**
CMT12X	1102,65	1102,65	15,30	1132,73	16,61	**1100,70**
CMT12Y	1102,12	1102,12	14,99	1148,82	18,48	**1100,70**
CMT14X	1102,61	1102,61	14,90	1132,73	16,61	**1100,70**
CMT14Y	1104,64	1102,03	13,59	1148,82	18,48	**1100,70**
CMT11X	1439,62	1439,62	18,62	1635,70	30,36	**1419,12**
CMT11Y	1427,65	1427,65	18,81	1874,76	39,36	**1419,12**
CMT13X	1426,63	1426,63	17,90	1635,70	30,36	**1419,12**
CMT13Y	1436,21	1433,59	19,27	1874,76	39,36	**1419,12**
CMT04X	1424,02	1419,76	16,93	1679,28	34,96	**1392,54**
CMT04Y	1433,03	1432,45	20,20	1630,31	28,50	**1392,54**
CMT09X	1434,21	1434,21	17,67	1761,13	37,98	**1392,54**
CMT09Y	1443,14	1436,85	23,87	1630,31	30,35	**1392,54**
CMT05X	1771,71	1764,46	22,85	2103,67	35,37	**1703,87**
CMT05Y	1758,88	1758,88	22,59	2171,37	37,35	**1703,87**
CMT10X	1779,19	1779,19	23,49	2103,67	35,37	**1703,87**
CMT10Y	1791,52	1791,52	24,00	2171,37	37,35	**1704,23**

Fonte: Produzido pelo autor

Os resultados demonstram que o VDNS proposto superou o CPLEX em todas as instâncias testadas, independentemente da configuração adotada. Mesmo quando o CPLEX partiu da mesma solução de entrada, foi incapaz de alcançar os mesmos valores obtidos pelo VDNS. A diferença se acentua nas instâncias de maior porte (como CMT05X, CMT09X e CMT10Y), indicando que o mecanismo de geração de subinstâncias do VDNS proporciona um balanceamento mais eficaz entre exploração local intensiva e custo computacional.

6 Conclusão

Este trabalho apresentou avanços na aplicação do algoritmo de Busca em Vizinhança com Profundidade Variável ao G-VRPSPD. As principais contribuições concentram-se na incorporação de estratégias voltadas à definição mais eficiente das vizinhanças, incluindo mecanismos sistemáticos de seleção de clientes, com base em critérios de cobertura e proximidade geográfica, e a construção de subinstâncias a partir de múltiplas soluções correlacionadas.

Essas estratégias promovem um equilíbrio eficaz entre intensificação e diversificação da busca, resultando em soluções de alta qualidade e maior robustez. Os experimentos computacionais confirmam a superioridade do VDNS proposto em relação ao HH-ILS e ao VDNS original, com destaque para as instâncias de maior porte. Além disso, a comparação direta com a resolução completa via CPLEX evidencia a eficiência do uso de subinstâncias, mesmo sem *warm start*, na obtenção de resultados superiores dentro do mesmo tempo de execução.

Como direções futuras, propõe-se investigar mecanismos adaptativos para o controle do parâmetro de profundidade da clique e o uso de estratégias dinâmicas na formação do conjunto de soluções. A aplicação do método a variantes mais complexas do VRP — como aquelas com múltiplos objetivos, janelas de tempo ou frota heterogênea — também representa uma linha promissora de pesquisa.

Referências

AHUJA, R. K. et al. A survey of very large-scale neighborhood search techniques. *Discrete Applied Mathematics*, v. 123, n. 1, p. 75–102, 2002. Citado 4 vezes nas páginas 13, 14, 22 e 23.

ALINAGHIAN, M.; NADERIPOUR, M. A novel comprehensive macroscopic model for time-dependent vehicle routing problem with multi-alternative graph to reduce fuel consumption: A case study. *Computers & Industrial Engineering*, v. 99, p. 210–222, 2016. Citado na página 27.

ALTNER, D. S. et al. Very large-scale neighborhood search. In: _____. *Search Methodologies: Introductory Tutorials in Optimization and Decision Support Techniques*. [S.l.]: Springer US, 2014. p. 339–367. Citado na página 13.

ASGHARI, M.; MIRZAPOUR AL-E-HASHEM, S. M. J. Green vehicle routing problem: A state-of-the-art review. *International Journal of Production Economics*, v. 231, p. 107899, 2021. Citado na página 12.

AVCI, M.; TOPALOGLU, S. A hybrid metaheuristic algorithm for heterogeneous vehicle routing problem with simultaneous pickup and delivery. *Expert Systems with Applications*, v. 53, p. 160–17, 2016. Citado na página 27.

BARROSO, D. O.; NEPOMUCENO, N. Um algoritmo de busca em vizinhança com profundidade variável para o problema de roteamento de veículos com coletas e entregas simultâneas. In: *Anais do LVI Simpósio Brasileiro de Pesquisa Operacional (SBPO 2024)*. Fortaleza, Brasil: Galoá, 2024. Citado 11 vezes nas páginas 13, 14, 15, 16, 27, 29, 30, 32, 33, 34 e 35.

BLUM, C. et al. Construct, merge, solve & adapt a new general algorithm for combinatorial optimization. *Computers & Operations Research*, v. 68, p. 75–88, 2016. ISSN 0305-0548. Citado 2 vezes nas páginas 15 e 24.

BLUM, C.; RAIDL, G. R. Hybridization based on problem instance reduction. In: _____. *Hybrid Metaheuristics: Powerful Tools for Optimization*. [S.l.]: Springer International Publishing, 2016. p. 45–62. Citado na página 15.

BONDY, J. A.; MURTY, U. S. R. *GRAPH THEORY WITH APPLICATIONS*. [S.l.]: Elsevier Science Publishing Co., Inc., 1976. Citado 2 vezes nas páginas 17 e 18.

DANNA, E.; ROTHBERG, E.; PAPE, C. L. Exploring relaxation induced neighborhoods to improve MIP solutions. *Mathematical Programming*, v. 102, n. 1, p. 71–90, 2005. Citado na página 15.

DANTZIG, G. B. Linear programming. *Operations Research*, v. 50, n. 1, p. 42–47, 2002. Citado na página 20.

DANTZIG, G. B.; RAMSER, J. H. The truck dispatching problem. *Management Science*, INFORMS, v. 6, n. 1, p. 80–91, 1959. ISSN 00251909, 15265501. Disponível em: <http://www.jstor.org/stable/2627477>. Citado na página 25.

DEKKER, R.; BLOEMHOF, J.; MALLIDIS, I. Operations research for green logistics – an overview of aspects, issues, contributions and challenges. *European Journal of Operational Research*, v. 219, n. 3, p. 671–679, 2012. Feature Clusters. Citado na página 12.

DELL'AMICO, M.; RIGHINI, G.; SALANI, M. A branch-and-price approach to the vehicle routing problem with simultaneous distribution and collection. *Transportation Science*, v. 40, n. 2, p. 235–247, 2006. Citado na página 27.

DEMIR, E.; BEKTAŞ, T.; LAPORTE, G. A comparative analysis of several vehicle emission models for road freight transportation. *Transportation Research Part D: Transport and Environment*, v. 16, n. 5, p. 347–357, 2011. Citado na página 27.

DETHLOFF, J. Vehicle routing and reverse logistics: The vehicle routing problem with simultaneous delivery and pick-up. *OR Spektrum*, v. 23, n. 1, p. 79–96, 2001. Citado na página 12.

ERDOĞAN, S.; MILLER-HOOKS, E. A green vehicle routing problem. *Transportation Research Part E: Logistics and Transportation Review*, v. 48, n. 1, p. 100–114, 2012. Select Papers from the 19th International Symposium on Transportation and Traffic Theory. Citado 2 vezes nas páginas 12 e 27.

FRANCESCHETTI, A. et al. A metaheuristic for the time-dependent pollution-routing problem. *European Journal of Operational Research*, v. 259, n. 3, p. 972–991, 2017. Citado na página 27.

GAJPAL, Y.; ABAD, P. An ant colony system (ACS) for vehicle routing problem with simultaneous delivery and pickup. *Computers & Operations Research*, v. 36, n. 12, p. 3215–3223, 2009. Citado 2 vezes nas páginas 27 e 32.

GAREY, M. R.; JOHNSON, D. S. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. [S.l.]: W. H. Freeman, 1979. Citado na página 26.

GOETSCHALCKX, M.; JACOBS-BLECHA, C. The vehicle routing problem with backhauls. *European Journal of Operational Research*, v. 42, n. 1, p. 39–51, 1989. ISSN 0377-2217. Disponível em: <https://www.sciencedirect.com/science/article/pii/037722178990057X>. Citado na página 25.

GOKSAL, F. P.; KARAOGLAN, I.; ALTIPARMAK, F. A hybrid discrete particle swarm optimization for vehicle routing problem with simultaneous pickup and delivery. *Computers & Industrial Engineering*, v. 65, n. 1, p. 39–53, 2013. Citado na página 27.

GOLDBARG, M.; GOLDBARG, E. *Grafos*: Conceitos, algoritmos e aplicações. Rio de Janeiro: Elsevier, 2012. ISBN 9788595155756. Disponível em: <https://integrada.minhabiblioteca.com.br/#/books/9788595155756/>. Citado na página 17.

GOLDBARG, M. C.; GOLDBARG, E. G.; LUNA, H. P. L. *Otimização Combinatória e Meta-heurísticas*: Algoritmos e aplicações. Rio de Janeiro: Elsevier, 2015. ISBN 9788595154667. Disponível em: <https://integrada.minhabiblioteca.com.br/#/books/9788595154667/>. Citado 2 vezes nas páginas 17 e 20.

HILLIER, F.; LIEBERMAN, G. *Introdução à Pesquisa Operacional*. AMGH, 2013. ISBN 9788580551198. Disponível em: <https://books.google.com.br/books?id=-A88a0-KxQ0C>. Citado 4 vezes nas páginas 17, 18, 19 e 21.

HUANG, Y. et al. A study on carbon reduction in the vehicle routing problem with simultaneous pickups and deliveries. In: *Proceedings of 2012 IEEE International Conference on Service Operations and Logistics, and Informatics*. [S.l.: s.n.], 2012. p. 302–307. Citado 2 vezes nas páginas 12 e 27.

INTERNATIONAL BUSINESS MACHINES CORPORATION. *IBM ILOG CPLEX Optimization Studio documentation*. [S.l.], 2022. Tópico: Emphasizing feasibility and optimality, Versão 22.1.1. Disponível em: <https://www.ibm.com/docs/en/icos/22.1.1?topic=optimizer-emphasizing-feasibility-optimality>. Citado na página 34.

KOÇ, Ç.; LAPORTE, G.; TÜKENMEZ, İ. A review of vehicle routing with simultaneous pickup and delivery. *Computers & Operations Research*, v. 122, p. 1–15, 2020. Citado na página 26.

MACRINA, G. et al. An energy-efficient green-vehicle routing problem with mixed vehicle fleet, partial battery recharging and time windows. *European Journal of Operational Research*, v. 276, n. 3, p. 971–982, 2019. Citado na página 27.

MCKINNON, A. *Decarbonizing Logistics: Distributing Goods in a Low Carbon World*. London: Kogan Page, 2018. ISBN 978-0749480486. Citado na página 12.

MIN, H. The multiple vehicle routing problem with simultaneous delivery and pick-up points. *Transportation Research Part A: General*, v. 23, n. 5, p. 377–386, 1989. ISSN 0191-2607. Disponível em: <https://www.sciencedirect.com/science/article/pii/019126078990085X>. Citado 2 vezes nas páginas 12 e 26.

MONTANÉ, F. T.; GALVÃO, R. D. A tabu search algorithm for the vehicle routing problem with simultaneous pick-up and delivery service. *Computers & Operations Research*, v. 33, n. 3, p. 595–619, 2006. Citado na página 27.

NEPOMUCENO, N. a.; SABOIA, R.; COELHO, A. A MILP-based very large-scale neighborhood search for the heterogeneous vehicle routing problem with simultaneous pickup and delivery. In: *Proceedings of the Genetic and Evolutionary Computation Conference*. New York, NY, USA: Association for Computing Machinery, 2023. (GECCO '23), p. 330–338. Citado na página 27.

NEPOMUCENO, N. V.; SABOIA, R. B.; PINHEIRO, P. R. A fast randomized algorithm for the heterogeneous vehicle routing problem with simultaneous pickup and delivery. *Algorithms*, Multidisciplinary Digital Publishing Institute, v. 12, n. 8, p. 158, 8 2019. Citado 2 vezes nas páginas 26 e 32.

OLGUN, B.; KOÇ, Ç.; ALTIPARMAK, F. A hyper heuristic for the green vehicle routing problem with simultaneous pickup and delivery. *Computers & Industrial Engineering*, v. 153, p. 107010, 2021. Citado 11 vezes nas páginas 12, 13, 14, 15, 16, 27, 29, 30, 34, 35 e 36.

PARRAGH, S. N.; DOERNER, K. F.; HARTL, R. F. A survey on pickup and delivery problems. *Journal für Betriebswirtschaft*, v. 58, n. 1, p. 21–51, 4 2008. ISSN 1614-631X. Disponível em: <https://doi.org/10.1007/s11301-008-0033-7>. Citado 2 vezes nas páginas 25 e 26.

RIECK, J.; ZIMMERMANN, J. Exact solutions to the symmetric and asymmetric vehicle routing problem with simultaneous delivery and pick-up. *Business Research*, v. 6, n. 1, p. 77–92, 2013. Citado 3 vezes nas páginas 14, 16 e 30.

ROGERS, D. S.; TIBBEN-LEMBKE, R. An examination of reverse logistics practices. *Journal of Business Logistics*, v. 22, n. 2, p. 129–148, 2001. Citado na página 12.

SALHI, S.; NAGY, G. A cluster insertion heuristic for single and multiple depot vehicle routing problems with backhauling. *J. Oper. Res. Soc.*, v. 50, n. 10, p. 1034–1042, 1999. Citado 4 vezes nas páginas 13, 15, 27 e 34.

SANTOS, M. J. et al. Green reverse logistics: Exploring the vehicle routing problem with deliveries and pickups. *Omega*, v. 118, p. 102864, 2023. Citado na página 26.

SARAIVA, R. D.; NEPOMUCENO, N. V.; PINHEIRO, P. R. The generate-and-solve framework revisited: Generating by simulated annealing. In: MIDDENDORF, M.; BLUM, C. (Ed.). *Evolutionary Computation in Combinatorial Optimization*. [S.l.]: Springer Berlin Heidelberg, 2013. p. 262–273. Citado 2 vezes nas páginas 15 e 24.

SUBRAMANIAN, A. et al. A parallel heuristic for the vehicle routing problem with simultaneous pickup and delivery. *Computers & Operations Research*, v. 37, n. 11, p. 1899–1911, 2010. Citado na página 27.

SUBRAMANIAN, A. et al. Branch-cut-and-price for the vehicle routing problem with simultaneous pickup and delivery. *Optim. Lett.*, v. 7, n. 7, p. 1569–1581, 2013. Citado na página 27.

TAHA, H. A. *Operations Research: An Introduction*. 9th. ed. Upper Saddle River, NJ: Pearson, 2011. ISBN 9780132555937. Citado na página 19.

TASAN, A. S.; GEN, M. A genetic algorithm based approach to vehicle routing problem with simultaneous pick-up and deliveries. *Computers & Industrial Engineering*, v. 62, n. 3, p. 755–761, 2012. Citado na página 27.

WOLSEY, L. A. *Integer Programming*. New York: Wiley, 1998. ISBN 9780471283660. Citado na página 20.