UNIVERSIDADE FEDERAL DE JUIZ DE FORA
INSTITUTO DE CIÊNCIAS EXATAS
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

Igor de Andrade Junqueira

Um *Iterated Greedy* e uma mateurística para o problema de roteamento de veículos elétricos com dois níveis e janela de tempo

Juiz de Fora
2024

Igor de Andrade Junqueira

Um *Iterated Greedy* e uma mateurística para o problema de roteamento de veículos elétricos com dois níveis e janela de tempo

Dissertação apresentada ao Programa de Pós-Graduação em Ciência da Computação da Universidade Federal de Juiz de Fora como requisito parcial à obtenção do título de Mestre em Ciência da Computação. Área de concentração: Ciência da Computação

Orientador: Heder Soares Bernardino

Coorientadores: Stênio Sã Rosário Furtado Soares, Luciana Brugiolo Gonçalves

Juiz de Fora

2024

Ficha catalográfica elaborada através do programa de geração
automática da Biblioteca Universitária da UFJF,
com os dados fornecidos pelo(a) autor(a)

de Andrade Junqueira, Igor.
Um Iterated Greedy e uma mateurística para o problema de roteamento de veículos elétricos com dois níveis e janela de tempo / Igor de Andrade Junqueira. -- 2024.
66 p.

Orientador: Heder Soares Bernardino
Coorientadores: Stênio Sã Rosário Furtado Soares, Luciana Brugiolo Gonçalves
Dissertação (mestrado acadêmico) - Universidade Federal de Juiz de Fora, Instituto de Ciências Exatas. Programa de Pós-Graduação em Ciência da Computação, 2024.

1. Roteamento em dois níveis. 2. Roteamento de veículos elétricos. 3. Iterated Greedy. 4. Mateurística. I. Soares Bernardino, Heder, orient. II. Sã Rosário Furtado Soares, Stênio, coorient. III. Brugiolo Gonçalves, Luciana, coorient. IV. Título.

Igor de Andrade Junqueira

Um Iterated Greedy e uma Matheurística para o Problema de Roteamento de Veículos Elétricos com Dois Níveis e Janela de Tempo

Dissertação apresentada ao Programa de Pós-Graduação em Ciência da Computação da Universidade Federal de Juiz de Fora como requisito parcial à obtenção do título de Mestre em Ciência da Computação. Área de concentração: Ciência da Computação.

Aprovada em 04 de abril de 2024.

BANCA EXAMINADORA

Prof. Dr. Heder Soares Bernadino - Orientador

Universidade Federal de Juiz de Fora

Prof. Dr. Stênio Sã Rosário Furtado Soares - Coorientador

Universidade Federal de Juiz de Fora

Profª. Drª. Luciana Brugiolo Gonçalves - Coorientadora

Universidade Federal de Juiz de Fora

Prof. Dr. Carlos Cristiano Hasenclever Borges

Universidade Federal de Juiz de Fora

Prof. Dr. Luiz Satoru Ochi

Universidade Federal Fluminense

Prof. Dr. André Gustavo dos Santos

Universidade Federal de Viçosa

Juiz de Fora, 19/03/2024.

Documento assinado eletronicamente por **Heder Soares Bernardino**, **Professor(a)**, em 03/05/2024, às 12:06, conforme horário oficial de Brasília, com fundamento no § 3º do art. 4º do Decreto nº 10.543, de 13 de novembro de 2020.

Documento assinado eletronicamente por **LUIZ SATORU OCHI**, **Usuário Externo**, em 03/05/2024, às 18:38, conforme horário oficial de Brasília, com fundamento no § 3º do art. 4º do Decreto nº 10.543, de 13 de novembro de 2020.

Documento assinado eletronicamente por **Luciana Brugiolo Goncalves**, **Professor(a)**, em 06/05/2024, às 13:41, conforme horário oficial de Brasília, com fundamento no § 3º do art. 4º do Decreto nº 10.543, de 13 de novembro de 2020.

Documento assinado eletronicamente por **Carlos Cristiano Hasenclever Borges**, **Professor(a)**, em 06/05/2024, às 15:36, conforme horário oficial de Brasília, com fundamento no § 3º do art. 4º do Decreto nº 10.543, de 13 de novembro de 2020.

Documento assinado eletronicamente por **André Gustavo dos Santos**, **Usuário Externo**, em 07/05/2024, às 10:19, conforme horário oficial de Brasília, com fundamento no § 3º do art. 4º do Decreto nº 10.543, de 13 de novembro de 2020.

Documento assinado eletronicamente por **Igor de Andrade Junqueira**, **Usuário Externo**, em 28/05/2024, às 12:04, conforme horário oficial de Brasília, com fundamento no § 3º do art. 4º do Decreto nº 10.543, de 13 de novembro de 2020.

Documento assinado eletronicamente por **Stênio Sã Rosário Furtado Soares**, **Usuário Externo**, em 07/06/2024, às 10:56, conforme horário oficial de Brasília, com fundamento no § 3º do art. 4º do Decreto nº 10.543, de 13 de novembro de 2020.

A autenticidade deste documento pode ser conferida no Portal do SEI-Ufjf (www2.ufjf.br/SEI) através do ícone Conferência de Documentos, informando o código verificador **1755179** e o código CRC **12E2DE7F**.

AGRADECIMENTOS

Aos professores Heder Bernardino, Stênio Sã, Luciana Brugiolo, e Lorenza Moreno, por todos os conselhos, pela ajuda e pela paciência com a qual guiaram o meu aprendizado.

Ao meu Pai e Irmã, por todo o incentivo, amor e carinho. À minha Mãe (*In memoriam*), que me ensinou como se reerguer diante das adversidades da vida.

Aos membros da banca examinadora, Carlos Cristiano, Luiz Satoru, e André Gustavo. Suas sugestões e questionamentos enriqueceram significativamente este trabalho.

O presente trabalho foi realizado com apoio da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES) – Código de Financiamento 88887.679888/2022-00, Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq), e da Fundação de Amparo à Pesquisa do Estado de Minas Gerais (FAPEMIG)

RESUMO

O Problema de Roteamento de Veículos Elétricos com Dois Níveis e Janela de Tempo compreende uma variante do problema clássico de Roteamento de Veículos, em que é necessário atender clientes de um depósito central por depósitos intermediários (satélites). Os satélites atendem os clientes usando Veículos Elétricos, sendo que os satélites são atendidos com Veículos a Combustão que partem do depósito central. Estações de Recarga podem ser utilizadas para estender o alcance dos Veículos Elétricos. Esse trabalho propõem duas abordagens baseadas em *Iterated Greedy* (IG) para resolver o problema de otimização. A primeira consiste em um IG que utiliza operadores de destruição e reconstrução. Além disso, a busca local *Random Variable Neighborhood Descent* é utilizada para obter um mínimo local e, assim, melhorar a solução após a fase de reconstrução. O segundo método proposto consiste em uma matheurística que utiliza o IG proposto inicialmente com um particionamento de rotas. A matheurística usa o IG proposto inicialmente para gerar um conjunto de rotas de Veículos Elétricos. A solução desse problema consiste em uma seleção de um subconjunto de rotas de forma que todos os clientes sejam atendidos, modelo de Programação Linear Inteira Mista, enquanto minimiza o somatório total das distâncias das rotas selecionadas. Os métodos propostos foram comparados com o método do estado da arte e obtiveram resultados significativamente melhores, considerando a distância (função objetivo) e tempo de processamento.

Palavras-chave: Roteamento em Dois Níveis; Roteamento de Veículos Elétricos; Matheurística; *Iterated Greedy*.

ABSTRACT

The Two-Echelon Electric Vehicle Routing Problem with Time Windows comprises a complex variant of the Vehicle Routing Problem that has to attend customs from a central deposit using intermediary depots (satellites). The satellites attend to customers using Electric Vehicles, and the satellites are attended by Combustion Vehicles that leave the central depot. Recharging stations can be used to extend the limited range of Electric Vehicles. The present work proposes two approaches based on Iterated Greedy (IG) to solve the optimization problem. The first one consists of an IG that's utilizes operators of destruction and rebuilding. The local search Random Variable Neighborhood Descent is also utilized to get a local minimal solution after the rebuilding phase. The second approach consists in a matheuristic that uses the proposed IG method with a set covering of routes. The matheuristic uses the proposed IG to create a pool of routes for the Electric Vehicles. This pool is used to formulate a Mixed Integer Linear Programming of the set covering problem. The solution to this problem consists in a selection of a sub-set of Electric Vehicles that attend all customs while minimizing the sum of the select routes distances. Both proposed methods were compared with the state of the art method and obtained significantly better results, concerning distance (objective function) and computational time.

Keywords: Two-Echelon Routing; Electric Vehicle Routing; Matheuristic; Iterated Greedy.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

2E-EVRP-TW	*Two-Echelon Electric Vehicle Routing Problem with Time Window*
ALNS	*Adaptive Large Neighborhood Search*
BSS	*Battery Swap Station*
CG	*Column Generation*
CO_2	Dióxido de Carbono
CV	*Combustion Vehicle*
EV	*Eletric Vehicle*
GRASP	*Greedy Randomized Adaptive Search Procedure*
HVRP	*Heterogeneous Vehicle Routing Problem*
IG	*Iterated Greedy*
LNS	*Large Neighborhood Search*
MDEVRP	*Multi-Depot Eletric Vehicle Routing Problem*
MDVRP	*Multi-Depot Vehicle Routing Problem*
MILP	*Mixed Integer Linear Programming*
MTZ	*Miller-Tucker-Zemli*
RMP	*Restricted Master Problem*
RS	*Recharging Station*
RVND	*Random Variable Neighborhood Descent*
SDEVRP	*Split Delivery Eletric Vehicle Routing Problem*
SPPRC	*Shortest Path Problem with Resource Constraints*
VND	*Variable Neighborhood Descent*
VNS	*Variable Neighborhood Search*
VRP	*The Vehicle Routing Problem*
VRPTW	*The Vehicle Routing Problem with Time Window*

SUMÁRIO

1 Introdução

O setor de transporte consiste na movimentação de mercadorias, pessoas, matéria-prima, entre outros. O planejamento eficiente permite a redução de custos operacionais e o aumento da satisfação do cliente. Além de reduzir custos, o planejamento torna-se essencial em operações complexas envolvendo um alto número de clientes. Formalmente, a distribuição de mercadorias foi definida em 1959 como *Truck Dispatching Problem* por *Dantzig* e *Ramser* (8). Esse problema é conhecido hoje como *Vehicle Routing Problem* (VRP).

O VRP (8) consiste em realizar entregas a um conjunto de clientes utilizando uma frota de veículos. Cada cliente possui uma demanda e existe um depósito central de onde os veículos iniciam e terminam as suas rotas. O depósito e os clientes possuem caminhos que os conectam entre si, e todo caminho possui uma distância associada. O problema consiste em atender a todos os clientes enquanto minimiza a distância total percorrida.

A partir do VRP várias variantes foram propostas. As mais importantes, relacionadas a esse trabalho, são descritas a seguir. O problema *Vehicle Routing Problem with Time Window* (VRPTW) foi introduzido por *Solomon* (25), e consiste em adicionar a restrição do tempo de chegada do veículo aos clientes. O tempo de chegada do veículo em todo cliente c deve estar dentro da sua respectiva janela de tempo $[e_c, l_c]$. O problema *Heterogeneous Vehicle Routing Problem* (HVRP) foi introduzido por Golden, Assad, Levy e Gheysens (12). Esse problema considera uma frota de veículos heterogênea, de forma que cada veículo tem características diferentes, como a capacidade de carga. O problema *Multiple Depot Vehicle Routing Problem* (MDVRP) foi introduzido por Kulkarni and Bhave (18), e consiste no atendimento dos clientes a partir de múltiplos depósitos.

A maioria dos trabalhos onde variantes do VRP são estudadas assume que os veículos utilizam motores a combustão, já que esses são os mais utilizados para o transporte de cargas, segundo Kalghatgi et al. (16). Entretanto, essas suposições estão mudando para considerar outras fontes de energia para o transporte de cargas, que não utilizam combustíveis fósseis, como Veículos Elétricos (EVs, do inglês *Eletric Vehicle*). Essas mudanças são realizadas por pressões de ordem econômica e ambiental [1](União Europeia aprova lei para cortar emissões de metano relacionados a combustíveis fósseis).

A logística verde considera o impacto ambiental no planejamento de distribuição. O impacto é relativo à emissão de gases do efeito estufa, como o CO_2 (principal gás gerado pela combustão). Além do impacto ambiental, a eficiência do sistema também é muito importante. A utilização de Veículos Elétricos tem se intensificado a partir da evolução da tecnologia das baterias e pelo fato desses veículos não produzirem CO_2 diretamente. Entretanto, a poluição dos EVs depende de como está organizada a matriz energética, em

[1] `https://www.theguardian.com/environment/2023/nov/15/eu-agrees-law-to-curb-methane-emissions-fossil-fuel-industry`

relação às emissões de CO_2.

A aplicação do problema VRP apresentado possui limitações, como o peso total que um veículo pode carregar, restrições de circulação em grandes cidades, e o alto custo operacional de veículos pesados, como destacado em (24). A utilização de dois níveis de roteamento, uma variante do VRP, permite a separação da distribuição de bens em dois níveis, e isso possibilita a remoção de veículos pesados dos centros urbanos.

Caminhões pesados são os veículos que possuem o maior custo por quilômetro (km) rodado. Apesar disso, esse tipo de veículo consegue movimentar elevadas quantidades de carga em rodovias. Entretanto, esse tipo de veículo não é adequado ao transporte de cargas em grandes cidades, tendo em vista seu alto custo operacional e limitações de circulação. Estes desafios podem ser contornados utilizando o modelo de dois níveis, que possibilita a combinação de caminhões pesados e leves eficientemente. A movimentação de carga nas cidades fica a cargo dos veículos leves, que realizam rotas rápidas nas cidades devido à sua capacidade máxima de carga ser pequena. Para viabilizar este modelo de distribuição, depósitos intermediários (satélites) são utilizados. No problema *Two-Echelon Electric Vehicle Routing Problem with Time Window* (2E-EVRP-TW) proposto por Akbay et al. (1), caminhões transportam bens do depósito para os satélites, e então EVs (que têm satélites como ponto de partida) realizam o atendimento das demandas dos clientes considerando requisitos de atendimento. A utilização dos satélites permite melhorar a operação logística ao utilizar eficientemente a combinação de veículos pesados e leves.

Este trabalho propõe duas abordagens baseadas em *Iterated Greedy* (20) para o problema 2E-EVRP-TW. O IG utiliza a estratégia de destruir parcialmente uma solução e reconstruí-la utilizando métodos construtivos. A segunda abordagem baseada em IG é uma *matheuristic*. Segundo (11), *matheuristic* é um método que consiste na combinação de metaheurísticas com programação matemática. Esse método utiliza o IG proposto inicialmente em combinação com problema de particionamento de conjuntos, baseado em Programação Linear Inteira Mista (MILP, do inglês *Mixed Integer Linear Programming*).

Experimentos computacionais são realizados utilizando as 92 instâncias de Akbay et al. (1) e os resultados indicam que ambos os métodos propostos possuem melhores resultados para as maiores instâncias quando comparados ao método baseado em *Variable Neighborhood Search* (VNS) (20). A *matheuristic* proposta obtém melhores resultados, considerando a distância, do que o IG proposto inicialmente. Entretanto, a abordagem híbrida consome mais tempo de processamento.

Esse trabalho está organizado da seguinte forma: o Capítulo 2 define formalmente o problema, no Capítulo 3, uma revisão da literatura é apresentada, o Capítulo 4 apresenta em detalhes os métodos propostos, no Capítulo 5 os resultados dos IGs propostos são comparados com um algoritmo da literatura, e o Capítulo 6 conclui o trabalho.

2 Definição do Problema

Esse capítulo descreve o problema *Two-Echelon Electric Vehicle Routing Problem with Time Window* (2E-EVRP-TW), assim como foi definido por Akbay et al. (1). Seja o grafo G (V, A), onde V= $\{V_c \cup V_s \cup V_r \cup V_d\}$ é o conjunto de vértices, e A o conjunto de arestas. O conjunto de vértices é composto pelos subconjuntos: V_c é o conjunto de clientes, V_s é o conjunto de satélites, V_r é o conjunto de estações de recarga (RS) e V_d é o conjunto de depósitos (esse trabalho considera somente um depósito). Todo cliente $i \in V_c$ possui uma demanda positiva d_i, um tempo de serviço s_i, e uma janela de tempo $[e_i, l_i]$. O conjunto de arestas $A = A_1 \cup A_2$, onde $A_1 = \{(i,j) \mid i \neq j \text{ e } i,j \in V_d \cup V_s\}$ são os arcos entre os depósitos e satélites e $A_2 = \{(i,j) \mid i \neq j \text{ e } i,j \in V_s \cup V_c \cup V_r\}$ são os arcos entre satélites, clientes e estações de recarga. Cada arco $a \in A$ está associado a uma distância positiva d_a.

O problema é definido com dois níveis de roteamento, sendo o primeiro envolvendo depósitos e satélites, enquanto o segundo trata do atendimento das demandas dos clientes a partir dos satélites. Os satélites são depósitos intermediários e atendem os clientes utilizando Veículos Elétricos (EVs, do inglês *Eletric Vehicle*). A demanda de cada satélite é definida pela soma das demandas dos clientes atendidos por ele. Os satélites são atendidos por veículos a combustão (CVs, do inglês *Combustion Vehicle*) que saem dos depósitos. Todo cliente deve ser atendido por um único EV e os satélites podem ser atendidos por múltiplos CVs, isso é conhecido como *Split Delivery* (9). Todo veículo possui uma capacidade máxima de carga, Q_{EV} ou Q_{CV}, de acordo com seu tipo.

O consumo da bateria para um EV é calculado como $tx_{bat} \times d_a$, onde tx_{bat} é a taxa de consumo da bateria por unidade de distância. O tempo para percorrer um arco $a \in A$ é definido como $d_a \times C_{dist}$, onde C_{dist} é uma constante. Estações de recarga (RSs, do inglês Recharging Stations) podem ser utilizadas para estender o alcance dos EVs, realizando a recarga completa da bateria. O tempo necessário para recarga da bateria é calculado como $(Ch_{full} - ch_{rs}) \times Ch_{tx}$, onde Ch_{full} é a capacidade total da bateria, ch_{rs} é a bateria restante do veículo ao chegar em uma RS e Ch_{tx} é a taxa de recarga do EV.

Outra característica do problema está relacionada às quantidades máximas de veículos que podem ser utilizados na solução, para EVs (m_{EV}) e CVs (m_{CV}). É importante observar que esses limites não impedem que uma solução utilize menos veículos. O limite relacionado aos EVs considera todos os satélites.

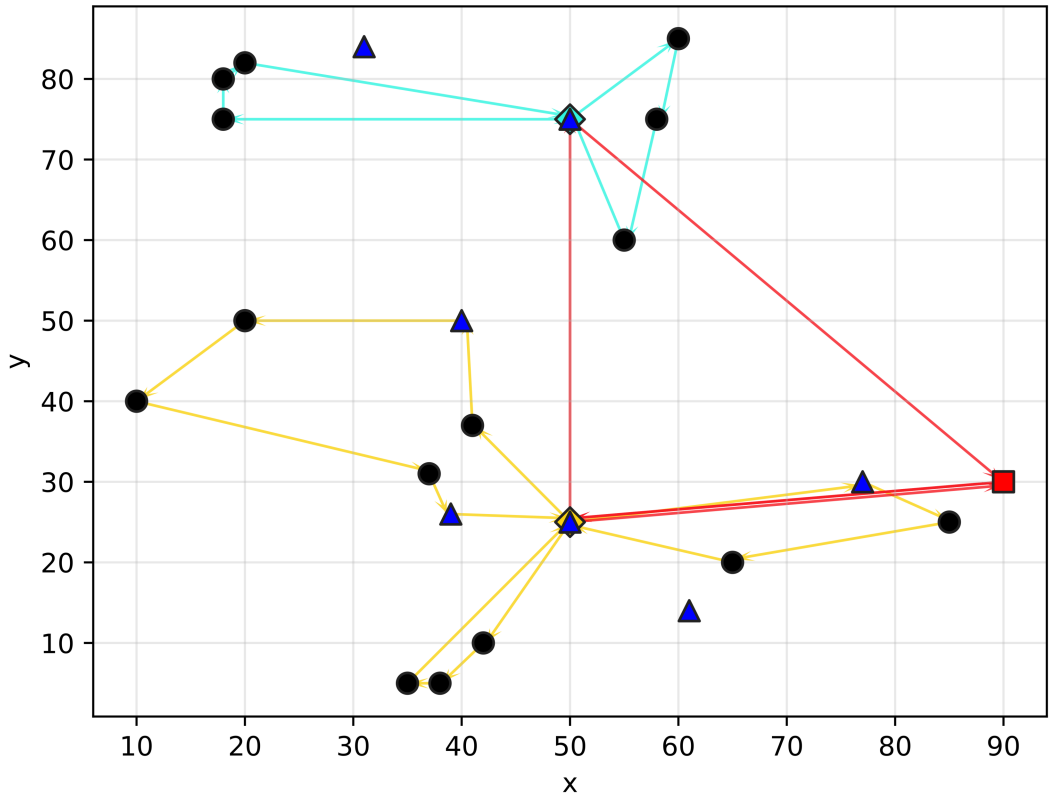
O problema 2E-EVRP-TW pode ser mais facilmente entendido quando o mesmo é separado em dois problemas. O primeiro consiste em associar todo cliente a um satélite, e esse problema é resolvido como um roteamento de veículos elétricos com múltiplos depósitos e janela de tempo (MDEVRP-TW). Após resolver o primeiro problema, é possível determinar a demanda e janela de tempo dos satélites. As demandas dos satélites

são definidas pelas demandas (composta pela soma das demandas dos clientes que são atendidos) das rotas dos EVs que partem do mesmo satélite. A janela de tempo do satélite é obtida da seguinte forma: primeiro calcula-se o maior tempo de saída, com todas as janelas de tempo respeitadas, para cada rota EV, e utiliza-se o menor tempo calculado como tempo de chegada máximo da janela de tempo. O tempo mínimo é calculado pelo tempo do deslocamento entre o depósito e cada satélite. O segundo problema é um Roteamento de Veículos com Janela de Tempo (VPRTW). Apesar do problema ser interpretado como um VRPTW, a diferença é que podem existir satélites com demanda igual a zero e esses não precisam ser visitados pelos CVs.

Uma solução para o problema 2E-EVRP-TW é composta por um conjunto de rotas de EVs e CVs. As rotas dos EVs têm uma sequência de clientes e RSs. Essas rotas devem iniciar e terminar com o mesmo satélite. Além disso, a carga restante da bateria deve ser positiva. Após deixar o satélite, a rota do EV atende um subconjunto dos clientes dentro das suas respectivas janelas de tempo. Considerando que os EVs do satélite i iniciam a rota em um dado momento de tempo tl_i^{sat}, esse satélite deve receber todas as suas demandas, por meio das rotas de CVs, em um instante de tempo menor ou igual a tl_i^{sat}. Outra característica das rotas dos CVs é que somente os satélites com demanda positiva precisam ser atendidos. As rotas dos CVs iniciam no depósito com tempo de saída igual a 0, atendem um subconjunto de satélites e retornam. A função objetivo do problema consiste em minimizar a distância total percorrida pelos veículos elétricos e a combustão.

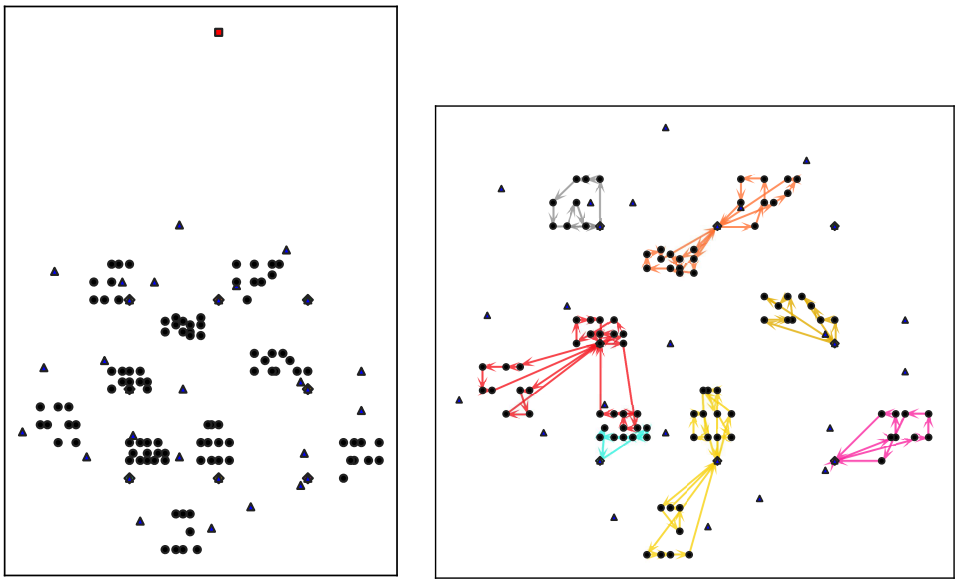
A Figura 1 mostra uma solução de uma instância com 15 clientes, 2 satélites e 7 estações de recarga. Do conjunto de clientes, 6 são atendidos pelo satélite azul e 9 pelo satélite amarelo. Duas das rotas do satélite amarelo utilizam estações de recarga de bateria para realizar o percurso. Para atender as demandas dos satélites, essa solução utiliza a entrega dividida para o satélite amarelo, que é atendido a partir do depósito por duas rotas dos veículos a combustão.

A Figura 2 mostra a instância C105_21x e sua solução. Esta instância, como mostra a Figura 2a, possui 100 clientes, 8 satélites, e 29 estações de recarga. A Figura 2b apresenta uma solução para o segundo nível (roteamento dos clientes com EV), enquanto a Figura 2c mostra uma solução para o primeiro nível (roteamento dos satélites usando CV). Uma solução completa é apresentada na Figura 2d. A solução apresentada respeita as restrições definidas anteriormente, como capacidade dos veículos, atendimento de todos os clientes, e o tempo de chegada dos veículos compatível com a janela de tempo dos clientes e satélites.

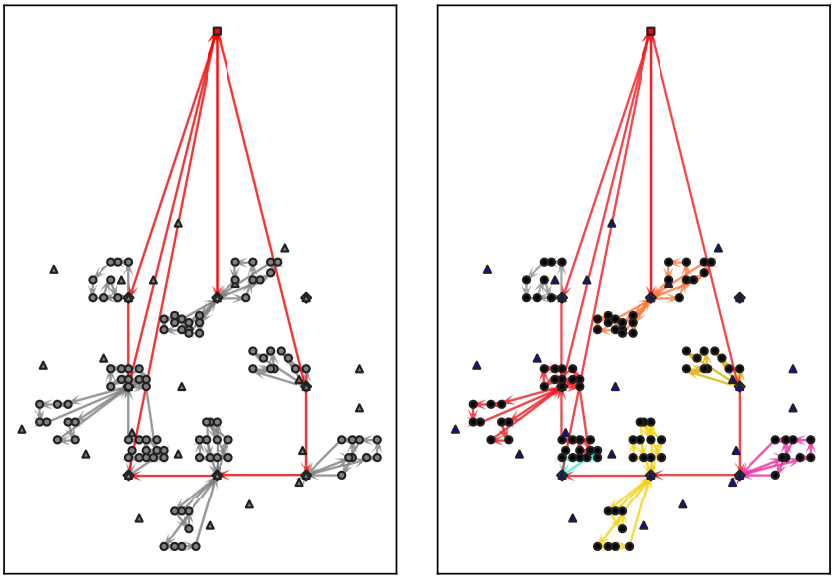


[h]

Figura 1 – Exemplo de uma solução do 2E-EVRP-TW. Os pontos são os clientes, triângulos são as RSs, losangos são os satélites, e o quadrado é o depósito.



(a) Instância. (b) Solução para o 2° nível.



(c) Solução para o 1° nível. (d) Solução completa.

Figura 2 – Exemplo da instância C105_21x e a sua solução. Os pontos são os clientes, triângulos são as RSs, losangos são os satélites, e o quadrado é o depósito.

3 Revisão Bibliográfica

O presente capítulo apresenta trabalhos fortemente relacionados ao problema 2E-EVRP-TW. Os trabalhos selecionados compreendem variações do problema 2E-EVRP. Como o problema estudado tem como referência a definição de Akbay et al. (1), o seu método de solução é apresentado. Após apresentar o primeiro método, os demais trabalhos, definidos em (5, 15, 30), são comparados com o primeiro em termos da definição do problema e métodos de solução.

A solução proposta por Akbay et al. (1) compreende o algoritmo *Variable Neighborhood Search* (VNS). Como o problema possui vários elementos com o potencial de gerar soluções inviáveis, tais como capacidade de carga, bateria dos EVs, e janela de tempo, a função objetivo é estendida para penalizar essas soluções. Para utilizar a penalização eficientemente, os pesos são dinamicamente ajustados em cada iteração com base nas iterações anteriores. A solução inicial do VNS é construída pela heurística de economias de Clarke e Wright (6).

Após gerar uma solução inicial, o VNS utiliza operadores de pertubação e a aplicação de uma busca local. Os operadores são divididos em padrão (*Random cyclic exchange* e *Random sequence relocation*), remoção/destruição (*Random customer removal*, *Random route removal*, e *Close satellite*) e construção (inserção gulosa do cliente, e inserção gulosa de estação de recarga). Os métodos propostos em (1) constituem dois VNSs (VNS_{red} e VNS_{full}), a diferença entre eles está nos operadores. Ambos os métodos, o VNS_{red} e VNS_{full}, utilizam os operadores padrões. Além desses operadores, o VNS_{full} também utiliza os operadores de remoção em conjunto com os operadores de construção.

Ambos VNSs utilizam o critério de aceitação baseado em temperatura, de acordo com Marte et al. (20), esse critério utiliza a temperatura para decidir se uma solução não aprimorante é aceita ou não. Esse critério aceita, com uma maior chance, soluções não aprimorantes quanto maior for a temperatura. Componentes importantes desse critério são a temperatura inicial e a função de decaimento.

O último componente do VNS é a busca local, a qual é o *Variable Neighborhood Descent* (VND). Essa busca utiliza diferentes movimentos para aprimorar uma solução. O critério de parada de ambos os VNSs é o tempo máximo de CPU. Entretanto, os resultados apresentados em (1) têm tempo de CPU inferior ao máximo, e por isso, os métodos devem ter um segundo critério que não é descrito. Esse segundo critério pode ser baseado em número máximo de iterações ou número máximo de iterações sem melhora, por exemplo.

Trabalhos semelhantes àquele em (1), como (5, 15, 30), são discutidos. Com relação à definição do problema, estações de troca de bateria (BSSs, do inglês *Battery Swap Stations*) são utilizadas em (15, 30) para estender o alcance dos EVs. Estações de recarga são utilizadas em (1, 5) com o mesmo propósito. Outra diferença com relação aos EVs é

a sua utilização em ambos os níveis em (15), enquanto os demais trabalhos utilizam-no somente no 2° nível. Restrições de janela de tempo só são utilizadas em (1, 30).

Outra diferença importante relacionada com a definição do problema é a função objetivo. A distância total é utilizada em (1). O custo total, que compreende custos em transporte e utilização do veículo é utilizado em (30). Em (15), também utiliza-se o custo total, entretanto, esse custo é composto pela distância e custos relacionados ao carregamento e descarregamento de carga.

Após discutir a definição do problema, os métodos propostos em (1, 5, 15, 30) são descritos. Esses métodos são baseados em uma única solução. VNS é utilizado em (1, 30). Os métodos baseados em VNS utilizam-se de operadores especiais, que têm como base a destruição e reconstrução parcial das soluções. *Large Neighborhood Search* (LNS) é utilizado em (5) e um *Adaptive Large Neighborhood Search* (ALNS) é combinado com geração de coluna (CG, do inglês *Column Generation*) em (15).

A metaheurística LNS consiste em um conjunto de operadores que destroem partes de uma solução e operadores que reconstroem a solução, de acordo com (20). Operadores de destruição e reconstrução são aplicados aos pares. Cada operador tem a mesma probabilidade de ser escolhido. A diferença para o ALNS é na escolha dos operadores, que consiste em calcular essas probabilidades com base nas iterações anteriores.

Jie et al. (15) dividem o problema em dois, *Multi-Depot Eletric Vehicle Routing Problem* (MDEVRP) e *Split Delivery Eletric Vehicle Routing Problem* (SDEVRP). A heurística construtiva *Sweep* é utilizada para gerar a primeira solução para o MDEVRP. Utilizando a solução corrente (criada pela heurística *Sweep*), o método consiste em utilizar um ALNS, no MDEVRP, para melhorar a solução corrente e CG para resolver o SDEVRP que utiliza os dados da solução do MDEVRP. Um método baseado em *label* é utilizado para encontrar a melhor inserção de BSS nas rotas de EVs, e assim, viabilizá-las. O critério de aceitação é baseado em temperatura no caso da solução corrente não melhorar a melhor solução já encontrada.

Para resolver o SDEVRP, Jie et al. (15) utilizam a decomposição de *Dantzig–Wolfe*. Uma introdução ao tópico pode ser verificada em *Wolsey* (31). O problema mestre consiste em um *set covering* de rotas EVs (primeiro nível). A quantidade de todas as rotas EVs é exponencial, por isso, o problema mestre possui somente um subconjunto de rotas EVs, este chamado de problema mestre restrito (RMP, do inglês *Restricted Master Problem*). O subproblema gera rotas que têm o potencial de entrar na base do RMP, e assim, melhorar o valor da sua função objetivo.

Após resolver o RMP, os valores das variáveis duais, relacionadas às restrições do RMP, são utilizados para formular o subproblema. Esse subproblema é formulado como caminho mínimo com restrição de recursos (SPPRC, do inglês *Shortest Path Problem with Resource Constraints*) (14). O SPPRC é resolvido por um algoritmo de *labeling* (14),

obtendo sempre a rota de menor custo reduzido possível para os valores duais do RMP. Se este custo reduzido for negativo, a rota é adicionada ao RMP. Se não for, nenhuma outra rota precisa ser acrescida ao RMP e a solução corrente é ótima. O método de GC permite obter formulações mais fortes do que a relaxação linear do SDEVRP (para o VRPTW, por *Lallehauge, Larsen, Madsen* e *Solomon* (17)). Como a GC produz soluções fracionadas, o algoritmo utiliza *branching* para alcançar soluções inteiras. Vale ressaltar que o algoritmo é heurístico, por não percorrer toda a árvore de *Branch and bound* (31), o que pode ser muito caro no que diz respeito ao tempo de computação. Além disso, os dados fornecidos para o SDEVRP são de uma heurística, que mesmo resolvido na otimalidade, não pode fornecer a solução ótima para o problema original.

Estratégias de penalização de soluções inviáveis são utilizadas em (1, 15, 30). Esses trabalhos realizam a penalização de uma solução quando existem rotas EV que utilizam mais do que a capacidade de bateria. As violações da janela de tempo dos clientes também são penalizadas em (1, 15). Penalidades relacionadas à violação da capacidade de carga dos veículos só são utilizadas em (1). Um esquema adaptativo dos pesos das penalidades é utilizado em (1, 15).

A solução inicial é uma etapa importante para métodos baseados em uma única solução, como os descritos aqui. Por esse motivo, a maioria dos trabalhos discutidos utilizam heurísticas clássicas para roteamento de veículos a fim de gerar uma primeira solução. A heurística de economias é utilizada em (1, 15). Uma heurística baseada no ângulo entre satélites e clientes é utilizada em (30). A heurística *sweep* é utilizada em (15). A solução inicial em (5) é gerada de forma aleatória, portanto, não se utiliza uma heurística como os demais trabalhos.

Os operadores de destruição da solução utilizados pelo LNS em (15) são: remoção aleatória de cliente, fechamento e abertura de satélite, remoção aleatória de rota, remoção de par BSS-cliente, remoção de BSS, e remoção de BSS baseado em custo. Os operadores de destruição utilizados em (5) são: remoção aleatória de rotas, fechamento de satélite, abertura de todos os satélites, e remoção de rotas com um único cliente.

Os operadores de reconstrução usados em (15) são: inserção gulosa básica, inserção básica de k-arrependimento, inserção gulosa avançada, e inserção avançada de k-arrependimento. A heurística de inserção mais barata é utilizada em (5). Uma abordagem baseada em algoritmo guloso também é utilizada em (1).

Todos os operadores de (1, 5, 15) utilizam variações de algoritmos gulosos. Esse tipo de algoritmo é determinístico, ou seja, sempre produz a mesma solução para uma dada entrada. Ao utilizar um algoritmo determinístico, os métodos apresentados podem ter dificuldade em gerar diferentes soluções. Essa dificuldade pode explicar o alto número de operadores utilizados por esses métodos.

A busca local *Random Variable Neighborhood Descent* (RVND) é utilizada em

(5). O *Variable Neighborhood Descent* (VND) é utilizado em (1, 30). Alguns operadores comuns de busca local utilizados são: *shift*, *swap*, *2-opt* e reinserção de RS. O trabalho de Jie et al. (15) não utiliza busca local.

Outro componente importante para problemas relacionados a veículos elétricos é a inserção de estações de recarga ou troca de bateria para viabilizar as rotas desses veículos. Operadores do VNS que utilizam a inserção mais barata são utilizados em (1, 30) para viabilizar essas rotas. Um algoritmo de *labeling*, que realiza inserções ótimas de estações, é utilizado em (5, 30).

Os trabalhos relacionados apresentados compartilham um problema em comum, entretanto, não é possível comparar diretamente os resultados obtidos por eles, e devido aos problemas estudados serem variações do 2E-EVRP, ou seja, existem pequenas mudanças nas definições de cada autor. Para resolver o 2E-EVRP a técnica mais comum utilizada na literatura é baseada em destruição e reconstrução da solução.

Os métodos propostos nesse trabalho, apresentados no próximo capítulo, têm como a principal diferença entre os trabalhos relacionados discutidos aqui, apenas a utilização de operadores de destruição e reconstrução. Os métodos baseados em VNS em (1) utilizam vários operadores de pertubação além daqueles relacionados a destruição e reconstrução.

4 Métodos de Otimização

Os métodos propostos nesse capítulo compreendem duas abordagens baseadas na metaheurística *Iterated Greedy* (IG) (20). A primeira abordagem trata-se de um IG que utiliza uma busca local *Random Variable Neighborhood Search* (RVND). O segundo método utiliza o IG para gerar um conjunto diverso de rotas e, a partir dessas rotas, um problema de particionamento de conjuntos seleciona o melhor subconjunto de rotas que atende todos os clientes, enquanto minimiza o somatório das distâncias. Não foi encontrado método similar ao proposto para o problema de roteamento de veículos em dois níveis. A Figura 3 mostra uma visão geral do método proposto. Os próximos parágrafos descrevem as abordagens em detalhes.

A metaheurística IG, baseada em uma única solução, utiliza a estratégia de destruir e reconstruir partes de uma solução com o intuito de melhorar a função objetivo. A fase de destruição remove componentes aleatoriamente da solução, e a fase de reconstrução utiliza um algoritmo guloso. Dois algoritmos baseados em IG são propostos aqui. O primeiro método utiliza um IG com busca local. O segundo método utiliza o IG proposto inicialmente em combinação com MILP para o problema de cobertura de conjuntos de rotas de EVs. Esse problema consiste em encontrar a melhor combinação de rotas de EVs enquanto minimiza o somatório total da distância.

O Algoritmo 1 sintetiza os métodos propostos. Um aspecto importante do método é a diferença entre instâncias pequenas (número de clientes menor ou igual a 15) e grandes (número de clientes maior que 15). Essa distinção entre os tipos de instâncias é importante devido a diferença entre a quantidade de EVs disponíveis em cada tipo. O método proposto utiliza a primeira solução viável (linha 1) para gerar os parâmetros (linhas 4, 5 e 6) utilizados no IG (linha 7). É importante notar que a segunda chamada do algoritmo IG, na Linha 10 do Algoritmo 1, tem um número de iterações (parâmetro `numItIGMilp`) muito menor do que a primeira (parâmetro `numItIG`). O parâmetro `execMilp` indica se o modelo de particionamento de rotas é utilizado. Os parâmetros associados ao resolvedor matemático foram agrupados em `MilpParam`, e controlam a intensidade do *presolver*, a intensidade dos cortes, e o *gap* de integralidade. Os demais parâmetros serão explicados ao longo do texto.

Nas próximas seções são apresentados cada um dos componentes do método proposto em detalhes. Na Seção 4.2 é descrita a Heurística de Inserção, na Seção 4.3 a estratégia para obtenção da Solução Inicial, o *Iterated Greedy* é descrito na Seção 4.1, sendo que os métodos para Destruição Parcial da Solução (Seção 4.4), Reparação da Solução (Seção 4.5) e Busca Local (Seção 4.6) apresentados nas seções seguintes. Para tratar da inviabilidade das rotas dos EVs em relação à carga da bateria, um procedimento é descrito na Seção 4.7. Um método baseado em programação matemática é descrito na Seção 4.9.

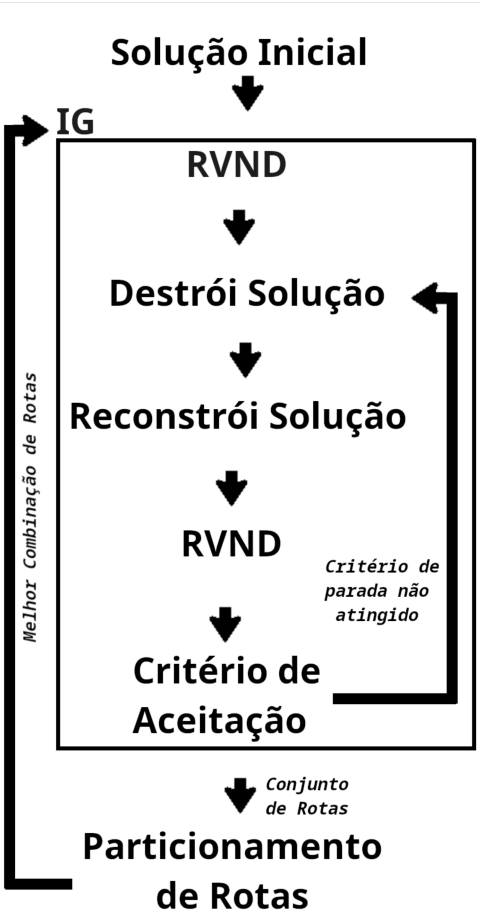


Figura 3 – Visão geral do método proposto.

Algorithm 1: Métodos Propostos.

Input: *numItIG*, $\overrightarrow{\alpha_{ini}}$, *numItMaxGreedy*, *numItUpdateProb*, α_{1n}, α_{2n}, *difBest*, *rmRate*, *estrategiaSelecao*, *multLimitCall*, *execMilp*, *MilpParam*, *numItIGMilp*

Output: Solucao

1 *SolBest* ← geraSolucaoInicial($\overrightarrow{\alpha_{ini}}$, *numItMaxGreedy*, *numItUpdateProb*);
2 **if** *!verrificaViabilidade(SolBest)* **then**
3 **return** SolVazia;
4 *numEvUsados* ← getNumEvUtilizados(*SolBest*);
5 *numRm* ← *rmRate* × *numEvUsados*;
6 *limitCall* ← $\lceil \frac{multLimitCall}{rmRate} \rceil$;
7 *SolBest* ← iteratedGreedy(SolBest, *numItIG*, α_{1n}, α_{2n}, *difBest*, *estrategiaSelecao*, *numRm*, *limitCall*);
8 **if** *execMilp* **then**
9 *SolBest* ← MILPSetCover(SolBest, MilpParam);
10 *SolBest* ← iteratedGreedy(SolBest, *numItIGMilp*, α_{1n}, α_{2n}, *difBest*, *estrategiaSelecao*, *numRm*, *limitCall*);
11 **return** SolBest;

4.1 *Iterated Greedy*

O Algoritmo 2 compreende o IG proposto. O IG utiliza a estratégia de destruição e reconstrução da solução para explorar o espaço de busca da solução. Esse algoritmo recebe uma solução viável e outros parâmetros. O processo iterativo do IG é controlado pela estrutura de repetição da linha 5.

A etapa de destruição ocorre entre as linhas 5 e 10, onde é possível observar que a estratégia de destruição varia dependendo da variável `numChamadas`. A estratégia principal de destruição envolve a remoção de `numRm` rotas de veículos elétricos (linha 6), mas é realizado uma alternância (determinado por `numChamadas`, e `limitCall`) com a estratégia de remover um satélite (linha 8). Essa remoção consiste em excluir todas as rotas que estão associadas a um determinado satélite (escolha aleatória).

A reconstrução da solução ocorre na linha 11. Em seguida, se a solução for viável, ocorre o refinamento via RVND (linha 13). O critério de aceitação (linha 17) decide se `SolC` é atualizada por `Sol'`.

Os componentes do IG proposto são: destruição parcial da solução, reparação da solução, busca local e critério de aceitação. Esses componentes são descritos em detalhes nas próximas seções.

Todas as rotas de EVs das soluções geradas pelo IG proposto são guardadas em um *pool* de rotas. Somente novas rotas são armazenadas, e as repetidas são descartadas.

Algorithm 2: Iterated Greedy para 2E-EVRP-TW.

Input: *SolBest*, *numItIG*, *numItUpdateProb*, α_{1n}, α_{2n}, *difBest*, *estrategiaSelecao*, *numRm*, *limitCall*
Output: Solucao

```
1  SolBest ← rvnd(SolBest);
2  SolC ← copia(SolBest);
3  numChamadas ← 0;
4  for i de 0 até (numItMaxIG-1) do
5      if numChamadas < limitCall then
6          Sol_p ← destroiEVs(SolC, numRm);
7          numChamadas ← numChamadas + 1;
8      else
9          Sol_p ← destroiSat(SolC);
10         numChamadas ← 0;
11     Sol' ← reparaSol(Sol_p, α_1n, α_2n, estrategiaSelecao);
12     if viavel(Sol') then
13         Sol' ← rvnd(Sol');
14         if custo(Sol') < custo(SolBest) then
15             SolBest ← Sol';
16             numChamadas ← 0;
17     SolC ← criterioAceitacao(difBest, Sol', SolC, SolBest);
18 return SolBest;
```

4.2 Heurística de Inserção

Essa seção consiste em explicar a heurística de inserção mais barata. Essa heurística tem como objetivo a construção de uma solução, preferencialmente factível. O método começa com uma solução vazia ou recebe uma solução parcial como parâmetro. Esse algoritmo consiste em inserir um cliente não visitado na posição que tem o menor incremento de distância. A ordem de inserção dos clientes é, usualmente, escolhida de acordo com um critério guloso, por exemplo, a distância entre o cliente e o depósito. A versão modificada utilizada é descrita pelo Algoritmo 3. O processo iterativo, que ocorre entre as linhas 2, e 13, consiste na criação de uma lista de candidatos e na escolha de um candidato para entrar na solução.

A lista de candidatos, criada na linha 1, contém clientes, suas posições, e incremento de distância para soluções candidatas. Essa lista é criada entre as linhas 9, e 12. O próximo candidato é escolhido aleatoriamente entre as linhas 4, e 7.

Pelo fato de um algoritmo guloso não conseguir criar soluções diferentes para uma mesma entrada, o Algoritmo 3 escolhe aleatoriamente o próximo candidato para entrar na solução (da linha 4 à 7). O candidato é escolhido de uma lista de candidatos, essa sendo ordenada crescentemente pelo incremento da distância. A escolha é feita de forma aleatória considerando uma seção da lista de candidatos `listaRestCand`. Os primeiros k candidatos da `listaCand` são considerados, sendo $k = \alpha * |\texttt{listaCand}|$, criando assim a `listaRestCand` definida em função do parâmetro $\alpha \in [0, 1]$.

Existem duas estratégias para escolher aleatoriamente o candidato da lista restrita;

Algorithm 3: Heurística de Inserção.

```
   Input: Sol, tipoHeur, α, estrategiaSelecao
   Output: Solucao
 1 listaCand ← getListaVazia();
 2 do
 3     if listaCand ≠ ∅ then
 4         listaCand ← ordenaPorIncDist(listaCand);
 5         listaRestCand ← getListaRestCand(listaCand, |listaCand|, α);
 6         cand ← escolheCandAleatorio(listaRestCand, estrategiaSelecao);
 7         Sol ← updateSol(Sol, cand);
 8     listaCand ← getListaVazia();
 9     foreach clie ∉ getCliAtend(Sol) do
10         foreach rota ∈ getTodasRotas(Sol) do
11             if (getDemAtual(rota) + getDem(clie)) <= getCapMax(rota) then
12                 listaCand ← atualizaLista(listaCand, tipoHeur, rota, clie);
13 while listaCand ≠ ∅;
14 return Sol;
```

escolha com probabilidade igual para cada candidato da lista restrita `estrategiaSelecao = probIgual`; ou torneio ternário `estrategiaSelecao = torneio`. O torneio ternário escolhe o candidato de menor incremento dentre três candidatos distintos escolhidos aleatoriamente da lista restrita.

A criação da lista de candidato ocorre entre as linhas 9, e 12. O `for` na linha 9 percorre todos os clientes, e o da linha 10, percorre todas as rotas, portanto, temos todos os pares `(clie, rota)`. O `if` na linha 11 verifica se a rota tem capacidade restante para atender o cliente *clie*. Caso exista capacidade, a linha 12 adiciona um conjunto de candidatos de *clie* na rota `rota`. A criação dos candidatos é realizada pela função `atualizaLista`, na linha 12. O parâmetro `tipoHeur` controla como a lista de candidatos é atualizada. Existem duas estratégias, sendo por rota de veículo `tipoHeur = tipoHeurPorRotaVeic` ou por arco `tipoHeur = tipoHeurPorArco`.

Na estratégia por rota de veículo `tipoHeur = tipoHeurPorRotaVeic`, a cada chamada da função `atualizaLista`, é inserido na `listaCand` um candidato que se refere à melhor posição de inserção do cliente na rota. Já na estratégia por arco `tipoHeur = tipoHeurPorArco`, visando conseguir uma quantidade maior de candidatos, importante para as instâncias com pequeno número de clientes, a cada chamada da função `atualizaLista` são inseridos como candidatas todas as posições onde é possível inserir o cliente na rota.

Para gerar uma solução do problema 2E-EVRP-TW, a heurística de inserção é utilizada duas vezes, uma vez para cada um dos níveis.

A primeira vez é usada para resolver o roteamento do segundo nível, um roteamento multi depósito de veículos elétricos com janela de tempo (MEVRPTW). A heurística verifica a viabilidade das soluções no que diz respeito à janela de tempo dos clientes e à capacidade de carga e de nível da bateria dos EVs. Quando uma rota de EV é inviável por conta do nível de bateria, o algoritmo da Seção 4.7 tenta viabiliza-lá realizando a inserção de uma estação de recarga.

Após resolver o segundo nível, a sua solução fornece os dados necessários para o roteamento do primeiro nível. Esses dados são compostos por: demanda de cada satélite (composta pelo somatório das demandas dos clientes atendidos por cada satélite) e janela de tempo de cada satélite (o tempo mais tarde que é possível sair do satélite e atender todos os clientes nas suas respectivas janelas de tempo). O problema de roteamento do primeiro nível, gerado pelos dados do segundo, constitui o problema de roteamento de veículos com entrega dividida e janela de tempo (SDVRPTW). A heurística de inserção para o primeiro nível permite a divisão de carga quando existe um satélite com demanda superior à capacidade do CV.

O algoritmo descrito aqui é utilizado para a construção de uma solução inicial e na fase de reconstrução.

4.3 Solução Inicial

O problema 2E-EVRP-TW pode ser muito desafiador para gerar uma solução viável devido às suas restrições. Para gerar uma primeira solução no método proposto, o Algoritmo 4 utiliza a Heurística de Inserção, descrita na Seção 4.2, e estratégias, que não estão representadas no Algoritmo 4 para a simplificação do mesmo, para contornar inviabilidades.

O Algoritmo 4 realiza várias iterações, no máximo `numItMaxGreedy`, utilizando a Heurística de Inserção e, para isso, são utilizados os parâmetros `tipoHeur = tipoHeurPorRotaVeic` e `estrategiaSelecao = probIgual` em cada um dos dois níveis. Como o Algoritmo 3 utiliza aleatoriedade na escolha do próximo candidato a entrar na solução, cada chamada pode gerar uma solução diferente.

Além dos parâmetros já citados, também é necessária a escolha de α para cada chamada da Heurística de Inserção. Para cada nível é selecionado um valor de $\overrightarrow{\alpha_{ini}}$, sendo α^i_{1n} para o primeiro nível e α^i_{2n} para o segundo, onde o índice i representa a iteração. Ambos os parâmetros são escolhidos aleatoriamente de acordo com uma probabilidade atualizada a cada `numItUpdateProb` iterações, para cada nível, baseado nos resultados obtidos nas iterações anteriores. Esse cálculo utiliza a distância acumulada das soluções parciais mais um termo de penalização. Esse termo é proporcional ao número de clientes não visitados. O peso da penalização utilizada consiste no valor da soma de todos os arcos de G. Essa estratégia de penalização pode aumentar a probabilidade em valores de α que geraram soluções com menor número de clientes não atendidos. Esse método consiste na versão reativa do *Greedy Randomized Adaptive Search Procedure* (Grasp) (22), com a diferença de não utilizar uma busca local e possuir estratégias extras para gerar uma solução viável.

A segunda estratégia visa favorecer clientes que não são atendidos nas soluções. Para isso, calcula-se a probabilidade de um cliente não estar na solução. Essa probabilidade é determinada pela sua frequência nas primeiras `numItUpdateProb` interações. Um cliente é escolhido para iniciar uma rota (utilizando uma versão modificada do problema do caminho mínimo), se um número pseudo-aleatório (entre [0,1)) é maior ou igual à probabilidade calculada. A rota de EV dedicada ajuda clientes que podem precisar de mais de uma estação de recarga para serem atendidos.

A limitação relacionada ao número máximo de rotas EV é tratada pela terceira estratégia. Essa estratégia consiste em inicializar uma rota com uma estação de recarga escolhida de forma aleatória. A segunda e terceira estratégias são necessárias devido à limitação do algoritmo que viabiliza rotas EVs, que não considera a inserção de múltiplas estações de recarga. Essas estratégias são utilizadas alternadamente após as primeiras `numItUpdateProb` iterações.

A versão modificada da inserção mais barata, explicada anteriormente, tem como principais características a utilização de todos os clientes como candidatos, e possui múltiplos candidatos para cada cliente. A modificação permite gerar um número maior de soluções factíveis e para um número maior de instâncias, quando comparado com uma versão que utiliza a ordem dos clientes com base na distância dos satélites.

Algorithm 4: Gera Solução Inicial Viável.

Input: $\overrightarrow{\alpha_{ini}}$, *numItMaxGreedy*, *numItUpdateProb*
Output: Solucao

```
1  vetSolAcumulada1n ← criaVet(|α_ini→|, 0.0);
2  vetSolAcumulada2n ← criaVet(|α_ini→|, 0.0);
3  sol2n ← heuristicaInsercao2n(geraSolVazia(), tipoHeurPorRotaVeic, get(α_ini→, 0),  probIgual);
4  solBest ← heuristicaInsercao1n(sol2n, tipoHeurPorRotaVeic, get(α_ini→, 0),  probIgual);
5  if verificaSolViavel(sol) then
6      return sol;
7  foreach α_i ∈ α_ini→ do
8      addVal(vetSolAcumulada2n, α_i, custoSol2n(solBest) + custoPenal2n(solBest));
9      addVal(vetSolAcumulada1n, α_i, custoSol(solBest) + custoPenal2n(solBest));
10 vetProb1n ← criaVet(|α_ini→|, 1/|α_ini→|);
11 vetProb2n ← criaVet(|α_ini→|, 1/|α_ini→|);
12 i ← 0;
13 while verificaSolInviavel(solBest) and i < numItMaxGreedy do
14     α^i_1n ← escolheRandRoleta(α_ini→, vetProb1n);
15     α^i_2n ← escolheRandRoleta(α_ini→, vetProb2n);
16     sol2n ← heuristicaInsercao2n(geraSolVazia(), tipoHeurPorRotaVeic, α^i_2n,  probIgual);
17     sol ← heuristicaInsercao1n(sol2n, tipoHeurPorRotaVeic, α^i_1n,  probIgual);
18     addVal(vetSolAcumulada2n, α^i_2n, custoSol2n(sol) + custoPenal2n(sol));
19     addVal(vetSolAcumulada1n, α^i_1n, custoSol(sol) + custoPenal2n(sol));
20     if sol < solBest then
21         solBest ← sol;
22     if (i mod numItUpdateProb) == 0 and i ≠ 0 then
23         vetProb2n ← atualizaProb(vetSolAcumulada2n, solBest);
24         vetProb1n ← atualizaProb(vetSolAcumulada1n, solBest);
25     i ← i + 1;
26 return solBest;
```

4.4 Destruição Parcial da Solução

A destruição parcial da solução corrente fica a cargo de dois operadores, sendo eles: destrói EVs (`destroiEVs`) e destrói satélite (`destroiSat`). Em ambos os casos, um ou mais componentes de uma solução são escolhidos aleatoriamente para serem removidos. É importante notar que somente uma pequena parte da solução é destruída no processo, caso contrário, se perderia toda a informação associada a uma solução, o que não é o objetivo dessa abordagem.

O operador `destroiEVs` remove `numRm` rotas de EVs de uma solução. Esse parâmetro é calculado como uma fração do número de rotas EVs utilizadas na primeira solução viável encontrada, tal como `numRm` = $rmRate \times numEvUsed$, sendo $rmRate$ um parâmetro. O operador `destroiSat` consiste em remover todas as rotas de EVs relacionadas ao satélite que está sendo excluído. Essa estratégia é necessária devido à dificuldade intrínseca do problema em se determinar o satélite ideal para um determinado cliente.

O operador `destroiEVs` é aplicado primeiro por `limitCall` iterações. Após atingir o limite especificado de iterações sem melhora, o operador `destroiSat` é utilizado para remover um único satélite da solução. Quando um satélite é removido, o operador `destroiEVs` continua a ser aplicado na próxima iteração. A expressão $\lceil \frac{multLimitCall}{rmRate} \rceil$ determina o número de chamadas necessárias para remover todas as rotas EVs, considerando que cada chamada remove `numRm` de uma solução.

Testes utilizando as instâncias de Akbay et al. (1) mostraram que aumentar o número de `limitCall` pode melhorar os resultados. Para isso, o número `limitCall` é multiplicado pelo parâmetro `multLimitCall`, onde `multLimitCall` $\in [1, 3]$.

4.5 Reparação da Solução

A fase de reparação (`reparaSol`) do IG proposto visa tratar a solução que saiu do processo de destruição parcial. Para isso, é utilizada a heurística de inserção descrita na Seção 4.2. Duas abordagens são utilizadas para reparar uma solução, onde a primeira consiste em aplicar duas vezes, sequencialmente, a heurística de inserção utilizando o parâmetro `tipoHeur = tipoHeurPorRotaVeic`. Cada chamada resolve um nível de roteamento. A aplicação da heurística ocorre semelhantemente a utilizada na solução inicial. A segunda abordagem realiza uma mudança no parâmetro `tipoHeur = tipoHeurPorArco`, para o segundo nível.

Ambas as abordagens usam parâmetros distintos para α para cada nível, sendo eles α_{1n} e α_{2n}. A primeira abordagem é utilizada para instâncias grandes e a segunda para as pequenas. As instâncias pequenas precisam de uma abordagem diferente porque elas possuem um número pequeno (quando comparado às instâncias grandes) de rotas EVs e isso faz com que a primeira abordagem produza as mesmas soluções.

4.6 Busca Local

O algoritmo *Variable Neighborhood Descent* (VND) é uma busca local que combina várias estruturas de vizinhança (29). O algoritmo utiliza uma sequência de vizinhanças, sequencialmente, para melhorar uma solução. A aplicação recomeça, da primeira estrutura, quando uma solução de melhora é encontrada, e se todas as vizinhanças são utilizadas sem que uma solução de melhora seja encontrada, o algoritmo atinge seu critério de parada.

O IG proposto utiliza uma versão modificada do VND, o *Random Variable Neighborhood Search* (RVND) (26, 28), como busca local. Essa variante usa, a cada chamada do algoritmo, uma sequência aleatória de vizinhanças. Com o RVND não é necessário escolher a ordem de aplicação dos movimentos. As vizinhanças utilizadas no RVND são definidas nos próximos parágrafos.

A Figura 4 ilustra os movimentos de vizinhança utilizados no RVND, detalhados em (27). A Figura 4a mostra a solução de referência. As Figuras 4b, 4c, 4d, 4e, 4f, e 4g mostram o resultado após aplicação de cada um dos movimentos na solução de referência.

A seguir são apresentadas as estruturas de vizinhança intra-rota, ou seja, aquelas que envolvem somente uma rota. A Figura 4b mostra o resultado da vizinhança *shift* que escolhe um cliente (1) e um arco $(0, 6)$. A aplicação do movimento consiste em remover o cliente (1) e inseri-lo na posição entre os clientes (0) e (6), criando a rota $(0-1-6-3-0)$. A Figura 4d mostra o resultado da vizinhança *swap*, dois clientes (2 e 7) são escolhidos e trocam as suas posições na rota. A Figura 4f mostra o resultado da vizinhança *2-opt*, duas arestas não consecutivas são removidas, (7,4) e (5,0), e a sub-rota entre os vértices 4 e 5 é invertida, dando origem a rota $(0-7-5-2-8-4-0)$.

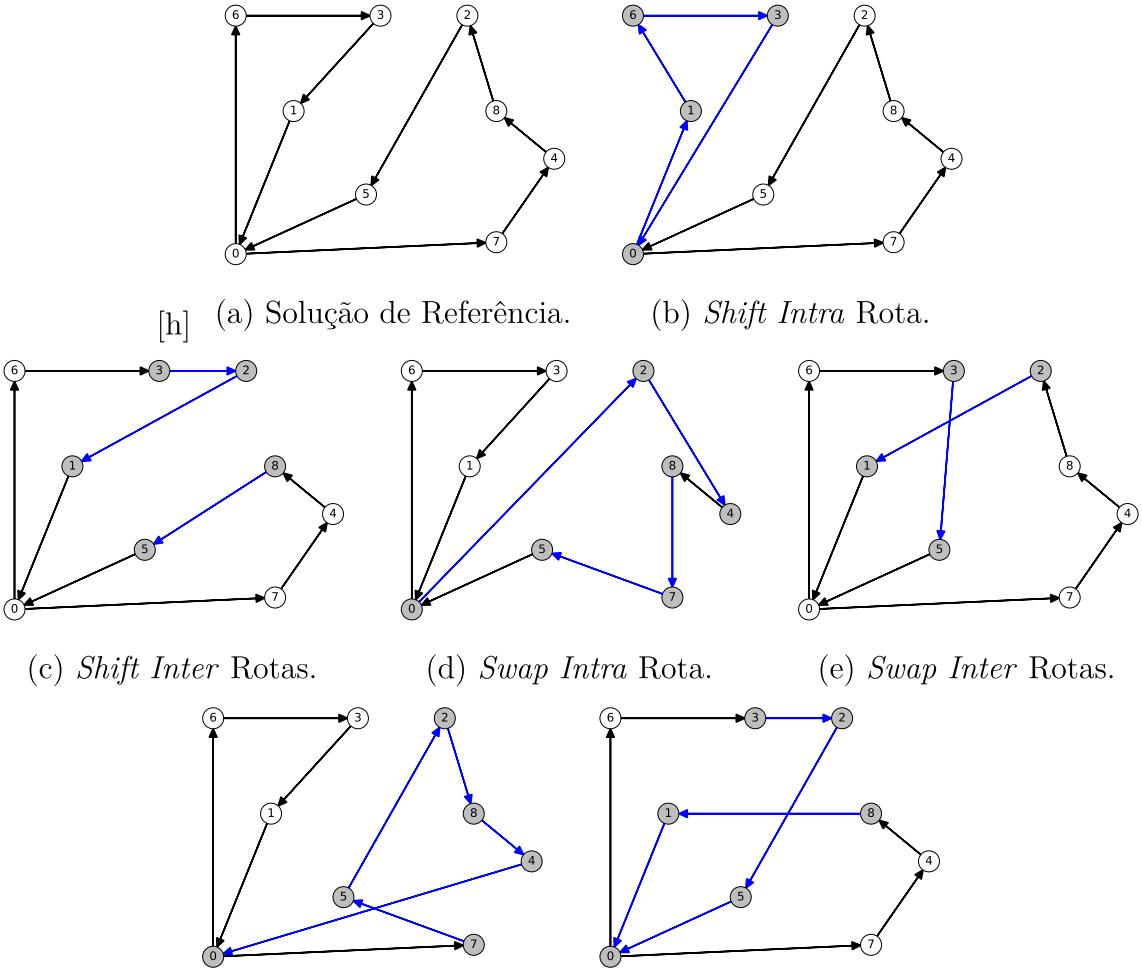
A seguir são apresentadas as vizinhanças inter-rotas, ou seja, aquelas que envolvem duas rotas. A Figura 4c apresenta o resultado da vizinhança *shift*, em que um cliente (2) e um arco $(3, 1)$ são escolhidos de rotas distintas, e a aplicação consiste em adicionar o cliente (2) entre os clientes do arco $(3, 1)$ da outra rota. A Figura 4e apresenta o resultado da vizinhança *swap* inter rotas, em que dois clientes, 1 e 5, são escolhidos de rotas distintas e trocam suas posições. A Figura 4g apresenta o resultado da vizinhança *cross*, em que dois clientes (2 e 1) são escolhidos de rotas distintas e suas rotas parciais, sendo elas $(1-0)$ e $(2-5-0)$, são trocadas.

Todas as vizinhanças são aplicadas somente às rotas do segundo nível, rotas dos EVs. Quando uma solução de melhora é encontrada, considerando o impacto no custo do segundo nível, o primeiro nível é refeito e verifica-se novamente se o resultado da aplicação da vizinhança é melhor do que a solução de referência.

4.7 Viablidade das Rotas EVs

Quando uma rota de veículo elétrico é inviável considerando a carga da bateria, um algoritmo é usado para tentar viabilizá-la. O algoritmo recebe uma rota EV inviável e tenta inserir uma estação de recarga para corrigir o problema. O primeiro passo consiste em identificar o arco (i, j) onde ocorreu a falha, ou seja, o esgotamento da bateria do veículo, e a posição b para iniciar as inserções. Essa posição pode ser i, ou uma posição anterior onde uma RS é utilizada, ou ainda o satélite onde começa a rota.

Antes do método proposto, existe um pré-processamento da instância que consiste



[h] (a) Solução de Referência. (b) *Shift Intra* Rota.

(c) *Shift Inter* Rotas. (d) *Swap Intra* Rota. (e) *Swap Inter* Rotas.

(f) *2-opt*. (g) *Cross*.

Figura 4 – Vizinhanças de Busca Local.

em encontrar, para cada aresta (i, j), as k estações de recarga mais próximas. Isso resulta em uma lista das k estações mais próximas para toda a aresta (i, j). Esse pré-processamento permite que a viabilização de rotas EVs possa ser realizada eficientemente, já que somente uma pequena parte das estações de recarga precisam ser testadas.

Após identificar o arco (i, j) com falha e o início b de inserções, esse algoritmo tenta inserir as `k` RSs mais próximas de cada arco entre b e i. Antes de se realizar a verificação da nova rota (com uma RS entre um arco), com relação à janela de tempo e carga de bateria, é necessário verificar se a distância da nova rota é menor do que o parâmetro `distMax`. Esse parâmetro é a distância máxima permitida para a rota viabilizada, recebida com a rota EV. Caso essa verificação falhe, passa-se para o próximo arco.

Quando uma rota é viabilizada, ela é imediatamente retornada. Apenas uma RS pode ser inserida, ou seja, não considera o caso em que uma rota pode precisar de duas RSs. A exceção é quando já existe uma RS na rota, o que pode levar a duas RSs a serem utilizadas entre um arco (i, j). Essa restrição à inserção de duas ou mais RSs é necessária, pois esse algoritmo é utilizado várias vezes ao longo dos métodos propostos e, com isso,

uma abordagem com maior custo computacional não seria o mais adequado.

4.8 Critério de Aceitação

Após a aplicação da busca local, existe a decisão de substituir a solução corrente, observando o critério de aceitação. Uma solução (`Sol'`) será aceita se for viável e se a sua diferença percentual ($100 * \frac{custo(Sol') - custo(SolBest)}{custo(SolBest)}$) entre a melhor solução (`SolBest`) for menor do que o limite estabelecido pelo parâmetro `difBest`.

4.9 Particionamento de Rotas para o 2E-EVRP-TW

De acordo com Fischetti and Fischetti (11), *matheuristics* são algoritmos desenvolvidos para resolver problemas de otimização e realizam a combinação de programação matemática (MILP) e metaheurísticas (ou heurísticas). Particularmente, *matheuristics* baseadas na estratégia de cobertura de rotas têm se mostrado eficientes na geração de boas soluções para diversas variantes do VRP (3). A ideia por trás desse método consiste em criar um conjunto de rotas utilizando uma heurística, e determinar a melhor combinação de rotas usando um MILP. O problema é resolvido selecionando um subconjunto de rotas que atendam a todos os clientes enquanto minimiza o somatório total da distância percorrida.

Matheuristics baseadas no problema de cobertura de conjuntos foram utilizadas em (2, 10, 32) para resolver variantes do problema de roteamento de veículos em dois níveis. Nesses trabalhos, uma metaheurística é utilizada para a criação de rotas dos dois níveis, e a cobertura de rotas é utilizada a fim de determinar a melhor combinação de rotas com um MILP. Como o problema tem dois níveis de roteamento, a formulação do MILP tem que assegurar que as rotas selecionadas de níveis distintos sejam compatíveis entre si.

O IG proposto cria soluções viáveis para o problema, onde são extraídas as rotas relativas ao segundo nível (rotas de EVs). Essas rotas são utilizadas para a formulação de um MILP do problema de particionamento de conjuntos. Além do segundo nível, o problema 2E-EVRP-TW também possui o primeiro nível (roteamento dos satélites), para isso, também são acrescentadas restrições relativas ao VRPTW. Ao combinar as formulações, particionamento de rotas EVs e VRPTW para o roteamento dos satélites, é necessário garantir a compatibilidade entre eles no que diz respeito à carga de transporte dos CVs, fornecidas pelo somatório das cargas dos EVs selecionados. Para garantir o correto dimensionamento das cargas entre os dois níveis, o problema de particionamento, onde exatamente uma rota EV é selecionada para cada cliente, é utilizado em detrimento ao problema de cobertura de conjuntos.

Quando as rotas de EVs são extraídas, elas são armazenadas em um conjunto de rotas. A *ELF Hash*, que pode ser verificada em (4), é utilizada para guardar rotas únicas em uma tabela. A eliminação de rotas repetidas contribui para reduzir o número

de variáveis do modelo, e consequentemente, o tempo de computação necessário para a criação e resolução do modelo matemático. Essa *Hash* foi originalmente criada para cadeias de caracteres, aqui, cada nó de uma rota é utilizado como um carácter.

A abordagem proposta tem uma vantagem em relação aos métodos de (2, 10, 32), porque o primeiro nível, CVs, possui as melhores rotas para os EVs selecionados, devido à criação das mesmas serem realizadas pelo modelo matemático, e não por um método heurístico, como realizado nos demais trabalhos.

Considere R o conjunto de rotas de EV gerado pela metaheurística IG, $R_i^{sat} \subset R$ contém as rotas iniciadas no satélite i, enquanto $R_c^{cust} \subset R$ corresponde ao subconjunto de rotas que visitam o cliente c. Para cada rota $r \in R$, defini-se d_r como a distância total percorrida pela rota r, q_r como a quantidade de carga levada pela rota r, e t_r como o último instante de tempo em que a rota r deve sair do satélite para atender a todos os seus clientes dentro de suas respectivas janelas de tempo.

Os conjuntos apresentados a seguir, que já foram previamente definidos, são necessários para descrição da formulação proposta: V_c (conjunto de clientes), V_s (conjunto de satélites), V_d (conjunto unitário com o depósito central), A_1 (conjunto de arcos que conectam o depósito e satélites).

A seguir são definidas as variáveis para o modelo matemático:

- y_r: Variável binária que indica se a rota r pertence à solução;
- z_i: Variável binária que indica se o satélite i é utilizado;
- $x_{i,j}$: Variável binária que indica se o arco (i, j) é percorrido por um CV;
- $f_{i,j}$: Variável contínua que indica a quantidade de carga de um CV ao percorrer o arco (i, j);
- t_i: Variável contínua que indica o instante de tempo em que um CV visita um satélite i;

A seguir o modelo matemático é definido:

$$min \quad \sum_{r \in R} d_r.y_r + \sum_{i,j \in A1} d_{i,j}.x_{i,j} \tag{4.1}$$

$$sujeito\ a \quad \sum_{r \in R_c^{cli}} y_r = 1 \qquad \forall c \in V_c \tag{4.2}$$

$$\sum_{r \in R_i^{sat} \cap R_c^{cli}} y_r \leq z_i \qquad \forall i \in V_s, \forall c \in V_c \tag{4.3}$$

$$\sum_{r \in R_i^{sat}} y_r \geq z_i \qquad \forall i \in V_s \tag{4.4}$$

$$\sum_{r \in R} y_r \leq m_{EV} \tag{4.5}$$

$$\left\lceil \frac{\sum_{i \in V_c} q_i}{Q_{CV}} \right\rceil \leq \sum_{j \in V_s} x_{i,j} \leq m_{CV} \qquad \forall i \in V_d \tag{4.6}$$

$$\sum_{j \in V_s, j \neq i} x_{i,j} = \sum_{j \in V_s, j \neq i} x_{j,i} \qquad \forall i \in V_s \cup V_d \tag{4.7}$$

$$\sum_{i \in V_s \cup V_d, i \neq j} x_{i,j} = z_j \qquad \forall j \in V_s \tag{4.8}$$

$$\sum_{j \in V_s \cup V_d, j \neq i} f_{j,i} - \sum_{j \in V_s \cup V_d, j \neq i} f_{i,j} = \sum_{r \in R_i^{sat}} q_r.y_r \qquad \forall i \in V_s \tag{4.9}$$

$$f_{i,j} \leq Q_{CV}.x_{i,j} \qquad \forall (i,j) \in A_1 \tag{4.10}$$

$$t_j \geq t_i + d_{i,j}.C_{dist}.x_{i,j} - T.(1 - x_{i,j}) \qquad \forall i \in V_s \cup V_d, \forall j \in V_s \tag{4.11}$$

$$t_i \leq \sum_{r \in R_c^{cust} \cap R_i^{sat}} T_r^{max}.y_r + \sum_{r \in R_c^{cust} \setminus R_i^{sat}} T.y_r \qquad \forall i \in V_s, \forall c \in V_c \tag{4.12}$$

$$y_r \in \{0,1\} \qquad \forall r \in R \tag{4.13}$$

$$z_i \in \{0,1\} \qquad \forall i \in V_s \tag{4.14}$$

$$x_{i,j} \in \{0,1\}; f_{i,j} \geq 0 \qquad \forall (i,j) \in A1 \tag{4.15}$$

$$t_i \geq 0 \qquad \forall i \in V_s \cup V_d \tag{4.16}$$

$$t_i = 0 \qquad \forall i \in V_d \tag{4.17}$$

$$f_{i,j} = 0 \qquad \forall i \in V_s, j \in V_d \tag{4.18}$$

A função objetivo, definida em 4.1, possui duas parcelas. A primeira está relacionada com as distâncias percorridas pelas rotas selecionadas de segundo nível (d_r), e a segunda considera a soma das distâncias $d_{i,j}$ de cada arco (i,j) utilizado no roteamento do primeiro nível.

As restrições de 4.2 a 4.5 estão relacionadas a seleção de rotas para os veículos elétricos. A primeira garante que cada cliente é visitado em uma única rota. A segunda garante que, para cada cliente, uma rota EV só é selecionada se o seu respectivo satélite é visitado por um CV. As restrições em 4.4 asseguram que um satélite não seja visitado caso não tenha rotas EV selecionadas. Restrições em 4.5 limitam o número total de rotas de EVs ao número de veículos elétricos disponíveis (m_{EV}).

As restrições de 4.6 a 4.12 estão relacionadas com o primeiro nível, dos CVs. As restrições em 4.6 asseguram que o número de CVs utilizados está entre o valor mínimo para transportar as mercadorias (quantidade total dividida pela capacidade do CV) e o

máximo de CVs disponíveis, m_{CV}. As restrições de conservação de fluxo em 4.7 controlam o caminho dos veículos pelo grafo. Restrições em 4.8 asseguram que todos os satélites com EVs selecionados sejam atendidos por um CV. Essa restrição limita o atendimento dos satélites a somente um CV, assim, impedindo a entrega dividida. Entretanto, isso possibilita o uso de apenas dois índices na varável x, reduzindo a quantidade de variáveis binárias, quando comparado a um modelo com três índices.

As restrições em 4.9 garantem, para todo satélite s, que a diferença entre a quantidade de carga de entrada e a de saída é igual ao somatório da quantidade de demanda das rotas EV selecionadas em s. As restrições em 4.10 garantem que a capacidade de carga dos CVs não seja excedida.

Restrições em 4.11 asseguram o tempo de chegada de i para o satélite j. O tempo de chegada em j deve ser pelo menos igual ao tempo de chegada em i mais o tempo de deslocamento de i para j. No caso de arco (i, j) não ser utilizado, a constante T é subtraída de t_i, e assim, a restrição tornasse válida. Essas restrições também eliminam subciclos, já que estabelecem a ordem de visitação dos satélites, semelhantemente às restrições *Miller-Tucker-Zemli* (MTZ) (21) de subciclo.

As restrições em 4.12 garantem que o tempo de chegada de um CV em cada satélite deve ser menor ou igual ao tempo de saída máximo permitido das rotas EVs selecionadas. Para cada satélite existe uma restrição que inclui todas as rotas de um mesmo cliente, então a restrição tem exatamente uma variável y_r igual a um, considerando os dois somatórios, assim: se a rota r atende um cliente a partir do satélite i, o tempo de saída da rota T_r^{max} limita t_i; caso contrário, se o cliente é visitado por uma rota de outro satélite, t_i é limitado pela constante de tempo máximo T.

As restrições 4.13 a 4.16 definem os domínios das variáveis, e restrições 4.17 e 4.18 definem os valores para as variáveis relacionadas ao depósito.

4.10 Estruturas de Dados Adicionais

A seção descreve um conjunto de estruturas de dados utilizadas na implementação dos algoritmos com o intuito de reduzir o seu tempo de computação.

No Algoritmo 3 é possível perceber uma limitação relacionada à remoção de todos os candidatos da iteração anterior, na Linha 8. A cada iteração é necessário refazer a lista de candidatos, o que é ineficiente. Entretanto, é possível realizar a atualização da lista, e assim, reduzir o número de chamadas a função `updateList`. Para melhorar o algoritmo de inserção são necessárias duas estruturas de dados adicionais. A primeira estrutura de dados é a matriz Cliente $\times$ Candidato de tamanho $n \times m$, sendo m igual ao maior número possível de candidatos para um cliente. Essa matriz permite acessar todos os candidatos de um dado cliente. A segunda é a matriz Rota $\times$ Candidato de tamanho $r \times p$, sendo r o

número total de rotas, e p o maior número possível de candidatos para uma rota. Essa matriz permite acessar todos os candidatos de uma dada rota.

A primeira iteração do algoritmo modificado é igual ao Algoritmo 3, ou seja, são criados os candidatos para todos os clientes. Quando um candidato é adicionado a lista, as duas estruturas auxiliares são atualizadas com a sua referência. A iteração (i) do algoritmo modificado, diferindo da primeira iteração, realiza a inserção de um candidato do cliente C_i em uma rota R_i. Os passos para atualizar a lista de candidatos são: remover todos os candidatos que tem como cliente C_i e atualizar os candidatos que utilizam a rota R_i.

A remoção dos candidatos que tem como cliente C_i é realizada utilizando-se a matriz Cliente $\times$ Candidato. A atualização dos candidatos referentes à rota R_i é realizada, diferentemente dependendo do parâmetro `tipoHeuri`. Caso `tipoHeur = tipoHeurPorRotaVeic`, somente esse caso é coberto, existe no máximo um candidato de um cliente não atendido da rota R_i. A atualização consiste em encontrar a nova melhor posição, na rota R_i, para os clientes associados aos candidatos encontrados na matriz Rota $\times$ Candidato.

No RVND uma estrutura auxiliar é utilizada para evitar recalcular movimentos já analisados, e que não produziram soluções de melhora. A estrutura de dados adicional consiste em uma matriz que possui o número de linhas igual ao número de rotas e o número de colunas igual à quantidade de movimentos. Sendo o k-ésimo movimento (intra-rota) e a i-ésima rota, e considerando que todas as possibilidades de aplicação do movimento foram tentadas, a matriz é alterada para (1) na posição relativa a k e a i, para sinalizar que o movimento já foi testado nessa rota. Caso o k-ésimo movimento volte a se repetir sem que a i-ésima rota seja alterada, esse movimento não é aplicado a essa rota, já que ele não melhora a solução atual e, assim, vários cálculos e verificações não têm que ser refeitos.

Quando um movimento modifica uma rota, a posição na matriz relativa a ele e a rota é alterado para `0`. Com isso, os demais movimentos são realizados para essa rota, dado que ele ainda não foi testado. Para os movimentos inter-rotas a estratégia é semelhante. Verificam-se as posições das duas rotas, e o movimento é realizado caso exista somente uma posição com `1` ou nenhuma. Caso existam duas posições marcadas com `1`, o movimento não é executado para aquele par de rotas.

Esse capítulo apresentou as abordagens propostas, que consistem em um Iterated Greedy e uma mateurística, baseado em um modelo de particionamento de rotas. O IG proposto possui as seguintes diferenças em relação ao algoritmo padrão: utiliza uma busca local RVND e a reconstrução da solução é feita via uma heurística de inserção mais barata. Essa heurística utiliza uma escolha aleatória da lista restrita de candidatos para escolher o próximo candidato a entrar na solução, permitindo que diferentes soluções sejam geradas e, assim, contribuindo para o aperfeiçoamento da solução. O particionamento de rotas utiliza as rotas de EVs geradas pelo IG proposto. As rotas são utilizadas para a criação de um modelo de particionamento. O modelo incorpora restrições adicionais para o primeiro

nível.

5 Experimentos Computacionais

Os resultados obtidos pelos dois métodos propostos, IG e IG_MILP, são comparados com os obtidos pelo VNS_{full} (Capítulo 3), apresentados em (1). Somente os resultados do VNS_{full} são apresentados, porque o mesmo apresenta resultados melhores do que o VNS_{red}. As seções a seguir detalham os experimentos.

5.1 Ambiente de Teste

Os Algoritmos propostos neste trabalho foram implementados usando C++, e os experimentos executados em um computador AMD Ryzen 7 5800X 3.8 GHz com 16 GB de memória RAM. Também foi utilizado *Debian* 12, *Irace* 3.5, e *g++* 12.2, e o código fonte foi compilado com a *flag* `-O3`. A formulação do MILP foi implementada usando o resolvedor de programação matemática *Gurobi* (10.0) (13) e configurado para executar com uma única *thread*.

Aakbay et al. (1) utiliza um processador Intel Xeon 5670 com 2.9 GHz e 32 GB de memória RAM. O *benchmark* (*PassMark*[1]) com uma única *thread*, possui as seguintes pontuações para cada processador: 3447 pontos para o processador AMD usado nos experimentos deste trabalho e 1392 pontos para o processador Intel usado em (1). Assim, observa-se uma razão de 0.4 entre os processadores, e esse valor é utilizado para converter os tempos de CPU. Como resultado, após essa conversão, os tempos de processamento podem ser considerados comparáveis. O código fonte e o material suplementar estão disponíveis publicamente[2].

5.2 Teste Estatístico

O teste de *Wilcoxon* é utilizado para testar se os resultados dos métodos IGs propostos e VNS_{full} são estatisticamente similares. O nível de significância utilizado nos testes é de 0.05. O teste de *Wilcoxon* é utilizado de duas formas diferentes. Para comparar os dois métodos IGs com VNS_{full}, o teste de *Wilcoxon signed-rank* é utilizado para comparar os resultados por grupos de instâncias. Ao comparar os métodos propostos IG e IG_MILP, o teste de *Wilcoxon rank-sum* é utilizado por instância. O teste de *Wilcoxon rank-sum* não é utilizado com o VNS_{full} porque o trabalho em (1) não disponibiliza os resultados por execução, com isso, a única opção foi o teste de *Wilcoxon signed-rank*.

O teste de *Wilcoxon signed-rank* compara o (IG ou IG_MILP) com VNS_{full} utilizando os valores das colunas: melhor distância (*dist best*), distância média (*dist avg*), e tempo de CPU (T). Esses valores estão disponíveis no material suplementar. *Wilcoxon*

[1] `www.cpubenchmark.net/compare/1307vs3869/Intel-Xeon-X5670-vs-AMD-Ryzen-7-5800X`

[2] `www.github.com/Routing-Problems-in-Green-Logistic/2E_EVRPTW_Paper`

rank-sum compara a distância e tempo de CPU do IG e IG_MILP com 30 execuções independentes. O teste estatístico fornece dois valores (p-value e *effect size*, r). O primeiro indica se existe uma diferença estatisticamente significativa entre duas amostras e a segunda indica a magnitude da diferença. A magnitude, segundo (7), pode ser pequena (r=[0.1,0.3)), média (r=[0.3,0.5)) ou grande (r=[0.5,1)). Será descrito ao longo do texto as porcentagens, para os métodos, das instâncias de um grupo que possue magnitude pequena, média, e grande.

Como os experimentos realizados em (1) foram executados somente 10 vezes, foram selecionadas as 10 primeiras execuções para realizar uma comparação mais justa com VNS_{full}, e os métodos IG e $\text{IG}^{\text{MILP}}_{\text{RVND}}$ foram comparados considerando as 30 execuções. Os resultados podem ser reproduzidos de forma independente utilizando as sementes (do gerador de números pseudo-aleatórios) presentes no material suplementar.

5.3 Instâncias

As instâncias foram criadas originalmente em (23) para o problema de roteamento de veículos elétricos com janela de tempo e estações de recarga (EVRP-TW) e foram adaptadas em (1) para o 2E-EVRP-TW. As instâncias estão disponíveis no repositório, juntamente com o código fonte.

O *dataset* compreende 92 instâncias, classificadas como pequenas (36 instâncias) ou grandes (56 instâncias). As instâncias pequenas são divididas em três grupos: C5, C10, e C15, e possuem, respectivamente, 5, 10, e 15 clientes. Já as instâncias grandes são divididas em outros três grupos: C100, R100, e RC100. O prefixo C indica que as posições dos clientes são clusterizadas, e R indica que as posições são randômicas. Toda instância grande possui 100 clientes. A Tabela 1 mostra o número de clientes e o número médio de RSs, satélites, EVs e CVs para cada grupo.

Tabela 1 – Características médias das instâncias.

Grupo	Cli	RS	Sat	EV	CV
C5	5	3.5	1	1.6	1.0
C10	10	4.3	1	3.3	1.0
C15	15	6.8	2	4.3	1.0
C100	100	29	8	26.3	2.5
R100	100	29	8	19.0	1.8
RC100	100	29	8	21.5	2.1

5.4 Variantes do IG

A Tabela 2 mostra as variantes do método proposto, que consistem na remoção de algum componente. O IG não utiliza o RNVD, IG_{RVND} utiliza a busca local, $\text{IG}^{\alpha=0}_{\text{RVND}}$

Tabela 2 – Métodos IGs.

Método	RVND	$\alpha = 0$	MILP
IG			
IG_{RVND}	X		
$\text{IG}^{\alpha=0}_{\text{RVND}}$	X	X	
$\text{IG}^{\text{MILP}}_{\text{RVND}}$	X		X

utiliza a busca local e os parâmetros α_{1n} e α_{2n} iguais a zero, e $\text{IG}^{MILP}_{\text{RVND}}$ utiliza o RVND e o particionamento de rotas (modelado como um problema de MILP). A Seção 5.6 mostra uma comparação entre essas variantes para comprovar o benefício de cada componente.

5.5 Ajuste de Parâmetros

Pelo fato de metaheurísticas serem muito sensíveis aos valores de seus parâmetros, o pacote *Irace* (19) é utilizado para realizar o ajuste de parâmetros. O resultado do *Irace* fornece os melhores valores de parâmetros considerando o conjunto de instâncias de teste, domínio dos parâmetros e número máximo de iterações. A lista de parâmetros e os valores testados pelo Irace são apresentados na Tabela 3. Os parâmetros do MILP são relativos ao resolvedor Gurobi.

Tabela 3 – Parâmetros analisados usando o Irace e os valores testados.

Parâmetros	Valores
α_{1n}	{0.15, 0.25, 0.4, 0.5, 0.65, 0.75, 0.8, 0.85, 0.9}
α_{2n}	{0.005, 0.01, 0.05, 0.1, 0.15, 0.25, 0.35, 0.45, 0.65, 0.9}
difBest	{0.005, 0.01, 0.015, 0.02, 0.03, 0.04, 0.05, 0.07, 0.1}
estrategiaSelecao	{*probIgual*, *torneio*}
rmRate	{0.1, 0.15, 0.2, 0.25, 0.3, 0.4, 0.5, 0.6}
multLimitCall	{1, 1.5, 2, 2.5, 3}
Milp_presolve	{-1, 1, 2}
Milp_cuts	{-1, 1, 2, 3}

A metodologia utilizada para o ajuste de parâmetros em (1) é dividir as instâncias em dois grupos, pequeno e grande (como previamente definido). Essa mesma metodologia é utilizada nesse trabalho. Para o grupo das instâncias pequenas, foram utilizadas 6 instâncias (16%) e máximo de 1500 execuções. Para o segundo grupo, foram utilizadas 13 instâncias (23%) e máximo de 4000 execuções. As instâncias foram escolhidas aleatoriamente e podem ser verificadas no material suplementar. Como (1) não apresenta as instâncias utilizadas no ajuste de parâmetros, não é possível garantir que as mesmas instâncias são utilizadas aqui.

A Tabela 4 mostra os parâmetros escolhidos baseados nos resultados do Irace. Somente os parâmetros do IG_{RVND} e $\text{IG}^{\text{MILP}}_{\text{RVND}}$ são apresentados; os parâmetros para as demais variantes encontram-se no material complementar. Os parâmetros (α_{1n} e *multLimitCall*)

não são usados para as instâncias pequenas porque essas instâncias têm somente um satélite (α_{1n} é igual a 0.0) e sendo possível utilizar apenas um dos métodos de destruição. Os parâmetros a seguir são necessários para os métodos IGs e não são escolhidos pelo Irace: `numItIG`= 3000, `numItIGMilp`= 100, $\overrightarrow{\alpha_{ini}} = \{0.05, 0.07, 0.1, 0.15, 0.2, 0.3, 0.35, 0.4, 0.5, 0.6\}$, `numItMaxGreedy`= 3000, `numItUpdateProb`= 250 e `MILP_gap`= 0.04.

Tabela 4 – Parâmetros escolhidos pelo Irace.

	IG_{RVND}		IG^{MILP}_{RVND}	
Parâmetro	Small	Large	Small	Large
α_{1n}	0.0	0.9	0.0	0.15
α_{2n}	0.9	0.1	0.9	0.35
difBest	0.07	0.015	0.1	0.05
estrategiaSelecao	*probIgual*	*torneio*	*probIgual*	*torneio*
rmRate	0.4	0.1	0.5	0.2
multLimitCall	2	2	1	1
Milp_presolve	-	-	2	2
Milp_cuts	-	-	2	1

5.6 Análise de Resultados

Aqui são apresentados os resultados de algumas variações (indicadas na Tabela 2) do IG proposto e do VNS_{full}. Foram realizadas inicialmente 10 execuções independentes (como realizado em (1)) para cada instância e para cada variante do IG, com o objetivo de comparar os resultados do IG proposto com aqueles obtidos pelo VNS_{full}. Para realizar uma comparação melhor entre as variantes do IG, foram realizadas mais 20 execuções independentes. As Tabelas 5 à 16 apresentam as colunas *Best*, Gap-Best, Gap-Avg e tempo de CPU em segundos.

A coluna Gap-Best é a diferença percentual em relação a distância da melhor execução e Best. A coluna *Best* é o melhor valor obtido por um dos algoritmos que estam sendo comparados. A coluna Gap-Avg é a diferença percentual em relação a distância média e Best. Para os métodos IGs propostos, o tempo de CPU é representado como T e para o VNS_{full} é representado como tempo de CPU convertido (TC), como definido na Seção 5.1. Para cada instância, os melhores valores de cada métrica são destacados em negrito. Os resultados são apresentados com uma única casa decimal. No material suplementar podem ser encontrado os valores de distância para cada algoritmo.

5.6.1 Análise de sensibilidade para α

Uma das principais diferenças do IG proposto para o IG padrão (conforme definido na literatura) é a escolha aleatória do próximo candidato a ser escolhido da lista restrita na fase de reconstrução. Para comprovar que essa alteração realmente produz melhores

resultados, o IG_{RVND} com os parâmetros do Irace é comparado com o $\text{IG}^{\alpha=0}_{\text{RVND}}$, onde os parâmetros α_{1n} e α_{2n} são iguais a zero. Os demais parâmetros foram ajustados com o Irace.

A Table 5 mostra os resultados dos dois IGs para os três grupos: C5, C10, e C15, respectivamente, com 5, 10, e 15 clientes. Para estes três grupos, o IG_{RVND} obteve melhores resultados do que $\text{IG}^{\alpha=0}_{\text{RVND}}$ para a distância. Em relação ao tempo de processamento, o $\text{IG}^{\alpha=0}_{\text{RVND}}$ obteve valores menores para os grupos C10 e C15.

Para o grupo C5, as médias de Gap-Best obtidas são: 0.0% para o $\text{IG}^{\alpha=0}_{\text{RVND}}$, e o IG_{RVND}. Em relação à Gap-Avg, os valores obtidos são: 1.4% ($\text{IG}^{\alpha=0}_{\text{RVND}}$) e 0.8 ($\text{IG}_{\text{RVND}}$). Existe uma diferença estatisticamente significativa (teste de Wilcoxon rank-sum por instância) para duas instâncias (16%), em relação a distância. A magnitude do teste é grande para as duas instâncias.

Para o grupo C10, as médias de Gap-Best obtidas são: 0.8% ($\text{IG}^{\alpha=0}_{\text{RVND}}$), e 0.0% ($\text{IG}_{\text{RVND}}$). Em relação à Gap-Avg, os valores obtidos são: 2.6% ($\text{IG}^{\alpha=0}_{\text{RVND}}$) e 0.5 ($\text{IG}_{\text{RVND}}$). Existe uma diferença estatisticamente significativa para 11 instâncias (92%), em relação a distância. A magnitude do teste é pequeno (25%), médio (8%), e grande (58%).

Para o grupo C15 as médias de Gap-Best obtidas são: 3.9% ($\text{IG}^{\alpha=0}_{\text{RVND}}$), e 0.0% ($\text{IG}_{\text{RVND}}$). Em relação à Gap-Avg, os valores obtidos são: 6.2% ($\text{IG}^{\alpha=0}_{\text{RVND}}$) e 1.1% ($\text{IG}_{\text{RVND}}$). Existe uma diferença estatisticamente significativa para 11 instâncias (83%), em relação a distância. A magnitude do teste é médio (17%), e grande (67%).

A Tabela 6 apresenta os resultados para os grupos C100 e R100, ambos com 100 clientes. O IG_{RVND} possui resultados melhores do que aqueles obtidos pelo $\text{IG}^{\alpha=0}_{\text{RVND}}$, considerando a distância. Em relação ao tempo de processamento, o $\text{IG}^{\alpha=0}_{\text{RVND}}$ é 1.2 vezes mais rápido do que o IG_{RVND} na média e para o grupo C100. Em relação ao grupo R100, o IG_{RVND} é 1.2 vezes mais rápido do que $\text{IG}^{\alpha=0}_{\text{RVND}}$. A inverção do *speedup* não pode ser explicada pelo parâmetro α, então a diferença nos demais parâmetros realiza essa inversão.

Para o grupo C100, as médias de Gap-Best obtidas são: 3.4% ($\text{IG}^{\alpha=0}_{\text{RVND}}$), e 0.0% ($\text{IG}_{\text{RVND}}$). Em relação à Gap-Avg, os valores obtidos são: 6.2% ($\text{IG}^{\alpha=0}_{\text{RVND}}$) e 1.8 ($\text{IG}_{\text{RVND}}$). Existe uma diferença estatisticamente significativa para todas as instâncias em relação a distância. A magnitude do teste é grande para todas as instâncias.

Para o grupo R100 as médias de Gap-Best obtidas são: 0.2% ($\text{IG}^{\alpha=0}_{\text{RVND}}$), e 0.0% ($\text{IG}_{\text{RVND}}$). Em relação à Gap-Avg, os valores obtidos são: 2.0% ($\text{IG}^{\alpha=0}_{\text{RVND}}$) e 1.5 ($\text{IG}_{\text{RVND}}$). Existe uma diferença estatisticamente significativa para 15 instâncias (65%), em relação a distância. A magnitude do teste é pequeno (13%), médio (48%), e grande (4%).

A Tabela 7 apresenta os resultados para o grupo RC100. O IG_{RVND} possui melhores

resultados do que o $\text{IG}^{\alpha=0}_{\text{RVND}}$ considerando a distância e tempo de processamento. As médias relativas à Gap-Best obtidas são: 1.0% ($\text{IG}^{\alpha=0}_{\text{RVND}}$), e 0.0% ($\text{IG}_{\text{RVND}}$). Em relação ao Gap-Avg, os valores obtidos são: 2.9% ($\text{IG}^{\alpha=0}_{\text{RVND}}$), e 1.6% ($\text{IG}_{\text{RVND}}$). Existe uma diferença estatisticamente significativa para 11 instâncias (69%) considerando a distância. A magnitude do teste é médio (19%), e grande (50%). O IG_{RVND} é 1.15 vezes mais rápido do que o $\text{IG}^{\alpha=0}_{\text{RVND}}$.

A análise dos resultados comparando o $\text{IG}^{\alpha=0}_{\text{RVND}}$ e IG_{RVND} permite concluir que a adoção de uma escolha randômica do próximo candidato na fase de reparo no IG melhora os resultados. Na metaheurística IG padrão pressupõe-se que remover diferentes componentes da solução gera diferentes soluções parciais e isso produz diferentes resultados na fase de reconstrução. Entretanto, isso pode não ser suficiente para criar soluções diversas, devido à viabilidade das soluções e quantidade de componentes removidos. A hipótese é que escolher o próximo candidato da lista restrita para entrar na solução de forma aleatória produz soluções mais diversas do que a escolha gulosa. Isso foi demonstrado realizando a comparação entre essas duas formas para o 2E-EVRP-TW.

Tabela 5 – Resultados para instâncias pequenas (30 execuções). Comparação com α_{1n} e α_{2n} iguais a 0 ($\text{IG}^{\alpha=0}_{\text{RVND}}$) com o IG_{RVND}. Teste de Wilcoxon rank-sum (nível de significância de 0.05) entre os algoritmos ($\text{IG}^{\alpha=0}_{\text{RVND}}$ e IG_{RVND}), por instância. *p-value* igual a `NA` (-). Diferenças não estatisticamente significativa não são marcadas. Resultados com diferença estatisticamente significativa tem a magnitude: pequena (*), média (**), ou grande (***). Tempos de CPU (colunas T) estão em segundos.

Instance	Best	$\text{IG}^{\alpha=0}_{\text{RVND}}$			IG_{RVND}		
		Gap Best	Gap Avg	T	Gap Best	Gap Avg	T
C101_C5x	385.5^{+}	**0.0%**	**0.0%**$^{-}$	**0.1**	**0.0%**	**0.0%**$^{-}$	**0.1*****
C103_C5x	341.3^{+}	**0.0%**	**0.0%**$^{-}$	**0.1**	**0.0%**	**0.0%**$^{-}$	**0.1****
C206_C5x	417.3^{+}	**0.0%**	**0.0%**$^{-}$	**0.1**	**0.0%**	**0.0%**$^{-}$	**0.1*****
C208_C5x	381.9^{+}	**0.0%**	**0.0%**$^{-}$	**0.1**	**0.0%**	**0.0%**$^{-}$	**0.1*****
R104_C5x	317.0^{+}	**0.0%**	**0.0%**$^{-}$	**0.1**	**0.0%**	**0.0%**$^{-}$	**0.1*****
R105_C5x	453.7^{+}	**0.0%**	5.0%	**0.2**	**0.0%**	**0.0%*****	**0.2**
R202_C5x	347.8^{+}	**0.0%**	**0.0%**	**0.1**	**0.0%**	**0.0%**	**0.1****
R203_C5x	371.3^{+}	**0.0%**	**0.0%**$^{-}$	**0.1**	**0.0%**	**0.0%**$^{-}$	**0.1*****
RC105_C5x	432.6^{+}	**0.0%**	3.1%	**0.1*****	**0.0%**	**0.0%*****	**0.1**
RC108_C5x	460.9^{+}	**0.0%**	**7.6%**	**0.1**	**0.0%**	8.3%	**0.1**
RC204_C5x	334.9^{+}	**0.0%**	**0.0%**$^{-}$	**0.1**	**0.0%**	**0.0%**$^{-}$	**0.1****
RC208_C5x	327.3^{+}	**0.0%**	**0.0%**$^{-}$	0.2	**0.0%**	**0.0%**$^{-}$	**0.1*****
Avg	385.9	**0.0%**	1.4%	**0.1**	**0.0%**	**0.8%**	**0.1**
C101_C10x	539.7^{+}	3.4%	3.4%	**0.3*****	**0.0%**	**0.0%*****	0.6
C104_C10x	484.3^{+}	**0.0%**	**0.0%**$^{-}$	**0.4*****	**0.0%**	**0.0%**$^{-}$	0.5
C202_C10x	425.5^{+}	2.1%	12.1%	**0.3*****	**0.0%**	**1.2%*****	0.4
C205_C10x	416.1^{+}	**0.0%**	0.4%	**0.3*****	**0.0%**	**0.0%***	0.5
R102_C10x	505.5^{+}	**0.0%**	3.2%	**0.3*****	**0.0%**	**0.0%*****	0.4
R103_C10x	436.1	**0.0%**	0.4%	**0.3*****	**0.0%**	**0.0%***	0.4
R201_C10x	460.7^{+}	**0.0%**	3.8%	**0.4*****	**0.0%**	**1.1%*****	0.6
R203_C10x	436.8^{+}	2.6%	2.6%	**0.7*****	**0.0%**	**2.4%*****	0.8
RC102_C10x	618.7	1.1%	2.1%	**0.4*****	**0.0%**	**1.0%****	0.5
RC108_C10x	559.9	**0.0%**	0.7%	**0.3*****	**0.0%**	**0.0%*****	**0.3**
RC201_C10x	495.5^{+}	0.1%	**0.1%**	**0.4*****	**0.0%**	**0.1%***	0.9
RC205_C10x	576.2^{+}	**0.0%**	0.8%	**0.4*****	**0.0%**	**0.1%*****	0.8
Avg	489.0	0.8%	2.6%	**0.4**	**0.0%**	**0.5%**	0.5
C103_C15x	578.7	**0.0%**	0.9%	**0.5*****	0.1%	**0.7%**	1.5
C106_C15x	505.0	2.3%	3.3%	**0.4*****	**0.0%**	**2.5%**	1.3
C202_C15x	555.4	10.3%	12.3%	**0.6*****	**0.0%**	**1.2%*****	1.0
C208_C15x	550.0	10.4%	14.8%	**0.6*****	**0.0%**	**2.0%*****	1.1
R102_C15x	703.9	1.9%	5.9%	**0.4*****	**0.0%**	**1.2%*****	0.6
R105_C15x	608.0	**0.0%**	3.0%	0.2	**0.0%**	**0.5%****	**0.1****
R202_C15x	593.7	**0.0%**	1.2%	**0.3*****	**0.0%**	**0.1%*****	0.4
R209_C15x	475.1	9.3%	12.8%	0.6	**0.0%**	**2.5%*****	**0.5****
RC103_C15x	616.3	**0.0%**	1.0%	**0.4*****	**0.0%**	**0.3%****	0.7
RC108_C15x	609.3	**0.0%**	3.7%	**0.4*****	**0.0%**	**0.9%*****	0.8
RC202_C15x	553.5	8.9%	9.3%	**0.6*****	**0.0%**	**0.3%*****	1.1
RC204_C15x	491.4	**0.0%**	11.7%	**0.6**	**0.0%**	**1.9%*****	**0.6**
Avg	577.2	3.9%	6.2%	**0.5**	**0.0%**	**1.1%**	0.8

Tabela 6 – Resultados para instâncias grandes (30 execuções). Comparação com α_{1n} e α_{2n} iguais a 0 ($\text{IG}^{\alpha=0}_{\text{RVND}}$) com o IG_{RVND}. Teste de Wilcoxon rank-sum (nível de significância de 0.05) entre os algoritmos ($\text{IG}^{\alpha=0}_{\text{RVND}}$ e IG_{RVND}), por instância. *p-value* igual a `NA` (-). Diferenças não estatisticamente significativa não são marcadas. Resultados com diferença estatisticamente significativa tem a magnitude: pequena (*), média (**), ou grande (***). Tempos de CPU (colunas T) estão em segundos.

		$\text{IG}^{\alpha=0}_{\text{RVND}}$			IG_{RVND}		
Instance	Best	Gap Best	Gap Avg	T	Gap Best	Gap Avg	T
C101_21x	1467.2	3.2%	8.9%	**16.6****	**0.0%**	**2.9%*****	17.3
C102_21x	1449.5	3.4%	5.2%	**18.4****	**0.0%**	**2.1%*****	22.5
C103_21x	1398.7	4.8%	7.1%	**20.7*****	**0.0%**	**1.5%*****	25.6
C104_21x	1383.5	3.2%	7.3%	**23.3*****	**0.0%**	**2.2%*****	31.3
C105_21x	1458.8	4.2%	6.6%	**29.4*****	**0.0%**	**2.3%*****	40.0
C106_21x	1428.8	4.5%	6.3%	**27.1*****	**0.0%**	**2.2%*****	31.2
C107_21x	1446.3	4.8%	7.3%	**26.2*****	**0.0%**	**2.4%*****	34.1
C108_21x	1413.1	3.2%	6.1%	**23.7*****	**0.0%**	**2.0%*****	29.5
C109_21x	1402.8	5.4%	6.9%	**22.7*****	**0.0%**	**1.7%*****	30.8
C201_21x	1204.0	1.9%	4.7%	**12.7****	**0.0%**	**1.6%*****	14.5
C202_21x	1183.3	2.8%	4.7%	**16.8**	**0.0%**	**1.8%*****	18.1
C203_21x	1170.4	3.5%	5.7%	**16.8***	**0.0%**	**1.4%*****	18.9
C204_21x	1141.7	4.4%	8.3%	**15.8****	**0.0%**	**1.5%*****	18.2
C205_21x	1197.5	1.4%	3.3%	**17.2**	**0.0%**	**0.8%*****	18.1
C206_21x	1177.1	2.0%	4.1%	**15.0*****	**0.0%**	**0.9%*****	18.8
C207_21x	1159.5	2.1%	6.1%	**14.3*****	**0.0%**	**1.1%*****	19.3
C208_21x	1157.7	4.6%	8.0%	**14.5****	**0.0%**	**1.2%*****	16.8
Avg	1317.6	3.4%	6.2%	**19.8**	**0.0%**	**1.8%**	24.3
R101_21x	1706.5	**0.0%**	**0.9%**	17.8	0.2%	1.6%	**16.1**
R102_21x	1575.3	0.1%	**0.9%**	20.2	**0.0%**	**0.9%**	**16.9****
R103_21x	1468.6	0.2%	3.4%	17.1	**0.0%**	**2.4%**	**14.6****
R104_21x	1406.9	1.6%	3.6%	14.4	**0.0%**	**2.6%****	**11.1*****
R105_21x	1591.6	0.1%	2.0%	14.6	**0.0%**	**1.0%****	**12.9**
R106_21x	1509.7	0.9%	2.1%	**13.0**	**0.0%**	**1.3%****	15.0
R107_21x	1436.0	0.1%	2.6%	16.2	**0.0%**	**1.4%***	**13.3***
R108_21x	1409.8	**0.0%**	2.8%	13.0	**0.0%**	**1.5%***	**11.4**
R109_21x	1469.6	0.4%	3.0%	13.9	**0.0%**	**1.9%***	**11.0****
R110_21x	1416.8	**0.0%**	4.8%	16.1	0.5%	**4.4%**	**8.4*****
R111_21x	1419.9	0.2%	4.8%	15.1	**0.0%**	**4.6%**	**8.8*****
R112_21x	1373.3	0.7%	2.6%	11.6	**0.0%**	**1.8%****	**9.3****
R201_21x	1198.4	0.3%	1.1%	19.9	**0.0%**	**0.5%****	**17.1****
R202_21x	1124.8	0.2%	1.8%	19.0	**0.0%**	**1.0%****	**14.3*****
R203_21x	1042.7	0.1%	**1.1%**	**27.0**	**0.0%**	1.2%	**27.0**
R204_21x	961.4	**0.0%**	0.4%	20.4	**0.0%**	**0.2%****	**16.8*****
R205_21x	1103.1	**0.0%**	1.2%	20.9	**0.0%**	**0.7%****	**19.7**
R206_21x	1074.7	0.4%	2.2%	24.6	**0.0%**	**1.2%*****	**21.1****
R207_21x	1013.8	**0.0%**	1.2%	25.9	**0.0%**	**0.7%****	**19.9*****
R208_21x	961.0	**0.0%**	0.5%	25.6	**0.0%**	**0.3%****	**17.8*****
R209_21x	1056.2	0.2%	**0.8%**	31.4	**0.0%**	**0.8%**	**24.8*****
R210_21x	1035.2	**0.0%**	1.2%	30.4	**0.0%**	**0.7%****	**25.9****
R211_21x	995.1	0.5%	1.1%	19.3	**0.0%**	**0.8%**	**16.0*****
Avg	1288.9	0.2%	2.0%	19.4	**0.0%**	**1.5%**	**16.0**

Tabela 7 – Resultados para instâncias grandes (30 execuções). Comparação com α_{1n} e α_{2n} iguais a 0 ($^{\alpha=0}_{Rvnd}$) com IG_{RVND}. Teste de Wilcoxon rank-sum (nível de significância de 0.05) entre os algoritmos ($^{\alpha=0}_{Rvnd}$ e $_{Rvnd}$), por instância. *p-value* igual a `NA` (-). Diferenças não estatisticamente significativa não são marcadas. Resultados com diferença estatisticamente significativa tem a magnitude: pequena (*), média (**), ou grande (***). Tempos de CPU (colunas T) estão em segundos.

Instance	Best	$\text{IG}^{\alpha=0}_{\text{RVND}}$			IG_{RVND}		
		Gap Best	Gap Avg	T	Gap Best	Gap Avg	T
RC101_21x	1826.6	1.4%	3.4%	14.9	**0.0%**	**2.7%**	**9.5*****
RC102_21x	1791.8	0.6%	2.6%	15.4	**0.0%**	**2.3%**	**10.6*****
RC103_21x	1707.2	1.3%	3.0%	15.7	**0.0%**	**2.0%****	**12.2****
RC104_21x	1623.9	3.1%	4.5%	17.8	**0.0%**	**2.4%*****	**15.4****
RC105_21x	1771.7	1.6%	4.1%	12.0	**0.0%**	**3.2%**	**7.4*****
RC106_21x	1731.4	0.1%	2.4%	10.7	**0.0%**	**1.8%**	**7.2****
RC107_21x	1669.8	2.0%	3.3%	15.8	**0.0%**	**2.1%*****	**11.6*****
RC108_21x	1644.7	1.6%	3.0%	17.8	**0.0%**	**1.6%*****	**12.2*****
RC201_21x	1289.2	**0.0%**	0.5%	**14.0*****	**0.0%**	**0.3%**	17.8
RC202_21x	1191.0	0.1%	1.1%	**17.5**	**0.0%**	**0.2%*****	18.8
RC203_21x	1072.4	0.2%	2.3%	16.8	**0.0%**	**0.7%*****	**16.0**
RC204_21x	1009.3	1.2%	3.1%	22.9	**0.0%**	**1.3%*****	**20.3****
RC205_21x	1173.7	0.5%	2.8%	25.1	**0.0%**	**1.8%****	**22.2**
RC206_21x	1163.2	1.1%	3.3%	16.3	**0.0%**	**0.9%*****	**15.8**
RC207_21x	1083.2	0.6%	3.4%	**15.0***	**0.0%**	**1.2%*****	18.2
RC208_21x	1011.7	0.7%	2.5%	21.0	**0.0%**	**1.8%****	**17.8****
Avg	1449.9	1.0%	2.9%	16.5	**0.0%**	**1.6%**	**14.4**

5.6.2 Resultados comparando o RVND

Outra diferença importante do IG proposto com o IG padrão (como descrito na literatura) é a utilização de busca local após a reconstrução da solução (nesse trabalho utiliza-se o RVND). Para confirmar que a alteração produz melhores resultados, o IG proposto com RVND é comparado o IG sem busca local, representado como IG_{Rvnd_0}. Os parâmetros foram ajustados como descrito na Seção 5.5. A seguir, os resultados são apresentados comparando variantes do método proposto.

A Tabela 8 mostra os resultados dos dois IGs para os três grupos: C5, C10, e C15, respectivamente, 5, 10, e 15 clientes. Para os três grupos o IG_{RVND} obteve melhores resultados do que *IG* para a distância. Em relação ao tempo de processamento, o *IG* obteve os menores valores para os grupos C10 e C15.

Para o grupo C5, as médias de Gap-Best obtidas são: 0.0% para o IG, e o IG_{RVND}. Em relação à Gap-Avg, os valores obtidos são: 1.6% (IG) e 0.8 (IG_{RVND}). Existe uma diferença estatisticamente (teste de Wilcoxon rank-sum por instância) significativa para 4 instâncias (33%), em relação a distância. A magnitude do teste é pequeno (8%), médio (8%), e grande (17%).

Para o grupo C15 as médias de Gap-Best obtidas são: 4% (IG), e 0.0% (IG_{RVND}). Em relação à Gap-Avg, os valores obtidos são: 8.3% (IG) e 1.1 (IG_{RVND}). Existe uma diferença estatisticamente significativa para 11 instâncias (92%), em relação a distância. A magnitude do teste é médio (25%), e grande (67%).

A Tabela 9 apresenta os resultados dos dois IGs para os grupos C100 e R100; ambos possuem 100 clientes. O IG_{RVND} possui melhores resultados do que o IG considerando a distância. O IG obteve menores tempos de processamento.

Para o grupo C100 as médias de Gap-Best obtidas são: 3.3% (IG), e 0.0% (IG_{RVND}). Em relação à Gap-Avg, os valores obtidos são: 7.1% (IG) e 1.8 (IG_{RVND}). O IG é 8 vezes mais rápido do que o IG_{RVND}. Existe uma diferença estatisticamente significativa para todas as instâncias, em relação a distância e tempo de processamento. A magnitude do teste é grande para todas as instâncias.

Para o grupo R100 as médias de Gap-Best obtidas são: 4.0% (IG), e 0.0% (IG_{RVND}). Em relação à Gap-Avg, os valores obtidos são: 8.7% (IG) e 1.5 (IG_{RVND}). O IG é 3 vezes mais rápido do que o IG_{RVND}. Existe uma diferença estatisticamente significativa para todas as instâncias, em relação a distância e tempo de processamento. A magnitude do teste é grande para todas as instâncias, com exceção de uma.

A Tabela 10 apresenta os resultados para o grupo RC100, também com 100 clientes. O IG_{RVND} possui melhores resultados do que o IG considerando a distância. O IG obteve menores tempos de processamento. As médias de Gap-Best obtidas são: 4.1% (IG), e 0.0% (IG_{RVND}). Em relação à Gap-Avg, os valores obtidos são: 9.7% (IG) e 1.6 (IG_{RVND}).

O IG é 4 vezes mais rápido do que o IG_{RVND}. Existe uma diferença estatisticamente significativa para todas as instâncias, em relação a distância e tempo de processamento. A magnitude do teste é grande para todas as instâncias.

A análise dos resultados comparando o IG e IG_{RVND} permite concluir que a inclusão da busca local RVND contribui significativamente para a melhora dos resultados.

Tabela 8 – Resultados para instâncias grandes (30 execuções). Comparação do IG com IG_{RVND}. Teste de Wilcoxon rank-sum (nível de significância de 0.05) entre os algoritmos (IG e IG_{RVND}), por instância. *p-value* igual a `NA` (-). Diferenças não estatisticamente significativa não são marcadas. Resultados com diferença estatisticamente significativa tem a magnitude: pequena (*), média (**), ou grande (***). Tempos de CPU (colunas T) estão em segundos. (+) Resultados iguais ao ótimo obtido pelo CPLEX[1].

Instance	Best	IG			IG_{RVND}		
		Gap Best	Gap Avg	T	Gap Best	Gap Avg	T
C101_C5x	385.5^{+}	**0.0%**	**0.0%**$^{-}$	**0.1*****	**0.0%**	**0.0%**$^{-}$	**0.1**
C103_C5x	341.3^{+}	**0.0%**	**0.0%**$^{-}$	**0.1**	**0.0%**	**0.0%**$^{-}$	**0.1**
C206_C5x	417.3^{+}	**0.0%**	0.4%	**0.1**	**0.0%**	**0.0%***	**0.1*****
C208_C5x	381.9^{+}	**0.0%**	**0.0%**$^{-}$	**0.1**	**0.0%**	**0.0%**$^{-}$	**0.1*****
R104_C5x	317.0^{+}	**0.0%**	**0.0%**$^{-}$	**0.1*****	**0.0%**	**0.0%**$^{-}$	**0.1**
R105_C5x	453.7^{+}	**0.0%**	5.0%	**0.1*****	**0.0%**	**0.0%*****	0.2
R202_C5x	347.8^{+}	**0.0%**	**0.0%**	**0.1**	**0.0%**	**0.0%**	**0.1*****
R203_C5x	371.3^{+}	**0.0%**	**0.0%**$^{-}$	**0.1**	**0.0%**	**0.0%**$^{-}$	**0.1*****
RC105_C5x	432.6^{+}	**0.0%**	3.5%	**0.1*****	**0.0%**	**0.0%*****	**0.1**
RC108_C5x	460.9^{+}	**0.0%**	8.4%	**0.1**	**0.0%**	**8.3%**	**0.1**
RC204_C5x	334.9^{+}	**0.0%**	0.2%	**0.1**	**0.0%**	**0.0%****	**0.1*****
RC208_C5x	327.3^{+}	**0.0%**	**0.0%**	0.2	**0.0%**	**0.0%**	**0.1*****
Avg	385.9	**0.0%**	1.6%	**0.1**	**0.0%**	**0.8%**	**0.1**
C101_C10x	539.7^{+}	3.4%	3.5%	**0.2*****	**0.0%**	**0.0%*****	0.6
C104_C10x	484.3^{+}	**0.0%**	4.1%	**0.4*****	**0.0%**	**0.0%*****	0.5
C202_C10x	425.5^{+}	**0.0%**	13.0%	**0.2*****	**0.0%**	**1.2%*****	0.4
C205_C10x	415.5^{+}	**0.0%**	0.5%	**0.2*****	0.2%	**0.2%**	0.5
R102_C10x	505.5^{+}	**0.0%**	4.8%	**0.2*****	**0.0%**	**0.0%*****	0.4
R103_C10x	436.1	**0.0%**	3.7%	**0.2*****	**0.0%**	**0.0%****	0.4
R201_C10x	460.7^{+}	0.3%	3.9%	**0.2*****	**0.0%**	**1.1%*****	0.6
R203_C10x	436.8^{+}	2.6%	2.6%	**0.6*****	**0.0%**	**2.4%*****	0.8
RC102_C10x	618.7	1.1%	2.1%	**0.2*****	**0.0%**	**1.0%*****	0.5
RC108_C10x	559.9	**0.0%**	0.7%	**0.2*****	**0.0%**	**0.0%*****	0.3
RC201_C10x	495.5^{+}	0.1%	0.5%	**0.3*****	**0.0%**	**0.1%****	0.9
RC205_C10x	576.2^{+}	**0.0%**	0.7%	**0.3*****	**0.0%**	**0.1%****	0.8
Avg	488.9	0.7%	3.6%	**0.3**	**0.0%**	**0.5%**	0.5
C103_C15x	579.3	**0.0%**	4.0%	**0.5*****	**0.0%**	**0.6%****	1.5
C106_C15x	505.0	2.3%	4.1%	**0.4*****	**0.0%**	**2.5%****	1.3
C202_C15x	555.4	10.3%	14.1%	**0.6*****	**0.0%**	**1.2%*****	1.0
C208_C15x	550.0	10.3%	14.3%	**0.6*****	**0.0%**	**2.0%*****	1.1
R102_C15x	703.9	3.1%	7.9%	**0.2*****	**0.0%**	**1.2%*****	0.6
R105_C15x	608.0	**0.0%**	5.9%	**0.1**	**0.0%**	**0.5%****	**0.1**
R202_C15x	593.7	**0.0%**	5.3%	0.8	**0.0%**	**0.1%*****	**0.4*****
R209_C15x	475.1	9.3%	15.0%	0.8	**0.0%**	**2.5%*****	**0.5*****
RC103_C15x	616.3	**0.0%**	1.0%	**0.4*****	**0.0%**	**0.3%**	0.7
RC108_C15x	609.3	**0.0%**	4.7%	**0.5*****	**0.0%**	**0.9%*****	0.8
RC202_C15x	553.5	8.9%	15.5%	**0.7*****	**0.0%**	**0.3%*****	1.1
RC204_C15x	491.4	12.2%	13.7%	1.0	**0.0%**	**1.9%*****	**0.6*****
Avg	577.2	4.0%	8.3%	**0.5**	**0.0%**	**1.1%**	0.8

Tabela 9 – Resultados para instâncias grandes (30 execuções). Comparação entre o IG e IG_{RVND}. Teste de Wilcoxon rank-sum (nível de significância de 0.05) entre os algoritmos (IG e IG_{RVND}), por instância. Resultados com diferença estatisticamente significativa tem a magnitude: pequena (*), média (**), ou grande (***). Tempos de CPU (colunas T) estão em segundos.

		IG			IG_{RVND}		
Instance	Best	Gap Best	Gap Avg	T	Gap Best	Gap Avg	T
C101_21x	1467.2	4.9%	8.2%	**2.4*****	**0.0%**	**2.9%*****	17.3
C102_21x	1449.5	3.6%	6.3%	**2.8*****	**0.0%**	**2.1%*****	22.5
C103_21x	1398.7	3.0%	6.3%	**3.6*****	**0.0%**	**1.5%*****	25.6
C104_21x	1383.5	2.9%	5.7%	**4.0*****	**0.0%**	**2.2%*****	31.3
C105_21x	1458.8	3.3%	6.6%	**2.5*****	**0.0%**	**2.3%*****	40.0
C106_21x	1428.8	3.2%	6.9%	**2.7*****	**0.0%**	**2.2%*****	31.2
C107_21x	1446.3	3.7%	7.0%	**2.8*****	**0.0%**	**2.4%*****	34.1
C108_21x	1413.1	1.6%	6.4%	**3.2*****	**0.0%**	**2.0%*****	29.5
C109_21x	1402.8	3.5%	6.1%	**3.2*****	**0.0%**	**1.7%*****	30.8
C201_21x	1204.0	3.0%	7.5%	**2.1*****	**0.0%**	**1.6%*****	14.5
C202_21x	1183.3	2.8%	8.7%	**3.1*****	**0.0%**	**1.8%*****	18.1
C203_21x	1170.4	4.5%	8.2%	**3.2*****	**0.0%**	**1.4%*****	18.9
C204_21x	1141.7	5.5%	8.5%	**4.0*****	**0.0%**	**1.5%*****	18.2
C205_21x	1197.5	2.2%	7.0%	**2.8*****	**0.0%**	**0.8%*****	18.1
C206_21x	1177.1	3.5%	5.9%	**2.9*****	**0.0%**	**0.9%*****	18.8
C207_21x	1159.5	2.0%	7.5%	**3.2*****	**0.0%**	**1.1%*****	19.3
C208_21x	1157.7	3.2%	7.2%	**3.4*****	**0.0%**	**1.2%*****	16.8
Avg	1317.6	3.3%	7.1%	**3.0**	**0.0%**	**1.8%**	24.3
R101_21x	1710.6	1.0%	3.9%	**3.4*****	**0.0%**	**1.4%*****	16.1
R102_21x	1575.3	1.4%	5.4%	**3.5*****	**0.0%**	**0.9%*****	16.9
R103_21x	1468.6	4.7%	10.0%	**6.4*****	**0.0%**	**2.4%*****	14.6
R104_21x	1406.9	5.4%	10.4%	**6.9*****	**0.0%**	**2.6%*****	11.1
R105_21x	1591.6	2.6%	6.2%	**4.6*****	**0.0%**	**1.0%*****	12.9
R106_21x	1509.7	3.8%	6.7%	**4.0*****	**0.0%**	**1.3%*****	15.0
R107_21x	1436.0	4.0%	9.5%	**5.7*****	**0.0%**	**1.4%*****	13.3
R108_21x	1409.8	4.1%	8.2%	**6.3*****	**0.0%**	**1.5%*****	11.4
R109_21x	1469.6	4.1%	8.7%	**4.5*****	**0.0%**	**1.9%*****	11.0
R110_21x	1424.1	4.6%	11.3%	**5.8*****	**0.0%**	**3.9%*****	8.4
R111_21x	1419.9	5.1%	11.0%	**6.8****	**0.0%**	**4.6%*****	8.8
R112_21x	1373.3	4.4%	7.0%	**3.7*****	**0.0%**	**1.8%*****	9.3
R201_21x	1198.4	4.8%	8.3%	**4.4*****	**0.0%**	**0.5%*****	17.1
R202_21x	1124.8	4.4%	9.6%	**3.3*****	**0.0%**	**1.0%*****	14.3
R203_21x	1042.7	3.6%	9.6%	**6.2*****	**0.0%**	**1.2%*****	27.0
R204_21x	961.4	2.8%	4.8%	**2.3*****	**0.0%**	**0.2%*****	16.8
R205_21x	1103.1	4.3%	9.8%	**6.1*****	**0.0%**	**0.7%*****	19.7
R206_21x	1074.7	4.8%	8.7%	**7.6*****	**0.0%**	**1.2%*****	21.1
R207_21x	1013.8	4.3%	8.8%	**1.9*****	**0.0%**	**0.7%*****	19.9
R208_21x	961.0	3.9%	6.5%	**2.7*****	**0.0%**	**0.3%*****	17.8
R209_21x	1056.2	5.0%	13.2%	**9.3*****	**0.0%**	**0.8%*****	24.8
R210_21x	1035.2	5.5%	13.0%	**9.2*****	**0.0%**	**0.7%*****	25.9
R211_21x	995.1	4.1%	7.2%	**2.1*****	**0.0%**	**0.8%*****	16.0
Avg	1289.4	4.0%	8.7%	**5.2**	**0.0%**	**1.5%**	16.0

Tabela 10 – Resultados para instâncias grandes (30 execuções). Comparação entre IG e IG_{RVND}. Teste de Wilcoxon rank-sum (nível de significância de 0.05) entre os algoritmos (IG e IG_{RVND}), por instância. Diferenças não estatisticamente significativa não são marcadas. Resultados com diferença estatisticamente significativa tem a magnitude: pequena (*), média (**), ou grande (***). Tempos de CPU (colunas T) estão em segundos.

Instance	Best	IG			IG_{RVND}		
		Gap Best	Gap Avg	T	Gap Best	Gap Avg	T
RC101_21x	1826.6	5.5%	8.3%	**2.2*****	**0.0%**	**2.7%*****	9.5
RC102_21x	1791.8	4.5%	8.2%	**2.5*****	**0.0%**	**2.3%*****	10.6
RC103_21x	1707.2	3.5%	9.5%	**2.9*****	**0.0%**	**2.0%*****	12.2
RC104_21x	1623.9	4.3%	8.7%	**2.6*****	**0.0%**	**2.4%*****	15.4
RC105_21x	1771.7	6.0%	12.0%	**3.9*****	**0.0%**	**3.2%*****	7.4
RC106_21x	1731.4	4.3%	7.4%	**2.8*****	**0.0%**	**1.8%*****	7.2
RC107_21x	1669.8	4.4%	8.2%	**2.4*****	**0.0%**	**2.1%*****	11.6
RC108_21x	1644.7	3.1%	7.1%	**2.3*****	**0.0%**	**1.6%*****	12.2
RC201_21x	1289.2	0.9%	9.3%	**3.7*****	**0.0%**	**0.3%*****	17.8
RC202_21x	1191.0	2.8%	9.1%	**2.0*****	**0.0%**	**0.2%*****	18.8
RC203_21x	1072.4	5.5%	9.8%	**3.6*****	**0.0%**	**0.7%*****	16.0
RC204_21x	1009.3	2.3%	6.5%	**10.5*****	**0.0%**	**1.3%*****	20.3
RC205_21x	1173.7	5.4%	20.3%	**6.8*****	**0.0%**	**1.8%*****	22.2
RC206_21x	1163.2	3.5%	11.2%	**3.7*****	**0.0%**	**0.9%*****	15.8
RC207_21x	1083.2	4.8%	10.5%	**5.5*****	**0.0%**	**1.2%*****	18.2
RC208_21x	1011.7	3.6%	11.7%	**6.7*****	**0.0%**	**1.8%*****	17.8
Avg	1449.9	4.1%	9.7%	**3.8**	**0.0%**	**1.6%**	14.4

5.6.3 Análise da influência do particionamento de rotas

Nessa seção, o IG_{RVND} e $\text{IG}_{\text{RVND}^{\text{MILP}}}$ são comparados. A Tabela 11 apresenta os resultados obtidos pelos dois algoritmos IGs propostos com 30 execuções independentes para os grupos C5, C10, e C15. Os resultados obtidos pelo $\text{IG}^{\text{MILP}}_{\text{RVND}}$ são melhores do que o IG_{RVND} quando a distância é considerada.

Para o grupo C5, as médias de Gap-Best obtidas são: 0.0% para o IG_{RVND}, e o $\text{IG}^{\text{MILP}}_{\text{RVND}}$. Para o Gap-Avg, os valores de média obtidos são: 1.6% (IG_{RVND}), e 0.8% ($\text{IG}^{\text{MILP}}_{\text{RVND}}$). O IG_{RVND} obteve tempos de processamento menores do que $\text{IG}^{\text{MILP}}_{\text{RVND}}$. Essa diferença não pode ser observada na tabela por causa da limitação adotada aqui de uma casa decimal de precisão. Os resultados das distâncias para duas abordagens são iguais (com três casas decimais) para a maior parte desse grupo, por conta disso, o *p-value* é igual à `NA`. Para a instância RC108_C5x, existe diferença estatisticamente significativa (teste de Wilcoxon rank-sum por instância) para a distância. A magnitude do teste é médio. Em relação ao tempo de processamento, existe uma diferença estatisticamente significativa para todas as instâncias, onde o IG_{RVND} possui uma média menor do que $\text{IG}^{\text{MILP}}_{\text{RVND}}$ para todas as instâncias. A magnitude do teste é grande.

Para o grupo C10, as médias de Gap-Best obtidas são: 0.0% (IG_{RVND}), e 0.1% ($\text{IG}^{\text{MILP}}_{\text{RVND}}$). Para o Gap-Avg, os valores de média obtidos são: 0.6% (IG_{RVND}), e 0.3% ($\text{IG}^{\text{MILP}}_{\text{RVND}}$). O IG_{RVND} é 1.4 vezes mais rápido do que $\text{IG}^{\text{MILP}}_{\text{RVND}}$. Existe uma diferença estatisticamente significativa para 58% das instâncias, relacionado a distância (função objetivo). A magnitude do teste é: médio (8%), e grande (50%). Em relação ao tempo de processamento, existe uma diferença estatisticamente significativa para todas as instâncias, onde o IG_{RVND} possui uma média menor do que $\text{IG}^{\text{MILP}}_{\text{RVND}}$ para todas as instâncias. A magnitude do teste é grande.

Para o grupo C15, as médias de Gap-Best obtidas são: 0.4% (IG_{RVND}), e 0.2% ($\text{IG}^{\text{MILP}}_{\text{RVND}}$). Para o Gap-Avg, os valores de média obtidos são: 1.6% (IG_{RVND}), e 0.8% ($\text{IG}^{\text{MILP}}_{\text{RVND}}$). O IG_{RVND} é 1.9 vezes mais rápido do que $\text{IG}^{\text{MILP}}_{\text{RVND}}$. Existe uma diferença estatisticamente significativa para 92% das instâncias, relacionado a distância. A magnitude do teste é médio (42%) e grande (50%). Em relação ao tempo de processamento, existe uma diferença estatisticamente significativa para todas as instâncias. A magnitude do teste é: pequeno (8%), médio (42%), e grande (50%).

A Tabela 12 apresenta os resultados obtidos pelos métodos IG_{RVND} e $\text{IG}^{\text{MILP}}_{\text{RVND}}$ para os grupos C100, e R100. O $\text{IG}^{\text{MILP}}_{\text{RVND}}$ proposto consegue resultados melhores para a distância mas, por outro lado, o IG_{RVND} consegue um melhor tempo de CPU.

Para o grupo C100, as médias de Gap-Best dos métodos são: 0.5% (IG_{RVND}), e 0.0% ($\text{IG}^{\text{MILP}}_{\text{RVND}}$). As médias em relação a Gap-Avg são: 2.3% (IG_{RVND}), e 0.6% ($\text{IG}^{\text{MILP}}_{\text{RVND}}$). O IG_{RVND} é na média 2 vezes mais rápido do que o $\text{IG}^{\text{MILP}}_{\text{RVND}}$. Existe uma diferença estatisticamente significativa na distância para todas as instâncias do grupo. A magnitude

dessa diferença é grande para 94% das instâncias e médio para o restante. Em relação ao tempo de CPU, existe uma diferença estatisticamente significativa para todas as instâncias. A magnitude dessa diferença é grande para toadas as instâncias.

Para o grupo R100, as médias de Gap-Best dos métodos são: 0.3% (IG_{RVND}), e 0.0% ($\text{IG}^{\text{MILP}}_{\text{RVND}}$). Em relação à média de Gap-Avg, temos os seguintes resultados: 1.8% (IG_{RVND}), e 1.1% ($\text{IG}^{\text{MILP}}_{\text{RVND}}$). O IG_{RVND} é na média 2.8 vezes mais rápido do que o $\text{IG}^{\text{MILP}}_{\text{RVND}}$. Para a distância, 74% das instâncias tem uma diferença estatisticamente significativa. A magnitude do teste é: pequeno (4%), médio (43%), e grande (26%). Em relação ao tempo de CPU, existe uma diferença estatisticamente significativa para todas as instâncias. A magnitude do teste é grande para todas as instâncias.

Tabela 13 apresenta os resultados dos métodos IGs propostos para o grupo RC100. O método $\text{IG}^{\text{MILP}}_{\text{RVND}}$ obteve melhores resultados para a distância, e o IG_{RVND} conseguiu melhores tempos de processamento.

As médias relativas à Gap-Best dos métodos são: 0.4% (IG_{RVND}), e 0.0% ($\text{IG}^{\text{MILP}}_{\text{RVND}}$). As médias em relação a Gap-Avg são: 2.1% (IG), e 1.0% ($\text{IG}^{\text{MILP}}_{\text{RVND}}$). O IG_{RVND} é na média 2.5 vezes mais rápido do que o $\text{IG}^{\text{MILP}}_{\text{RVND}}$. Considerando a distância, os resultados de 81% das instâncias têm uma diferença estatisticamente significativa. As magnitudes do teste são: pequeno (6%), médio (25%), e grande (50%). Para o tempo de processamento, 87% das instâncias têm uma diferença estatisticamente significativa. A magnitude é grande para todas essas instâncias.

O acréscimo do particionamento de rotas ao IG contribui de forma significativa para a melhora dos resultados, relacionados a distância, segundo a análise dos métodos IG_{RVND} e $\text{IG}^{MILP}_{\text{RVND}}$. Os testes estatísticos corroboram com essa conclusão.

Existe uma diferença do componente IG dos métodos IG_{RVND} e $\text{IG}^{MILP}_{\text{RVND}}$ realizada pela parametrização. A diferença consiste em como a busca é realizada, no IG_{RVND} a metaheurística busca uma melhor solução, já o $\text{IG}^{MILP}_{\text{RVND}}$ busca adicionar novas rotas de EVs ao conjunto de rotas. Essa diferença sutil faz com que o resultado do componente IG do $\text{IG}^{MILP}_{\text{RVND}}$ gere um resultado pior, relativo à distância, do que o IG_{RVND}.

Tabela 11 – Resultados para instâncias pequenas (30 execuções).Teste de Wilcoxon rank-sum (nível de significância de 0.05) entre os algoritmos (IG_{RVND} e IG^{MILP}_{RVND}), por instância. *p-value* igual a `NA` (-). Diferenças não estatisticamente significativa não são marcadas. Resultados com diferença estatisticamente significativa tem a magnitude: pequena (*), média (**), ou grande (***). Tempos de CPU (colunas T) estão em segundos.
(+) Resultados iguais ao ótimo obtido pelo CPLEX[1].

Instance	Best	IG_{RVND}			IG^{MILP}_{RVND}		
		Gap Best	Gap Avg	T	Gap Best	Gap Avg	T
C101_C5x	385.5^{+}	**0.0%**	**0.0%**$^{-}$	**0.1*****	**0.0%**	**0.0%**$^{-}$	**0.1**
C103_C5x	341.3^{+}	**0.0%**	**0.0%**$^{-}$	**0.1*****	**0.0%**	**0.0%**$^{-}$	**0.1**
C206_C5x	417.3^{+}	**0.0%**	**0.0%**$^{-}$	**0.1*****	**0.0%**	**0.0%**$^{-}$	**0.1**
C208_C5x	381.9^{+}	**0.0%**	**0.0%**$^{-}$	**0.1*****	**0.0%**	**0.0%**$^{-}$	0.2
R104_C5x	317.0^{+}	**0.0%**	**0.0%**$^{-}$	**0.1*****	**0.0%**	**0.0%**$^{-}$	0.2
R105_C5x	453.7^{+}	**0.0%**	**0.0%**$^{-}$	**0.2*****	**0.0%**	**0.0%**$^{-}$	**0.2**
R202_C5x	347.8^{+}	**0.0%**	**0.0%**	**0.1*****	**0.0%**	**0.0%**	**0.1**
R203_C5x	371.3^{+}	**0.0%**	**0.0%**$^{-}$	**0.1*****	**0.0%**	**0.0%**$^{-}$	**0.1**
RC105_C5x	432.6^{+}	**0.0%**	**0.0%**$^{-}$	**0.1*****	**0.0%**	**0.0%**$^{-}$	**0.1**
RC108_C5x	460.9^{+}	**0.0%**	8.3%	**0.1*****	**0.0%**	**3.8%****	**0.1**
RC204_C5x	334.9^{+}	**0.0%**	**0.0%**$^{-}$	**0.1*****	**0.0%**	**0.0%**$^{-}$	**0.1**
RC208_C5x	327.3^{+}	**0.0%**	**0.0%**$^{-}$	**0.1*****	**0.0%**	**0.0%**$^{-}$	0.2
Avg	-	**0.0%**	0.8%	**0.1**	**0.0%**	**0.4%**	**0.1**
C101_C10x	538.3^{+}	0.3%	0.3%	**0.6*****	**0.0%**	**0.1%*****	0.8
C104_C10x	484.3^{+}	**0.0%**	**0.0%**$^{-}$	**0.5*****	**0.0%**	**0.0%**$^{-}$	0.7
C202_C10x	425.5^{+}	**0.0%**	1.2%	**0.4*****	**0.0%**	**0.0%*****	0.6
C205_C10x	415.5^{+}	0.2%	0.2%	**0.5*****	**0.0%**	**0.0%*****	0.6
R102_C10x	505.5^{+}	**0.0%**	**0.0%**$^{-}$	**0.4*****	**0.0%**	**0.0%**$^{-}$	0.5
R103_C10x	436.1	**0.0%**	**0.0%**$^{-}$	**0.4*****	**0.0%**	**0.0%**$^{-}$	0.5
R201_C10x	460.7^{+}	**0.0%**	1.1%	**0.6*****	**0.0%**	**0.0%*****	1.0
R203_C10x	436.8^{+}	**0.0%**	2.4%	**0.8*****	**0.0%**	**2.3%*****	0.9
RC102_C10x	618.7	**0.0%**	**1.0%**	**0.5*****	1.1%	1.1%	0.6
RC108_C10x	559.9	**0.0%**	**0.0%**	**0.3*****	**0.0%**	**0.0%**	0.4
RC201_C10x	495.5^{+}	**0.0%**	0.1%	**0.9*****	**0.0%**	**0.0%*****	1.2
RC205_C10x	576.2^{+}	**0.0%**	**0.1%**	**0.8*****	**0.0%**	**0.1%****	1.0
Avg	-	**0.0%**	0.6%	**0.5**	0.1%	**0.3%**	0.7
C103_C15x	575.2	0.7%	1.3%	**1.5*****	**0.0%**	**1.2%**	3.3
C106_C15x	505.0	**0.0%**	2.5%	**1.3*****	2.3%	**2.3%****	2.6
C202_C15x	550.3	0.9%	2.2%	**1.0*****	**0.0%**	**0.6%*****	1.8
C208_C15x	550.0	**0.0%**	2.0%	**1.1****	**0.0%**	**0.3%****	2.1
R102_C15x	698.7	0.7%	2.0%	**0.6****	**0.0%**	**0.4%*****	**0.6**
R105_C15x	608.0	**0.0%**	0.5%	**0.1*****	**0.0%**	**0.1%*****	0.2
R202_C15x	593.7	**0.0%**	0.1%	**0.4*****	**0.0%**	**0.0%****	0.6
R209_C15x	475.1	**0.0%**	2.5%	**0.5****	**0.0%**	**1.8%****	0.8
RC103_C15x	607.4	1.5%	1.8%	**0.7***	**0.0%**	**1.3%*****	0.9
RC108_C15x	603.9	0.9%	1.8%	**0.8****	**0.0%**	**0.3%*****	0.9
RC202_C15x	552.7	0.2%	0.5%	**1.1****	**0.0%**	**0.2%****	2.3
RC204_C15x	485.3	1.2%	3.2%	**0.6*****	**0.0%**	**1.1%*****	0.9
Avg	-	0.4%	1.6%	**0.8**	**0.2%**	**0.8%**	1.5

Tabela 12 – Resultados para instâncias grandes (30 execuções). Teste de Wilcoxon rank-sum (nível de significância de 0.05) entre os algoritmos (IG_{RVND} e IG^{MILP}_{RVND}), por instância. Diferenças não estatisticamente significativa não são marcadas. Resultados com diferença estatisticamente significativa tem a magnitude: pequena (*), média (**), ou grande (***). Tempos de CPU (colunas T) estão em segundos. Tempo de CPU (colunas T) estão em segundos.

		IG_{RVND}			IG^{MILP}_{RVND}		
Instância	Best	Gap Best	Gap Avg	T	Gap Best	Gap Avg	T
C101_21x	1465.8	0.1%	3.0%	**17.3*****	**0.0%**	**1.2%*****	29.9
C102_21x	1437.2	0.9%	3.0%	**22.5*****	**0.0%**	**1.0%*****	39.8
C103_21x	1398.7	**0.0%**	1.5%	**25.6*****	**0.0%**	**0.2%****	51.0
C104_21x	1383.5	**0.0%**	2.2%	**31.3*****	**0.0%**	**0.6%*****	74.7
C105_21x	1448.4	0.7%	3.1%	**40.0*****	**0.0%**	**0.6%*****	75.1
C106_21x	1426.9	0.1%	2.3%	**31.2*****	**0.0%**	**0.6%*****	60.1
C107_21x	1443.0	0.2%	2.6%	**34.1*****	**0.0%**	**0.6%*****	76.3
C108_21x	1413.1	**0.0%**	2.0%	**29.5*****	**0.0%**	**0.5%*****	63.3
C109_21x	1399.8	0.2%	2.0%	**30.8*****	**0.0%**	**0.4%*****	67.3
C201_21x	1199.5	0.4%	2.0%	**14.5*****	**0.0%**	**0.4%*****	48.3
C202_21x	1177.6	0.5%	2.3%	**18.1*****	**0.0%**	**0.5%*****	51.7
C203_21x	1155.6	1.3%	2.7%	**18.9*****	**0.0%**	**0.6%*****	59.1
C204_21x	1129.3	1.1%	2.7%	**18.2*****	**0.0%**	**0.4%*****	55.8
C205_21x	1186.6	0.9%	1.7%	**18.1*****	**0.0%**	**0.7%*****	55.0
C206_21x	1168.8	0.7%	1.6%	**18.8*****	**0.0%**	**0.6%*****	55.0
C207_21x	1152.3	0.6%	1.8%	**19.3*****	**0.0%**	**0.8%*****	59.7
C208_21x	1156.1	0.1%	1.3%	**16.8*****	**0.0%**	**0.5%*****	51.1
Avg	-	0.5%	2.3%	**24.3**	**0.0%**	**0.6%**	57.6
R101_21x	1704.0	0.4%	1.8%	**16.1*****	**0.0%**	**0.7%****	24.0
R102_21x	1572.4	0.2%	1.1%	**16.9*****	**0.0%**	**0.5%****	23.2
R103_21x	1460.5	0.6%	3.0%	**14.6*****	**0.0%**	**2.5%****	20.5
R104_21x	1394.7	0.9%	3.5%	**11.1*****	**0.0%**	**1.7%*****	22.4
R105_21x	1584.8	0.4%	1.4%	**12.9*****	**0.0%**	**1.0%***	18.1
R106_21x	1507.5	0.1%	1.4%	**15.0**	**0.0%**	**0.7%*****	16.9
R107_21x	1428.0	0.6%	2.0%	**13.3*****	**0.0%**	**1.3%****	25.1
R108_21x	1403.3	0.5%	2.0%	**11.4*****	**0.0%**	**1.3%****	19.6
R109_21x	1462.3	0.5%	2.4%	**11.0*****	**0.0%**	**1.4%****	21.6
R110_21x	1403.5	1.5%	5.4%	**8.4*****	**0.0%**	**3.5%**	27.0
R111_21x	1416.3	0.3%	4.8%	**8.8*****	**0.0%**	**3.1%****	25.5
R112_21x	1373.2	**0.0%**	1.8%	**9.3*****	**0.0%**	**1.0%*****	20.0
R201_21x	1198.4	**0.0%**	**0.5%**	**17.1*****	**0.0%**	**0.5%**	52.0
R202_21x	1121.4	0.3%	1.3%	**14.3*****	**0.0%**	**0.6%*****	58.7
R203_21x	1042.0	0.1%	1.2%	**27.0*****	**0.0%**	**0.5%*****	84.1
R204_21x	961.4	**0.0%**	0.2%	**16.8*****	**0.0%**	**0.1%**	70.9
R205_21x	1103.1	**0.0%**	0.7%	**19.7*****	**0.0%**	**0.6%**	63.4
R206_21x	1074.7	**0.0%**	1.2%	**21.1*****	**0.0%**	**0.7%****	77.7
R207_21x	1011.9	0.2%	0.9%	**19.9*****	**0.0%**	**0.5%****	81.6
R208_21x	961.0	**0.0%**	**0.3%**	**17.8*****	**0.0%**	**0.3%**	75.5
R209_21x	1056.1	**0.0%**	0.8%	**24.8*****	**0.0%**	**0.4%****	82.3
R210_21x	1035.2	**0.0%**	0.7%	**25.9*****	**0.0%**	**0.6%**	82.9
R211_21x	994.7	**0.0%**	0.8%	**16.0*****	**0.0%**	**0.4%*****	60.0
Avg	-	0.3%	1.8%	**16.0**	**0.0%**	**1.1%**	45.1

Tabela 13 – Resultados para instâncias grandes (30 execuções).Teste de Wilcoxon rank-sum (nível de significância de 0.05) entre os algoritmos (IG_{RVND} e IG^{MILP}_{RVND}), por instância. Diferenças não estatisticamente significativa não são marcadas. Resultados com diferença estatisticamente significativa tem a magnitude: pequena (*), média (**), ou grande (***). Tempos de CPU (colunas T) estão em segundos.

		IG_{RVND}			IG^{MILP}_{RVND}		
Instâncias	Best	Gap Best	Gap Avg	T	Gap Best	Gap Avg	T
RC101_21x	1807.0	1.1%	3.8%	**9.5*****	**0.0%**	**1.8%*****	24.4
RC102_21x	1763.1	1.6%	3.9%	**10.6*****	**0.0%**	**1.4%*****	27.5
RC103_21x	1699.0	0.5%	2.5%	**12.2*****	**0.0%**	**1.0%*****	23.0
RC104_21x	1620.8	0.2%	2.6%	**15.4*****	**0.0%**	**1.3%*****	21.1
RC105_21x	1756.6	0.9%	4.1%	**7.4*****	**0.0%**	**2.6%*****	30.5
RC106_21x	1716.8	0.8%	2.7%	**7.2*****	**0.0%**	**1.0%*****	16.8
RC107_21x	1665.9	0.2%	2.3%	**11.6**	**0.0%**	**1.1%*****	13.7
RC108_21x	1637.3	0.5%	2.1%	**12.2**	**0.0%**	**1.2%****	13.4
RC201_21x	1289.0	**0.0%**	**0.3%**	**17.8*****	**0.0%**	**0.3%**	47.1
RC202_21x	1191.0	**0.0%**	**0.2%**	**18.8*****	**0.0%**	**0.2%**	48.8
RC203_21x	1072.4	**0.0%**	0.7%	**16.0*****	**0.0%**	**0.3%**	46.7
RC204_21x	1002.1	0.7%	2.0%	**20.3*****	**0.0%**	**1.3%****	70.8
RC205_21x	1173.4	**0.0%**	1.8%	**22.2*****	**0.0%**	**0.4%*****	50.7
RC206_21x	1163.2	**0.0%**	0.9%	**15.8*****	**0.0%**	**0.1%****	51.5
RC207_21x	1083.2	**0.0%**	1.2%	**18.2*****	0.1%	**0.6%****	49.4
RC208_21x	1011.3	**0.0%**	1.9%	**17.8*****	**0.0%**	**1.1%***	58.5
Avg	-	0.4%	2.1%	**14.4**	**0.0%**	**1.0%**	35.7

5.6.4 Comparação das propostas com VNS_{full}

Essa subseção discute os resultados dos IGs propostos aqui quando comparados com os obtidos pelo VNSfull. As variantes do IG comparadas são: IG_{RVND} e $\text{IG}^{MILP}_{\text{RVND}}$. As demais variantes não são comparadas com o VNS, pois o propósito delas era demonstrar a contribuição de cada componente (alteração do construtivo na fase de reconstrução e o RVND). Além disso, essas variantes do IG alcançaram resultados piores do que aqueles encontrados pelo IG_{RVND}.

A Tabela 14 apresenta os resultados obtidos pelos métodos: VNS_{full}, IG_{RVND}, e $\text{IG}^{\text{MILP}}_{\text{RVND}}$ com 10 execuções independentes para os grupos C5, C10, e C15. Os métodos IGs propostos, com relação a distância, são competitivos com o VNS_{full} para as instâncias pequenas. Além disso, os IGs obtém menor tempo de processamento.

Para o grupo C5, as médias de Gap-Best obtidas são: 0.0% (VNS_{full}), 0.1% (IG_{RVND}), e 0.1% ($\text{IG}^{\text{MILP}}_{\text{RVND}}$). Em relação à Gap-Avg as médias obtidas são: 0.1% (VNS_{full}), 0.9% (IG_{RVND}), e 0.4% ($\text{IG}^{\text{MILP}}_{\text{RVND}}$). Os métodos IGs são 27 vezes mais rápidos do que o VNS_{full}. O resultado dos testes estatísticos (teste de Wilcoxon signed-rank por coluna) não apontam diferença estatisticamente significativa entre os resultados de distância e tempo entre o VNS_{full} e os IGs.

Para o grupo C10, as médias de Gap-Best obtidas são: 0.0% (VNS_{full}), 0.1% (IG_{RVND}), e 0.3% ($\text{IG}^{\text{MILP}}_{\text{RVND}}$). Em relação à Gap-Avg as médias obtidas são: 0.4% (VNS_{full}), 0.6% (IG_{RVND}), e 0.3% ($\text{IG}^{\text{MILP}}_{\text{RVND}}$). Em relação ao tempo de processamento, o IG_{RVND} e $\text{IG}^{\text{MILP}}_{\text{RVND}}$ são respectivamente, 6 e 4 vezes mais rápidos do que o VNS_{full}. Os resultados dos testes estatísticos são que não existe diferença estatisticamente significativa entre os resultados de distância entre o VNS_{full} e os IGs. Em relação ao tempo de processamento, existe uma diferença estatisticamente significativa entre o VNS_{full} e o IG_{RVND}. A magnitude para esse teste é grande. Com relação ao $\text{IG}^{\text{MILP}}_{\text{RVND}}$, o resultado do teste é que não existe diferença estatisticamente significativa entre o VNSfull e o $\text{IG}_{\text{RVND}^{\text{MILP}}}$.

Para o grupo C15, as médias de Gap-Best obtidas são: 0.4% (VNS_{full}), 0.5% (IG_{RVND}), e 0.0% ($\text{IG}^{\text{MILP}}_{\text{RVND}}$). Em relação à Gap-Avg as médias obtidas são: 1.2% (VNS_{full}), 1.2% (IG_{RVND}), e 0.6% ($\text{IG}^{\text{MILP}}_{\text{RVND}}$). Em relação ao tempo de processamento, o IG_{RVND} e $\text{IG}^{\text{MILP}}_{\text{RVND}}$ são respectivamente, 10 e 6 vezes mais rápidos do que o VNS_{full}. Os resultados dos testes estatísticos indicam não existir diferença estatisticamente significativa entre os resultados de distância entre o VNS_{full} e os IGs. Em relação ao tempo de processamento, existe uma diferença estatisticamente significativa entre o VNS_{full} e os IGs. A magnitude do teste é grande.

A Tabela 15 mostra os resultados para as instâncias grandes, com 100 clientes, dos grupos C100 e R100. Para ambos os grupos, as duas abordagens propostas conseguiram melhores resultados do que o VNS_{full}, em relação a distância e ao tempo de processamento.

Para o grupo C100, o IG_{RVND} as médias de Gap-Best obtidas são: 1.6%(VNS_{full}), 0.8% (IG_{RVND}), e 0.0% ($\text{IG}^{\text{MILP}}_{\text{RVND}}$). Em relação à Gap-Avg, as médias obtidas são: 3.9% (VNS_{full}), 2.2% (IG_{RVND}), e 0.6% ($\text{IG}^{\text{MILP}}_{\text{RVND}}$). Em relação ao tempo de processamento, os métodos IG_{RVND} e $\text{IG}^{\text{MILP}}_{\text{RVND}}$ são, respectivamente, 10 e 4 vezes mais rápidos do que VNS_{full}. Existe diferença estatisticamente significativa, para a distância *best*, distância média, e tempo de processamento, entre o VNS_{full} e os IGs. A magnitude do teste é grande.

Para o grupo R100, o IG_{RVND} as médias de Gap-Best obtidas são: 6.6%(VNS_{full}), 0.5% (IG_{RVND}), e 0.0% ($\text{IG}^{\text{MILP}}_{\text{RVND}}$). Em relação à Gap-Avg, as médias obtidas são: 11.9% (VNS_{full}), 1.7% (IG_{RVND}), e 1.0% ($\text{IG}^{\text{MILP}}_{\text{RVND}}$). O Gap-Avg obtido pelo VNS_{full} de 35%, para a instância R101_21x, é o maior valor de gap obtido por um método, considerando todos os testes. Em relação ao tempo de processamento, os métodos IG_{RVND} e $\text{IG}^{\text{MILP}}_{\text{RVND}}$ são, respectivamente, 11 e 4 vezes mais rápidos do que VNS_{full}. Existe diferença estatisticamente significativa, para a distância *best*, distância média, e tempo de processamento, entre o VNS_{full} e os IGs. A magnitude do teste é grande.

A Tabela 16 apresenta os resultados do grupo RC100. Os métodos IGs propostos obtêm melhores resultados para quase todas as instâncias, considerando Gap-Best, Gap-Avg, e tempo de CPU, quando comparados com VNS_{full}. As médias de Gap-Best obtidas são: 2.4% (VNS_{full}), e 0.1 para o IG_{RVND} e o $\text{IG}^{\text{MILP}}_{\text{RVND}}$. Em relação à Gap-Avg, as médias obtidas são: 6.8% (VNS_{full}), 2.2 (IG_{RVND}), e 1.0% ($\text{IG}^{\text{MILP}}_{\text{RVND}}$). O tempo de CPU para os métodos propostos, IG_{RVND}, e $\text{IG}^{\text{MILP}}_{\text{RVND}}$ são, respectivamente, 14 e 5 vezes mais rápidos do que o VNS_{full}. Existe uma diferença estatisticamente significativa entre os dois métodos IGs e VNS_{full} para a média da Dist-Best, Dist-Avg, e tempo de CPU para o grupo RC100 de instâncias. Essas diferenças possuem uma magnitude grande.

A análise dos resultados comparando os IGs propostos com o VNS_{full} permite concluir que o IG_{RVND} e $\text{IG}^{\text{MILP}}_{\text{RVND}}$ conseguem ser melhores em termos de distância e tempo de processamento, quando comparados com o VNS_{full} para as instâncias maiores. Já para as menores, os resultados dos IGs são próximos aos obtidos pelo VNS_{full} para os grupos C5, e C10. Para o grupo C15, ambos os métodos propostos conseguem resultados melhores do que o VNS_{full}.

Tabela 14 – Resultados para instâncias pequenas (10 execuções). Teste de Wilcoxon signed-rank (nível de significância de 0.05) entre as colunas (por grupos) entre VNS_{full} e IG e VNS_{full} com IG_MILP. Diferenças não estatisticamente significativa não são marcadas. Resultados com diferença estatisticamente significativa tem a magnitude: pequena (*), média (**), ou grande (***). Tempos de CPU (colunas TC e T) estão em segundos. (+) Resultados iguais ao ótimo obtido pelo CPLEX[1].

Instância	Best	VNS_{full}[1]			IG			IG_MILP		
		Gap Best	Gap Avg	TC	Gap Best	Gap Avg	T	Gap Best	Gap Avg	T
C101_C5x	385.5^{+}	**0.0%**	**0.0%**	5.0	**0.0%**	**0.0%**	**0.1**	**0.0%**	**0.0%**	**0.1**
C103_C5x	341.3^{+}	**0.0%**	**0.0%**	0.2	**0.0%**	**0.0%**	**0.1**	**0.0%**	**0.0%**	**0.1**
C206_C5x	417.3^{+}	**0.0%**	**0.0%**	**0.0**	**0.0%**	**0.0%**	0.1	**0.0%**	**0.0%**	0.1
C208_C5x	381.9^{+}	**0.0%**	**0.0%**	**0.0**	**0.0%**	**0.0%**	0.1	**0.0%**	**0.0%**	0.2
R104_C5x	317.0^{+}	**0.0%**	**0.0%**	**0.0**	**0.0%**	**0.0%**	0.1	**0.0%**	**0.0%**	0.2
R105_C5x	453.7^{+}	**0.0%**	**0.0%**	11.9	**0.0%**	**0.0%**	**0.2**	**0.0%**	**0.0%**	**0.2**
R202_C5x	347.8^{+}	**0.0%**	**0.0%**	**0.0**	**0.0%**	**0.0%**	0.1	**0.0%**	**0.0%**	0.1
R203_C5x	371.3^{+}	**0.0%**	**0.0%**	2.9	**0.0%**	**0.0%**	**0.1**	**0.0%**	**0.0%**	**0.1**
RC105_C5x	432.6^{+}	**0.0%**	1.1%	8.6	**0.0%**	**0.0%**	**0.1**	**0.0%**	**0.0%**	**0.1**
RC108_C5x	460.9^{+}	**0.0%**	**0.0%**	1.3	**0.0%**	9.0%	**0.1**	**0.0%**	4.3%	**0.1**
RC204_C5x	332.9^{+}	**0.0%**	**0.0%**	**0.0**	0.6%	0.6%	0.1	0.6%	0.6%	0.1
RC208_C5x	327.3^{+}	**0.0%**	**0.0%**	6.1	**0.0%**	**0.0%**	**0.1**	**0.0%**	**0.0%**	0.2
Avg	-	**0.0%**	**0.1%**	2.7	0.1%	0.9%	**0.1**	0.1%	0.4%	**0.1**
C101_C10x	538.3^{+}	**0.0%**	**0.0%**	5.3	0.3%	0.3%	**0.6**	**0.0%**	0.1%	0.8
C104_C10x	484.3^{+}	**0.0%**	**0.0%**	2.8	**0.0%**	**0.0%**	**0.5**	**0.0%**	**0.0%**	0.7
C202_C10x	425.5^{+}	**0.0%**	**0.0%**	0.8	**0.0%**	1.2%	**0.4**	**0.0%**	**0.0%**	0.6
C205_C10x	415.5^{+}	**0.0%**	**0.0%**	**0.4**	0.2%	0.2%	0.5	**0.0%**	**0.0%**	0.6
R102_C10x	505.5^{+}	**0.0%**	3.8%	9.0	**0.0%**	**0.0%**	**0.5**	**0.0%**	**0.0%**	**0.5**
R103_C10x	436.1	**0.0%**	0.3%	1.4	**0.0%**	**0.0%**	**0.4**	**0.0%**	**0.0%**	0.5
R201_C10x	460.7^{+}	**0.0%**	**0.0%**	6.6	**0.0%**	1.0%	**0.6**	**0.0%**	**0.0%**	1.0
R203_C10x	436.5^{+}	**0.0%**	**0.0%**	**0.0**	0.1%	2.2%	0.8	2.4%	2.4%	1.0
RC102_C10x	618.8	**0.0%**	**0.0%**	4.4	1.1%	1.1%	**0.5**	1.1%	1.1%	0.6
RC108_C10x	559.9	**0.0%**	**0.0%**	**0.0**	**0.0%**	**0.0%**	0.3	**0.0%**	**0.0%**	0.4
RC201_C10x	495.5^{+}	**0.0%**	**0.0%**	1.0	**0.0%**	0.1%	**0.9**	**0.0%**	**0.0%**	1.2
RC205_C10x	576.2^{+}	**0.0%**	**0.0%**	6.2	**0.0%**	0.1%	**0.8**	**0.0%**	0.1%	1.0
Avg	-	**0.0%**	0.4%	2.9	0.1%	0.6%	**0.5*****	0.3%	**0.3%**	0.7
C103_C15x	575.2	**0.0%**	**0.0%**	**0.2**	0.7%	1.2%	1.4	**0.0%**	1.4%	3.3
C106_C15x	516.6	**0.0%**	**0.0%**	**0.4**	**0.0%**	0.3%	1.3	**0.0%**	**0.0%**	2.6
C202_C15x	550.3	**0.0%**	**0.0%**	5.0	0.9%	2.4%	**1.0**	**0.0%**	0.8%	2.1
C208_C15x	550.0	**0.0%**	**0.0%**	8.8	**0.0%**	2.3%	**1.1**	**0.0%**	0.2%	1.8
R102_C15x	698.7	2.6%	2.6%	4.8	1.1%	2.1%	**0.6**	**0.0%**	**0.5%**	**0.6**
R105_C15x	608.0	**0.0%**	**0.0%**	10.2	**0.0%**	0.4%	**0.1**	**0.0%**	0.1%	0.2
R202_C15x	593.7	**0.0%**	**0.0%**	24.4	**0.0%**	**0.0%**	**0.4**	**0.0%**	**0.0%**	0.6
R209_C15x	475.1	**0.0%**	1.5%	31.0	**0.0%**	**0.8%**	**0.5**	**0.0%**	1.2%	0.9
RC103_C15x	607.4	1.5%	**1.5%**	**0.7**	1.5%	1.8%	**0.7**	**0.0%**	**1.5%**	1.0
RC108_C15x	603.9	**0.0%**	1.9%	**0.1**	0.9%	1.8%	0.7	**0.0%**	**0.3%**	0.9
RC202_C15x	552.7	**0.0%**	6.2%	4.6	0.2%	0.2%	**0.9**	**0.0%**	**0.1%**	2.0
RC204_C15x	485.3	**0.0%**	**0.0%**	6.2	1.2%	2.0%	**0.5**	**0.0%**	1.4%	0.9
Avg	-	0.4%	1.2%	8.2	0.5%	1.2%	**0.8*****	**0.0%**	**0.6%**	1.4***

[1] Akbay et al. (2022)

Tabela 15 – Resultados para instâncias grandes (10 execuções). Teste de Wilcoxon signed-rank (nível de significância de 0.05) entre as colunas (por grupos) entre VNS_{full} e IG e VNS_{full} com IG_MILP. Resultados com diferença estatisticamente significativa tem a magnitude: pequena (*), média (**), ou grande (***). Tempos de CPU (colunas TC e T) estão em segundos.

		VNS_{full}[1]			IG			IG_MILP		
Instâncias	Best	Gap Best	Gap Avg	TC	Gap Best	Gap Avg	T	Gap Best	Gap Avg	T
C101_21x	1470.4	1.6%	4.6%	232.0	1.4%	3.5%	**16.7**	**0.0%**	**1.2%**	32.7
C102_21x	1437.2	0.7%	3.5%	228.8	1.1%	2.8%	**22.5**	**0.0%**	**1.1%**	35.7
C103_21x	1398.7	**0.0%**	1.9%	262.6	**0.0%**	1.4%	**26.0**	**0.0%**	**0.2%**	54.6
C104_21x	1383.5	1.2%	4.1%	216.4	1.7%	2.6%	**32.4**	**0.0%**	**0.8%**	76.0
C105_21x	1448.5	3.1%	5.0%	186.4	0.7%	3.0%	**39.9**	**0.0%**	**0.6%**	76.0
C106_21x	1426.9	0.2%	3.5%	214.8	0.6%	2.1%	**31.6**	**0.0%**	**0.6%**	61.2
C107_21x	1443.0	3.0%	4.9%	233.0	0.2%	2.6%	**34.3**	**0.0%**	**0.4%**	74.0
C108_21x	1413.3	2.7%	5.4%	209.6	0.7%	2.2%	**29.5**	**0.0%**	**0.6%**	62.5
C109_21x	1400.3	0.7%	4.0%	269.5	0.2%	1.7%	**31.8**	**0.0%**	**0.3%**	69.5
C201_21x	1199.5	0.8%	2.9%	218.0	0.5%	2.1%	**14.6**	**0.0%**	**0.3%**	46.2
C202_21x	1177.6	0.9%	4.7%	281.2	0.8%	2.3%	**17.8**	**0.0%**	**0.6%**	53.3
C203_21x	1157.1	3.8%	5.1%	307.0	1.6%	2.1%	**19.2**	**0.0%**	**0.6%**	59.1
C204_21x	1129.3	2.8%	4.6%	231.2	1.1%	2.4%	**18.6**	**0.0%**	**0.5%**	54.9
C205_21x	1186.6	1.6%	3.1%	265.8	0.9%	1.7%	**18.8**	**0.0%**	**0.7%**	53.4
C206_21x	1170.4	1.0%	2.4%	222.4	0.7%	1.7%	**18.3**	**0.0%**	**0.4%**	57.9
C207_21x	1155.4	1.6%	2.9%	225.2	1.0%	1.6%	**18.9**	**0.0%**	**0.5%**	58.4
C208_21x	1156.8	1.1%	2.8%	261.2	0.2%	1.2%	**17.0**	**0.0%**	**0.3%**	50.2
Avg	-	1.6%	3.9%	237.7	0.8%***	2.2%***	**24.4*****	**0.0%*****	**0.6%*****	57.8***
R101_21x	1706.9	27.7%	35.2%	235.9	0.2%	2.6%	**16.2**	**0.0%**	**0.5%**	24.2
R102_21x	1575.0	17.0%	28.6%	120.0	0.1%	1.2%	**16.8**	**0.0%**	**0.4%**	23.1
R103_21x	1460.5	18.4%	25.3%	140.3	1.5%	3.0%	**13.4**	**0.0%**	**2.3%**	19.3
R104_21x	1401.2	4.9%	16.2%	214.1	1.6%	3.6%	**11.0**	**0.0%**	**0.5%**	21.6
R105_21x	1585.7	20.4%	24.6%	185.4	0.4%	**1.2%**	**13.4**	**0.0%**	1.3%	19.4
R106_21x	1507.5	14.4%	25.2%	61.3	0.6%	1.4%	**14.2**	**0.0%**	**0.5%**	18.2
R107_21x	1428.0	4.3%	16.9%	129.4	0.9%	2.1%	**13.8**	**0.0%**	**1.5%**	27.0
R108_21x	1409.2	2.8%	11.4%	73.7	**0.0%**	1.2%	**12.7**	**0.0%**	**0.9%**	18.4
R109_21x	1462.3	4.6%	15.1%	154.5	0.9%	2.6%	**11.4**	**0.0%**	**1.3%**	22.5
R110_21x	1410.8	4.2%	7.3%	264.3	1.0%	4.5%	**9.7**	**0.0%**	**2.8%**	30.0
R111_21x	1418.6	7.3%	12.3%	193.3	1.4%	**3.7%**	**8.8**	**0.0%**	4.6%	26.3
R112_21x	1373.7	2.9%	5.8%	25.8	1.1%	2.1%	**8.1**	**0.0%**	**1.0%**	22.2
R201_21x	1198.4	1.7%	4.5%	255.9	**0.0%**	**0.4%**	**17.7**	0.3%	0.6%	49.9
R202_21x	1121.4	1.3%	4.0%	225.6	0.3%	1.5%	**13.0**	**0.0%**	**0.7%**	64.3
R203_21x	1042.6	2.4%	5.2%	217.0	0.2%	1.3%	**28.6**	**0.0%**	**0.4%**	84.3
R204_21x	961.4	0.4%	1.6%	236.4	0.1%	0.2%	**15.8**	**0.0%**	**0.1%**	70.8
R205_21x	1103.1	2.8%	4.7%	180.8	0.1%	0.8%	**21.0**	**0.0%**	**0.6%**	64.8
R206_21x	1074.7	1.7%	4.0%	205.1	0.1%	1.2%	**23.8**	**0.0%**	**0.7%**	76.2
R207_21x	1011.9	1.3%	4.3%	205.2	0.2%	0.9%	**18.6**	**0.0%**	**0.4%**	82.1
R208_21x	961.0	1.0%	3.7%	207.6	**0.0%**	**0.3%**	**17.1**	**0.0%**	0.4%	80.5
R209_21x	1056.4	2.0%	3.1%	239.1	0.1%	0.7%	**26.1**	**0.0%**	**0.4%**	84.2
R210_21x	1035.2	1.0%	3.2%	192.2	**0.0%**	0.7%	**25.5**	**0.0%**	**0.6%**	80.7
R211_21x	994.7	0.5%	4.0%	168.8	**0.0%**	0.9%	**16.9**	**0.0%**	**0.4%**	61.4
Avg	-	6.6%	11.9%	180.1	0.5%***	1.7%***	**16.2*****	**0.0%*****	**1.0%*****	45.9***

[1] Akbay et al. (2022)

Tabela 16 – Resultados para instâncias grandes (10 execuções). Teste de Wilcoxon signed-rank (nível de significância de 0.05) entre as colunas (por grupos) entre VNS_{full} e IG e VNS_{full} com IG_MILP. Resultados com diferença estatisticamente significativa tem a magnitude: pequena (*), média (**), ou grande (***). Tempos de CPU (colunas TC e T) estão em segundos.

		VNS_{full}[1]			IG			IG_MILP		
Instância	Best	Gap Best	Gap Avg	TC	Gap Best	Gap Avg	T	Gap Best	Gap Avg	T
RC101_21x	1807.0	5.6%	16.6%	242.1	1.5%	4.1%	**9.4**	**0.0%**	**1.8%**	24.1
RC102_21x	1763.1	4.1%	16.1%	158.8	1.9%	4.2%	**9.4**	**0.0%**	**1.5%**	29.7
RC103_21x	1701.5	1.6%	8.5%	188.1	0.9%	2.1%	**12.2**	**0.0%**	**1.0%**	22.1
RC104_21x	1620.8	1.5%	4.2%	149.2	1.3%	2.8%	**16.4**	**0.0%**	**1.4%**	20.1
RC105_21x	1773.1	1.7%	9.2%	202.7	0.5%	3.1%	**7.3**	**0.0%**	**1.6%**	25.6
RC106_21x	1716.8	2.0%	5.3%	180.2	1.1%	2.9%	**7.0**	**0.0%**	**1.2%**	20.7
RC107_21x	1665.9	1.3%	3.2%	272.5	1.3%	2.3%	**11.6**	**0.0%**	**1.1%**	13.6
RC108_21x	1622.8	**0.0%**	**2.0%**	304.3	1.4%	3.2%	**12.0**	0.9%	**2.0%**	13.2
RC201_21x	1289.0	2.3%	5.4%	113.0	0.3%	**0.4%**	**17.4**	**0.0%**	**0.4%**	46.9
RC202_21x	1191.0	0.8%	3.4%	212.8	**0.0%**	**0.2%**	**18.3**	**0.0%**	0.4%	51.4
RC203_21x	1072.4	2.9%	6.2%	249.9	**0.0%**	0.6%	**17.8**	**0.0%**	**0.2%**	50.8
RC204_21x	1002.1	3.8%	5.3%	188.1	1.4%	2.4%	**20.2**	**0.0%**	**1.3%**	75.5
RC205_21x	1173.4	3.8%	6.1%	142.7	0.3%	2.2%	**22.6**	**0.0%**	**0.3%**	49.5
RC206_21x	1163.2	2.6%	4.6%	244.3	**0.0%**	1.1%	**15.1**	**0.0%**	**0.0%**	52.7
RC207_21x	1083.2	2.2%	5.8%	176.8	**0.0%**	1.3%	**17.3**	0.1%	**0.5%**	47.1
RC208_21x	1011.3	3.8%	5.6%	206.7	**0.0%**	2.1%	**16.5**	**0.0%**	**1.3%**	60.1
Avg	-	2.4%	6.8%	201.7	0.8%***	2.2%***	**14.3*****	**0.1%*****	**1.0%*****	36.2***

[1] Akbay et al. (2022)

6 Conclusões e Trabalhos Futuros

O 2E-ERVP-TW consiste em atender demandas de clientes por meio de satélites (depósitos intermediários) e então EVs partem dos satélites para atender os clientes. Esse problema lida com desafios na distribuição de bens, como *e-commerce*, o qual é a coordenação de entregas usando dois níveis de roteamento. O processo de solução do 2E-EVRP-TW pode ser difícil devido às suas restrições operacionais, como a capacidade dos veículos, janela de tempo, e capacidade e recarga da bateria.

O presente trabalho propôs duas abordagens baseadas em *Iterated Greedy* (IG). O IG, com a notação: IG_{RVND}, proposto possui características diferentes de algoritmos propostos para o mesmo problema, por exemplo: utiliza somente componentes de destruição e reconstrução, a fase de reparo considera múltiplos candidatos, e a seleção do próximo candidato é aleatória na lista restrita de candidatos.

A segunda abordagem consiste em uma *matheuristic* que utiliza o IG_{RVND} proposto inicialmente com um particionamento de rotas. Esse método é designado aqui como $\text{IG}^{MILP}_{\text{RVND}}$, e não foi encontrado um algoritmo semelhante na literatura para o problema alvo deste trabalho. O $\text{IG}^{MILP}_{\text{RVND}}$ proposto utiliza as rotas EVs geradas pelo método IG_{RVND} para criar um conjunto de rotas e um problema de particionamento, formulado como um MILP, que seleciona um subconjunto de rotas que atendem todos os clientes. A formulação proposta também determina as rotas dos CVs do primeiro nível enquanto minimiza a distância total percorrida.

Experimentos computacionais foram realizados para comparar os dois métodos IGs com o algoritmo estado da arte. Também foram testados algumas variações do IG, para estudar a contribuição de cada componente adicionado ao método IG. Os experimentos foram divididos em quatro etapas. A primeira compara a influência do parâmetro α, considerando o valor igual a zero e o definido pelo *irace*. Esse experimento evidencia que a escolha aleatória do próximo candidato na fase de reconstrução gera melhores resultados do que a versão gulosa.

O segundo experimento compara a influência da busca local RVND no método proposto. Para isso foram comparados o IG_{RVND} com o IG que não utiliza o RVND. Esse experimento evidencia que o acréscimo do RVND tem um impacto muito significativo nos resultados.

O terceiro experimento compara a influência do particionamento de rotas no método proposto. Para isso, foram comparados o IG_{RVND} com o $\text{IG}^{MILP}_{\text{RVND}}$. Essa etapa evidencia a contribuição do particionamento com MILP.

O quarto experimento compara os dois métodos IGs propostos, IG_{RVND} e $\text{IG}^{MILP}_{\text{RVND}}$, com o algoritmo estado da arte. Os resultados do teste estatístico mostram que os dois métodos IGs obtiveram melhores resultados quando comparados com o algoritmo estado

da arte para as instâncias maiores. Os resultados do teste estatístico mostraram que os métodos propostos têm uma diferença estatística significativa para a maioria dos grupos de instâncias grandes quando comparado com o estado da arte. Além dos resultados da função objetivo, os dois IGs propostos conseguiram melhores tempos de processamento do que o estado da arte.

As instâncias, com 100 clientes, adaptadas em (1) possuem a seguinte limitação: todos os satélites têm as mesmas posições para todas as instâncias. Em trabalhos futuros, pretende-se criar instâncias em que as posições dos satélites sejam distintos entre as instâncias. Outra limitação está relacionado a quantidade fixa de satélites, para isso, pretende-se estudar como a solução muda conforme a alteração das posições e o número de satélites. Uma limitação do modelo de particionamento está relacionada a não utilização do modelo de divisão de carga. Em trabalhos futuros pretende-se adaptar o modelo para permitir a divisão de carga. Um experimento interessante é executar os testes do $\mathrm{IG}_{\mathrm{RVND}}$ com o mesmo tempo de processamento do $\mathrm{IG}^{\mathrm{MILP}}_{\mathrm{RVND}}$, para verificar se o aumento do tempo de processamento faz com que o $\mathrm{IG}_{\mathrm{RVND}}$ consiga resultados melhores.

REFERÊNCIAS

1 M. A. Akbay, C. B. Kalayci, C. Blum, and O. Polat. Variable neighborhood search for the two-echelon electric vehicle routing problem with time windows. *Applied Sciences*, 12(3):1014, 2022.

2 Y. Amarouche, R. N. Guibadj, and A. Moukrim. A neighborhood search and set cover hybrid heuristic for the two-echelon vehicle routing problem. In *18th Workshop on Algorithmic Approaches for Transportation Modelling, Optimization, and Systems (ATMOS 2018)*, pages 1–11. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2018.

3 C. Archetti and M. Speranza. A survey on matheuristics for routing problems. *EURO Journal on Computational Optimization*, 2(4):223–246, 2014.

4 A. Binstock and J. Rex. *Practical algorithms for programmers*. Addison-Wesley Longman Publishing Co., Inc., 1995.

5 U. Breunig, R. Baldacci, R. F. Hartl, and T. Vidal. The electric two-echelon vehicle routing problem. *Computers & Operations Research*, 103:198–210, 2019.

6 G. Clarke and J. W. Wright. Scheduling of vehicles from a central depot to a number of delivery points. *Operations research*, 12(4):568–581, 1964.

7 J. Cohen. *Statistical Power Analysis for the Behavioral Sciences*. Lawrence Erlbaum Associates, 1988.

8 G. B. Dantzig and J. H. Ramser. The truck dispatching problem. *Management science*, 6(1):80–91, 1959.

9 M. Dror and P. Trudeau. Split delivery routing. *Naval Research Logistics (NRL)*, 37 (3):383–402, 1990.

10 D. Dumez, C. Tilk, S. Irnich, F. Lehuédé, K. Olkis, and O. Péton. A matheuristic for a 2-echelon vehicle routing problem with capacitated satellites and reverse flows. *European Journal of Operational Research*, 305(1):64–84, 2023.

11 M. Fischetti and M. Fischetti. *Matheuristics*, pages 121–153. Springer International Publishing, Cham, 2018. ISBN 978-3-319-07124-4.

12 B. Golden, A. Assad, L. Levy, and F. Gheysens. The fleet size and mix vehicle routing problem. *Computers & Operations Research*, 11(1):49–66, 1984.

13 Gurobi Optimization, LLC. Gurobi Optimizer Reference Manual, 2023.

14 S. Irnich and G. Desaulniers. *Shortest Path Problems with Resource Constraints*, pages 33–65. Springer US, Boston, MA, 2005. ISBN 978-0-387-25486-9.

15 W. Jie, J. Yang, M. Zhang, and Y. Huang. The two-echelon capacitated electric vehicle routing problem with battery swapping stations: Formulation and efficient methodology. *European Journal of Operational Research*, 272(3):879–904, 2019.

16 G. Kalghatgi, H. Levinsky, and M. Colket. Future transportation fuels. *Progress in Energy and Combustion Science*, 69:103–105, 2018.

17 B. Kallehauge, J. Larsen, O. B. Madsen, and M. M. Solomon. *Vehicle Routing Problem with Time Windows*, pages 67–98. Springer US, Boston, MA, 2005.

18 R. Kulkarni and P. Bhave. Integer programming formulations of vehicle routing problems. *European Journal of Operational Research*, 20(1):58–67, 1985.

19 M. López-Ibáñez, J. Dubois-Lacoste, L. P. Cáceres, M. Birattari, and T. Stützle. The irace package: Iterated racing for automatic algorithm configuration. *Operations Research Perspectives*, 3:43–58, 2016.

20 R. Martí, P. M. Pardalos, and M. G. C. Resende, editors. *Handbook of Heuristics*. Springer, 2018. ISBN 978-3-319-07123-7.

21 C. E. Miller, A. W. Tucker, and R. A. Zemlin. Integer programming formulation of traveling salesman problems. *Journal of the ACM (JACM)*, 7(4):326–329, 1960.

22 M. G. C. Resende and C. C. Ribeiro. *Greedy Randomized Adaptive Search Procedures*, pages 219–249. Springer US, Boston, MA, 2003. ISBN 978-0-306-48056-0.

23 M. Schneider, A. Stenger, and D. Goeke. The electric vehicle-routing problem with time windows and recharging stations. *Transportation science*, 48(4):500–520, 2014.

24 N. Sluijk, A. M. Florio, J. Kinable, N. Dellaert, and T. Van Woensel. Two-echelon vehicle routing problems: A literature review. *European Journal of Operational Research*, 2022.

25 M. M. Solomon. Algorithms for the vehicle routing and scheduling problems with time window constraints. *Operations research*, 35(2):254–265, 1987.

26 M. J. Souza, I. M. Coelho, S. Ribas, H. G. Santos, and L. H. d. C. Merschmann. A hybrid heuristic algorithm for the open-pit-mining operational planning problem. *European Journal of Operational Research*, 207(2):1041–1051, 2010.

27 A. Subramanian. *Heuristic, Exact and Hybrid Approaches for Vehicle Routing Problems*. Phd thesis, Universidade Federal Fluminense, Niterói, RJ, March 2012.

28 A. Subramanian, L. M. Drummond, C. Bentes, L. S. Ochi, and R. Farias. A parallel heuristic for the vehicle routing problem with simultaneous pickup and delivery. *Computers & Operations Research*, 37(11):1899–1911, 2010.

29 E.-G. Talbi. *Metaheuristics: from design to implementation*. John Wiley & Sons, 2009.

30 D. Wang and H. Zhou. A two-echelon electric vehicle routing problem with time windows and battery swapping stations. *Applied Sciences*, 11(22), 2021. ISSN 2076-3417.

31 L. A. Wolsey. *Integer programming*. John Wiley & Sons, 2020.

32 M. A. Zamal, A. H. Schrotenboer, and T. V. Woensel. The two-echelon vehicle routing problem with pickups, deliveries, and deadlines. *SSRN*, 2023.