UNIVERSIDADE FEDERAL RURAL DE PERNAMBUCO
DEPARTAMENTO DE PÓS GRADUAÇÃO EM INFORMÁTICA APLICADA - PPGIA

GLEYDSON ALVES DE BRITO

Vehicle Priority-aware Dynamic Timeline Routing: Uma nova abordagem para resolução do problema do roteamento de veículos

DISSERTAÇÃO DE MESTRADO

Recife

2023

GLEYDSON ALVES DE BRITO

Vehicle Priority-aware Dynamic Timeline Routing: Uma nova abordagem para resolução do problema do roteamento de veículos

Dissertação apresentada ao Programa de Pós Graduação em Informática Aplicada, como parte dos requisitos necessários à obtenção do título de Mestre.

Orientador: Prof. Dr. Cícero Garrozi

Recife
2023

Dados Internacionais de Catalogação na Publicação
Universidade Federal Rural de Pernambuco
Sistema Integrado de Bibliotecas
Gerada automaticamente, mediante os dados fornecidos pelo(a) autor(a)

B862v Brito, Gleydson Alves de
Vehicle Priority-aware Dynamic Timeline Routing: Uma nova abordagem para resolução do problema do roteamento de veículos / Gleydson Alves de Brito. - 2023.
153 f. : il.

Orientador: Cicero Garrozi.
Inclui referências e apêndice(s).

Dissertação (Mestrado) - Universidade Federal Rural de Pernambuco, Programa de Pós-Graduação em Informática Aplicada, Recife, 2023.

1. VRP. 2. VRPTW. 3. NSGA-III. 4. Multiobjective Genetic Algorithm. 5. Genetic Algorithm. I. Garrozi, Cicero, orient. II. Título

CDD 004

Dedico este trabalho a minha família, sobretudo a minha esposa Elane França, quem me deu toda força, amor e coragem, iluminando meus caminhos para que eu pudesse chegar até aqui. Sem ela não me imagino indo tão longe. Dedico ainda a meus filhos, João Vinícius e Vicente França, que me incentivam e me motivam a ser melhor a cada dia.

Agradecimentos

Gostaria de agradecer ao meu orientador, Prof. Cícero Garrozi, por todo companheirismo, dedicação e conhecimento. Não é possível descrever aqui tudo que aprendi com ele ao longo desses anos. Sua atuação foi fundamental, não só tecnicamente, mas como pessoa humana, empática e tranquilizadora. Contei com toda sua compreensão nos momentos mais difíceis dessa trajetória, que não foram poucos.

Gostaria de agradecer mais uma vez a minha esposa, que tanto me ensinou e ensina todo o tempo. Agradecer por sua paciência e por ter sempre acreditado em mim, mesmo quando nem eu conseguia acreditar. Sua força é pedra fundamental na minha vida.

Gostaria de agradecer a todos os professores e coordenadores do PPGIA, programa que me possibilitou conhecimentos sem precedentes. Toda colaboração, e tudo que foi construído, consolidam meu alicerce acadêmico e científico.

Gostaria de agradecer aos amigos, que, independente da distância física, estão sempre comigo, na minha memória e no meu coração.

Gostaria de agradecer ao meu gestor, Prof. Severino Mendes de Azevedo Júnior, que com toda empatia me forneceu sempre a oportunidade de desempenhar minha pesquisa da melhor forma possível. Sua contribuição também foi fundamental.

Gostaria de agradecer a todos os amigos da Progesti que, a todo momento, torceram pelo meu sucesso, e também colaboraram com essa conquista.

Por fim, minha gratidão a todos que, exercendo democraticamente sua cidadania, extirparam do poder o verme inominável que aqui habitava e corroía a democracia. Que a democracia se fortaleça, e que o amor nunca se cale diante do ódio.

I
N
E
L
E
G - Í - V - E - L

“O sol há de brilhar mais uma vez. A luz há de chegar aos corações. Do mal, será queimada a semente. O amor será eterno novamente . . . ” (Nelson Cavaquinho)

“Presentemente, eu posso me
considerar um sujeito de sorte.
Porque apesar de muito moço,
me sinto são, e salvo, e forte.
E tenho comigo pensado,
Deus é Brasileiro e anda do meu lado.
E assim já não posso sofrer no ano passado.
Tenho sangrado demais,
tenho chorado pra cachorro.
Ano passado eu morri,
mas esse ano eu não morro. . . “ (Belchior)

Abstract

With the advancement and technological development, industries that operate under a business model reliant on the logistics of goods and raw materials encounter a substantial challenge in optimizing and mitigating their operational costs. Over time, changes in consumer habits have led to an increasing need for accessing products and services through digital platforms. The COVID-19 pandemic further exacerbated this scenario, impacting the market with a sudden surge in demand due to circulation restrictions. This sudden increase highlighted the challenges in logistics management and underscored the positive impacts of efficient logistical processes. Supply chain management encompasses two key variables that significantly influence operational costs: the total route distance traveled by the fleet and the fixed costs associated with fleet size. Optimizing these variables is a complex and extensively studied task that emerged from the research on the Vehicle Routing Problem (VRP). VRP assumes that optimizing these variables can substantially reduce operational costs by providing more efficient distribution of routes and fleets. Given the relevance of this field and the industry's needs, this study proposes a novel approach to address the Vehicle Routing Problem. This approach introduces a new concept termed "urgency of service" and incorporates the implementation of novel evolutionary operators and a multiobjective genetic algorithm within a framework that considers fleet timelines. The research validates that the developed approach efficiently resolves vehicle routing problems with time windows and vehicle capacity constraints. The Solomon benchmark was employed for validation purposes. The obtained results demonstrate that the proposed approach achieved competitive solutions in problems with restricted time windows, outperforming other solutions in some cases concerning fleet sizing. Moreover, in problems with more flexible time windows, the approach yielded significantly improved results compared to the benchmark, with reductions ranging from 16.67% to 66.67%. This showcases the approach's high capability in identifying efficient solutions that employ smaller fleets. Such reductions have an extremely positive impact, potentially leading to a drastic reduction in overall operational costs for smaller enterprises. Additionally, this cost reduction presents an opportunity for smaller companies to participate competitively in the market and improves efficiency for larger companies.

Keywords: VPR, VRPTW, vehicle route problem. time-window, Multi-Objective Genetic Algorithm, NSGA-III, Algoritmo genético, Programação multiobjetivo

Lista de ilustrações

Lista de tabelas

Lista de abreviaturas e siglas

CCMO	Coevolutionary Framework for Constrained Multiobjective Optimization Problems
COA	Chaos Optimization Algorithm
CVRP	Capacitated Vehicle Routing Problem
CVRPPDTW	Capacited Vehicle Route Problems Pickup and Delivery with Time Window
CVRPTW	Capacited Vehicle Route Problem with Time Window
EC	Evolutionary Computation
GA	Genetic Algorithm
MDVRP	Multi-Depot Vehicle Routing Problem
MOEA	Multi Objective Evolutionary Algorithms
NSGA I	Nondominated Sorting Genetic Algorithm I
NSGA II	Nondominated Sorting Genetic Algorithm II
NSGA III	Nondominated Sorting Genetic Algorithm III
PSO	Particle Swarm Optimization
SGA	Simple Genetic Algorithm
TSP	Traveling Salesman Problem
VEGA	vector-evaluated genetic algorithm
VND	Variable Neighborhood Descendent
VPDTR	Vehicle Priority-Aware Dynamic Timeline Routing
VRP	Vehicle Route Problem
VRPPD	Vehicle Routing Problem with Pickup and Delivery
VRPTWPD	Vehicle Route Problem with Time Window and Pickup and Delivery

Sumário

1 Introdução

1.1 A Covid-19 e seu impacto na gestão logística

A pandemia de Covid-19 (Sars-Cov-2) que se desencadeou a partir do ano de 2020 criou profundas modificações no mundo, promovendo impactos significativos nas relações pessoais, profissionais e econômicas. De março de 2020 a até meados de 2023, foram registrados cerca de 767 milhões de casos, com aproximadamente 6,95 milhões de mortes em todo o mundo[1]. Segundo o Ministério da Saúde, até o mês de junho de 2023, o Brasil registrava aproximadamente 37,6 milhões de casos com mais de 700 mil mortes[2].

O ano de 2020 pode ser considerado um dos períodos mais difíceis da pandemia. O vírus possuía uma alta capacidade de propagação e contágio, enquanto a capacidade para a produção de medicamentos e vacinas ainda era muito limitada. Rapidamente o vírus se espalha pelo mundo, e faz com que a Organização Mundial de Saúde decrete o estado pandêmico[3]. Com cada vez mais pessoas se contaminando, muitos países adotaram estratégias de prevenção de contágio, como o *lockdown*, para mitigar a disseminação do vírus. Nesse momento, a ideia era desacelerar o contágio e ganhar tempo para a produção de vacinas ou medicações eficientes contra o vírus. O *lockdown* consistia em manter o maior número de pessoas possível dentro de suas residências, permitindo que apenas os profissionais dos serviços essenciais se locomovessem até seus locais de trabalho.

Toda cadeia produtiva foi impactada, e grande parte dos setores econômicos necessitou adotar uma forma de trabalho que possibilitasse a continuidade das atividades, sem que houvesse o deslocamento das pessoas até os postos de trabalho. Rapidamente as empresas são obrigadas a migrar suas atividades presenciais para as plataformas digitais para permitir a continuidade dos seus negócios. O setor hoteleiro e turístico foi completamente impossibilitado de realizar suas atividades. Os bares e restaurantes, agora fechados, para evitar o encerramento das atividades, precisaram implementar os serviços no formato *delivery*. Com o amplo fechamento do comércio, os consumidores passaram a adotar as plataformas *on-line* para adquirir produtos. Tanto o setor alimentício quanto o setor comercial, agora, passam a depender fortemente dos serviços de entregas e coletas para poder viabilizar seus negócios.

Em momentos mais críticos do isolamento social, algumas cidades chegaram a registrar um aumento de 100% no transporte individual de mercadorias (SETTEY *et al.*, 2021). Com o aumento do fluxo de mercadorias, toda a cadeia logística é impactada. Com cada vez mais clientes para serem atendidos, os desafios para a gestão logística aumentam

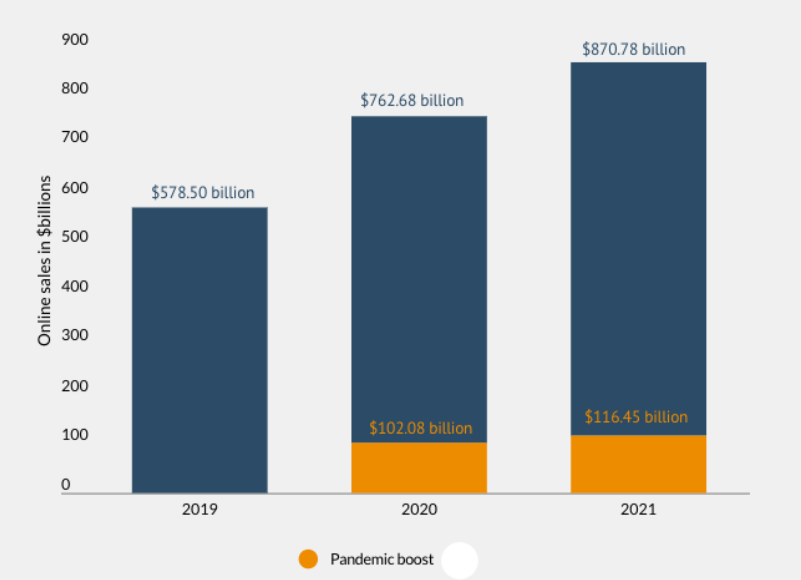
[1] https://ourworldindata.org/covid-cases

[2] https://www.gov.br/saude/pt-br/coronavirus/informes-diarios-covid-19/covid-19-situacao-epidemiologica-do-brasil-ate-a-se-25-de-2023

[3] https://www.who.int/director-general/speeches/detail/who-director-general-s-opening-remarks-at-the-media-briefing-on-covid-19---11-march-2020

e diante desse aumento é necessário que a forma de distribuição de produtos seja eficiente para que isso não impacte negativamente nos custos operacionais. Grande parte dos custos logísticos se concentram na manutenção da estocagem, transporte e gestão logísticas (EMAD, 2022), assim, observar esses pilares se torna parte essencial para obter resultados positivos. Segundo dados do Digital Commerce 360 2022[4], a pandemia adicionou mais de 200bi extras em vendas nos comércios eletrônicos.

Figura 1 – Pandemia adiciona US$ 218,53 bilhões extras às vendas de comércio eletrônico nos EUA em 2020-2021 (Tradução livre)



Digital Commerce 360 2022

Como a capacidade de estocagem e o tamanho da frota são elementos limitados dentro de uma cadeia de distribuição, a gestão logística precisa ser otimizada para que os custos operacionais possam ser reduzidos. É necessário que haja um fluxo eficiente de produtos, evitando exceder a capacidade de estocagem dos armazéns ou as condições de transporte. A correta distribuição dos produtos, em um tempo aceitável e de modo que as necessidades dos clientes sejam atendidas, é uma etapa crucial do processo logístico. Os produtos devem ser distribuídos entre os diversos veículos, priorizando o tempo de entrega limite e otimizando as rotas, de modo que os veículos possam atender todos os clientes dentro do tempo requerido, utilizando as rotas que minimizem as distâncias. Quanto mais eficiente o processo logístico, menor será o impacto nos custos operacionais do negócio.

1.2 O problema de pesquisa

Todo o cenário de gestão logística remete a um conhecido problema no campo da computação e da pesquisa operacional: o problema do roteamento de veículos. O problema

[4] https://www.digitalcommerce360.com/article/coronavirus-impact-online-retail/

do roteamento de veículos, ou do inglês *vehicle routing problem - VRP,* consiste na identificação da melhor distribuição de clientes, para uma determinada frota, que possa minimizar os custos operacionais otimizando as distâncias percorridas e o tempo gasto. Essa é a abordagem canônica do *VRP* e é a base para diversas outras variações existentes na literatura. Ao longo do tempo, diversas outras variáveis vêm sendo consideradas e adicionadas para adaptar a abordagem geral do *VRP* a problemas e necessidades específicas.

O *VRP* é um problema permutacional complexo, e dentro da classificação de problemas é considerado um problema da classe NP-difícil (KIM *et al.*, 2015; JAYARATHNA *et al.*, 2021; ESTRADA-MORENO *et al.*, 2019). Definir uma rota ótima para que apenas um veículo possa visitar vinte e cinco clientes, por exemplo, já nos dá um problema da ordem de $1,551121 * 10^{25}$ combinações possíveis. Assim, considerando um cenário simples, já é possível notar que identificar soluções otimizadas utilizando uma abordagem algorítmica de força bruta seria uma tarefa, no mínimo, extremamente difícil. Supondo que um computador pudesse testar um trilhão de permutações (10^{12}) por segundo. Testar todas as combinações possíveis, levaria $1,55 * 10^{13}$ segundos, o que equivale a aproximadamente 491 mil anos. Assim, é possível observar que não é factível a utilização de um algoritmo de força bruta para a resolução do *VRP*, ainda que de menor dimensão.

Dessa forma, é um grande desafio identificar soluções viáveis para o problema do roteamento de veículos, sobretudo para o Problema de Roteamento de Veículos que consideram Janelas de Tempo e Capacidade (CVRPTW), proposta abordada neste trabalho. Estabelecer estratégias que possam viabilizar essas soluções é um passo extremamente relevante para o desenvolvimento de serviços baseados em distribuições logísticas. Atualmente, existem muitas pesquisas relacionadas ao *VPR* que utilizam abordagens diversas, visando definir estratégias que facilitem a definição de rotas adequadas. Essas estratégias vão desde algoritmos exatos, até a utilização de heurísticas especializadas. Contudo, o campo de pesquisa ainda é vasto, e dada a complexidade e as variações dos tipos de problemas relacionados ao *VRP*, existe uma enorme necessidade de buscar cada vez mais abordagens que possam atender aos mais diversos desafios da área.

1.3 Justificativa

Considerando que o processo logístico para distribuição de materiais, insumos, alimentos, dentre outros, é uma atividade com impacto direto na economia, o desabastecimento de materiais e matérias-primas essenciais a indústrias e comércios pode atingir direta e indiretamente a vida das pessoas. É necessário que haja esforços no sentido de contribuir para a construção de soluções que possam impactar positivamente essa área do conhecimento. O aprimoramento dos conhecimentos sobre a gestão eficiente dos processos logísticos atinge tanto o setor público, quanto o setor privado, com extrema relevância para

a manutenção eficiente das redes de distribuição locais e até globais.

Com a migração de muitas atividades econômicas para o universo digital, cada vez mais, antigos e novos negócios tornam-se dependentes de sistemas de entregas de produto. As transformações digitais implementadas pelo varejo virtual e os novos hábitos de consumo vêm aumentando a cada ano. Somente em 2020, impulsionada pelo isolamento social e as restrições impostas devido à Covid-19, o Brasil registrou um aumento de 18% nas compras realizadas em comércios virtuais, movimentando cerca de 106 bilhões de reais (REZENDE *et al.*, 2020). Mesmo tendo que lidar com os desafios da captação de clientes dentro do universo digital, pequenos e médios negócios podem tirar vantagem da redução dos custos inerentes ao comércio tradicional, para implantar negócios totalmente on-line. Deltoro *et al.* (2012) argumenta que em muitos negócios a adoção da estratégia on-line pode ser bastante atrativa para pequenas e médias empresas, uma vez que, em diversas situações, os investimentos podem ser mais baixos que no comércio tradicional.

Todos os modelos de negócios baseados em comércio eletrônico têm como ponto em comum a necessidade de estabelecer uma cadeia de distribuição de produtos de forma eficiente. Como a rede de distribuição representa um fator preponderante nos custos operacionais, utilizar ferramentas que possam viabilizar um adequado processo de entrega é crucial para a sobrevivência dentro desse mercado. Sendo assim, é possível compreender que obter informações precisas sobre a adequada distribuição e roteamento da frota pode trazer um grande diferencial competitivo.

Por isso é tão importante perceber a importância dos estudos realizados no campo do roteamento de veículos. Por se tratarem de problemas de extrema complexidade, que ao mesmo tempo podem trazer impactos tão positivos nos mais diversos campos econômicos, a busca por abordagens cada vez mais eficientes pode contribuir diretamente para o desenvolvimento de negócios baseados na logística de distribuição. Ferramentas de otimização de rotas podem alavancar negócios extraindo maior eficiência das frotas, além de contribuírem diretamente para a redução dos custos operacionais, com um impacto direto nos lucros ou receitas das empresas.

1.4 Objetivos

1.4.1 Objetivo geral

Esse trabalho se propõe a definir uma nova abordagem de resolução do *VRP*. A abordagem denominada *Vehicle Priority-Aware Dynamic Timeline Routing - VPDTR* é uma heurística baseada em algoritmos genéticos multiobjetivos que implementa uma série de estruturas inovadoras para lidar com o processo de otimização de rotas em frotas homogêneas.

1.4.2 Objetivo específico

O objetivo específico deste trabalho é construir uma nova abordagem para resolver o problema do roteamento de veículos considerando janelas de tempo e capacidade. Um algoritmo robusto que capaz de lidar com as dificuldades do *CVRPTW* adaptando-se de forma eficiente às mais diversas disposições geográficas de clientes. Ao fim, essa abordagem identifica um conjunto de rotas ótimas, ou soluções que se aproximam das soluções ótimas (ótimas proximais), que minimizam a distância total percorrida pela frota e, simultaneamente, respeita as restrições do problema, os limites temporais de atendimento e capacidades dos veículos, demonstrando uma distribuição de veículos de forma eficiente que minimize os custos operacionais totais.

Para validar a abordagem proposta, utiliza-se o *benchmark* proposto por Solomon (1987). Esse *benchmark* consiste em um conjunto de problemas específicos para lidar com abordagens CVRPTW, sendo uma das referências mais utilizadas para validar esse tipo de abordagem.

Para demonstrar a eficiência da abordagem, os resultados serão apresentados e discutidos em capítulo específico e poderão apontar os pontos fortes, fracos e melhorias alcançadas para a heurística proposta neste trabalho.

1.4.3 Limitações do trabalho

Esse trabalho se limita a propor uma nova abordagem para a resolução do problema de roteamento de veículos considerando janelas temporais e capacidade (*CVRPTW*). Isso implica dizer que as demais variações não serão contempladas por essa abordagem. Cabe ainda salientar que a abordagem apresentada lida apenas com o *VRP* de frotas homogêneas e de depósito único. No *benchmark* utilizado, todos os veículos partem e retornam ao mesmo ponto fixo e possuem a mesma capacidade de transporte.

Nesta versão, os veículos simulam situações apenas de coleta ou entrega de itens aos clientes, mas não os dois simultaneamente. A versão que implementa entrega e coletas simultâneas é a variação *VRP pickup and delivery* (*VRPTWPD*).

2 Revisão da Literatura

2.1 Problema do Roteamento de Veículos (*Vehicle Route Problem - VRP*)

O problema do roteamento de veículos (*VRP*) é um conhecido problema da literatura e sua formalização foi definida pela primeira vez por George Dantzig e John Ramser no ano de 1959 em um artigo intitulado *The Truck Dispatching Problem* (DANTZIG; RAMSER, 1959). O *VRP* pode ser considerado uma generalização do problema do caixeiro viajante (T*raveling Salesman Problem - TSP*), e embora possuam similaridades, as características que o difere do *TSP* adicionam camadas extras de complexidade ao problema (KIM *et al.*, 2015). Podemos entender o *TSP*, de forma simplificada, utilizando a seguinte metáfora: considerando um conjunto de cidades, as quais o caixeiro precisa visitar, e supondo que para quaisquer pares de cidades possa haver uma conexão entre elas, o objetivo do problema é encontrar a rota com menor distância total que, obrigatoriamente, visite cada uma das cidades apenas uma vez, e retorne ao ponto de partida.

O problema do roteamento de veículos possui semelhanças com o *TSP* e como ele, também é um problema combinatório. Diferente do *TSP*, no *VRP* todos os veículos partem de um mesmo ponto inicial, denominado depósito. A partir do depósito, precisam visitar um conjunto de pontos, usualmente denominados clientes, visitando cada ponto apenas uma vez, retornando, ao final, ao seu ponto de partida. De forma ampla, o problema de roteamento consiste na identificação de um conjunto de rotas que minimizem uma determinada função de custo em uma rede de transporte. Dessa forma, seja $G = \{V, A\}$ um grafo composto por um conjunto de vértices $V = \{v_1, v_2, v_3, \ldots, v_n\}$, interligados por arestas $A = \{ a_{ij} : i, j \in V, i \neq j\}$. O conjunto R define uma rota, tal que $R = \{a_{12}(v_1, v_2), a_{23}(v_2, v_3), \ldots, a_{mk}(v_m, v_k)\}$. O conjunto $S = \{ R_1, R_2, \ldots, R_n \}$ é considerado uma solução para o problema quando contém um conjunto de rotas R, capaz de visitar todos os vértices de V sem repetição. De forma mais objetiva, o *VRP* consiste na otimização de um conjunto de rotas para uma frota de veículos visando a minimização dos custos de operação.

O objetivo do *VRP* é distribuir determinado conjunto de clientes entre a quantidade de veículos disponíveis em uma frota, de modo que cada veículo atenda de forma ótima seu conjunto de clientes. O problema traz um alto grau de complexidade devido ao grande número de combinações possíveis para a distribuição dos pontos de atendimento. É necessário encontrar o melhor arranjo, entre a quantidade e a sequência de atendimento dos clientes para cada veículo, de modo que os custos totais operacionais sejam minimizados. O *VRP* é considerado um problema da classe NP-Difícil (KIM *et al.*, 2015; JAYARATHNA *et al.*, 2021; ESTRADA-MORENO *et al.*, 2019). A classe NP abrange um conjunto de problemas que não possuem solução algorítmica em tempo polinomial no que diz respeito à complexidade computacional. Além da complexidade computacional, o grande número de variações, ou versões, do problema acaba aumentando sua complexidade.

2.2 Principais Variações do Problema do Roteamento de Veículos

Ao longo do tempo, variantes da versão clássica do *VRP* surgiram para solucionar problemas reais. Para resolver determinados problemas, conjuntos de restrições precisaram ser adicionados para que fosse possível lidar com as situações encontradas em ambientes reais. Mesmo sendo adaptações do problema clássico, as variações do *VRP* continuam sendo problemas da classe NP-Difícil, uma vez que, apenas, adicionam camadas extras de complexidade (CACERES-CRUZ *et al.*, 2014).

2.2.1 Problema do Roteamento de Veículos com Entregas e Coletas (*Vehicle Routing Problem with Pickup and Delivery -VRPPD*)

Extensão do problema clássico, o *VRPPD* adiciona ao problema a obrigatoriedade de entregar ou coletar itens a cada cliente visitado. Nesta variação do problema, cada veículo precisa respeitar a necessidade de cada cliente. Cada ponto de atendimento exige uma demanda específica, seja a entrega ou coleta de um item. Então, além de otimizar as rotas, é mandatório fazê-lo respeitando as restrições de entrega ou coleta. Os veículos precisam cumprir todo seu itinerário, coletando e entregando os itens, de modo que ao retornar ao depósito não tenha visitado um cliente mais de uma vez nem haja itens remanescentes nos veículos. O objetivo também é a minimização do custo operacional total.

2.2.2 Problema do Roteamento de Veículos com Janela de Tempo (*Vehicle Routing Problem with Time Window - VRPTW)*

Para a variação *VRPTW*, cada veículo precisa atender os clientes dentro das suas respectivas janelas de tempo. Cada cliente possui um horário inicial e final de atendimento e deve ser atendido dentro de sua janela temporal. Ao chegar para atendimento antes do tempo inicial, o veículo deve aguardar, somente podendo iniciar seu atendimento a partir do horário mínimo previsto. De outra forma, a janela superior de tempo define o horário máximo de atendimento, que uma vez ultrapassada, o cliente não poderá mais ser atendido. Muitas das abordagens que utilizam janelas de tempo também definem um período para atendimento, usualmente denominado tempo de serviço. Tempo de serviço é o tempo necessário para a realização do serviço no cliente, e no decorrer do problema é considerado para o cálculo do tempo total da operação. Outras abordagens desse problema permitem a existência de janelas de tempo com limites superiores flexíveis, aplicando uma penalização proporcional ao atraso (ESTRADA-MORENO *et al.*, 2019). A depender da abordagem utilizada, ao longo das execuções, essas violações vão deteriorando gradativamente as soluções, criando uma tendência de convergência para soluções que não violam, ou que violem menos, as restrições.

2.2.3 Problema do Roteamento de Veículos considerando Capacidade (*Capacitated Vehicle Routing Problem - CVRP*)

Nos *CVRP,* a capacidade de carga de cada veículo deve ser considerada. Os veículos não podem coletar itens além de sua capacidade de transporte. As rotas devem ser elaboradas observando a capacidade dos veículos, de modo que ao longo do problema essa restrição não seja violada. Existem duas sub variações principais dessa abordagem: problemas que consideram a capacidade com uma frota homogênea ou problemas que consideram a capacidade com uma frota heterogênea (PARRAGH, 2011). Por frota homogênea, podemos entender uma frota em que todos os veículos possuem as mesmas características (neste caso, a mesma capacidade). Frotas heterogêneas são frotas em que os veículos apresentam diferentes capacidades, ou características, de transporte. Em quaisquer abordagens, a capacidade de cada veículo deve ser respeitada para se obterem soluções válidas.

2.2.4 Problema do Roteamento de Veículos Multi Depósitos (*Multi-Depot Vehicle Routing Problem - MDVRP*)

Na sua definição clássica, o *VRP* utiliza o mesmo ponto de partida para todos os veículos. Mas, na variação *MDVRP*, há a existência de mais de um ponto de partida. Assim os veículos podem partir de pontos diferentes para realizar o atendimento aos clientes. É uma restrição comum a essa abordagem que os veículos retornem ao seu ponto original de partida, embora haja variações como o Problema do Roteamento de Veículos Aberto (*Open Vehicle Route Problem*), em que os veículos não ficam restritos a retornar ao seu ponto inicial (SARIKLIS; POWELL, 2000; BRANDÃO, 2004).

2.2.5 Variações complexas do *VRP*

Todas essas variações do *VRP* surgiram ao longo do tempo com o intuito de solucionar problemas difíceis do mundo real. O campo prático de maior aplicabilidade das abordagens do *VRP* é o campo da logística, e o transporte de cargas e materiais é a principal atividade de empresas que atuam nessa área. Contudo, é menos comum que um problema real se utilize apenas da versão clássica, ou de uma abordagem específica de forma isolada. O que ocorre, na maioria dos casos, é uma combinação de abordagens para poder solucionar um problema prático de forma efetiva. É muito comum que corporações do ramo da logística realizem o serviço de coleta e entrega de bens. Dessa maneira, a abordagem *VRPPD* poderia ser utilizada.

A Tabela 1 apresenta as principais variações do problema do roteamento de veículos, mas não pretende esgotar todas as suas possibilidades.

Tabela 1 – Sumário das variações do problema de roteamento de veículos

Abordagem	Minimizar distância total	Depósito	Considera Capacidade?	Considera Entregas e Coletas?	Considera Janelas de Tempo?
VRP	Sim	Único	Não	Não	Não
VRPPD	Sim	Único	Não	Sim	Não
VRPTW	Sim	Único	Não	Não	Sim
CVRP	Sim	Único	Sim	Não	Não
MDVRP	Sim	Múltiplos	Não	Não	Não
CVRPPD	Sim	Único	Sim	Sim	Não
CVRPTW	Sim	Único	Sim	Não	Sim
VRPPDTW	Sim	Único	Não	Sim	Sim
CVRPPDTW	Sim	Único	Sim	Sim	Sim

Elaborado pelo autor

Também é comum que haja um determinado prazo para realizar essa entrega ou coleta. Nesse caso seria necessário incorporar a abordagem *VRPTW*. Então, agora, já teríamos um problema que poderia ser denominado como: problema do roteamento de veículos com entregas e coletas considerando janelas de tempo, ou *VRPPDTW*. Indo mais além, em problemas reais não é possível ignorar a capacidade de transporte dos veículos, seja em uma frota homogênea ou heterogênea. Para tal, acrescentar-se-ia a abordagem que considera a capacidade do veículo também como uma restrição, assim teríamos *CVRPPDTW*. Ou considerar uma abordagem que considera a capacidade e as restrições de janelas temporais, como a CVPRTW, que é a abordagem utilizada neste trabalho. Na prática, o que vai definir a variação do *VRP* é a complexidade do problema e quais as restrições que serão consideradas. Quanto mais próximo de problemas reais, mais restrições necessitam ser adicionadas e maior é a complexidade para resolução do problema. A Tabela 1 sumariza as abordagens populares do *VRP*.

O sumário é apenas exemplificativo, e enumera as principais variantes do problema. Outras variações podem ser encontradas na literatura, ainda sendo possível encontrar outras adaptações relacionadas ao *VRP*, mas que não foram originalmente concebidas como um problema de roteamento de veículos.

2.3 Técnicas de Resolução do Problema do Roteamento de Veículos

Na literatura, as técnicas de resolução do problema de roteamento de veículos e suas variações são muito vastas, e incluem algoritmos exatos, algoritmos híbridos, heurísticas especializadas e metaheurísticas. Lozano *et al.* (2016)l utiliza um algoritmo exato para resolver o problema do roteamento de veículos. O algoritmo denominado *Pulse Algorithm* é inspirado na propagação de um pulso em um grafo e atua em dois estágios: no primeiro, o algoritmo identifica caminhos parciais e estima seus custos. No segundo, o caminho ótimo encontrado é retornado a partir de uma busca recursiva sobre os caminhos identificados. Zhang *et al.* (2019) também utilizou um algoritmo exato para encontrar rotas otimizadas para a distribuição de petróleo na *National Petroleum Corporation,* na China, onde caminhões tanque precisam transportar o petróleo até as estações de óleo dentro de um intervalo de tempo específico. A abordagem *Branch-and-Price-and-Cut* utilizada busca minimizar os custos operacionais sendo baseada no algoritmo de geração de colunas (*Column Generation Algorithm*), onde cada coluna representa uma solução factível do problema. Um relaxamento de programação linear é aplicado a um subconjunto de colunas, lavando a um problema linear restrito que pode ser rapidamente resolvido aplicando o método simplex.

Errico *et al.* (2018) também utiliza o *Branch-and-Price-and-Cut algorithm* para solucionar o *VRPTW* com tempo de serviço estocástico. Nessa variante, o tempo de atendimento de cada cliente é gerado estocasticamente para simular as necessidades dinâmicas de problemas reais. Kallehauge (2008) aponta que, dentre os algoritmos exatos, as abordagens de formulação de caminhos, como algoritmos de caminhos mínimos por exemplo, têm demonstrado as maiores contribuições em pesquisas relacionadas ao VRPTW, embora outras abordagens, como árvores geradoras, também tenham apresentado bons resultados. Shen *et al.* (2020) utiliza um algoritmo híbrido denominado *Brain Storm Optimization - BSO*, que combina inteligência de enxame de partículas com um sistema de colônia de formigas para resolver o problema de roteamento de veículos com janelas de tempo .

Xu *et al.* (2015) utiliza uma combinação entre algoritmos genéticos (*Genetic Algorithm - GA)* e algoritmos de enxame de partículas (*Particle Swarm Optimization - PSO*). O *PSO* é utilizado para codificar as soluções e um operador de cruzamento do algoritmo genético é utilizado para obter maior diversidade das partículas, evitar convergência prematura e proporcionar maior capacidade de exploração de espaço de soluções . Já Hu *et al.* (2013) propõe um algoritmo híbrido baseado em enxame de partículas para resolver o *VRPTW*. Utiliza *Chaos Optimization Algorithm - COA* e um processo de mutação gaussiana para evitar que as soluções fiquem presas a ótimos locais, e possam aumentar a diversidade da população. Tan *et al.* (2006) propõe um algoritmo evolucionário multiobjetivo híbrido com operadores genéticos especializados focados no problema *VRPTW*. O autor propõe um operador de cruzamento denominado r*oute-exchange crossover*, que consegue compartilhar suas melhores rotas com os indivíduos filhos. Esse processo permite transmitir para

gerações posteriores os trechos do cromossomo, avaliados como bons e com isso favorecer o processo de melhoria dos indivíduos da população. Nesse processo de cruzamento, os nós presentes nas rotas transmitidas são automaticamente excluídos de suas posições originais no indivíduo que as recebe, evitando, com isso, que haja duplicidade de nós no indivíduo filho. Já o processo de mutação pode ocorrer a partir de três abordagens distintas, em que uma delas é escolhida aleatoriamente para proceder uma pequena alteração no indivíduo para explorar ótimos locais .

Hedar e Bakr (2014) utiliza *Tabu Search Optimization* para resolver o *VRPTW*. O algoritmo *Tabu Search* realiza buscas locais em soluções previamente codificadas, com intuito de identificar ótimos locais. O processo se repete buscando encontrar aproximações de uma solução ótima global ao longo das iterações. Frifita e Masmoudi (2020) propõe uma nova abordagem do *VRP* aplicada ao atendimento domiciliar de pacientes. A abordagem denominada problema do roteamento de veículos com janelas de tempo, dependência temporal, multi estrutura e multi especialidade utiliza estratégias de busca de vizinhança e programação inteira como métodos de resolução. Vincent *et al.* (2021) apresenta uma nova variante do *VRP*, denominada *two-echelon vehicle routing problem with time windows, covering options, and occasional drivers*. Nesta abordagem, a frota possui dois tipos de veículos: veículos de carga ou motorista ocasionais, e também há dois tipos de entrega: em domicílio ou entrega alternativa. Os clientes escolhem uma opção de entrega, e sua demanda é atendida baseada na disponibilidade das frotas nos locais de cobertura. Neste trabalho, são utilizadas técnicas de programação inteira e busca adaptativa em ampla vizinhança para resolução do problema.

Harzi e Krichen (2017) utiliza *Variable Neighborhood Descendent - VND* para resolver o *VRPTW*. Essa abordagem é considerada uma metaheurística relativamente nova na literatura, mas tem sido aplicada com sucesso em problemas de otimização combinatorial. A abordagem *VND* realiza uma busca local baseada na vizinhança, até que o melhor vizinho seja identificado, a partir de um nó de referência. Esse processo continua até que todos os nós tenham sido verificados e uma solução válida tenha sido construída . O algoritmo *Simulated Annealing* foi utilizado por Liang *et al.* (2022) para lidar com o problema do roteamento de veículos com armário de encomendas, uma nova variante do *VRP* proposta pelos autores. Nesta variante, os clientes podem optar por receber os itens, não só em suas residências, como é mais comum, mas também em seus armários de encomendas. Segundo os autores, isso facilita o processo de entrega ou coleta de bens porque permite uma maior flexibilidade de horários para realização dos serviços.

Heurísticas bioinspiradas, como algoritmos de colônia de formigas, colônia de abelhas e enxame de partículas, são abordagens amplamente utilizadas para resolução do problema do roteamento de veículos. Silva Junior *et al.* (2021) utiliza uma combinação do sistema de colônia de formigas múltiplas com uma busca local realizada a partir da abordagem r*andom variable neighborhood descent* para resolver o *VRPTW* dinâmico. Yao

et al. (2017) propõe um algoritmo de colônia de abelhas aprimorado, introduzindo um novo operador de cruzamento e uma estratégia de escaneamento para otimização local. A estratégia de escaneamento considera o ângulo de cada cliente em relação ao depósito, dentro de um raio de ação e o operador de cruzamento age baseado na estratégia de escaneamento ordenando os clientes pelo valor de ângulo formado em relação ao depósito. Os clientes são ordenados em ordem crescente do grau do ângulo formado, e um caminho é formado a partir desses clientes. Alzaqebah *et al.* (2018) investiga o algoritmo de colônias de abelhas para identificar os pontos fortes e fracos desse tipo de abordagem. O autor identifica como ponto forte sua capacidade de exploração, com as abelhas batedoras, e a performance de explotação das abelhas recrutadoras. Também aponta que a principal fraqueza do algoritmo é ser fortemente dependente de parâmetros, uma vez que cada instância pode requerer uma configuração de parâmetros diferentes para que o algoritmo alcance uma solução efetiva .

Dong *et al.* (2018) propõe uma abordagem denominada *tissue P system* baseado em algoritmos genéticos multiobjetivos. A abordagem utiliza *discrete glowworm evolution mechanism e variable neighborhood evolution mechanism* como sub etapas do algoritmo para balancear os processos de exploração e explotação. Lan *et al.* (2020) usou de um algoritmo de descida de vizinhança variável multi objetivo baseado em decomposição para resolver o VRPTW. O algoritmo tri objetivo considera a minimização do custo total da rota, o tempo de espera para atendimento e o número total de veículos. A abordagem utiliza os operadores evolutivos para realizar seleção, mutação e cruzamento das soluções e o algoritmo de descida de vizinhança como técnica de busca local. Além disso, foram utilizadas estratégias de inicialização heurísticas e arquivos externos de suporte para melhorar o desempenho .

Por ser um problema reconhecidamente difícil, a resolução do *VRP* por algoritmos exatos em tempo factível torna-se muito complexa. Nesse sentido, abordagens heurísticas e metaheurísticas têm sido amplamente utilizadas buscando aproximações das soluções ótimas globais. Em essência, o VRP e suas variantes são problemas multiobjetivos, pois envolvem a otimização de um conjunto de objetivos, que muitas vezes são conflitantes entre si. A minimização do custo operacional total, por exemplo, passa por minimizar a distância total das rotas, tempo total de percurso, tempo de espera, o número de veículos utilizados e outros aspectos aplicados a problemas específicos. É comum a utilização de abordagens que adaptam a função de custo a um único objetivo, mas suas características multiobjetivas intrínsecas, sem dúvida, favorecem o uso de heurísticas e metaheurísticas que consideram objetivos múltiplos.

Dentre as abordagens heurísticas mais utilizadas na literatura, as bioinspiradas possuem um grande destaque, com bons resultados no que diz respeito aos problemas de roteamento (OMBUKI *et al.*, 2006). Nesse sentido, os algoritmos genéticos multiobjetivos são fortes candidatos para lidar com esse tipo de problema, uma vez que, por utilizar a fronteira

de pareto para identificar as melhores soluções, não necessitam de uma classificação de peso ou importância aplicada aos objetivos das funções de custo para definir a aptidão de uma determinada solução, evitando assim o enviesamento da avaliação (OMBUKI *et al.*, 2006).

2.4 Otimização Multiobjetivo e o Problema do Planejamento de Rotas

2.4.1 Problemas de Otimização Multiobjetivo

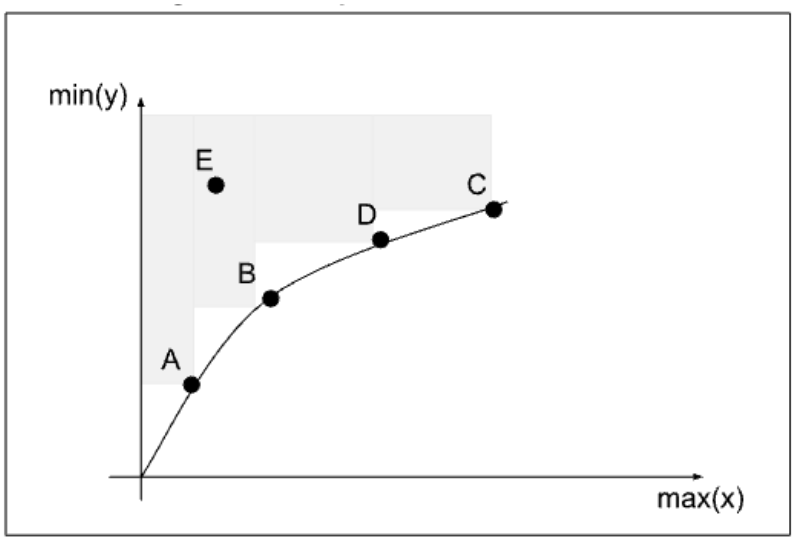
Problemas de otimização têm como principal objetivo a maximização, ou minimização, de um determinado objetivo. Esses tipos de problemas utilizam técnicas e abordagens capazes de identificar os valores mínimos ou máximos de uma determinada função de custo. Ocorre que para muitas classes de problemas, a otimização de apenas um único objetivo não é capaz de identificar uma solução que seja abrangente e eficaz o suficiente para resolver o problema em toda sua complexidade. Denominamos problemas de otimização multiobjetivo, os problemas para os quais a análise e resolução de uma única dimensão do problema não seja suficiente para obter uma solução eficaz que mantenha o equilíbrio entre todos os seus objetivos. Nesses casos, a qualidade de uma solução é definida pelo relacionamento entre os diversos objetivos, que podem, ou não, ser conflitantes entre si. Assim, não há apenas uma única função para otimizar, mas sim um conjunto de funções, bem como não há apenas uma única solução para o problema, mas um conjunto de soluções. E esse conjunto de soluções pode ser identificado a partir do uso da teoria do ótimo de Pareto (COELLO, 2007).

O ótimo de Pareto é uma região do espaço objetivo onde está localizado o conjunto de soluções que resolvem o problema de forma otimizada. Para identificar esse conjunto, é necessária a compreensão do conceito de dominância. Dentro da teoria do ótimo de Pareto, as soluções podem ser dominadas, dominantes ou não dominadas. Dadas duas soluções x e x', é possível afirmar que a solução x' é uma solução dominada, em relação a x, se, e somente se, cada um dos valores do seu conjunto de objetivos forem iguais ou inferiores aos valores dos respectivos objetivos da solução x (assumindo um problema de maximização). Já a solução x, em relação a x', é considerada uma solução dominante se não perde em nenhum dos objetivos, podendo vencer em todos, ou ao empatar, vence em ao menos um dos objetivos de x' (EIBEN; SMITH, 2015). As soluções ditas não dominadas são aquelas soluções que, em relação às demais soluções, apresentam o melhor resultado em ao menos um dos objetivos. Isso quer dizer que não são soluções dominadas, pois, em ao menos um objetivo, essa solução será mais otimizada que as demais. A esse conjunto de soluções não dominadas damos o nome de Fronteira de Pareto. A fronteira composta por soluções ótimas que resolvem o problema, normalmente, é desconhecida. O que ocorre na prática é que a cada iteração, uma frente de Pareto é identificada e definida em relação

à geração atual. O objetivo é aproximar essa fronteira gerada cada vez mais próximo da fronteira dos pontos ótimos para o problema.

Embora haja outras abordagens para resolução de problemas de otimização multi-objetivos, GOLDBERG (1989) sugere a abordagem baseada em dominância ao invés de considerar os valores absolutos de cada objetivo. O autor salienta que essa abordagem utilizada com métodos de nicho ou de especialização contribuem para a manutenção da diversidade e tem demonstrado grande eficiência para lidar com esse tipo de problema. A Figura 2 demonstra a representação gráfica relacionada à compra de um automóvel fictício considerando as seguintes características: conforto e custo. Neste exemplo, um veículo tende a ter menos conforto à medida que seu custo diminui. O problema apresenta dois objetivos, o custo que deve ser minimizado e o conforto que deve ser maximizado, porém esses objetivos são conflitantes entre si. Assim, nesse exemplo, não é possível identificar uma solução que possua máximo conforto e mínimo custo ao mesmo tempo, uma vez que a melhoria de um objetivo causa necessariamente a degradação do outro. O que existe é uma relação entre custo e conforto que deve ser avaliada para identificar a opção mais adequada para um caso específico, considerando as soluções que compõem a frente de Pareto. Dado que o eixo Y representa a custo e o eixo X representa o conforto, a figura abaixo apresenta a representação gráfica das soluções no espaço objetivo.

Figura 2 – Exemplo da fronteira de Pareto



Inspirado em Deb (2018)

Os pontos *A, B, C* e *D* formam a frente de Pareto, pois cada uma das soluções é uma solução não dominada, ou seja, vence as demais soluções em ao menos um dos objetivos e não é dominada por nenhuma outra solução. A área acinzentada representa a região onde todas as soluções são dominadas e consequentemente não são úteis ao problema. A solução *E* é uma solução dominada e não pode fazer parte da fronteira de Pareto. Para problemas multiobjetivos, a fronteira de pareto auxilia na tomada de decisão

frente a um problema que necessite da avaliação entre mais de um aspecto ou característica, diferentemente de problemas mono objetivo, onde o algoritmo encontra a única solução ótima, ou sua melhor aproximação.

2.4.2 Computação Evolucionária (*Evolutionary Computation - EC*)

Coello (2007) aponta que os algoritmos de otimização podem ser classificados em três categorias principais: algoritmos enumerativos, determinísticos e estocásticos. Embora os algoritmos enumerativos também sejam determinísticos, o autor salienta que as diferenças, nessa classificação, é que os algoritmos determinísticos podem utilizar heurísticas, ao contrário dos enumerativos que buscam avaliar todas as possibilidades de soluções para identificar a solução ótima, atuando como um algoritmo de força bruta. Por motivos óbvios, abordagens enumerativas, embora sejam de simples implementação, são mais ineficientes quanto maior é o espaço de busca, uma vez que quanto maior o espaço de busca maior a complexidade computacional para resolvê-lo. Já os algoritmos estocásticos tentam incorporar conhecimentos do domínio do problema para identificar soluções próximas do ótimo global com menor custo computacional. Dentre os métodos de busca estocásticos, temos a computação evolucionária como a área que descreve os métodos que utilizam a metáfora do processo de evolução natural como base para a descoberta de soluções ótimas proximais. Dentre as abordagens evolutivas mais comuns estão os algoritmos genéticos, estratégias evolucionárias e a programação evolucionária. Todas essas técnicas baseiam-se no conceito *Darwiniano* de sobrevivência do mais apto. As soluções são codificadas como indivíduos, que submetidos a uma função de avaliação, têm seu nível de aptidão classificado. Cada solução está associada a uma aptidão, e essa aptidão é quem vai guiar o processo evolutivo. Um conjunto de indivíduos representa uma população.

Essa população é submetida a um processo de seleção natural simulado, para que ao longo das gerações, uma solução ótima possa ser identificada. Ao longo dessas gerações, esses indivíduos são submetidos a variações, recombinações e competições em busca de serem selecionados para propagarem seu genótipo para as gerações seguintes. Os algoritmos evolucionários são amplamente utilizados e têm se demonstrado uma ferramenta muito eficiente e capazes de resolver problemas complexos com um ou muitos objetivos (LÜCKEN *et al.*, 2014). Dentre eles, os algoritmos genéticos são amplamente utilizados para resolução de problemas de otimização de forma geral, e têm sido aplicados a problemas nas áreas de roteamento de veículos, computação em nuvem, processamento de imagem, computação paralela, dentre outros (DEVI *et al.*, 2014). O trabalho realizado por Xu *et al.* (2015) aponta que, de 2007 a 2015, algoritmos genéticos e algoritmos de enxame de partículas foram as abordagens mais utilizadas para lidar com problemas de seleção de características para algoritmos de classificação. Os autores indicam ainda que, dentre as

técnicas de computação evolucionária, os algoritmos genéticos são a abordagem mais amplamente utilizada para esse tipo de problema, com trabalhos que datam do final da década de 1980 .

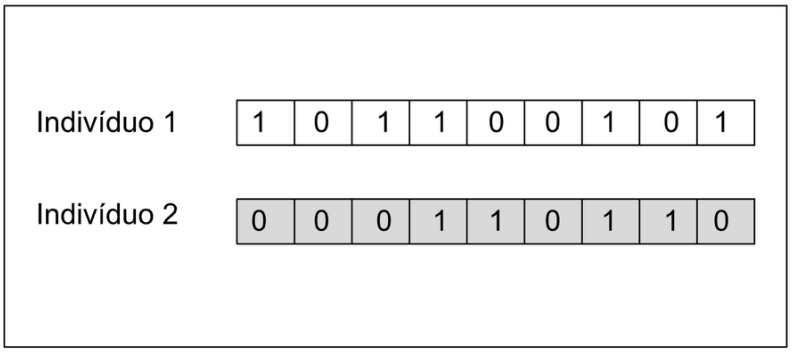
2.4.3 Visão geral sobre o Algoritmo Genético Simples

Os algoritmos genéticos são os mais amplamente conhecidos e, dentre as abordagens evolucionárias, seus conceitos basilares foram inicialmente propostos por Holland em um estudo sobre comportamento adaptativo, no trabalho denominado: *Adaptation in Natural and Artificial Systems* (HOLLAND, 1992). Os conceitos elaborados por Goldberg, juntamente com o trabalho desenvolvido por Jong (1975) em sua tese de doutorado, definiram o que hoje se compreende como a versão canônica do algoritmo genético ou S*imple Genetic Algorithm - SGA*. O *SGA* segue um conjunto de operações padrão que o caracterizam e para compreender seu funcionamento é necessário entender os processos utilizados e como esses processos evoluíram ao longo do tempo. As principais operações que definem o *SGA* são sua representação, processo de recombinação, processo de mutação e seleção de sobreviventes.

2.4.3.1 Representação binária

Na representação binária, os indivíduos são codificados em termos de zeros e uns. O tamanho do cromossomo é definido pela necessidade de representação do problema e cada gene apresenta um valor que está associado a alguma característica do problema no mundo real. Por exemplo, 1 pode representar o estado ativo de uma máquina, enquanto 0 pode representar o estado inativo. A quantidade de genes pode representar a quantidade de máquinas em um determinado sistema e o cromossomo inteiro representa o sistema completo. A Figura 3 apresenta a representação binária em um *SGA.*

Figura 3 – Exemplo da representação binária



Elaborado pelo autor

O cromossomo então pode representar o estado global de um sistema complexo com

base nas atividades individuais de cada uma das máquinas. Outras representações foram elaboradas ao longo do tempo, tais como: representação inteira, representações com valores reais, representação permutacional e representação em árvore (EIBEN; SMITH, 2015), além de representações customizadas que combinam versões clássicas para adaptarem-se a problemas específicos. Em relação a inicialização da população inicial, no *SGA* o processo mais comum é de forma aleatória. Nesses casos, os valores que cada gene vai assumir, 0 ou 1, é escolhido aleatoriamente utilizando uma probabilidade uniforme de 50% de chance para o valor 1 e 50% de chance para o valor 0. Os operadores de mutação e recombinação é que vão determinar as alterações a serem realizadas nesses indivíduos.

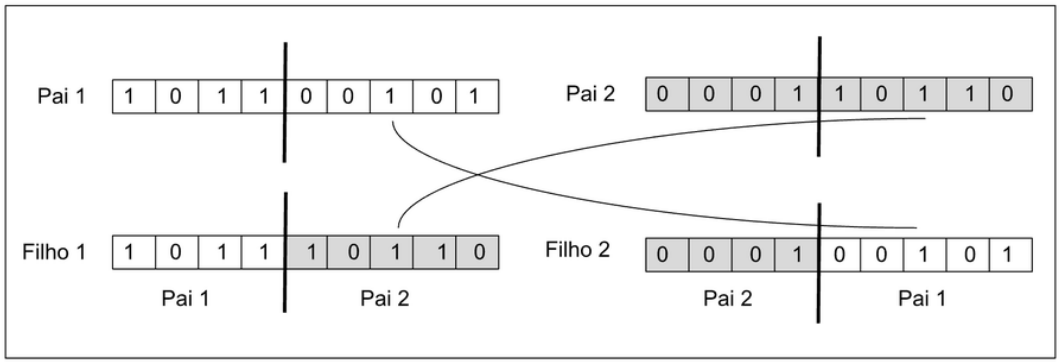
2.4.3.2 Inicialização da população

Nos *SGAs*, usualmente as populações são definidas com indivíduos criados aleatoriamente. Durante o processo evolutivo, esses indivíduos sofrem variações e recombinações que possibilitam que os novos indivíduos que surjam possam se adaptar melhor ao ambiente simulado. Em situações de maior complexidade, a utilização puramente aleatória de geração dos indivíduos pode não ser a forma mais eficiente de lidar com o problema. Diante de cenários mais complexos, o uso de heurísticas especializadas pode ser uma abordagem que auxilie no processo de geração dos indivíduos e da população inicial. Esse tipo de abordagem usualmente utiliza informações do contexto ou domínio para projetar soluções que possam acelerar de forma eficiente o processo evolucionário. Outro fator relevante ao adotar heurísticas para a geração da população se dá em relação ao tempo de execução. Uma heurística bem projetada pode levar a geração a um patamar que, de forma aleatória em um *SGA*, acontecesse em um tempo de execução muito mais elevado. Dessa forma, a escolha entre criação aleatória, ou heurística, deve ser definida conforme a complexidade do problema e a qualidade da heurística utilizada.

2.4.3.3 Processo de recombinação

A recombinação é o processo que permite que indivíduos compartilhem seus materiais genéticos com intuito de gerar indivíduos filhos que possam herdar características dos indivíduos pais. No *SGA*, as recombinações dos indivíduos são definidas com o cruzamento de um ponto (*one-point crossover*). Nesse tipo de recombinação, um ponto de corte é sorteado aleatoriamente dentro do intervalo $[1, t-1]$, sendo t o tamanho do cromossomo. Esse ponto de corte é aplicado a ambos os indivíduos e o bloco de genes anteriores (ou posteriores) ao ponto de corte são trocados entre os indivíduos, como demonstrado na Figura 4.

Figura 4 – Exemplo de cruzamento de um ponto



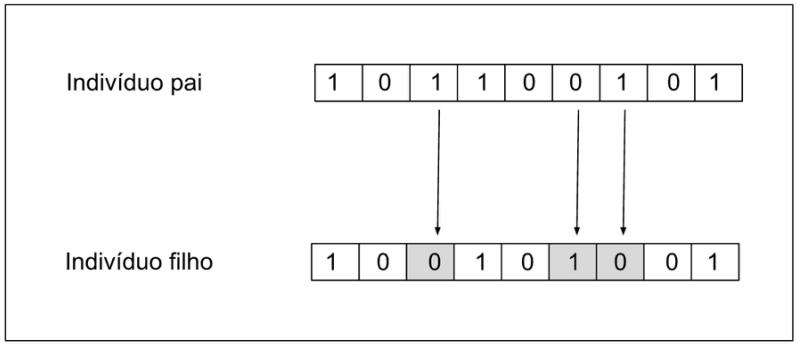
Elaborado pelo autor

Além dessa versão mais clássica de recombinação, outra estratégia semelhante é o cruzamento de *n-pontos*. Nesse tipo de cruzamento, ao invés de um ponto ser aleatoriamente sorteado, a quantidade de pontos de corte também é sorteada dentro de um intervalo definido. Dessa forma vários pontos de corte são selecionados e os trechos genéticos recombinados para gerar os indivíduos filhos. Outra estratégia bastante utilizada é o cruzamento uniforme. Nesse tipo de cruzamento cada gene tem uma probabilidade uniforme de ser selecionado de um, ou outro indivíduo pai. Então, para cada gene a ser atribuído a um indivíduo filho, realiza-se um sorteio para identificar de qual pai será herdado. Assim como no processo de representação do indivíduo, os processos de recombinação possuem ampla diversificação na literatura, uma vez que nem todos os tipos de recombinação são aplicáveis a qualquer tipo de representação. Também é vasta a literatura sobre operadores de recombinação inéditos construídos para problemas específicos. Esses operadores podem utilizar conhecimento do domínio para formular procedimentos que possibilitem a transmissão de trechos genéticos considerados bons. Ao final, o grande objetivo do processo de recombinação é promover uma exploração do espaço de soluções em buscas de novos locais promissores.

2.4.3.4 Processo de mutação

O processo de mutação mais tradicional para um problema de representação binária é a mutação *bit-flip*. Nesse processo, cada gene é considerado separadamente e a ele é aplicado uma pequena probabilidade de mutação. Na mutação *bit-flip*, o gene, caso selecionado para ser variado, terá seu valor alterado (i.e de 1 para 0, ou de 0 para 1) de acordo com seu valor atual. O número de genes modificados não é fixo e vai depender das probabilidades individuais que serão avaliadas. A mutação é apresentada na Figura 5.

Figura 5 – Exemplo da mutação *bit-flip*



Elaborado pelo autor

A Figura 5 demonstra o processo de mutação *bit-flip* onde apenas os genes das posições 3, 6 e 7 foram selecionados para a variação a partir do indivíduo "pai". Essa variação permite gerar um novo indivíduo "filho". Diferente do processo de recombinação, a mutação visa realizar um processo de exploração, também conhecido como busca global. A busca local objetiva identificar novas soluções nas proximidades do indivíduo que estamos variando. Usualmente as probabilidades de um indivíduo sofrer o processo de mutação são baixas, variando de 0,5% a 5%, embora não seja regra. Taxas maiores podem ser utilizadas a depender do objetivo. Além disso, existem abordagens de taxas de mutação adaptativas, que podem favorecer o processo evolucionário, aplicando taxas maiores no início do processo, com uma redução progressiva ao longo das gerações.

2.4.3.5 Processo de avaliação dos indivíduos

A avaliação é o processo pelo qual cada indivíduo é verificado em relação a sua adaptação ao ambiente. No processo biológico, os indivíduos melhor adaptados ao ambiente tendem a sobreviver e passar seus genes adiante, gerando mais indivíduos com as mesmas características. Similarmente, no processo computacional, os indivíduos melhores avaliados têm maior probabilidade de transferir seus códigos genéticos para gerações futuras. Nas simulações computacionais, a avaliação é dada por uma função, conhecida como função objetivo, que calcula a aptidão (*fitness*) de cada indivíduo. No *SGA* não existe uma função objetivo padrão, uma vez que essa função é completamente dependente do problema. Problemas distintos têm formas de avaliação diferentes, mas a aptidão é peça fundamental para guiar o processo de seleção de pais e de sobreviventes.

2.4.3.6 Seleção de pais

Para poder realizar o *pool* de acasalamento, é necessário desenvolver mecanismos que indiquem quais dos indivíduos serão selecionados para serem submetidos ao processo de recombinação. Dentre as estratégias encontradas na literatura, podemos citar, como

abordagens comuns utilizadas no *SGA:* a seleção proporcional a aptidão, seleção através de ranking, seleção através de torneio e a seleção uniforme. Na seleção proporcional à aptidão, a probabilidade de um determinado indivíduo ser selecionado é obtida pela função:

$$P(i) = \frac{fi}{\sum_{j=1}^{\mu} fi} \tag{2.1}$$

Dessa forma, a probabilidade de um indivíduo *i* ser selecionado é dada pela razão entre a aptidão fi e o somatório de todas as aptidões dos demais indivíduos da população (HOLLAND, 1975). Embora seja uma abordagem muito popular, ela traz consigo alguns problemas. Como os indivíduos de melhor aptidão têm maior probabilidade de serem selecionados, ocorre que em poucas gerações esses indivíduos dominam a população e acabam inviabilizando uma busca mais abrangente no espaço de soluções. E terminam por concentrar as soluções em um pequeno local do espaço de soluções. Esse fenômeno é conhecido como convergência prematura. Outro problema comum a esse tipo de abordagem ocorre quando as aptidões dos indivíduos estão em patamares muito semelhantes. À medida que os valores absolutos das aptidões se aproximam, as probabilidades tendem, também, a se aproximar de tal forma que a distribuição se torna quase uniforme ocasionando a perda da pressão de seleção. Isso faz com que indivíduos que se afastem da aptidão média tendam a não serem selecionados mesmo possuindo aptidão melhor que os demais indivíduos (EIBEN; SMITH, 2015) . A seleção baseada em ranking ordena os indivíduos da população a partir de suas aptidões e, a cada um, atribui a probabilidade proporcional à sua posição no ranking. Essa abordagem mantém a pressão de seleção por considerar a colocação do indivíduo no ranking ao invés de considerar o valor absoluto da aptidão (EIBEN; SMITH, 2015).

As duas abordagens citadas consideram toda a população para realização da seleção de Pais, porém em populações demasiadamente grandes, pode haver um alto consumo computacional, ou até mesmo a impossibilidade de realizar a análise. Dessa forma, para lidar com populações muito grandes, é possível realizar a seleção por torneio. Esse tipo de operador facilita a seleção, pois não necessita de conhecimento sobre a aptidão da população inteira. Ele realiza somente a comparação entre dois, ou um pequeno grupo de indivíduos, selecionados aleatoriamente para competir entre si pela participação no *pool* de acasalamento. Por não necessitar verificar a população inteira, essa abordagem pode melhorar consideravelmente a eficiência computacional. Por fim, é possível citar a seleção uniforme, que pode ser considerada, entre as já mencionadas, uma abordagem mais simples, uma vez que atribui a todos os indivíduos a mesma probabilidade de seleção. À primeira vista, isso pode levar a um enfraquecimento da pressão de seleção, pois qualquer indivíduo, independente de sua aptidão, pode ser selecionado. Para que isso não ocorra, é sugerido que esse tipo de operador seja acompanhado de um mecanismo de seleção de

sobreviventes robusto e fortemente baseado na aptidão (EIBEN; SMITH, 2015).

2.4.3.7 Seleção de sobreviventes

Após o processo de variação, recombinação e mutação, é preciso selecionar os indivíduos que irão compor a população da geração seguinte. Usualmente, a seleção de sobreviventes é obtida a partir de alguma estratégia de seleção, que é aplicada à população inicial. No *SGA*, o gerenciamento da população é comumente obtido pelo modelo generacional. No modelo generacional, tem-se uma população inicial de tamanho λ de onde um conjunto de pais é selecionado para o *pool* de acasalamento, gerando uma prole de tamanho μ. No modelo generacional tanto λ e μ, quanto o *pool* de acasalamento têm o mesmo tamanho. Dessa forma toda população é substituída por sua prole ao final da geração. Existem diversas abordagens para realizar a seleção de sobreviventes. As abordagens podem ser baseadas na aptidão ou na idade dos indivíduos. Na seleção baseada na idade, a aptidão não é considerada e cada indivíduo é projetado para sobreviver pelo mesmo número de gerações do processo evolucionário.

2.4.4 Algoritmos Evolucionários Multiobjetivo (*Multi Objective Evolutionary Algorithms - MOEA*)

Os algoritmos evolucionários multiobjetivos são as abordagens da área da computação evolucionária capazes de lidar com problemas que exigem a avaliação de objetivos múltiplos. Nesse sentido, esses algoritmos vêm se aprimorando ao longo dos anos e têm se demonstrado altamente robustos para resolver problemas de otimização, embora isso não implique afirmar que essas técnicas são soluções universais para quaisquer problemas de otimização. A primeira abordagem a utilizar essas técnicas foi proposta por Schaffer *et al.* (1985) em sua tese de doutorado, embora isso não seja um consenso na literatura, uma vez que há autores que indicam que o trabalho proposto por Ito et al. (1983) seja anterior ao proposto por Schaffer. Em sua abordagem, conhecida como *VEGA* (*vector-evaluated genetic algorithm*), Schaffer propõe a divisão da população em subpopulações, onde cada uma das subpopulações eram avaliadas de acordo com uma função objetivo, mas a seleção de pais e as recombinações eram realizadas de forma global (ITO *et al.*, 1983). Essa abordagem apresentou boa aproximação da frente de pareto em poucas gerações, mas essa aproximação não permanece eficiente indefinidamente (EIBEN; SMITH, 2015).

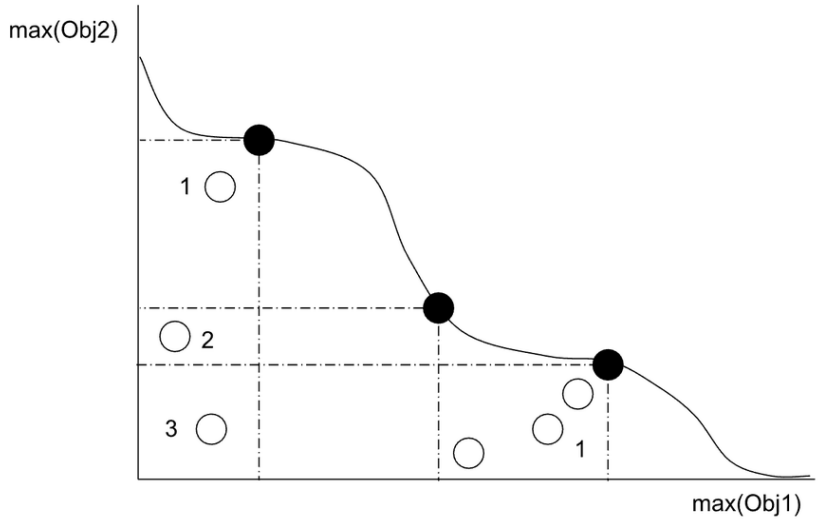
O trabalho de Goldberg (1989) foi o primeiro a introduzir a ideia de avaliação baseada na frente de Pareto. Foi Goldberg que sugeriu a ideia de mover a população para a fronteira das soluções não dominadas. O autor sugere ainda a utilização de técnicas de nicho, para evitar que o GA (Genetic Algorithm) convergisse para um único ponto da fronteira (COELLO, 2007). Uma revisão da literatura realizada por (LIANG *et al.*, 2022) aponta que as principais abordagens utilizadas para resolver problemas de planejamento de rotas como o *VRPTW*,

planejamento de rotas para robôs, problemas do caixeiro viajante e planejamento de rotas para veículos aéreos não tripulados, são os algoritmos genéticos multiobjetivos (*Multi Object Genetic Algorithm - MOGA*) e suas variações, o *NSGA-II* (*Nondominated Sorting Genetic Algorithm*) e um framework coevolucionário para problema de otimização multi objetivo restrito (*Coevolutionary Framework for Constrained Multiobjective Optimization Problems - CCMO*).

2.4.4.1 Algoritmo Genético Multiobjetivo - *MOGA*

Um algoritmo genético multiobjetivo segue todo processo do algoritmo genético tradicional no que diz respeito à mutação, cruzamento e representação, mas se diferencia no momento de realizar a avaliação desses indivíduos. Em um *GA* tradicional, existem diversas técnicas de definição da aptidão dos indivíduos considerando apenas um único objetivo. Já no *MOGA*, proposto por (FONSECA; FLEMING, 1995) como uma variação do trabalho de Goldberg, a aptidão de cada indivíduo é dada utilizando um ranqueamento baseado na dominância de Pareto. Nessa abordagem, cada indivíduo é inserido em um *ranking r* baseado na seguinte regra: $r(x_i, t) = 1 + p_i^t$, onde x_i é um indivíduo em uma geração t que é dominado por uma solução p_i^t . Quer dizer que o valor de ranqueamento atribuído a cada indivíduo é dado pelo número de soluções que o domina mais um e as soluções não dominadas possuem valor 1. Nesse ranqueamento as soluções de menor pontuação representam as soluções de maior qualidade. A Figura 6 apresenta a frente de Pareto e as soluções dominadas com seus respectivos valores de ranqueamento.

Figura 6 – Exemplo da definição do ranking das soluções



Inspirado em COELLO et. al. (2007)

Os procedimentos do *MOGA* são demonstrados no pseudocódigo abaixo na Figura 7.

Figura 7 – Pseudocódigo do MOGA

```
Procedimento MOGA(N', g, fk(x)) N' membros envolvidos em g gerações para resolver fk(x)
     Inicialize a população P
     Avaliar os indivíduos em relação aos seus objetivos
     Atribuir os indivíduos ao ranking baseado na dominância de Pareto
     Calcular a quantidade de nichos
     Atribuir aptidão em escala linear
     Atribuir fitness Compartilhado
     para i=1 até g faça
          Seleção via Amostragem Estocástica Universal
          Efetuar o cruzamento de um ponto
          Proceder mutação
          Avaliar os indivíduos em relação aos seus objetivos
          Atribuir os indivíduos ao ranking baseado na dominância de Pareto
          Calcular a quantidade de nichos
          Atribuir aptidão em escala linear
          Atribuir fitness Compartilhado
     fim para
fim procedimento
```

Inspirado em FONSECA (1995)

2.4.4.2 *Nondominated Sorting Genetic Algorithm (NSGA I)*

O primeiro modelo do *NSGA* foi proposto por Srinivas e Deb em 1994 (SRINIVAS; DEB, 1994). Esse algoritmo trouxe uma modificação no procedimento de ranqueamento proposto do Goldberg (1989). Nessa nova abordagem, a população é classificada em termos da dominância das soluções. Todos os indivíduos não dominados são classificados em uma categoria utilizando uma aptidão fictícia, proporcional ao tamanho da população. Os indivíduos já classificados são ignorados, e outra rodada de classificação é realizada para identificar um novo conjunto de indivíduos não dominados (uma nova frente de pareto). Essa classificação ocorre até que não haja mais indivíduos para serem classificados e categorizados. Os indivíduos que foram classificados na primeira frente de pareto são os indivíduos de aptidão máxima, logo, terão mais cópias dentro da população, auxiliando para que o processo evolutivo possa convergir para as regiões mais promissoras do espaço de soluções (COELLO, 2007). Os procedimentos adotados no *NSGA I* estão descritos no pseudocódigo da Figura 8.

Figura 8 – Pseudocódigo do *NSGA-I*

```
Procedimento MOGA(N', g, fk(x)) N' membros envolvidos em g gerações para resolver fk(x)
     Inicialize a população P
     Avaliar os indivíduos em relação aos seus objetivos
     Atribuir os indivíduos ao ranking baseado na dominância de Pareto
     Calcular a quantidade de nichos
     Atribuir aptidão em escala linear
     Atribuir fitness Compartilhado
     para i=1 até g faça
          Seleção via Amostragem Estocástica Universal
          Efetuar o cruzamento de um ponto
          Proceder mutação
          Avaliar os indivíduos em relação aos seus objetivos
          Atribuir os indivíduos ao ranking baseado na dominância de Pareto
          Calcular a quantidade de nichos
          Atribuir aptidão em escala linear
          Atribuir fitness Compartilhado
     fim para
fim procedimento
```

Inspirado em Deb (1994)

É importante notar que a variação dos indivíduos (mutação e recombinação) pode seguir os processos descritos no *SGA*, mas não se limitam a estes. Outras abordagens podem ser implementadas de acordo com o problema e a representação utilizada. Nessa implementação no *NSGA I*, o que se destaca é a forma de realizar o processo de avaliação dos indivíduos, mas todo restante do processo evolutivo é conservado. A mudança no processo de avaliação é imprescindível para que os múltiplos objetivos possam ser considerados.

2.4.4.3 *Nondominated Sorting Genetic Algorithm (NSGA II)*

Em 2002, (DEB, 2001) publica o trabalho onde apresenta a versão intitulada *NSGA-II*. Essa versão surge como uma melhoria do *NSGA* originário. Para o autor, as principais críticas ao *NSGA* eram relacionadas à sua ineficiência quanto à complexidade computacional, com tempo de execução da ordem de $O(mn^3)$ (sendo m o número de objetivos e n o tamanho da população), a utilização de uma abordagem não elitista e a necessidade de especificar um parâmetro compartilhado entre as soluções. Uma das principais melhorias apresentadas no *NSGA II* foi a redução da complexidade computacional, que nessa versão II é da ordem de $O(mn^2)$, o que tornou o algoritmo significativamente mais rápido. O parâmetro compartilhado, denominado σ_{share}, que denota a distância máxima em que duas soluções compartilham sua aptidão, era um parâmetro definido pelo usuário, e em larga medida, responsável pela manutenção da diversidade da população. Dessa forma, a definição de valores imprecisos para esse parâmetro comprometia a eficiência geral do algoritmo. Na nova versão, o parâmetro de compartilhamento é substituído por um processo que o autor chama de comparação de aglomeração (*crowded-comparison*). Segundo o

autor, essa abordagem elimina a necessidade de parametrização do usuário e mantém a diversidade da população. Além disso, ao utilizar a abordagem de preservação elitista, os melhores indivíduos não dominados encontrados tanto na geração inicial quanto na prole são selecionados para a próxima geração. Conforme o autor, isso garante a participação dos melhores indivíduos ao longo do processo evolutivo, favorecendo a convergência do algoritmo. Os procedimentos dos *NSGA-II* podem ser demonstrados no pseudocódigo da Figura 9.

Figura 9 – Pseudocódigo do *NSGA-II*

```
Procedimento NSGA-II(N ,g,fk(xk))N N membros envolvidos na geração g para resolver fk(x)
     Inicialize a população P
     Gerar uma população aleatório de tamanho N
     Avaliar os indivíduos de acordo com seus objetivos
     Atribuir o ranking baseado na dominância de Pareto - Ordenados
     Gerar uma população de filhos
          Seleção binária por torneio
          Recombinação e mutação
     para i = 1 até g faça
          para cada pai e filho na população faça
               Atribuir o ranking baseado na dominância de Pareto - Ordenados
               Gerar conjuntos de vetores não dominados ao longo dos pontos de referência
               Iterar nesse conjunto adicionando soluções para as próximas gerações iniciando
               do início da fronteira até os N indivíduos encontrados para determinar a
               distância de aglomeração entre os pontos da fronteira de Pareto
          fim para
          Pontos selecionados (elitistas) na frente inferior (com classificação inferior) e
          estão fora de uma distância de aglomeração
          Gerar uma população de filhos
               Seleção binária por torneio
               Recombinação e mutação
     fim para
fim procedimento
```

Inspirado em Deb (2002)

2.4.4.4 *Nondominated Sorting Genetic Algorithm (NSGA-III)*

O *NSGA-III* também foi proposto por Deb e Jain (2014) e é baseado no *NSGA-II*. Essa versão do *NSGA* é otimizada para lidar com problemas com muitos objetivos, usualmente quatro ou mais e traz mudanças significativas no processo de seleção de sobreviventes e é o algoritmo multiobjetivo baseado em não dominância mais eficiente e robusto conhecido (JAIN; DEB, 2013). À medida que os objetivos aumentam, uma fração maior da população será de indivíduos não dominados (KALYANMOY DEB, 2001), e esse aumento demasiado dos indivíduos na frente de Pareto limita o espaço para a criação de novas soluções em uma mesma geração. Isso tende a prejudicar o processo de busca, tornando o algoritmo, de forma geral, ineficiente. Ao contrário do *NSGA-II*, no *NSGA-III*, pontos de referência adaptativos são criados para guiar o processo de identificação dos indivíduos não dominados que formarão a frente de Pareto. Esses pontos de referência

são criados para lidar com alguns impactos negativos típicos em problemas com muitos objetivos. Com a definição dos pontos de referência, a frente de Pareto será formada com o conjunto de soluções não dominadas que estão próximas a esta referência, que tem a capacidade de se atualizar de forma adaptativa. A Figura 10 apresenta o pseudocódigo do *NSGA-III.*

Figura 10 – Pseudocódigo do *NSGA-III*

```
Procedimento NSGA-III(N, G, fk(xk)):
     P ← GerarPopulaçãoAleatória(N)
     AvaliarPopulação(P)
     CalcularDistânciaDensidade(P)
     ConjuntosFronteira ← GerarConjuntosFronteira(P)
     Para i de 1 até G faça:
          PopulaçãoFilhos ← Vazia()
          Para cada pai e filho na população faça:
               AtribuirRanking(P)
               ConjuntosFronteira ← GerarConjuntosFronteira(P)
               Selecionados ← SelecionarPontosElitistas(ConjuntosFronteira)
               Filho ← RealizarRecombinaçãoEMutação(Selecionados)
               AdicionarFilho(PopulaçãoFilhos, Filho)

          PopulaçãoAmpliada ← MesclarPopulações(P, PopulaçãoFilhos)
          AvaliarPopulação(PopulaçãoAmpliada)
          CalcularDistânciaDensidade(PopulaçãoAmpliada)
          P ← SelecionarPopulação(PopulaçãoAmpliada, N)
          ConjuntosFronteira ← GerarConjuntosFronteira(P)

     Retornar MelhorSolução(P)
Fim do procedimento
```

Inspirado em Deb (2013)

3 Metodologia

Este capítulo propõe demonstrar de forma detalhada as técnicas e abordagens utilizadas na Vehicle Priority-Aware Dynamic Timeline Routing (*VPDTR*), uma nova abordagem para resolução do problema de roteamento de veículos com janelas de tempo e capacidade (*CVRPTW*).

Para validação da solução proposta, foi utilizado o *benchmark* de Solomon[1], um conjunto de dados muito utilizado na literatura para testar e validar soluções relacionadas ao *VRPTW*. Na primeira parte deste capítulo, são apresentados as instâncias utilizadas e como elas foram incorporadas ao algoritmo. Em seguida, são explicadas detalhadamente as etapas da heurística proposta, tais como: a heurística de criação da população inicial, o operador de mutação que atua como um algoritmo de reparação da solução, o operador de cruzamento otimizado baseado no *Order Crossover* e a função de avaliação adotada. Neste trabalho a seleção de sobreviventes é realizada com a implementação utilizada no *NSGA III* proposta por Deb (KALYANMOY DEB, 2001).

3.1 O Problema do Roteamento de Veículos com Restrição de Capacidade e Janelas de Tempo (CVRPTW)

O CVPRTW é um problema de otimização multiobjetivo complexo que busca minimizar a distância total de uma frota que tem como restrição as capacidades dos veículos e a janelas de atendimento dos clientes. E pode ser formulado a partir das seguintes equações:

$$min_{td} = \sum_{k=1}^{k} \sum_{i=0}^{n} \sum_{j=0}^{n} (X_{ij}^{k} C_{ij}) \quad (3.1)$$

$$X_{ii}^{k} = 0 (\forall i \in \{1, \ldots, n\}, \forall k \in \{1, \ldots, k\}) \quad (3.2)$$

$$X_{ij}^{k} \in \{0, 1\} (\forall i, j \in \{1, \ldots, n\}, \forall k \in \{1, \ldots, k\}) \quad (3.3)$$

$$\sum_{k=1}^{k} \sum_{i=1}^{n} X_{i,j}^{k} = 1 (\forall j \in \{2, \ldots, n\}) \quad (3.4)$$

[1] https://www.sintef.no/projectweb/top/vrptw/25-customers/

$$\sum_{i=1}^{n}\sum_{j=2}^{n} X_{i,j}^{k} d_j \leq Q^k (\forall k \in \{1,\ldots,k\}) \tag{3.5}$$

$$\sum_{k=1}^{n}\sum_{j=2}^{n} X_{1,j}^{k} \leq k \tag{3.6}$$

$$\sum_{j=2}^{n} X_{1,j}^{k} - \sum_{j=2}^{n} X_{j,1}^{k} = 0 (\forall k \in \{1,\ldots,k\}) \tag{3.7}$$

$$s_{ki} + C_{ij} - L\left(1 - X_{ij}^{k}\right) \leq s_{kj} (\forall i,j \in \{1,\ldots,N\}, \forall k \in \{1,\ldots,K\}) \tag{3.8}$$

$$a_j \leq s_{kj} \leq b_j (\forall i,j \in \{1,\ldots,N\}, \forall k \in \{1,\ldots,K\}) \tag{3.9}$$

A função objetivo é definida na equação (3.1), onde $C_{i,j}$ representa o custo de deslocamento do nó i para o nó j ($C_{i,j}$ é considerado como a distância para viajar do nó i para o nó j), k é o número máximo de veículos que podem ser utilizados, n é o número de clientes mais o depósito. A equação (3.1) impõe que um veículo não pode viajar de um cliente para o mesmo cliente. A equação (3.3) restringe o valor de $X_{i,j}^{k}$ a 0 ou 1; quando é definido como um, significa que o veículo k viaja do nó i para o nó j, caso contrário, é definido como 0. A equação (3.4) garante que um cliente seja visitado apenas uma vez. A equação (3.5) assegura que a soma das capacidades dos clientes visitados por um veículo não exceda a capacidade do veículo ($Q^k = Q$ para todos os veículos), onde Q^k é a capacidade de carga do veículo k (Q é igual para todos os veículos) e d_j é a demanda do cliente j. A equação (3.6) requer que um máximo de k rotas iniciem em um depósito. A equação (3.7) garante que todas as rotas iniciem e terminem no depósito. A equação (3.8) significa que, se o veículo k está viajando do cliente i para o cliente j, não podemos chegar ao cliente j antes de $S_{k,i} + C_{i,j}$, onde $S_{k,j}$ é o tempo de visita do cliente j pelo veículo k e L é um grande escalar ($L \geq \sum_{i=1}^{n}\sum_{j=1}^{n} C_{i,j}$) . Finalmente, a Equação (3.9) assegura que as janelas de tempo sejam cumpridas, onde a_j é o horário mais cedo para o cliente j permitir o serviço e b_j é o horário mais tardio para o cliente j permitir o serviço (GONZÁLEZ *et al.*, 2018).

3.2 Instâncias de Solomon

O *benchmark* de Solomon é um dos mais utilizados na literatura quando se trata do *VRPTW*, e conta com um conjunto de problemas organizados em 3 tipos de instâncias diferentes: as instâncias R, RC e C. As do tipo R são instâncias onde os dados são distribuídos geograficamente de forma esparsa, aleatória e sem agrupamentos; as do tipo RC são definidas como uma combinação de dados esparsos e agrupamentos; e nas instâncias do tipo C os dados são definidos como vários agrupamentos geograficamente distribuídos. Além disso, cada tipo de instância conta com duas variações: a variação 1, que define janelas de tempo hiper restritas, que permitem poucos clientes por rota, e a variação 2, com janelas de tempo mais relaxadas, o que permite abranger mais clientes por rota.

Segundo Solomon, as coordenadas dos clientes foram criadas de forma aleatória e são idênticas dentro de cada tipo de instância (R, RC e C), e o que as difere é a distribuição de janelas de tempo, que para cada cliente pode ser mais, ou menos, restrita. As instâncias ainda são distribuídas em relação a quantidade de clientes: 25, 50 e 100 clientes por instância. Para este trabalho, foram simuladas apenas as instâncias com 25 clientes. A Figura 11 demonstra a apresentação dos dados da instância R101.

Figura 11 – Exemplo de apresentação da instância R101

```
1   R101
2
3   VEHICLE
4   NUMBER     CAPACITY
5     25         200
6
7   CUSTOMER
8   CUST NO.   XCOORD.   YCOORD.   DEMAND   READY TIME   DUE DATE   SERVICE TIME
9
10      0         35        35         0          0         230          0
11      1         41        49        10        161         171         10
12      2         35        17         7         50          60         10
13      3         55        45        13        116         126         10
14      4         55        20        19        149         159         10
15      5         15        30        26         34          44         10
16      6         25        30         3         99         109         10
17      7         20        50         5         81          91         10
```

Elaborado pelo autor

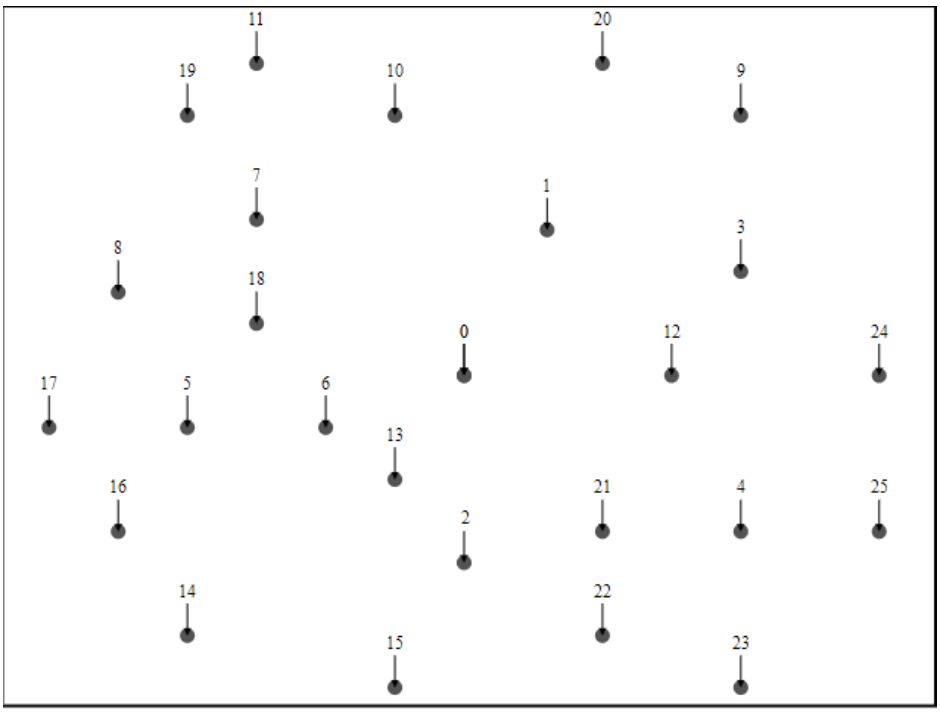
A Figura 11 demonstra um padrão de apresentação geral que se aplica a todas as instâncias. Na linha 1, temos a referência da instância, neste caso a instância do tipo R, variação 1 problema 01. Das linhas 3 a 5, tem-se o número máximo de veículos que podem ser utilizados para a solução (*vehicle number*) e a capacidade máxima de carga de cada veículo (*capacity*). Nas linhas 7 e 8 tem-se respectivamente: na coluna 1 (*cust no*), o número identificador de cada cliente, onde o 0 representa o depósito, e os identificadores

de 1 a *n* representam cada um dos clientes; na coluna 2, tem-se a coordenada do eixo X (*xcoord*); na coluna 3, a coordenada do eixo Y (*ycoord*); na coluna 4, a quantidade de recurso que será demandada do veículo (*demand*); na coluna 5, a janela de tempo inferior que representa o tempo de início do serviço (*ready time*); na coluna 6, a janela de tempo superior que representa o tempo limite de atendimento (*due date*); e na coluna 7, o tempo de serviço, que representa a duração de realização do serviço para cada cliente (*service time*).

3.2.1 Características das disposições geográficas dos clientes nas instâncias R, RC e C

Nas Figuras 12, 13 e 14 visualiza-se as características das disposições geográficas dos clientes, considerando as instâncias R, RC e C. A marcação em laranja denota o depósito e marca o ponto de partida inicial de todos os veículos, e os pontos azuis marcam cada um dos clientes que devem ser atendidos. A partir do depósito, cada veículo necessita construir uma rota para visitar um conjunto de clientes buscando otimizar a distância percorrida.

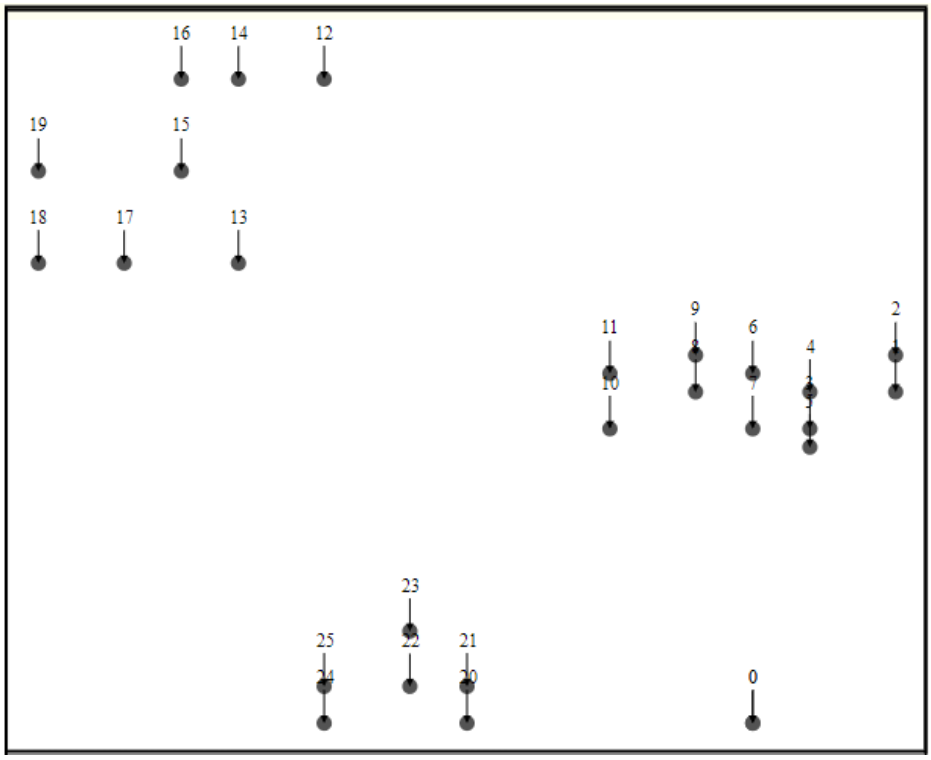
Figura 12 – Disposição geográfica dos clientes na instância R101



Elaborado pelo autor

Na Figura 13 observa-se que os clientes são espalhados ao longo do espaço geográfico de forma esparsa sem a formação de agrupamentos.

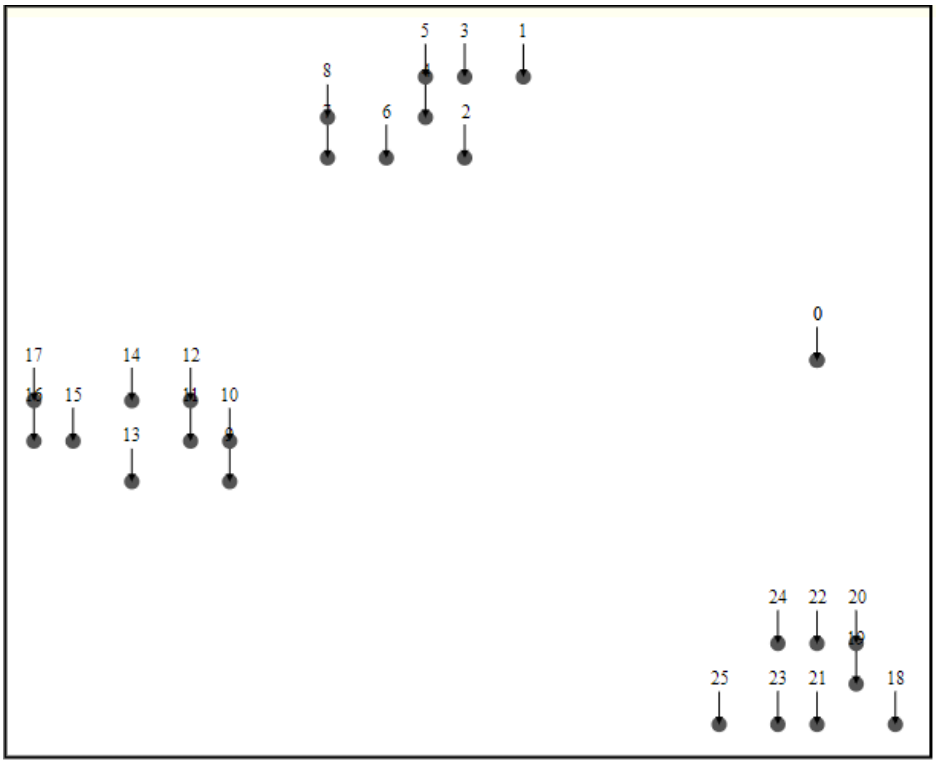
Figura 13 – Disposição geográfica dos clientes na instância RC101



Elaborado pelo autor

Já na Figura 13, observa-se que existe a combinação de clientes esparsos e alguns agrupamentos, criando uma paisagem levemente diferente da instância R.

Figura 14 – Disposição geográfica dos clientes na instância C101



Elaborado pelo autor

A instância C é formada apenas por aglomerados. Os clientes são dispostos em aglomerados, que estão espalhados pela paisagem.

De forma geral, as instâncias buscam reproduzir cenários com diferentes disposições geográficas para simular as diversas distribuições de clientes em cenários reais. Isso tende a forçar a construção de algoritmos com maior capacidade de abstração, ao invés da construção de heurísticas hiper especializadas, que conseguem gerar boas soluções para apenas um, ou outro, cenário, mas que não apresentam a mesma eficiência aplicada aos demais cenários. Como a disposição de clientes em situações reais são imprevisíveis e dinâmicas, um bom algoritmo deve ser capaz de encontrar soluções suficientemente boas em quaisquer um dos cenários. Neste sentido, este trabalho visa apresentar um algoritmo robusto que seja capaz de lidar com as dificuldades do *VRPTW* adaptando-se de forma eficiente às mais diversas disposições geográficas de clientes.

3.3 *Vehicle Priority-Aware Dynamic Timeline Routing - VPDTR* - Uma nova abordagem para resolução do *CVRPTW*

O presente trabalho propõe uma nova heurística para resolução do problema do roteamento de veículos com janelas de tempo e capacidade. O *CVRPTW* é usualmente abordado como um problema de minimização do custo total operacional. Neste trabalho, foram definidos três objetivos a serem minimizados: a distância total da rota, o número

de violações da janela de tempo final e a quantidade de veículos. Tanto a distância total da rota quanto o número de veículos devem ser os menores possíveis, mas o número de violações deve ser obrigatoriamente zero, dado que é uma restrição intrínseca ao problema. Embora o número de violações deva ser zero, essa métrica foi inserida como um objetivo para ampliar as possibilidades de soluções, evitando o excesso de restrições, que poderia levar ao descarte de soluções promissoras que ao longo do processo evolutivo pudessem ser reparadas. Assim, a ideia principal foi manter a possibilidade de violação para corrigir a solução ao longo do processo evolutivo.

A heurística *VPDTR-CVRPTW* é baseada no *NSGA-III* (JAIN; DEB, 2013), mas incorpora uma abordagem inovadora no processo de inicialização da população. Essa inovação introduz o conceito de *Prioridade*, uma medida relativa entre distância e tempo que fornece ao algoritmo uma forma eficiente e adaptativa para lidar com o processo de distribuição dos clientes na criação da população inicial. Essa abordagem objetiva, prioritariamente, deve mitigar a possibilidade do algoritmo gerar soluções inválidas. Além dessa inovação, *VPDTR-CVRPTW* também implementa um algoritmo de reparação como processo de mutação. Esse algoritmo visa corrigir os indivíduos que, porventura, violem as janelas de tempo, efetuando correções pontuais nas violações de cada solução. Com isso, estima-se que, ao longo das gerações, os indivíduos potencialmente eficientes possam se tornar soluções competitivas dentro do processo evolutivo. O processo de recombinação também foi otimizado e une características novas com o conhecido operador de recombinação *Order Crossover* (GOLDBERG, 1989). Esse novo operador possibilita a manutenção dos possíveis trechos de rotas eficientes, conservando as vizinhanças entre os nós clientes, além de ter a capacidade de se adaptar ao *VRPTW* que, por suas características particulares, não possibilita o uso tradicional do operador em questão.

3.3.1 Representação dos indivíduos

Os indivíduos são representados como múltiplas listas de números inteiros, onde cada uma das listas representa uma parte do cromossomo total. Esses cromossomos parciais podem apresentar tamanhos variáveis, mas o cromossomo total final terá sempre o mesmo tamanho, definido pela configuração da instância. Os valores inteiros representam o identificador de um cliente ou o ponto de partida. O identificador 0 foi utilizado para representar o ponto de partida, e os demais valores representam cada um dos clientes da respectiva instância. Na solução final, cada um dos subcromossomos representará um veículo, e a sequência do cromossomo representará a ordem da rota a ser percorrida. Por convenção será identificado como depósito o marcador 0. Esse marcador corresponde ao ponto de partida para todas as rotas nessa representação. Dessa forma, todas as rotas construídas devem partir do depósito e retornar a ele ao final. Na Figura 15 podemos observar um exemplo da representação de uma solução (indivíduo) para o problema.

Figura 15 – Exemplo de representação de um indivíduo

v1	0	1	7	13	19	24	0
v2	0	2	8	14	20	25	0
v3	0	3	0				
v4	0	4	10	16	22	0	
v5	0	5	11	17	0		
v6	0	6	0				

Elaborado pelo autor

Em resumo, cada veículo contém a ordem de visitação dos clientes, e essa ordem compõe sua rota. O conjunto de rotas define um indivíduo. As rotas podem variar de 3 até 27 pontos de visitação, uma vez que as instâncias utilizadas possuem 25 clientes, mais as respectivas menções aos depósitos no início e no fim. Então a menor rota possível para um veículo é sair do depósito, visitar um único cliente e retornar. E a rota máxima é uma solução com um único veículo, parte do depósito, visita os 25 clientes e retorna. Por se tratar de um problema permutacional, cada cliente somente pode figurar uma única vez dentro da solução. Ao partir do depósito, os veículos saem com capacidade máxima, e sempre que retornam ao depósito recuperam a capacidade total. Isso pode ocorrer ao final da rota, ou até mesmo entre os clientes, uma vez que o veículo necessite retornar ao depósito para recuperar sua capacidade total. Ao longo do trajeto, os veículos decrementam sua capacidade em função da demanda de cada cliente.

3.3.2 Compreendendo o conceito de *Prioridade*

A população inicial de um algoritmo que visa resolver o problema do roteamento de veículos pode ser um fator decisivo para identificar soluções com alto nível de qualidade. Neste trabalho, para construir a população inicial, parte dos indivíduos são gerados aleatoriamente, e parte é construída usando a heurística *Vehicle Priority-Aware Dynamic Timeline Routing*. A combinação entre geração aleatória e geração heurística pode reduzir significativamente o tempo de um algoritmo genético conseguir atingir mínimos locais razoáveis (GHOSEIRI; GHANNADPOUR, 2010) . A *VPDTR* constrói uma solução baseada no conceito de “Prioriade” que foi concebido neste trabalho. Considerando que um dos objetivos principais do *VRPTW* é minimizar a distância total percorrida, e que para isso é necessário que as janelas de tempo sejam respeitadas, podemos observar que há uma relação importante entre distância e tempo. Observando essa importância, foi adotada uma abordagem em que fosse possível relacionar as duas grandezas simultaneamente. A medida relativa que se extrai dessas grandezas é a velocidade, resultado da razão entre distância e tempo. Assim, a medida de Prioridade de um veículo para um determinado

cliente é dada pelo valor da velocidade necessária para alcançá-lo até o limite da sua janela de tempo final, considerando o tempo disponível.

Desta forma, quanto maior o valor da velocidade a ser empregada, mais urgente se torna esse cliente. Outros trabalhos utilizam a velocidade como variável de decisão, como, por exemplo, em Goeke (2015) que incorpora a velocidade dos veículos em funções de consumo realistas para calcular o custo total operacional utilizando frotas mistas entre veículos elétricos e veículos a combustão (GOEKE; SCHNEIDER, 2015). Zhao (2019) analisa como a velocidade dos veículos é afetada a partir dos atrasos nos tempos de partida, aplicada a um algoritmo bi objetivo dependente de tempo (ZHAO *et al.*, 2019). Já Kramer (2015) utiliza a velocidade como uma variável de decisão para resolver um *VRPTW* que busca minimizar o custo operacional e os impactos ambientais. Observando as publicações que consideram a medida de velocidade como variável de decisão, não foram identificados trabalhos que utilizam essa medida para definir a prioridade de atendimento dos clientes (KRAMER *et al.*, 2015).

A equação (3.1) descreve o cálculo de Prioridade de um determinado veículo para um cliente.

$$\beta_i = \begin{cases} \frac{d_{vi}}{ftw_i - ctt_v}, se(ftw_i - ctt_v) \geq 0 \\ -\infty, \ para\ os\ demais\ casos \end{cases} \tag{3.11}$$

Sendo a *Prioridade* βi a razão entre a distância d_{vi} de um $veculo_v$ até um $cliente_i$, pela diferença entre a janela de tempo final ftw do cliente i e o tempo atual de viagem ctt do veículo v. Essa diferença representa o tempo restante que o veículo v tem para alcançar um cliente i antes de ultrapassar o limite máximo de atendimento. O resultado da diferença somente será atribuído caso esse valor seja maior que zero. Do contrário, é atribuído o valor infinito negativo para a *Prioridade*, denotando que, da posição atual do veículo, o cliente alvo é inalcançável considerando o tempo disponível. O tempo atual de viagem (ctt_v) é definido com base no somatório do tempo decorrido do ponto de partida do veículo até o cliente atual c_n, como descrito a seguir (3.2).

$$c_{tt} = \sum_{i=0}^{k-1} tt_{(i,i+1)} + tempo\ de\ servico \tag{3.13}$$

Sendo o c_{tt} a representação do somatório do tempo de viagem tt do depósito dep, com $i = 0$, até o cliente k, passando por todos os clientes anteriores na rota.

3.3.3 Definindo a quantidade de veículos para uma solução

Para administrar as atribuições de clientes aos veículos, é necessário que uma quantidade de veículos seja selecionada. Para manter a característica estocástica inerente aos algoritmos genéticos, a quantidade de veículos de cada solução foi definida aleatoriamente com base na quantidade máxima permitida de veículos, segundo as definições das instâncias de Solomon. Para evitar que seja sorteada uma quantidade de veículos que impossibilite a construção de uma solução válida, uma etapa de cálculo de demanda total foi realizada para identificar limite inferior para a quantidade de veículos dada sua capacidade para atender determinada instância do problema. A equação (3.3) apresenta o cálculo da quantidade mínima necessária de veículos.

$$min_v = \left(\lfloor \left(\frac{\sum_{i=1}^{max_v} demanda_i}{max_{capacidade}} \right) \rfloor \right) + 1 \tag{3.15}$$

Então, o número mínimo de veículos min_v é obtido a partir do cálculo da parte inteira da razão entre a demanda total, $\sum_{i=1}^{max_v} demanda$, e a capacidade máxima permitida para cada veículo $max_{capacidade}$, acrescido de uma unidade. Assim podemos garantir que não haverá solução inexequível com uma quantidade de veículos menor que o mínimo necessário. Assim, a quantidade de veículos deverá ser maior ou igual ao limite inferior identificado, e menor ou igual ao limite superior definido pela instância.

3.3.4 Atribuição de clientes aos veículos

Uma vez definido o limite inferior de veículos necessários, é realizado um sorteio aleatório de probabilidade uniforme, considerando o intervalo entre a quantidade mínima e máxima de veículos. Esse sorteio define a quantidade de veículos da solução. Definida a quantidade de veículos, a etapa da primeira atribuição de clientes a cada veículo é realizada através de uma seleção aleatória com probabilidade inversamente proporcional a soma entre a distância do depósito até o cliente alvo e a janela de tempo inicial, conforme a equação

(3.5). Para definir as probabilidades, são considerados apenas os clientes alcançáveis (equação 3.4).

Um cliente somente é considerado alcançável, quando o tempo acumulado deslocamento para alcançá-lo não é superior a sua janela de tempo final. Para decidir qual cliente será atribuído ao veículo, é realizada uma seleção aleatória, considerando a probabilidade de seleção expressa pela equação (3.5), a partir da lista formada com os clientes de maior probabilidade. Neste caso, serão selecionados 10 clientes de maior probabilidade, quando essa lista possuir mais de 10 clientes, ou a lista contendo todos os clientes, quando essa lista possuir menos de dez clientes. A opção por dez clientes, ou menos, se dá para evitar que as probabilidades de seleção se diluam a tal ponto de se aproximarem demasiadamente e com isso se tornem quase uma probabilidade uniforme. Essa estratégia evitar que o algoritmo caia no problema de seleção da roleta, quando uma quantidade grande de elementos acaba fazendo com que a probabilidade se aproxime de uma probabilidade uniforme.

$$rc_i = \begin{cases} verdadeiro,\ se\ dist_{vi} \leq ftw_i \\ falso,\ para\ demais\ casos \end{cases} \tag{3.16}$$

$$P(i) = \begin{cases} \frac{1}{d_{vi}+stw_i}, se\ (3.4)\ \ verdadeiro \\ -\infty, demais\ casos \end{cases} \tag{3.17}$$

Calcular a probabilidade considerando distância e o limite inferior da janela de tempo traz a vantagem de privilegiar os clientes que estão mais próximos do depósito e possuem menor janela inicial. Quanto menor a janela inicial, menor será o tempo de espera do veículo para início do atendimento. Essa otimização proporciona um melhor tempo total de rota. O procedimento completo para identificar os dez clientes de maior probabilidade pode ser encontrado na Figura 16.

Figura 16 – Procedimento para identificar a probabilidade de seleção dos clientes

```
Procedimento encontre_clientes_candidatos(cliente_atual, clientes_remanescentes)
      criar lista vazia de clientes_alcançáveis
      criar uma lista vazia de clientes_candidatos
      para cada cliente_remanscente dentre clientes_remanescentes faça:
            calcular a distância do cliente_atual para o cliente_remanescente
            se distância < janela_tempo_superior_cliente_remanescente:
                  adicionar o cliente_remanescente a lista de clientes_alcançáveis
      fim para
      para cada cliente_candidato em clientes_alcançáveis faça:
            calcular a distância do cliente_atual para o cliente_candidato
            probabilidade de seleção ← 1/(distância + janela_inferior_candidato)
            adicionar o cliente e a probabilidade a lista de clientes_candidatos
      fim para
      ordenar a lista em ordem decrescente
      retorne os dez melhores clientes candidatos
fim procedimento
```

Elaborado pelo autor

Após a atribuição do cliente inicial, é necessário alocar os clientes restantes, os distribuindo entre os veículos. A forma mais tradicional de realizar essa etapa é criar sucessivas iterações para que para cada veículo, dado um critério previamente estabelecido, atenda um determinado cliente. Em problemas teóricos, essa pode ser uma abordagem viável, mas considerando problemas reais esse tipo de procedimento não é factível. Distribuir os clientes baseando-se apenas nas quantidade, de modo que cada um dos veículos atendam quantidades iguais de clientes pode levar a um problema de distorção temporal e especial. Se algum dos veículos é selecionado para atender os clientes mais distantes, sua rota será desproporcionalmente maior que os demais, necessitando de um tempo muito maior de serviço.Enquanto um cliente atende um conjunto de clientes e percorre pequenas distâncias, outro poderá percorrer grandes distância atendendo a mesma quantidade de clientes.

A Figura 17 demonstra uma simulação de uma situação em que esse tipo de distorção pode ser observado.

Figura 17 – Representação hipotética de uma rota distorcida

	Pontos percorridos	Distância em km	Tempo em horas
Veículo A	p0-p1	600	12
	p1-p4	50	1
	p4-p8	100	2
	p8-p9	150	3
		900	18

	Pontos percorridos	Distância em km	Tempo em horas
Veículo B	p0-p2	30	0,6
	p2-p5	25	0,5
	p5-p6	15	0,3
	p6-p7	60	1,2
		130	2,6

Elaborado pelo autor

No exemplo hipotético da Figura 17, é possível observar que ambos os veículos possuem a mesma quantidade de clientes visitados, mas com grande distorção em relação às variáveis de distância e tempo. Caso esse exemplo fosse aplicado ao mundo real, o veículo A teria percorrido uma distância total de 900 km gastando aproximadamente 18 horas. O veículo B teria percorrido 130 km em 2,6 horas. Tudo isso dentro de uma mesma jornada laboral. Essa distorção se dá porque ao atribuir indistintamente a mesma quantidade de clientes para todos os veículos, as distâncias já percorridas e o tempo gasto para percorrê-las não são levados em consideração e isso pode gerar um desbalanceamento das rotas individuais.

Outra abordagem possível é definir a quantidade de clientes por veículo de forma aleatória, usando uma probabilidade uniforme. Nos experimentos realizados, a distribuição aleatória não se mostrou eficaz. Constatou-se empiricamente que esse processo não representava uma abordagem promissora, pois por um lado permitia a mesma distorção temporal, e por outro, contava majoritariamente com a sorte para promover bons resultados. Ambas situações citadas não são capazes de lidar com problemas reais de forma eficiente, pois em situações aplicadas ao contexto real a diferença de tempo individual das rotas não deve variar significativamente, uma vez que os veículos possuem jornada de trabalho semelhantes.

Considerando os desafios logísticos reais, foi desenvolvido um procedimento para contornar os problemas das abordagens citadas. Nesse sentido, esse trabalho realiza a sincronização do tempo de deslocamento entre os veículos. Com o *tempo sincronizado* há determinado controle sobre quais veículos poderão ter atribuição de clientes a cada iteração baseado em um limiar temporal. Esse limiar é capaz de identificar que um determinado veículo ainda está em deslocamento e que não deve atender outros clientes naquela iteração. Então, um veículo que precisa atender um cliente distante, só deve atender outros clientes quando concluir o atendimento mais distante.

Tal qual o mundo real, os “relógios” dos veículos estão relativamente sincronizados. O termo mais preciso é “relativamente” sincronizado, pois, não há uma sincronização restrita

como no tempo cronológico, mas sim, uma referência que simula o tempo cronológico e não permite que um veículo atue de forma desproporcional, como exemplificado na Figura 17. O limiar pode ser obtido a partir das equações (3.18), (3.19) e (3.20).

$$ST(v) = \{ctt_{v1}, ctt_{v2}, ..., ctt_{vn}\} \tag{3.18}$$

$$MT(st) = min(ST(v)) \tag{3.19}$$

$$limiar = \begin{cases} MT + (2 * \sigma), \ se\ n(ST(v)) > 2 \\ MT + \sigma, \ para\ demais\ casos \end{cases} \tag{3.20}$$

Sendo $ST(v)$ o conjunto formado pelo tempo atual de viagem de cada um dos veículos de uma solução e $MT(st)$ é o valor mínimo do conjunto $ST(v)$. O $limiar$ indica o valor de referência que inclui ou exclui um determinado veículo da atribuição de clientes em uma iteração. Esse limiar é definido pelo ctt_v mínimo acrescido de dobro do desvio padrão , para soluções que contenham mais de dois veículos, e o ctt_v mínimo acrescido do desvio padrão para os demais casos. A cada iteração um novo limiar é calculado e define quais os veículos podem entrar na rodada de atribuição de clientes. Há uma diferença sutil na definição do limiar, que ocorre baseada na quantidade de veículos existentes na solução. Quando há mais de dois veículos, a expressão $MT + (2 * \sigma)$ garante que apenas os veículos que estejam dentro dessa margem de tempo participem da rodada de atribuições. Já quando há apenas dois veículos, considerar $MT + (2 * \sigma)$ faria com que os dois veículos sempre estivessem na rodada de atribuição, mesmo que estivessem temporalmente distantes. Assim, para os casos em que há apenas dois veículos na solução, a expressão $MT + \sigma$ garante que , quando um dos veículos estiver temporalmente distante, ele não entre na rodada de atribuição de clientes. Isso permite conservar o equilíbrio no tempo de trabalho de cada um dos veículos.

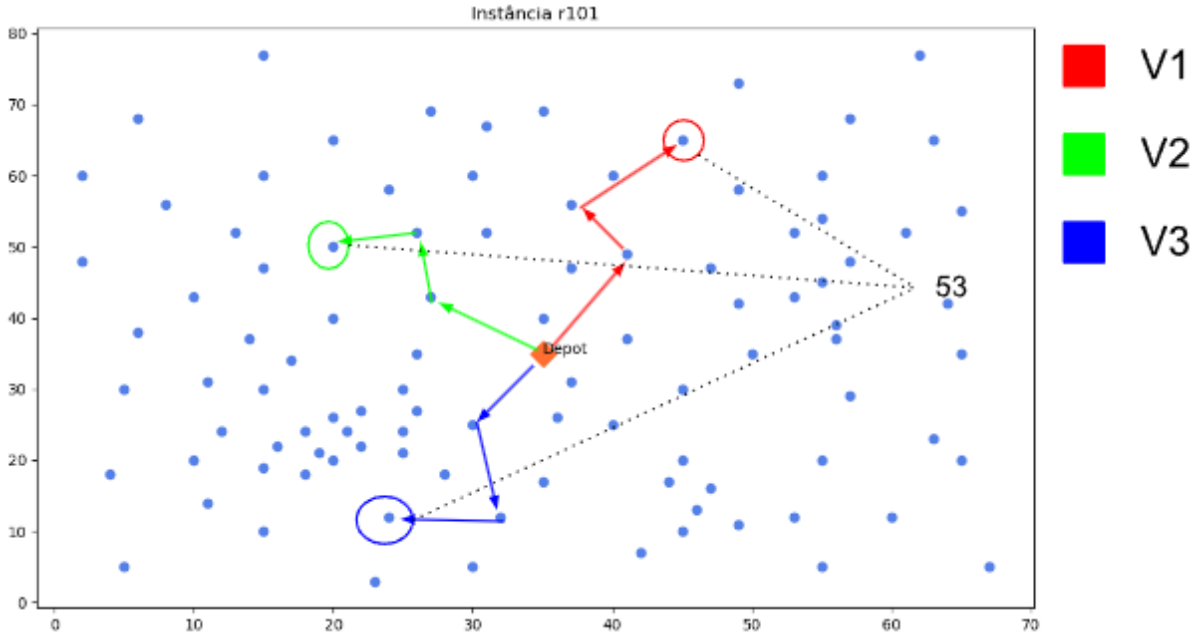
3.3.5 Utilizando *Prioridade* para definir o atendimento dos clientes

Uma vez definido o limiar da iteração e selecionados os veículos que poderão atender os clientes, é necessário monitorar os clientes com base na *Prioridade*, para que os veículos não se distanciem de algum cliente que de tal forma não seja mais possível atendê-lo. Então, a cada iteração, a *Prioridade* entre o veículo e todos os clientes não atendidos é calculada. A partir desse cálculo, o conjunto com os dez clientes mais urgentes é formado. Esse processo se repete para cada um dos veículos. Com a análise desses conjuntos de

clientes urgentes, é definido se algum cliente necessita prioridade no atendimento. O que faz com que um cliente tenha prioridade de atendimento é encontrá-lo como elemento do conjunto de todos os veículos simultaneamente.

O fato de um cliente figurar na lista de clientes urgentes de todos os veículos ao mesmo tempo, significa que os veículos estão se distanciando desse cliente e em breve não poderão mais conseguir atendê-lo, e isso invalidará toda a solução. A Figura 18 ilustra a situação em que os veículos se distanciam de um cliente.

Figura 18 – Priorizando um cliente com base na *Prioridade*



Elaborado pelo autor

Na simulação (Figura 18) é possível identificar três veículos que já atenderam três clientes cada um. Ocorre que o cliente de número 53 possui uma janela de atendimento restrita, e figurou a lista de clientes urgentes dos veículos v_1, v_2, v_3. Isso indica que na corrente iteração esse cliente deve ter prioridade de atendimento. E para atender esse cliente antes que o tempo limite seja ultrapassado, é designado o veículo mais próximo, que no exemplo da Figura 18 é o v_1. O procedimento completo de construção dos conjuntos de clientes urgentes é apresentado na Figura 19.

Figura 19 – Procedimento para identificar clientes urgentes

```
Procedimento encontre_clientes_urgentes(clientes_remanescentes, ponto_partida, tempo_atual_rota):
     Criar uma lista vazia de clientes_urgentes
     para cada cliente em clientes_remanescentes faça:
          calcular a distância entre o ponto_partida ao cliente
          tempo_restante ← cliente.janela_tempo_superior - tempo_atual_rota
          velocidade_de_urgência ← distância/tempo_restante caso tempo_restante > 1 senão -infinito
          adicionar o cliente e a velocidade_de_urgência a lista de clientes_urgentes
     fim para
     Ordenar a lista de clientes_urgentes de forma não decrescente
     se a lista de clientes_urgentes tem mais de dez elementos faça:
          retorne os dez clientes mais urgentes
     senão
          retorne a lista de clientes_urgentes
fim procedimento
```

Elaborado pelo autor

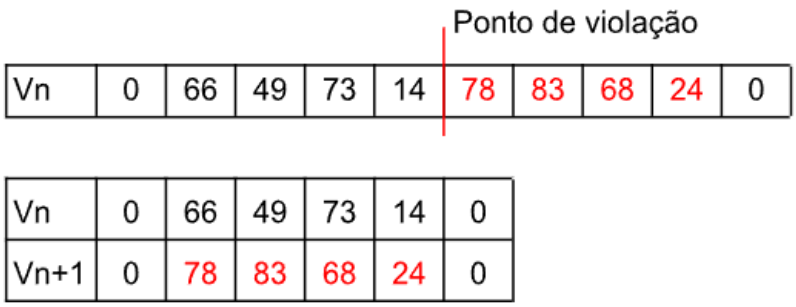
Assim, sempre que um cliente urgente é identificado, o seu atendimento é priorizado e procedido pelo veículo que se encontrar mais próximo. Porém, em muitos casos não existirá a figura do cliente urgente. Quer dizer que todos os clientes ainda estão em uma situação que não correm o risco de não serem atendidos. Então, não havendo clientes urgentes, outro procedimento é adotado. Para atender os clientes não urgentes, um *ranking* é criado com no máximo os quatro clientes mais próximos do veículo, esse *ranking* vai definir as probabilidades de cada um desses clientes serem escolhidos aleatoriamente. A probabilidade de cada um dos clientes serem escolhidos é dada pelo inverso de sua posição no *ranking*.

Para um *ranking* com quatro clientes, o primeiro tem 40% de chance de ser selecionado, o segundo 30%, o terceiro 20% e o último 10%. A partir daí um dos clientes é selecionado para ser atendido pelo veículo de referência.

3.3.6 Corrigindo as violações de capacidades

A heurística de criação dos indivíduos utiliza a Prioridade como forma de mitigar as violações de janelas de tempo, porém, não restringe o algoritmo quanto a possibilidade de violar as restrições de capacidade. Sempre que a capacidade máxima de um veículo é ultrapassada pelas demandas do conjunto de clientes que ele atende, gera-se uma penalização do ponto de vista da capacidade. Portanto, assim que um indivíduo é formado, um procedimento específico é executado para eliminar as violações de capacidade. Cada veículo que viola a capacidade é dividido em dois ou mais veículos conforme a necessidade, eliminando suas violações de carga.

Figura 20 – Correção das violações de capacidade de um veículo



Elaborado pelo autor

A Figura 20 demonstra o procedimento de correção de violação de capacidade em um veículo k. Encontrado o ponto onde a capacidade máxima do veículo é ultrapassada, cria-se um ponto de divisão, e a partir dele outro veículo é criado com os clientes remanescentes. Nesse exemplo, ao ultrapassar do cliente 14 para o cliente 78 houve uma violação, então o veículo k passa a atender até o cliente 14, e os demais são atribuídos a um veículo $k + 1$. Esse procedimento é repetido para todos os veículos, incluindo-se os veículos recém criados, para garantir que não haja nenhuma violação de capacidade.

3.3.7 Heurística de criação da população inicial

O procedimento completo de criação do indivíduo é apresentado no pseudocódigo da Figura 21.

Figura 21 – Pseudocódigo da heurística de criação dos indivíduos

```
Procedimento criação_de_individuo_baseado_na_urgência(instância):
      carregar os dados da instância
      calcular a demanda total para instância
      calcular o mínimo de veículos necessários para atender a demanda
      escolher o número de veículos de forma aleatória uniforme considerando o mínimo e o máximo de veículos
      para cada veículo  dentro de veículos faça:
            recuperar os clientes candidatos a partir da probabilidade da distância e janela de tempo
            selecionar um cliente aleatoriamente a partir dos clientes candidatos
            Cris um veículo e adicionar o cliente selecionado
            excluir o clientes da lista de clientes
      fim para
      enquanto houver clientes para atender faça:
            criar um conjunto com os tempos_atuais de viagem para todos os veículos
            calcular o mínimo e o desvio padrão do conjunto criado
            calcular o limiar
            filtrar os veículos a partir do limiar
            criar os conjuntos de clientes_candidatos a partir da urgência por veículo
            se há intersecção entre os clientes candidatos faça:
                  para cada cliente urgente na lista de clientes_urgentes faça:
                        adicionar o cliente prioritário ao veículo mais próximo
                  fim para
            senão faça:
                  para cada veículo dentro dos veículos atuais faça:
                        calcular a distância entre o cliente atual para todos os clientes remanescentes
                        criar um ranking com os quatro clientes mais próximos
                        selecionar um cliente aleatoriamente a partir do inverso da sua posição dentro do ranking
                        adicionar o cliente ao veículo
                  fim para
            fim enquanto
            corrigir os veículos que violam as restrições de capacidade
fim procedimento
```

Elaborado pelo autor

Para cada um dos veículos é necessário definir a probabilidade de atendimento de cada um dos clientes ainda não atendidos. Estabelecendo a lista com os 10 clientes de maior probabilidade de atendimento para o veículo, uma seleção aleatória é realizada com base na probabilidade de atendimento de cada dos 10 clientes listados, e atribui esse cliente ao veículo. Esse processo se repete para cada um dos veículos, até que todos tenham definido o primeiro cliente que deverá se atendido.

Após a atribuição do primeiro cliente, os demais clientes de cada um dos veículos é realizado a partir da probabilidade de urgência. O tempo total de viagem de cada um dos veículos é contabilizado. O limiar de atendimento é definido, e apenas os clientes que possuem tempo total de viagem menor que o limiar irão participar da rodada de atribuição de clientes.Após identificar quais os veículos poderão participar da rodada de atribuições, serão identificados os níveis de urgência partindo de cada um dos veículos para todos os demais clientes não atendidos. Cada veículo possui um conjunto de clientes e seus respectivos níveis de urgência. Esses conjuntos são avaliados e é identificado se há alguma cliente que pertença a intersecção de todos os conjuntos. Caso haja um cliente, esse cliente é considerado urgente e é atendido pelo veículo que estiver mais próximo. Caso não haja intersecção, um cliente é selecionado aleatoriamente para cada um dos veículos conforme

a probabilidade contabilizada pela distância e a janela de tempo inicial.

3.3.8 Abordagens de recombinação

Os operadores de recombinação são extremamente importantes, pois proporcionam o processo de explotação do espaço de busca. A exploração mais efetiva do espaço de busca possibilita um maior grau de diversidade na população e mitiga as possibilidades do algoritmo convergir precocemente para mínimos ou máximos locais, a depender do objetivo do problema. Embora o *VRPTW* ainda seja uma generalização do problema do caixeiro viajante (*TSP*), sua estrutura base não permite que os operadores permutacionais comuns possam ser utilizados da mesma forma que são aplicados no *TSP*.

O cromossomo no *VRPTW* é representado por múltiplas listas de pequenos pedaços do cromossomo, e não apenas uma lista contendo o cromossomo inteiro. Como no *VRP*, um operador tradicional não consegue realizar o processo de recombinação sem que seja necessário algum grau de customização, é muito comum que as abordagens de resolução do *VRP* se utilizem de processos de recombinação otimizados ou customizados para lidar com as características específicas de cada problema.

Srivastava (2021) utilizou o *NSGA II* para resolver o problema do roteamento de veículos com janelas de tempo implementando operadores de cruzamento e mutação projetados especificamente para o problema. O operador de recombinação desenvolvido pelo autor atua diretamente relacionado a cada objetivo e não ao problema como um todo. O procedimento é dividido em cinco fases, onde em cada uma delas a solução é avaliada sobre aspectos diferentes relacionados aos objetivos (SRIVASTAVA *et al.*, 2021). Men (2020) resolve o *VRPTW* utilizando um algoritmo evolucionário híbrido com um operador de cruzamento de troca de rotas (*Route Exchange Crossover*). Nesse operador, a melhor sequência de genes em uma rota é compartilhada com outros indivíduos na população, a partir da avaliação dos bons trechos do cromossomo, que se dá conforme o critério do risco de transporte médio da rota (MEN *et al.*, 2020). Já Yao (2017) utiliza um algoritmo de colônia de formigas que é otimizado a partir de um operador de cruzamento especializado. Nesse operador, duas rotas são selecionadas aleatoriamente, e em cada uma delas um nó é selecionado, também aleatoriamente. Após selecionar os dois nós, ambos são trocados de lugar, gerando novas rotas (YAO *et al.*, 2017).

Pratama (2017) aplica um *OX Crossover* tradicional para proceder o processo de recombinação usando um algoritmo genético. Na abordagem adotada pelo autor, é possível utilizar a versão tradicional do operador uma vez que as rotas serão realizadas por único veículo (PRATAMA; MAHMUDY, 2017). Xu (2015) utiliza um algoritmo genético combinado a um algoritmo de enxame de partículas para solucionar o *VRPTW*. O operador de cruzamento é integrado ao algoritmo para prevenir a convergência prematura e evitar que o algoritmo fique preso a mínimos locais. O operador é projetado para atuar sobre

a velocidade e a posição das partículas. O indivíduo filho gerado pelo operador somente substituirá o indivíduo pai caso sua aptidão seja melhor que aptidão do pai (XU *et al.*, 2015). As possibilidades de operadores de cruzamento são inúmeras e a cada abordagem, ou contexto em que o operador é aplicado, uma necessidade específica pode surgir fazendo com que um novo operador seja desenvolvido ou um operador já conhecido seja otimizado ou adaptado.

3.3.8.1 Algoritmo de recombinação otimizado

As abordagens de recombinação utilizadas neste trabalho são uma otimização do consolidado *OX Crossover* (Order Crossover) proposto por GOLDBERG (1989) e um novo operador desenvolvido com o intuito de explorar novas soluções a partir da união de trechos de cromossomos que contenham um único alelo para formar um novo trecho de cromossomo válido. O operador de cruzamento *OX* otimizado atua em quatro fases.

Na primeira fase, ocorre um processo de planificação do indivíduo pai. Como os indivíduos são representados como um conjunto de listas contendo trechos menores do cromossomo, com os veículos representados pelas listas, e os clientes representados pelos elementos dessas listas, é necessário transformar esse cromossomo em uma única lista, para poder aplicar o *OX Crossover* tradicional. Nesse processo também é removida a referência ao depósito de todos os veículos, pois a referência do depósito não deve ser considerada na construção das rotas. A Figura 22 demonstra o processo de planificação do indivíduo

Figura 22 – Processo de planificação dos indivíduos

i1	0	66	49	73	14	0			
i2	0	78	83	68	24	57	0		
Planificado	66	49	73	14	78	83	68	24	57

Elaborado pelo autor

Após o processo de planificação, a fase dois embaralha aleatoriamente os indivíduos planificados para possibilitar o surgimento de indivíduos mais promissores.

Figura 23 – Processo de embaralhamento dos indivíduos

planificado	66	49	73	14	78	83	68	24	57
embaralhado	73	57	24	78	49	68	83	66	14

Elaborado pelo autor

Embaralhada a ordem de atendimento, a terceira fase consiste na aplicação do *OX* tradicional. Nessa fase cada um dos indivíduos pai irá gerar um indivíduo filho que é uma recombinação do cromossomo dos indivíduos pais (Figura 24).

Figura 24 – Exemplo hipotético do cruzamento

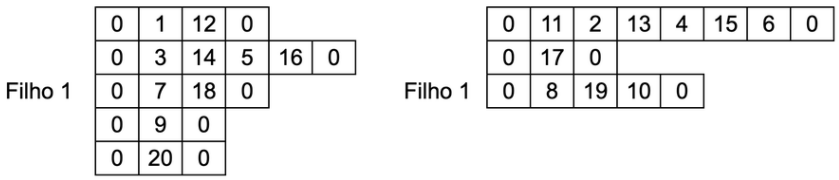
Pai 1	1	2	3	4	5	6	7	8	9	10
Pai 2	11	12	13	14	15	16	17	18	19	20

Filho 1	1	12	3	14	5	16	7	18	9	20
Filho 2	11	2	13	4	15	6	17	8	19	10

Elaborado pelo autor

Na quarta e última etapa, os indivíduos são reconstruídos e têm seu formato restabelecido. Para recriar os indivíduos, os clientes são atribuídos aos veículos partindo do primeiro até que seja identificado que irá haver uma violação de alguma restrição, seja de capacidade ou janela de tempo.

Figura 25 – Reconstrução hipotética dos indivíduos



Elaborado pelo autor

Assim, sempre que for identificado que irá ocorrer uma violação, um novo veículo é criado. Esse processo ocorre até que todos os clientes estejam alocados em algum dos veículos. Os procedimentos do *OX* e operador de cruzamento otimizado são apresentados nas figuras 26 e 27.

Figura 26 – Pseudocódigo do *order* crossover

```
Procedimento order_crossover(pai1, pai2):
      tamanho t ← tmax(pai1, pai2)
      filho1 ← criar uma nova lista vazia de tamanho t
      filho2 ← criar uma nova lista vazia de tamanho t
      posição1 ← número inteiro aleatório entre 0 e t-1
      posição2 ← número inteiro aleatório entre 0 e t-1
      início ← min(posição1, posição2)
      fim ← max(posição1, posição2)
      para i do início ao fim faça:
            filho1[i] ← pai1[i]
            filho2[i] ← pai2[i]
      fim para
      j ← fim + 1
      k ← início + 1
      enquanto j ≠ início faça
            se k ⩾ t então k ← 0
            se filho1 não contém pai2[k] então
                  filho1[j] ← pai2[k]
                  j ← j + 1
            se filho2 nãocontém  pai1[k] então
                  filho2[j] ← pai1[k]
                  j ← j + 1
            k ← k + 1
      fim enquanto
      retorne filho1, filho2
fim procedimento
```

Inspirado em GOLDBERG (1989)

Figura 27 – Pseudocódigo do algoritmo de cruzamento otimizado

```
Procedimento cruzamento(paiA, paiB):
      criar cópias de ambos os pais
      planificar ambas as cópias dos pais
      embaralhar aleatoriamente as cópias dos pais
      aplicar o OX às cópias dos pais para gerar dois filhos
      para cada filho na lista de filhos faça:
            enquanto houver clientes no filho faça:
                  se o cliente candidato a atribuição viola alguma restrição então:
                        criar um novo veículo e adicione o cliente
                  senão:
                        adicionar o cliente ao veículo atual
            fim enquanto
      fim para
      retorne ambos os filhos
fim procedimento
```

Elaborado pelo autor

Já o novo cruzamento proposto neste trabalho, denominado *cruzamento por agregação*, consiste em tentar reduzir o número total de veículos, um dos objetos globais do problema. Esse operador busca unir os trechos de cromossomos muito pequenos (veículos que atendem apenas um cliente) para com isso otimizar a solução com um número menor de veículos. Esse operador possui probabilidade de atuação de 20% e, quanto utilizado, somente realiza o procedimento nos casos em que haja veículos atendendo apenas um cliente. Do contrário, apenas retorna uma cópia dos indivíduos pais. A Figura apresenta o pseudocódigo do *cruzamento por agregação*.

Figura 28 – Pseudocódigo do algoritmo de *cruzamento_por_agregação*

```
Procedimento cruzamento_por_agregação(inda, indb):
    lista_de_individuos ← [inda, indb]
    para cada individuo em lista_de_individuos faça:
        clientes_para_agregaçao ← []
        para cada veículo no indivíduo atual faça:
            se o veículo possui apenas um cliente faça:
                adiciona o cliente a clientes_para_agregação
                exclui o veículo do indivíduo atual
            fim se
        fim para
        se clientes_para_agregação possui clientes faça:
            adicione as referências ao depósito
            adicione o veículo novamente ao indivíduo
        fim se
    retorne inda, indb
fim procedimento
```

Elaborado pelo autor

Com esse procedimento, é possível reduzir o número total de veículos da solução unindo os veículos que atendem apenas um cliente. Essa redução do número total de veículos, tanto otimiza o número de veículos, como pode contribuir para redução da distância total da rota. Essa estratégia foi desenvolvida em harmonia com os operadores de mutação, que, para resolver as violações relacionadas ao tempo limite de atendimento, podem ter a necessidade de criar novos veículos. Então, para prevenir que a otimização de um objetivo não deteriore o outro, o operador foi implementado.

3.3.9 Abordagens de mutação

Os processos de mutação, ou variação, objetivam proceder pequenas alterações no indivíduo com o intuito de realizar uma busca local. A busca local é a avaliação do espaço de busca em regiões próximas a este indivíduo, que pretende identificar indivíduos com melhores aptidões nestas regiões. O processo de variação soma-se ao processo de recombinação para conseguir explorar o máximo possível o espaço de busca. Além de operadores tradicionais, muitos trabalhos desenvolvem operadores de mutação com foco na reparação. Esse tipo de operador, usualmente, utiliza conhecimentos específicos do domínio para promover melhoria nas soluções através de correções pontuais. Essas correções buscam ao longo do processo evolucionário, considerando algoritmos evolutivos, transformar soluções inviáveis em soluções viáveis.

Srivastava (2021) utiliza um operador de mutação com a estratégia de destruição e

reconstrução. Nessa estratégia, a solução é parcialmente destruída e reconstruída utilizando uma abordagem baseada nos objetivos específicos, que utiliza uma junção adequada entre uma abordagem aleatória e gulosa (SRIVASTAVA *et al.*, 2021). Wang (2020) introduz a ideia de um operador de mutação adaptativo para o *NSGA II* aplicados ao reperfilamento de rodas para o sistema ferroviário. Esse operador evita a convergência prematura e, à medida que o processo evolutivo avança (WANG *et al.*, 2020).

Buba (2018) introduz um mecanismo de reparo reverso em sub-rotas para corrigir rotas inviáveis em problemas de roteamento em trânsitos urbanos (BUBA; LEE, 2018) . Zein (2017) utiliza algoritmos genéticos para projetar estruturas compostas, modelando as espessuras e percentuais de orientação das fibras nas zonas das estruturas. Um operador de reparação é utilizado para corrigir o empilhamento das fibras e garantir que exista um empilhamento admissível conforme as regras de fabricação (ZEIN *et al.*, 2017).

3.3.9.1 Algoritmo de mutação como operador de reparação

Um algoritmo de reparação, como já visto, é um procedimento que utiliza de algum conhecimento do processo ou domínio, e tem por principal objetivo promover uma melhoria na solução em que atua (VAIRA; KURASOVA, 2014). Muitos dos operadores de mutação mais tradicionais atuam realizando pequenas alterações nos indivíduos buscando encontrar melhorias para uma solução, dentro do espaço de busca. Não há necessariamente um operador de mutação padrão para problemas de roteamento de veículos. Como um problema permutacional, o *VRP* pode utilizar operadores específicos para problemas permutacionais, mas de forma adaptada. O *VRPTW* possui uma ampla gama de aplicações e por isso incorpora a complexidade das áreas em que são utilizados como base para a resolução de problemas. Considerando que o *VRPTW* é um problema amplamente estudado e reconhecidamente difícil, este trabalho propõe um operador de reparação capaz de corrigir as violações de janelas de tempo.

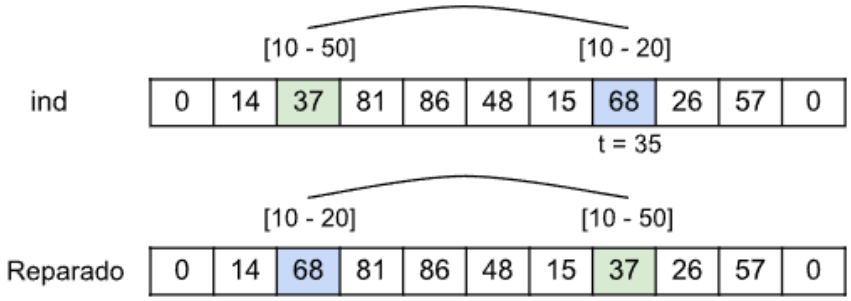
Essas violações normalmente ocorrem na construção das soluções, pois o processo evolutivo foi construído sem restringir violações de janela de tempo. Esse procedimento foi permitido para que o processo evolucionário pudesse explorar o espaço de soluções de uma forma mais livre. O objetivo é definir um processo que pudesse, ao longo do tempo, ir corrigindo as violações que porventura existissem nos indivíduos. Uma violação de janela de tempo se dá quando um veículo chega ao cliente para o atendimento após a janela limite de atendimento deste cliente. Algumas situações específicas podem levar às violações. Esse cliente pode ter uma janela de atendimento muito restrita, ou até mesmo, o cliente pode ser incompatível com a rota em que aparece. A posição em que aparece na rota também pode ser incompatível com os limites de sua janela temporal.

Em qualquer caso, para que essa violação seja corrigida, é necessário que a janela de atendimento fosse maior. É justamente nesse ponto que o operador atua. O operador

visa localizar um outro cliente que possua uma janela de atendimento maior, e realiza uma troca de posição entre os dois clientes, alterando assim a ordem de atendimento. Um outro cliente é procurado dentro do mesmo veículo para realizar a troca. Esse processo se dá dentro do mesmo veículo para que não haja um impacto significativo na rota total e para que a estrutura atual do indivíduo não sofra grande impacto, mantendo a ideia de um operador de mutação: pequenas transformações com intuito de proporcionar buscas locais.

O primeiro processo de tentativa de reparação se dá em duas etapas: a primeira é uma correção que busca um cliente que possua uma janela de atendimento maior que a do cliente que está violando, mas que esteja localizado em posições anteriores; a segunda etapa, realiza um processo semelhante, mas buscando nos clientes à frente do que está violando. A Figura 29 apresenta visualmente o processo de reparação por permuta.

Figura 29 – Processo de reparação por permuta



Elaborado pelo autor

Na Figura 29, o cliente 68 tem sua janela de tempo superior violada. O período máximo para atendê-lo são às 20 unidades de tempo. A chegada a este clientes se deu no instante de tempo 35, logo isso viola a restrição de janela de tempo. Porém, o cliente 37 possui uma janela com maior folga, variando de 10 a 50 unidades. Sendo assim, o cliente 37 é um candidato a realizar uma permuta com o cliente 68. E essa permuta acontece, pois este cliente foi o primeiro cliente que atende ao critério de substituição. Alternando os dois de posição, a violação está resolvida. Dentro do processo evolutivo não há garantias que esse processo irá sempre resolver o problema. Haverá situações em que mesmo realizando a troca, a violação permaneça; mas ao longo das gerações a tendência é que somente as soluções que conseguem ser reparadas permaneçam no processo. Já o processo de reparação para frente segue a mesma lógica, buscando clientes posteriores para realização da permuta, quando não foi possível reparar olhando os clientes antecedentes. O processo de mutação visa corrigir violações relacionadas a janelas de tempo, mas essa correção pode gerar um impacto na distância da rota. Esse impacto poderá ser positivo ou negativo, de forma que naturalmente o processo evolutivo vá descartando as soluções mais deterioradas. A Figura 30 apresenta o algoritmo de reparação por permuta e a FIgura 31 aprensenta o algoritmo de reparação para frente.

Figura 30 – Pseudocódigo de reparação por permuta

```
Procedimento reparar_por_permuta(indivíduo, veículo, cliente):
      encontrado ← falso
      i ← índice em que ocorre a violação
      enquanto encontrado == falso faça:
            se i < 2 então:
                  reparar_para_frente(indivíduo, veículo, cliente)
            fim se
            se cliente[i].janela_final < cliente[i-1].janela_final:
                  cliente[i] troca de posição com cliente[i-1]
                  encontrado = verdadeiro
            fim se
            i=i-1
      fim enquanto
      retorne indivíduo reparado
fim procedimento
```

Elaborado pelo autor

Figura 31 – Pseudocódigo de reparação para frente

```
Procedimento reparar_para_frente(indivíduo, veículo, cliente):
      encontrado ← falso
      i ← índice onde ocorre a violação
      enquanto encontrado == falso faça:
            se i == veículo.tamanho - 2 então:
                  retorne indivíduo pois não há como reparar
            fim se
            se cliente[i].janela_final < cliente[i+1].janela_final:
                  cliente[i] troca de posição com cliente[i-1]
                  encontrado = verdadeiro
            fim se
            i=i+1
      fim enquanto
      retorne indivíduo reparado
fim procedimento
```

Elaborado pelo autor

Já o segundo processo de reparação acontece com a análise dos clientes em que está ocorrendo violação da restrição de janela temporal. Para eliminar essas violações, o operador realiza a remoção daqueles clientes que estão violando, e os realoca para outros

veículos, onde eles não violem a restrição. Porém, análises empíricas mostraram que esse operador só é efetivo à medida que haja poucas violações na solução, pois ao não conseguir realocar um cliente em nenhum outro veículo, o operador cria um novo veículo para realizar a alocação. Quando esse operador é aplicado no início do processo evolutivo, ele tende a deteriorar muito as soluções. Nos estágios iniciais do processo evolutivo, as soluções tendem a ter um grande número de violações, e isso aumenta a probabilidade de muitos clientes não conseguirem ser realocados. Dessa forma, o operador pode acabar criando um grande número de veículos que atendem apenas um cliente, e isso gera soluções com uma qualidade baixa.

O operador trabalha de forma simples e pode ser detalhado como segue: cada um dos veículos é visitado para identificar em quais veículos e em quais clientes ocorreu violação. Nos veículos em que ocorrem violações, o primeiro cliente que está violando é removido e armazenado em uma lista auxiliar. O processo é repetido dessa forma, retirando sempre o primeiro cliente que viola a cada iteração, até que não haja mais nenhuma violação em nenhum veículo. Em seguida, a partir da lista auxiliar em que estão armazenados os clientes que foram removidos, o algoritmo tenta, um a um, alocá-los em outra posição em que eles possam se encaixar sem provocar violação. Isso pode ocorrer em outro veículo, ou até mesmo no próprio veículo do qual o cliente saiu, mas em outra posição. O cliente é alocado na primeira posição em que ele possa ser inserido sem violar nenhuma restrição.

Após tentar realocar todos os clientes, duas situações podem ocorrer: no primeiro caso, todos os clientes foram realocados, e a solução foi completamente reparada. No segundo caso, alguns clientes não puderam ser realocados, porque nenhum ponto da solução permitia o encaixe sem violar as restrições. Nesse caso, um novo veículo é criado com todos os clientes remanescentes não alocados. Neste novo veículo a ordem de alocação não é observada, e os clientes remanescentes apenas são inseridos nele da mesma forma em que estavam armazenados na lista auxiliar. A partir disso, esse novo indivíduo é integrado novamente à população. A Figura 32 mostra o algoritmo de reparação por realocação.

Figura 32 – Pseudocódigo do algoritmo de reparação por realocação

```
Procedimento reparação_por_realocação(indivíduo):
    se há violação faça:
        removidos ← []
        enquanto houver violações faça:
            remova o primeiro cliente que viola em cada veículo
        para cada cliente_removido na lista removidos faça:
            mudança ← falso
            para cada veículo do indivíduo faça:
                para cliente no veículo faça:
                    insira o cliente_removido na posição atual do veículo
                    verifique se o veículo ainda apresenta violações
                    se não há violação então:
                        mudança ← verdadeiro
                        interrompa
                    senão:
                        desfaça a operação
                se houve mudança então:
                    interrompa
        se ainda há clientes na lista removidos faça:
            aloque os clientes remanescentes em novo veículo
        retorne o indivíduo reparado
    senão:
        retorne uma cópia do indivíduo
fim procedimento
```

Elaborado pelo autor

Como já referido anteriormente, esse operador não é indicado para ser utilizado no início do processo evolucionário. Porém, à medida que o processo avança e as soluções possuem poucas violações, esse operador se demonstrou altamente eficiente. Então, o processo de mutação somente utiliza esse operador quando a solução possui menos de dez violações. Não é possível afirmar que dez seja o valor ideal, mas diante das simulações e das análises empíricas, identificou-se que esse valor alcançou resultados efetivos. Com até dez violações os indivíduos tendiam a ser reparados de forma eficiente, o que se confirmou com os resultados obtidos pelo algoritmo geral. Como já mencionado, a utilização desse operador precocemente trazia o efeito colateral negativo de gerar um excesso de veículos com apenas um cliente, o que degrada, consideravelmente, a solução final. Além disso, em todas as simulações executadas sem a utilização do operador de reparação por realocação, o algoritmo genético não conseguiu convergir para soluções válidas, o que indica que esse operador contribui significativamente com o processo evolutivo.

Um procedimento específico condensa os algoritmos das Figuras 30, 31 e 32, em uma única função de mutação geral. A Figura 33 apresenta o pseudocódigo do algoritmo

de mutação.

Figura 33 – Pseudocódigo da mutação

```
Procedimento mutação(indivíduo):
      se há violações faça:
            se há menos de 10 violações faça:
                  retorne reparação_por_realocação(indivíduo)
            senão
                  retorne reparação_por_permuta(indivíduo)
      senão:
            retorne uma cópia do indivíduo
fim procedimento
```

Elaborado pelo autor

4 Simulações

4.1 Informações gerais sobre as simulações

Para validar a eficiência da abordagem *VPDTR*, foi utilizado o *benchmark* de Solomon, que é uma das referências mais utilizadas para problemas de roteamento de veículos com restrição de janelas de tempo. Nele estão contidos os melhores resultados encontrados por pesquisadores de todo o mundo[1]. Cada um dos resultados encontrados na plataforma é considerado o melhor resultado já obtido até o momento. Portanto, os resultados disponibilizados representam o estado da arte em relação aos problemas propostos por Solomon (SOLOMON, 1987). Para realizar as simulações, foram utilizados todos os 58 problemas para as instâncias com 25 clientes. Os resultados estão compilados no capítulo 5 e serão discutidos posteriormente.

A heurística *VPDTR* é baseada no algoritmo genético multiobjetivo *NSGA-III*. O *NSGA-III* foi adotado como algoritmo base para implementação da heurística por ser uma versão otimizada do *NSGA-II* e ser extremamente eficiente quando utilizado em problemas com muitos objetivos. Embora a heurística lide com 3 objetivos neste momento, a adoção do *NSGA-III* permite que a heurística esteja preparada para trabalhar em *CVRPTW* com quatro ou mais objetivos com poucos ajustes de implementação.

Para validação dos resultados, foram aplicados, a cada um dos problemas das instâncias do tipo $R1$ *e* $R2$; $RC1$ *e* $RC2$; $C1$ *e* $C2$, 5 execuções utilizando os parâmetros apresentados na Figura 34:

Figura 34 – Parâmetros gerais do *NSGA-III*

Número de gerações: 6000
Tamanho da população: 700
Taxa de mutação: 100%
Taxa de cruzamento: 100%
Pontos de referência do hiperplano: 18

Elaborado pelo autor

Um dos fatores mais difíceis na utilização de algoritmos genéticos é a configuração dos parâmetros de inicialização. Alguns trabalhos tentam otimizar e identificar a configuração de parâmetros ideal para determinados tipos de problemas. Angelova (2011) analisa o ajuste de parâmetros de algoritmo genético aplicado a problemas de processo de fermentação. O autor argumenta que o número de gerações foi o parâmetro que mais exerceu influência sobre o tempo de convergência do algoritmo (ANGELOVA *et al.*, 2011) . Nisrina (2022) identifica que não existe um valor exato para cada parâmetro do AG, mas identifica que algumas combinações entre tamanho de população, taxa de mutação, taxa de cruzamento

[1] https://www.sintef.no/projectweb/top/vrptw/solomon-benchmark/

e número de gerações são mais eficientes quando aplicados ao contexto do problema do caixeiro viajante para problemas logísticos de grande dimensão (NISRINA *et al.*, 2022).

Já Herdiana (2022) aponta que uma abordagem adaptativa dos valores dos parâmetros trouxe bons resultados no que diz respeito a melhora da aptidão, mas uma piora na eficiência relacionada ao tempo de execução na solução do problema do caixeiro viajante. Dos quatro parâmetros analisados, o autor aponta que a taxa de mutação foi o que menos influenciou no resultado final da aptidão (HERDIANA *et al.*, 2022). Assim, é possível perceber que a definição dos valores dos parâmetros dos algoritmos genéticos é tarefa não trivial. Obter configurações reconhecidamente ótimas dependeria de um extenso estudo sobre o tema, o que não é o objetivo deste trabalho. Os valores definidos para os parâmetros foram obtidos através de diversos testes e análises empíricas do comportamento dos algoritmos considerando os resultados obtidos.

Embora não seja uma abordagem comum, a utilização dos operadores de mutação e cruzamento com taxas em 100% apresentou-se como a composição paramétrica mais eficiente. Essa taxa de mutação e cruazamento diz respeito a probabilidade de um indivíduo ser selecionado para o processo de variação. Dessa maneira, todos os indivíduos passam necessariamente pelo processo de variação. Como a heurística deixa o algoritmo livre para promover violações do ponto de vista da janela de tempo, é necessário que o processo de reparação seja eficiente e constante, do contrário o algoritmo não converge e não alcança soluções válidas. Uma solução válida é aquela que não viola nenhuma das restrições do problema.

Para a abordagem desenvolvida neste trabalho, o algoritmo só começou a apresentar resultados positivos, convergindo para soluções válidas, a partir do momento em que os parâmetros e os operadores foram ajustados na configuração apontada na Figura 34. Ao final de cada simulação, com a parametrização ajustada conforme a Figura 34, a heurística proposta consegue produzir de 1 a 5 soluções válidas, um resultado que pode parecer um número relativamente pequeno, mas que no contexto complexo do problema se mostra um resultado bastante relevante. Para chegar a esses parâmetros, foram testados valores que variaram de 4000 a 12000 gerações, populações de 200 a 1800 indivíduos, taxas de mutação e cruzamento de variaram de 1 a 100% e pontos de hiper planos variando de 12 a 24. Para observar o nível de dificuldade do problema, em testes realizados a partir da instância R101 com 25 clientes, foram executadas cinco rodadas de simulação, gerando 100.000 soluções completamente aleatórias em cada rodada. Das 500.000 soluções geradas, nenhuma solução válida foi encontrada. Isso pode indicar que de forma aleatória, a probabilidade de produzir uma solução válida é pequena ou improvável.

Teste semelhante foi realizado utilizando apenas o algoritmo de geração da população inicial da heurística *VPDTR* que, no mesmo cenário, gerou apenas três soluções válidas em um total de 500.000 soluções, o que representa uma probabilidade de 0,0006% de chance de gerar uma solução válida utilizando apenas o algoritmo de geração da popu-

lação inicial, sem considerar os operadores e o processo evolucionário. Embora seja uma probabilidade pequena, comparada com uma abordagem aleatória, a heurística demonstra ser uma solução mais eficiente. Ao final, é possível observar que, o que proporciona, de fato, os resultados efetivos é a combinação desse algoritmo dentro do contexto evolutivo, que permite que as soluções sejam reparadas durante o processo evolucionário. Nesse caso, as soluções inválidas, mas promissoras, ainda têm a oportunidade de serem reparadas e apresentarem bons resultados. Além disso, diversas outras configurações de parâmetros também foram testadas e avaliadas. Ainda assim, os resultados não foram tão promissores quanto com a configuração descrita na Figura 34.

4.2 Ambiente de execução

Todas as simulações foram executadas em um computador Macbook air M1 ano 2020, equipado com o chip apple M1, CPU de 8 núcleos, memória integrada de 8 GB e 512 GB de armazenamento do tipo SSD, utilizado com o sistema operacional macOS Ventura na versão 14.4.

5 Resultados e discussão

5.1 Apresentação dos resultados

O presente capítulo apresenta os resultados obtidos com a *VPDTR*. Para fins de validação da abordagem desenvolvida, os resultados são comparados com o *benchmark* de Solomon. O *benchmark* reúne os melhores resultados identificados, até o presente momento, por diversos pesquisadores do mundo inteiro[1], e representam o estado da arte para os problemas propostos por Solomon. As tabelas de resultados deste Capítulo apresentam uma comparação entre os resultados obtidos neste trabalho e a referência do *benchmark* para cada um dos 58 problemas. As soluções são comparadas a partir da distância total da rota e o número total de veículos de cada solução. O objetivo final do problema é minimizar as duas métricas, indicando que quanto menores os valores, maior qualidade tem a solução.

5.1.1 Resultados para as instâncias do tipo R1

Como mencionado na Seção 3.2.1, as instâncias do tipo R são aquelas em que a distribuição dos clientes é concebida de forma aleatória e com janelas de tempo hiper restritas. Para este tipo de instância, nenhuma regra específica de distribuição de clientes foi aplicada, diferente das demais instâncias em que existem regras que definem a forma de distribuir os clientes dentro do espaço geográfico. Possuir janelas de tempo hiper restritas é uma característica que torna a identificação das soluções eficientes uma tarefa consideravelmente mais difícil, se comparado aos problemas com janelas de tempo mais flexíveis. Com janelas de tempo muito restritas, as rotas apresentam pouca possibilidade de variação, pois, nesses cenários, as violações ocorrem com muito mais facilidade. Nas análises empíricas, foi identificado que, com frequência, um determinado indivíduo chegava a um estágio onde não havia mais possibilidade de reparação, e consequentemente este indivíduo acabava por ser descartado durante o processo evolutivo.

Gerar soluções incorrigíveis não é um problema por si só. Os algoritmos genéticos lidam bem com esse tipo de problema, uma vez que eliminam esses indivíduos e os substituem durante o processo evolucionário. Mas quanto menor a flexibilidade das janelas, menor a probabilidade de construir boas soluções. A tabela da Figura 35 apresenta a compilação dos resultados obtidos para os 12 problemas da instância R1 e a comparação entre os resultados publicados no *benchmark* de Solomon.

[1] https://www.sintef.no/projectweb/top/vrptw/solomon-benchmark/

Figura 35 – Resultados para as instâncias R1

Instâncias R1	Solomon *Benchmark*		Heurística VPDTR		Análises	
	Menor distância conhecida	Número total de veículos	Menor distância encontrada	Número total de veículos	Resultado comparativo entre as distâncias	Resultado comparativo entre a quantidade de veículos
r101	617,10	8	618,32	8	0,20%	0,00%
r102	547,10	7	562,73	7	2,86%	0,00%
r103	454,60	4	476,77	4	4,88%	0,00%
r104	416,90	4	440,83	4	5,74%	0,00%
r105	530,50	6	531,53	6	0,19%	0,00%
	530,50	6	556,71	5	4,94%	-16,67%
r106	465,40	3	470,33	5	1,06%	66,67%
r107	424,30	4	438,75	4	3,41%	0,00%
r108	397,30	4	427,67	4	7,64%	0,00%
r109	441,30	5	456,01	5	3,33%	0,00%
	441,30	5	460,52	4	4,36%	-20,00%
r110	444,10	4	449,82	4	1,29%	0,00%
r111	428,80	5	442,47	4	3,19%	-20,00%
r112	393,00	4	418,65	4	6,53%	0,00%

Elaborado pelo autor

A tabela Figura 35 exibe, na coluna 1, o identificador do problema, nas colunas 2 e 3 exibe os resultados do *benchmark*, nas colunas 3 e 4 exibe os resultados identificados pela abordagem *VPDTR* e nas 5 e 6 exibe os comparativos e melhorias entre a abordagem proposta e os resultados publicados. Na tabela da Figura 35 é possível identificar que os resultados obtidos pela *VPDTR* são competitivos, sobretudo, considerando que os resultados apresentados no *benchmark* é um compilado de diversos autores, registrando-se para cada problema apenas o melhor resultado geral obtido.

Para os problemas R105, R109 e R111, a *VPDTR* identificou soluções mais eficientes em relação ao tamanho da frota, conseguindo frotas com, respectivamente, 16,67%, 20% e 20% menos veículos que os resultados consolidados. Assim, os resultados para esses três problemas apresentam soluções não dominadas, ou seja, não há possibilidade de apontar qual das soluções é mais eficiente, de forma geral, sem aplicar as soluções a um contexto específico. A exceção do problema R106, os demais problemas apresentam frotas de tamanho igual ou inferior aos resultados comparados, o que pode apontar que, relacionado aos tamanhos das frotas, a *VPDTR* consegue identificar soluções altamente eficientes.

Os problemas R105 e R109 apresentam dois resultados para o mesmo problema, no primeiro apresenta um resultado mais expressivo no tamanho total da rota com a mesma quantidade de veículos do problema de referência, no segundo o tamanho total da rota é menos eficiente, porém a solução obtida pela heurística é mais otimizada que a solução do problema de referência em relação ao tamanho da frota. Em relação às distâncias totais das rotas, a *VPDTR* se demonstrou extremamente eficiente e, embora não tenha identificado uma solução com a menor distância total, foi bastante consistente nos resultados. A partir dos resultados apresentados, é possível identificar que a heurística proposta possui satisfatória na resolução dos problemas, dada a pequena variação entre os percentuais de distâncias totais comparadas.

5.1.2 Resultados para as instâncias do tipo R2

As instâncias do tipo R2, assim como a R1, tem uma disposição geográfica aleatória. Porém, diferem das instâncias R1 em relação às janelas de tempo, que nesta variação apresenta janelas do tipo flexíveis. Por janelas de tempo flexíveis, pode-se entender janelas nas quais os clientes possuem maior disponibilidade de tempo de atendimento. Aqui, como as janelas são mais amplas, há uma maior possibilidade de variar a ordem de atendimento dos clientes, podendo explorar de forma mais eficiente o espaço de soluções e consequentemente obter menos violações. A tabela da Figura 36 apresenta os resultados e análise comparativa para os 11 problemas para a instância R2.

Figura 36 – Resultados para as instâncias R2

Instâncias R2	Solomon *Benchmark*		Heurística VPDTR		Análises	
	Menor distância conhecida	Número total de veículos	Menor distância encontrada	Número total de veículos	Resultado comparativo entre as distâncias	Resultado comparativo entre a quantidade de veículos
r201	463,30	4	534,50	2	15,37%	-50,00%
r202	410,50	4	504,68	2	22,94%	-50,00%
r203	391,40	3	476,14	2	21,65%	-33,33%
r204	355,00	2	450,12	1	26,79%	-50,00%
r205	393,00	3	425,20	2	8,19%	-33,33%
r206	374,40	3	444,64	2	18,76%	-33,33%
r207	361,60	3	409,53	1	13,25%	-66,67%
r208	328,20	1	351,25	1	7,02%	0,00%
r209	370,70	2	418,25	1	12,83%	-50,00%
r210	404,60	3	484,71	2	19,80%	-33,33%
r211	350,90	2	415,85	1	18,51%	-50,00%

Elaborado pelo autor

Diferente da instância do tipo R1, os resultados obtidos para a instância R2 apontam que a *VPDTR* aplicada a problemas de janelas flexíveis, possui uma capacidade maior de obter soluções mais eficientes em relação ao tamanho da frota. A *VPDTR* obteve resultados melhores, em relação ao tamanho da frota, que os demais resultados consolidados no *benchmark*, a exceção do problema R208, que por possuir apenas um veículo na frota não possibilita melhoria, e mesmo sendo menos eficiente do ponto de vista da distância total, apresenta soluções não dominadas para 10 dos 11 problemas analisados. As melhorias nos tamanhos das frotas vão de 33,33% a 66,67% de redução na quantidade de veículos utilizados.

Esses resultados são expressivos, pois, em determinadas situações, a redução dos custos com o tamanho da frota possibilita que empresas de menor porte atuem de forma competitiva no mercado. Com uma drástica redução da frota, é possível reduzir de forma significativa parte dos custos fixos de um negócio. Nesse sentido, a *VPDTR* aplicada a cenários de maior flexibilidade de janelas de atendimento pode obter resultados extremamente eficientes, podendo ser altamente atraente para negócios logísticos que possuam maior parte dos custos fixos concentrados no tamanho da frota. Os resultados apresentados na Figura 36, mais uma vez, apontam para uma consistência da heurística

VPDTR, tanto no que diz respeito às distâncias totais, quanto para o tamanho da frota, uma vez que ela apresentou resultados eficientes e robustos ao longo de todos os problemas. É possível notar também que, quando os resultados não são mais eficientes no tamanho da frota de um problema, eles tendem a se aproximar mais dos resultados ótimos em relação às distâncias totais. Esses resultados também se refletem nos resultados obtidos na instância R1.

5.1.3 Resultados para as instâncias do tipo RC1

Com uma distribuição levemente diferente das instâncias R, nas instâncias do tipo RC, os clientes são distribuídos tanto de forma aleatória, quanto por agrupamentos (Figura 13). Esse tipo de distribuição forma uma paisagem que simula uma disposição esparsa, com alguns locais de maior concentração de clientes. Nesse cenário, é possível explorar a capacidade do algoritmo em se adaptar a disposições irregulares, situações que podem ser facilmente encontradas em ambientes de problemas reais. Além disso, os problemas da instância RC1 apresentam as características inerentes aos problemas do tipo 1, que são as janelas de tempo hiper restritas. Outro fator relevante a observar é que, por possuir as limitações temporais de atendimento, uma abordagem algorítmica gulosa, por exemplo, pode não ser eficiente, uma vez que esse tipo de abordagem tem melhor adaptação quando considera variáveis de decisão bem definidas. Por exemplo, ao considerar apenas a distância até os clientes mais próximos, desconsiderar as janelas de tempo pode levar a soluções inválidas, violando o tempo limite de atendimento, e ao considerar apenas as janelas de tempo, há prejuízo nas distâncias totais da rota.

Então, embora existam estudos relacionados a algoritmos gulosos multi-objetivos, como em (NAKAI *et al.*, 2022), onde o autor utilizou um algoritmo guloso multi-objetivo baseado em soluções não dominadas de Pareto para resolver um problema de seleção de sensores, obtendo resultados equivalentes ou superiores a abordagens gulosas puras ou de grupo; é mais comum que as estratégias gulosas sejam aplicadas em etapas específicas, ou processos especializados, do que aplicados como uma abordagem mais generalista, como demonstra (CHANG *et al.*, 2014) utilizando um algoritmo genético multi-objetivo baseado em busca gulosa aplicado a distribuição eficiente de recursos a vítimas de desastres , ou em (LU *et al.*, 2022), onde o autor apresentou um algoritmo híbrido baseado em Pareto com iterações gulosas aplicado para o problema de escalonamento de fluxo híbrido em ambientes de produção. Então, cabe notar que para problemas com características relacionadas a instância do tipo RC, a abordagem precisa manter uma relação equilibrada em abordagens mistas, do tipo aleatória e de aglomerados. A tabela da Figura 37 consolida os resultados obtidos para os oito problemas da instância RC1.

Figura 37 – Resultados para as instâncias RC1

Instâncias RC1	Solomon *Benchmark*		Heurística VPDTR		Análises	
	Menor distância conhecida	Número total de veículos	Menor distância encontrada	Número total de veículos	Resultado comparativo entre as distâncias	Resultado comparativo entre a quantidade de veículos
rc101	461,10	4	462,15	4	0,23%	0,00%
rc102	351,80	3	355,08	3	0,93%	0,00%
rc103	332,80	3	360,42	3	8,30%	0,00%
rc104	306,60	3	338,72	3	10,48%	0,00%
rc105	411,30	4	412,37	4	0,26%	0,00%
rc106	345,50	3	353,46	3	2,30%	0,00%
rc107	298,30	3	325,21	3	9,02%	0,00%
rc108	294,50	3	342,33	3	16,24%	0,00%

Elaborado pelo autor

É possível observar na Figura 37 que a *VPDTR* atinge um desempenho satisfatório, obtendo resultados muito competitivos e próximos dos resultados ótimos conhecidos. Para a maioria dos problemas, a abordagem se aproxima dos valores ótimos, mantendo a quantidade mínima de veículos. A exceção dos problemas RC104, RC107 e RC108, os demais resultados estão extremamente próximos dos resultados ótimos, embora os problemas RC104 e RC107 ainda permaneçam dentro da margem de 10% acima dos valores ótimos. Apenas o problema RC108 apresenta um resultado menos eficiente, destoando dos resultados médios obtidos. Os motivos pelos quais a *VPDTR* não encontrou uma solução aproximada para o problema RC108 não foram identificados. Uma hipótese, é que o resultado apresentado pode ser o resultado ótimo, ou muito próximo e que seja extremamente difícil de ser obtido, uma vez que os demais resultados se apresentaram consistentes ao longo de todos os problemas.

5.1.4 Resultados para as instâncias do tipo RC2

As instâncias RC2 possuem a mesma disposição geográfica das instâncias RC1, mas com às características de janelas temporais flexíveis, como nos demais problemas do tipo 2. Assim, o que difere as duas instâncias é que problemas do tipo 2 permitem rotas com mais clientes, possibilitando assim que as soluções sejam construídas com menos veículos. A tabela da Figura 38 consolida os resultados obtidos para os oito problemas da instância RC2.

Figura 38 – Resultados para as instâncias RC2

Instâncias RC2	Solomon *Benchmark*		Heurística VPDTR		Análises	
	Menor distância conhecida	Número total de veículos	Menor distância encontrada	Número total de veículos	Resultado comparativo entre as distâncias	Resultado comparativo entre a quantidade de veículos
rc201	360,20	3	467,95	2	29,91%	-33,33%
rc202	338,00	3	441,28	2	30,56%	-33,33%
rc203	326,90	3	395,08	2	20,86%	-33,33%
rc204	299,70	3	429,39	1	43,27%	-66,67%
rc205	338,00	3	435,37	2	28,81%	-33,33%
rc206	324,00	3	408,62	2	26,12%	-33,33%
rc207	298,30	3	454,98	2	52,52%	-33,33%
rc208	269,10	2	358,41	1	33,19%	-50,00%

Elaborado pelo autor

Os resultados apontados na tabela da Figura 38 evidenciam um comportamento já identificado em outros resultados, como os apresentados na tabela da Figura 36. Para a instância RC2, a *VPDTR* é mais eficiente que os melhores resultados conhecidos, em relação ao tamanho da frota, quando os problemas possuem janelas temporais mais relaxadas. Assim, para todos os problemas da instância RC2, a abordagem proposta apresentou resultados mais eficientes, reduzindo o número de veículos de cada um dos problemas, como é possível observar. Com reduções de variaram de 33,33% a 66,67% menos veículos, a *VPDTR* identificou soluções com frotas altamente otimizadas. Mesmo assim, não é possível afirmar qual das soluções é mais eficiente para cada problema, uma vez que as soluções são não dominadas entre si.

Os dois conjuntos de soluções podem atuar de forma eficiente em cenários distintos, e em cenários mais propensos a menor capacidade de investimento em frotas, a *VPDTR* demonstrou ser mais eficiente que o conjunto de soluções apresentado no benchmark. Em alguns cenários, uma diminuição de 66,67% no tamanho da frota pode representar uma redução drástica nos custos operacionais, sobretudo com a possibilidade de redução dos custos fixos, a exemplo da quantidade de veículos, manutenções, contratação de mão de obra especializada, combustível, seguros etc. Portanto, tanto em cenários de disposição aleatória de clientes, quanto num cenário híbrido (aleatório + agrupamentos), a *VPDTR* demonstra alta capacidade de generalização, resolvendo os problemas de forma eficiente em ambos cenários, sem a necessidade de qualquer ajuste ou adaptação específica.

5.1.5 Resultados para as instâncias do tipo C1

Dentre todos os conjuntos de problemas, os relacionados às instâncias do tipo C são os mais particulares. Nesse tipo de instância, os clientes são dispostos sob a forma de aglomerados de clientes distribuídos em pontos distintos do espaço geográfico, como é possível observar na Figura 14. À primeira vista, esse tipo de configuração pode induzir à aplicação de técnicas mais especializadas nesse tipo de distribuição, como algoritmos de aglomeração baseados em centróides, *k-means*; ou algoritmo de aglomeração baseados

em densidade, como o *DBSCAN*; ou algoritmos de aglomeração Gaussiana, como o *Gaussian Mixture Model algorithm* que permite aglomerados de contornos não circulares, lidando com um dos problemas do *k-means*, que permite apenas aglomerados circulares; ou até *Affinity Propagation clustering algorithm*, algoritmo que diferente dos demais, constrói os aglomerados por afinidade ou similaridade e não requer parâmetros de inicialização contendo quantidade nem tamanhos dos aglomerados.

Contudo, como já mencionado na seção 5.1.4, as janelas de tempo competem com as distâncias como variáveis de decisão, e nem sempre é possível conectar todos os pontos de um mesmo aglomerado, dado que, a depender da sequência de clientes, a rota pode se tornar inaplicável. Além disso, nos cenários onde todos os clientes são estritamente dispostos em aglomerados bem definidos, como na Figura 14, não é exatamente factível, e, portanto o uso puro de algoritmo de aglomeração possui menor capacidade de abstração, quando comparado a outras técnicas de resolução do *VRP*. Ainda assim, alguns testes com o algoritmo *x-means*, que identifica aglomerados sem a necessidade da indicação da quantidade de grupos, foram realizados, para tentar encontrar aglomerados promissores e definir uma quantidade de veículos mais adequada. Contudo, para a identificação eficiente de um aglomerado, era necessário realizar um cálculo de densidade do aglomerado e definir um limiar que pudesse indicar se o aglomerado identificado era de fato um aglomerado coeso e denso.

Portanto, o aumento da complexidade dos cálculos relacionados à identificação dos aglomerados e às verificações de densidade não se mostraram eficientes e promissores a ponto de justificar a compensação da complexidade inserida no problema, pois as janelas de tempo, sobretudo as hiper restritas, não permitem, para todos os casos, a formação de aglomerados bem definidos, e isso pode depreciar as soluções geradas por essas técnicas. A Tabela 39 consolida os resultados obtidos para os nove problemas da instância C1.

Figura 39 – Resultados para as instâncias C1

Instâncias C1	Solomon *Benchmark*		Heurística VPDTR		Análises	
	Menor distância conhecida	Número total de veículos	Menor distância encontrada	Número total de veículos	Resultado comparativo entre as distâncias	Resultado comparativo entre a quantidade de veículos
c101	191,30	3	191,81	3	0,27%	0,00%
c102	190,30	3	192,09	3	0,94%	0,00%
c103	190,30	3	227,02	4	19,30%	33,33%
c104	186,90	3	225,00	4	20,39%	33,33%
c105	191,30	3	226,93	4	18,63%	33,33%
c106	191,30	3	191,81	3	0,27%	0,00%
c107	191,30	3	216,03	4	12,93%	33,33%
c108	191,30	3	225,56	4	17,91%	33,33%
c109	191,30	3	268,73	3	40,48%	0,00%

Elaborado pelo autor

Dentre todos os resultados, os relacionados à instância C1 foram os menos eficientes, como é possível verificar na tabela da Figura 39. Como os problemas para esta instância são restritos a conjuntos de aglomerados, é possível que uma das hipóteses da diminuição

da eficiência da *VPDTR* se deva ao fato da heurística possuir uma característica mais generalista que especialista em cenários desta natureza. É possível identificar ainda que os problemas C101, C102 e C106 obtiveram resultados próximos do ótimo, mas os demais problemas tiveram resultados menos expressivos, inclusive com uma degradação das soluções no que diz respeito aos tamanhos das frotas.

Outro fato que é importante notar é a consistência dos resultados do *benchmark*, que apresenta resultados com um valor exato e igual para vários problemas. Uma hipótese a ser levantada é que esses valores sejam de fato os valores ótimos, e dessa forma são valores mais complexos de serem alcançados.

5.1.6 Resultados para as instâncias do tipo C2

Ao avaliar os resultados das instâncias C2, é possível identificar que o mesmo padrão referente aos problemas com janelas de tempo flexíveis se confirma. Para quase todos os problemas, há uma redução no tamanho da frota. A tabela da Figura 40 consolida os resultados obtidos para os nove problemas da instância C2.

Figura 40 – Resultados para as instâncias C2

Instâncias C2	Solomon *Benchmark*		Heurística VPDTR		Análises	
	Menor distância conhecida	Número total de veículos	Menor distância encontrada	Número total de veículos	Resultado comparativo entre as distâncias	Resultado comparativo entre a quantidade de veículos
c201	214,70	2	215,54	2	0,39%	0,00%
c202	214,70	2	231,88	1	8,00%	-50,00%
c203	214,70	2	233,07	1	8,56%	-50,00%
c204	213,10	2	247,71	1	16,24%	-50,00%
c205	214,70	2	299,69	1	39,59%	-50,00%
c206	214,70	2	299,92	1	39,69%	-50,00%
c207	214,50	2	226,37	2	5,53%	0,00%
	214,50	2	279,58	1	30,34%	-50,00%
c208	214,50	2	253,96	1	18,40%	-50,00%

Elaborado pelo autor

A exceção do resultado do problema C201, todos os resultados da *VPDTR* apresentados na Tabela 40 são mais eficientes que as soluções de referência. Esses resultados são consistentes ao longo de todas as instâncias cujas janelas de tempo são mais relaxadas. Para estes problemas as reduções foram de 50%, o que representa, mais uma vez, uma diminuição significativa nos custos de operação, objetivo maior do *VRP*. O problema C207 apresenta duas soluções e corrobora novamente a hipótese de que a heurística proposta é mais eficiente que as soluções de referência quando lida com janelas de tempo relaxadas. Por outro lado, o mesmo problema C207, quando identifica uma solução com mesmo tamanho de frota, se aproxima bem mais do valor ótimo de referência.

Além disso, os resultados das instâncias C202, C203 e C207 ainda encontram valores relativamente próximos ao valor ótimo, mesmo com a drástica redução da frota, o que significa que mesmo com uma performance menos eficiente sobre as distâncias totais

em problemas de janelas relaxadas, a abordagem ainda identifica soluções que podem ser extremamente eficientes nas duas frentes, tanto com uma distância total da rota eficiente, quanto com uma frota altamente otimizada.

6 Conclusão

De forma geral, a proposta apresentada neste trabalho teve comportamento consistente e bastante eficiente. Ao longo dos problemas tratados, foi possível demonstrar a alta capacidade da VPDTR em identificar soluções com frotas extremamente otimizadas em todos os cenários trabalhados. Outro ponto que é possível inferir dos resultados, é o fato da abordagem proposta conseguir uma generalização eficiente nos três tipos de cenários simulados, demonstrando que além da consistência, possui uma boa flexibilidade diante de cenários tão distintos.

Um ponto extremamente importante para destacar é que os resultados fornecidos pelo *benchmark* reúnem as melhores soluções conhecidas, porém, esses resultados são provenientes de diversos estudos e autores diferentes. Sendo assim, alguns estudos podem ter foco em um tipo de instância, obtendo resultaos menos eficientes nas demais. Enquanto que a *VPDTR* consegue transitar de forma eficiente e competitiva por todos os cenários com a mesma implementação. Por exemplo, em relação aos problemas com janelas de tempo hiper restritas, é possível destacar o bom desempenho da heurística, com a identificação de resultados competitivos, se comparados aos melhores resultados conhecidos. Já em relação à otimização da frota, a *VPDTR* é notadamente mais eficiente que quase todos os resultados com os quais foi comparada.

Além disso, a partir da análise da evolução dos resultados durante o processo de concepção teórica e implementação, foi possível observar que o conceito de Prioridade, elaborado neste trabalho, contribuiu significativamente na construção de soluções promissoras, permitindo a geração de indivíduos muito mais eficientes que em uma abordagem aleatória. Estabelecer essa medida relativa entre distância e tempo, denominada *Prioridade*, previne que na construção das rotas, os veículos se afastem demasiadamente de algum cliente a ponto de não conseguir mais atendê-lo. Isso mitiga a possibilidade de criar soluções com um alto número de violações, o que tornaria a capacidade de reparação dos operadores genéticos menos eficiente.

Outro ponto que pode ser destacado é o algoritmo de reparação como operador de mutação. Com uma abordagem que permite a violação das janelas temporais, foi necessário desenvolver um operador eficiente para efetuar as correções durante o processo evolutivo. Assim, esse operador desempenhou papel fundamental dentro do processo global da heurística. Além disso, a sincronização da linha de tempo dos veículos pode resolver questões relacionadas ao desbalanceamento das rotas, mantendo todos os veículos equilibrados em relação ao tempo total de operação.

Por fim, uma grande contribuição da heurística *VPDRT* é sua eficiência na identificação de solução com uma reduzida quantidade de veículos. Essa redução tem um impacto significativo em empresas de médio e pequeno, que podem se beneficiar com a diminuição da quantidade de veículos necessários para atender suas demandas. Essa

redução possibilita um menor capital investido com frota e recursos humanos. Assim, a solução proposta tem grande potencial de uso prático, pois possui a capacidade atender não só pequenas e médias empresas, que representam maior parte do mercado, mas também grandes empresas, quando aplicada ao atendimento de clientes em regiões próximas às filiais. Parte dignificativa do mercado pode se beneficiar da abordagem proposta e com isso obter vantagem competitiva, possibilitando uma cadeia logística mais eficiente.

7 Trabalhos futuros

Como trabalhos futuros, pode-se destacar a realização de experimentos em instâncias com 50 e 100 clientes, para verificar o desempenho da abordagem em cenários de maior complexidade. Outro cenário que pode se apresentar promissor é a utilização de inserção dinâmica de clientes. Como o algoritmo calcula a direção das rotas em tempo de execução, uma possibilidade que se abre é aplicar a *VPDTR* a cenários dinâmicos, onde os clientes sejam inseridos dinamicamente durante o processo evolutivo.

Outro ponto a ser explorado são as técnicas de adaptação para um operador de cruzamento mais eficiente, especificamente aplicado aos problemas do roteamento de veículos. O operador de cruzamento é um passo altamente importante para a manutenção da diversidade da população e para a conservação de rotas ou trechos de rotas que se apresentem promissores, transferindo para novos indivíduos os cromossomos mais relevantes, além de proporcionar uma exploração ampla do espaço de soluções.

Também é possível aprimorar o operador de correção de violações de janelas de tempo, construindo um operador que seja mais preciso durante a correção das violações, possibilitando que os indivíduos inválidos sejam corrigidos de forma mais rápida e eficiente.

Além disso, o armazenamento de trechos de cromossomos incorrigíveis pode subsidiar uma rotina de eliminação de soluções não factíveis. Guardar os padrões de cromossomo incorrigíveis pode agilizar o processo de identificação de soluções promissoras, uma vez que não é necessário dispender tempo de processamento com soluções que não podem mais serem ajustadas.

Referências

ALZAQEBAH, M. *et al.* Bees algorithm for vehicle routing problems with time windows. **International Journal of Machine Learning and Computing**, v. 8, n. 3, p. 234 – 240, 2018.

ANGELOVA, M. *et al.* Tuning genetic algorithm parameters to improve convergence time. **International Journal of Chemical Engineering**, Hindawi, v. 2011, 2011.

BRANDÃO, J. A tabu search algorithm for the open vehicle routing problem. **European Journal of Operational Research**, Elsevier, v. 157, n. 3, p. 552 – 564, 2004.

BUBA, A. T.; LEE, L. S. Differential evolution with improved sub-route reversal repair mechanism for multiobjective urban transit routing problem. **Numerical Algebra, Control and Optimization**, v. 8, n. 3, p. 351 – 376, 2018.

CACERES-CRUZ, J. *et al.* Rich vehicle routing problem: Survey. **ACM Computing Surveys (CSUR)**, ACM New York, NY, USA, v. 47, n. 2, p. 1 – 28, 2014.

CHANG, F. *et al.* Greedy-search-based multi-objective genetic algorithm for emergency logistics scheduling. **Expert Systems with Applications**, Elsevier, v. 41, n. 6, p. 2947 – 2956, 2014.

COELLO, C. A. C. **Evolutionary algorithms for solving multi-objective problems**. [S.l.]: Springer, 2007.

DANTZIG, G. B.; RAMSER, J. H. The truck dispatching problem. **Management science**, Informs, v. 6, n. 1, p. 80 – 91, 1959.

DEB, K. **Multi-Objective Optimization using Evolutionary Algorithms**. 1. ed. Chichester: JOHN WILEY & SONS, LTD, 2001.

DELTORO, M. F. *et al.* Factores determinantes y consecuencias de la adopción del comercio electrónico B2C: una comparativa internacional. **Estudios Gerenciales**, Elsevier, v. 28, n. 123, p. 101 – 120, 2012.

DEVI, R. *et al.* Survey on evolutionary computation tech techniques and its application in different fields. **International Journal on Information Theory (IJIT)**, Citeseer, v. 3, n. 3, p. 73 – 82, 2014.

DONG, W. *et al.* A tissue P system based evolutionary algorithm for multi-objective VRPTW. **Swarm and evolutionary computation**, Elsevier, v. 39, p. 310 – 322, 2018.

EIBEN, A. E.; SMITH, J. E. **Introduction to evolutionary computing**. [S.l.]: Springer, 2015.

EMAD, M. **Logistics costs in Iceland**. 2022. Tese (Doutorado).

ERRICO, F. *et al.* The vehicle routing problem with hard time windows and stochastic service times. **EURO Journal on Transportation and Logistics**, Springer, v. 7, p. 223 – 251, 2018.

ESTRADA-MORENO, A. *et al.* A biased-randomized algorithm for redistribution of perishable food inventories in supermarket chains. **International Transactions in Operational Research**, Wiley Online Library, v. 26, n. 6, p. 2077 – 2095, 2019.

FONSECA, C. M.; FLEMING, P. J. Multiobjective genetic algorithms made easy: selection sharing and mating restriction. In: IET, 1995. **First International Conference on Genetic Algorithms in Engineering Systems**: Innovations and applications. [S.l.], 1995. p. 45 – 52.

FRIFITA, S.; MASMOUDI, M. VNS methods for home care routing and scheduling problem with temporal dependencies, and multiple structures and specialties. **International transactions in operational research**, Wiley Online Library, v. 27, n. 1, p. 291 – 313, 2020.

GHOSEIRI, K.; GHANNADPOUR, S. F. Multi-objective vehicle routing problem with time windows using goal programming and genetic algorithm. **Applied Soft Computing**, Elsevier, v. 10, n. 4, p. 1096 – 1107, 2010.

GOEKE, D.; SCHNEIDER, M. Routing a mixed fleet of electric and conventional vehicles. **European Journal of Operational Research**, Elsevier, v. 245, n. 1, p. 81 – 99, 2015.

GOLDBERG, D. Genetic algorithms in search. **optimization and machine learning**, Addison Wesley, 1989.

GONZÁLEZ, O. M. *et al.* A parallel memetic algorithm to solve the capacitated vehicle routing problem with time windows. **International Journal of Combinatorial Optimization Problems and Informatics**, v. 9, n. 1, 2018.

HARZI, M.; KRICHEN, S. Variable neighborhood descent for solving the vehicle routing problem with time windows. **Electronic Notes in Discrete Mathematics**, Elsevier, v. 58, p. 175 – 182, 2017.

HEDAR, A.; BAKR, M. A. Three strategies tabu search for vehicle routing problem with time windows. **Computer Science and Information Technology**, v. 2, n. 2, p. 108 – 119, 2014.

HERDIANA, I. K. *et al.* Optimization of Adaptive Genetic Algorithm Parameters in Traveling Salesman Problem. **Journal of Computer Networks, Architecture and High Performance Computing**, v. 4, n. 2, p. 169 – 176, 2022.

HOLLAND, J. Adaption in Natural and Artificial Systems. **MIT Press, Cambridge, MA, 1992. 1st edition**, 1975.

HOLLAND, J. H. **Adaptation in natural and artificial systems Cambridge**. [S.l.]: Massachusetts-London, 1992.

HU, W. *et al.* A hybrid chaos-particle swarm optimization algorithm for the vehicle routing problem with time window. **Entropy**, MDPI, v. 15, n. 4, p. 1247 – 1270, 2013.

ITO, K. *et al.* A multiobjective optimization approach to a design problem of heat insulation for thermal distribution piping network systems. **Journal of Mechanisms, Transmissions, and Automation in Design**, v. 105, n. 2, p. 206 – 213, 1983.

JAIN, H.; DEB, K. An evolutionary many-objective optimization algorithm using reference-point based nondominated sorting approach, part II: Handling constraints and extending to an adaptive approach. **IEEE Transactions on evolutionary computation**, IEEE, v. 18, n. 4, p. 602 – 622, 2013.

JAYARATHNA, D. *et al.* Survey on ten years of multi-depot vehicle routing problems: mathematical models, solution methods and real-life applications. **Sustainable Development Research**, v. 3, n. 1, p. p36 – p36, 2021.

JONG, K. D. **An Analysis of the Behaviour of a Class of Genetic Adaptive Systems**. 1975. Tese (Doutorado) — University of Michigan.

KALLEHAUGE, B. Formulations and exact algorithms for the vehicle routing problem with time windows. **Computers & Operations Research**, Elsevier, v. 35, n. 7, p. 2307 – 2330, 2008.

KALYANMOY DEB. **Multi-Objective Optimization Using Evolutionary Algorithms**: An Introduction. 1. ed. New York: Wiley, 2001. 544 p. ISBN 9780470743614.

KIM, G. *et al.* City vehicle routing problem (city VRP): A review. **IEEE Transactions on Intelligent Transportation Systems**, IEEE, v. 16, n. 4, p. 1654 – 1666, 2015.

KRAMER, R. *et al.* A matheuristic approach for the pollution-routing problem. **European Journal of Operational Research**, Elsevier, v. 243, n. 2, p. 523 – 539, 2015.

LAN, Y. *et al.* Decomposition based multi-objective variable neighborhood descent algorithm for logistics dispatching. **IEEE Transactions on Emerging Topics in Computational Intelligence**, IEEE, v. 5, n. 5, p. 826 – 839, 2020.

LIANG, J. *et al.* A survey on evolutionary constrained multiobjective optimization. **IEEE Transactions on Evolutionary Computation**, IEEE, v. 27, n. 2, p. 201 – 221, 2022.

LOZANO, L. *et al.* An exact algorithm for the elementary shortest path problem with resource constraints. **Transportation Science**, INFORMS, v. 50, n. 1, p. 348 – 357, 2016.

LU, C. *et al.* A Pareto-based hybrid iterated greedy algorithm for energy-efficient scheduling of distributed hybrid flowshop. **Expert Systems with Applications**, Elsevier, v. 204, 2022.

LÜCKEN, C. V. *et al.* A survey on multi-objective evolutionary algorithms for many-objective problems. **Computational optimization and applications**, Springer, v. 58, p. 707 – 756, 2014.

MEN, J. *et al.* Robust multi-objective vehicle routing problem with time windows for hazardous materials transportation. **IET Intelligent Transport Systems**, Wiley Online Library, v. 14, n. 3, p. 154 – 163, 2020.

NAKAI, K. *et al.* Nondominated-solution-based multi-objective greedy sensor selection for optimal design of experiments. **IEEE Transactions on Signal Processing**, IEEE, v. 70, p. 5694 – 5707, 2022.

NISRINA, N. *et al.* The effect of genetic algorithm parameters tuning for route optimization in travelling salesman problem through general full factorial design analysis. **Evergreen Joint Journal of Novel Carbon Resource Sciences and Green Asia Strategy**, v. 09, n. 01, p. 163 – 203, 2022.

OMBUKI, B. *et al.* Multi-objective genetic algorithms for vehicle routing problem with time windows. **Applied Intelligence**, Springer, v. 24, p. 17 – 30, 2006.

PARRAGH, S. N. Introducing heterogeneous users and vehicles into models and algorithms for the dial-a-ride problem. **Transportation Research Part C: Emerging Technologies**, Elsevier, v. 19, n. 5, p. 912 – 930, 2011.

PRATAMA, R. Y.; MAHMUDY, W. F. Optimization of vehicle routing problem with time window (VRPTW) for food product distribution using genetics algorithm. **Journal of Information Technology and Computer Science**, v. 2, n. 2, 2017.

REZENDE, A. A. de *et al.* A reinvenção das vendas: as estratégias das empresas brasileiras para gerar receitas na pandemia de covid-19. **Boletim de Conjuntura (BOCA)**, v. 2, n. 6, p. 53 – 69, 2020.

SARIKLIS, D.; POWELL, S. A heuristic method for the open vehicle routing problem. **Journal of the Operational Research Society**, Springer, v. 51, p. 564 – 573, 2000.

SCHAFFER, J. D. *et al.* Multiple objective optimization with vector evaluated genetic algorithms. In: **Proceedings of an international conference on genetic algorithms and their applications**. [S.l.: s.n.], 1985. p. 93 – 100.

SETTEY, T. *et al.* The growth of e-commerce due to COVID-19 and the need for urban logistics centers using electric vehicles: Bratislava case study. **Sustainability (Switzerland)**, v. 13, n. 10, 2021. ISSN 20711050.

SHEN, Y. *et al.* A hybrid swarm intelligence algorithm for vehicle routing problem with time windows. **Ieee Access**, IEEE, v. 8, p. 93882 – 93893, 2020.

SILVA JUNIOR, O. S. da *et al.* A multiple ant colony system with random variable neighborhood descent for the dynamic vehicle routing problem with time windows. **Soft Computing**, Springer, v. 25, p. 2935 – 2948, 2021.

SOLOMON, M. M. Algorithms for the vehicle routing and scheduling problems with time window constraints. **Operations research**, Informs, v. 35, n. 2, p. 254 – 265, 1987.

SRINIVAS, N.; DEB, K. Muiltiobjective optimization using nondominated sorting in genetic algorithms. **Evolutionary computation**, MIT Press, v. 2, n. 3, p. 221 – 248, 1994.

SRIVASTAVA, G. *et al.* NSGA-II with objective-specific variation operators for multiobjective vehicle routing problem with time windows. **Expert Systems with Applications**, Elsevier, v. 176, 2021.

TAN, K. C. *et al.* A hybrid multiobjective evolutionary algorithm for solving vehicle routing problem with time windows. **computational optimization and applications**, Springer, v. 34, p. 115 – 151, 2006.

VAIRA, G.; KURASOVA, O. Genetic algorithm for VRP with constraints based on feasible insertion. **Informatica**, Vilnius University Institute of Mathematics and Informatics, v. 25, n. 1, p. 155 – 184, 2014.

VINCENT, F. Y. *et al.* Design of a two-echelon freight distribution system in last-mile logistics considering covering locations and occasional drivers. **Transportation Research Part E: Logistics and Transportation Review**, Elsevier, v. 154, 2021.

WANG, X. *et al.* Optimization of Wheel Reprofiling Based on the Improved NSGA-II. **Complexity**, Hindawi Limited, v. 2020, p. 1 – 13, 2020.

XU, S. *et al.* A combination of genetic algorithm and particle swarm optimization for vehicle routing problem with time windows. **Sensors**, MDPI, v. 15, n. 9, p. 21033 – 21053, 2015.

YAO, B. *et al.* Improved artificial bee colony algorithm for vehicle routing problem with time windows. **PloS one**, Public Library of Science San Francisco, CA USA, v. 12, n. 9, 2017.

ZEIN, S. *et al.* A repair operator for the preliminary design of a composite structure using genetic algorithms. **Structural and Multidisciplinary Optimization**, Springer, v. 55, p. 2099 – 2110, 2017.

ZHANG, Z. *et al.* Exact algorithms for the vehicle routing problem with time windows and combinatorial auction. **Transportation Science**, Informs, v. 53, n. 2, p. 427 – 441, 2019.

ZHAO, P. *et al.* Time-dependent and bi-objective vehicle routing problem with time windows. **Advances in Production Engineering & Management**, v. 14, n. 2, p. 201 – 212, 2019.

Apêndices

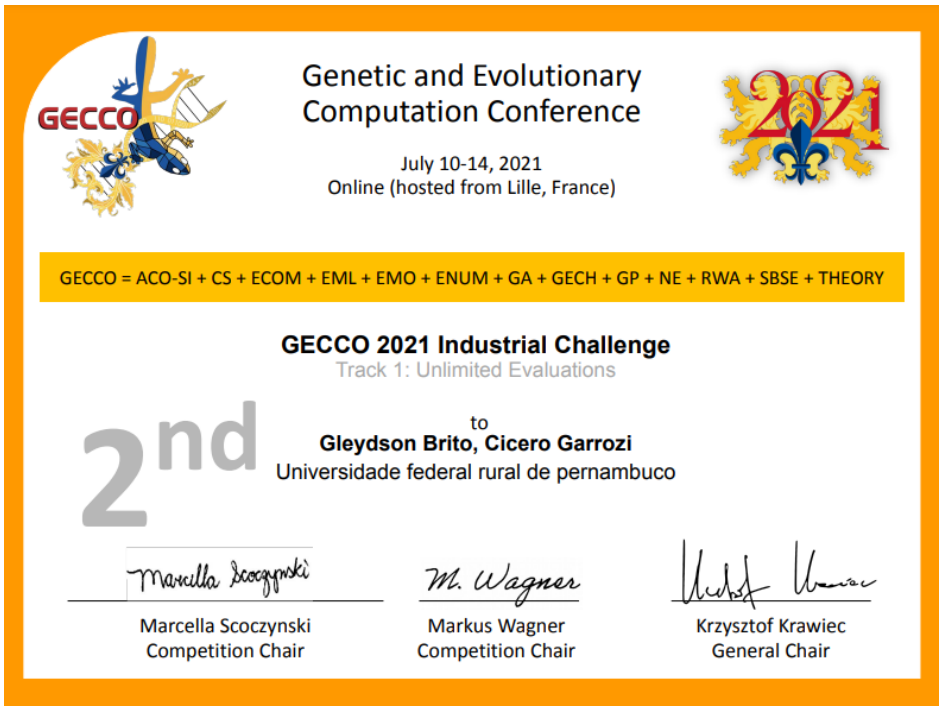
APÊNDICE A – Resultado da competição internacional GECCO 2021

A equipe composta pelo Prof. Dr. Cícero Garrozi e o Mestrando Gleydson Brito participou da competição internacional ***GECCO 2021 In***dustrial Challenge da *Genetic and Evolutionary Computation Conferrence*

Nesta participação a equipe obteve a 2° colocação na trilha 1 da competição. O resultado pode ser visualizado na página *Competition-Awards Gecco-2021*[1]. Para resolver o problema, foi utilizado um algoritmo genético com operadores otimizados .

Visão Geral:

> *"Modelos de simulação são ferramentas valiosas para a estimativa de uso de recursos e planejamento de capacidade. O simulador BaBSim.Hospital aborda explicitamente as dificuldades enfrentadas pelos hospitais devido à pandemia de COVID-19. O simulador pode lidar com muitos aspectos do planejamento de recursos em hospitais, como leitos de UTI, ventiladores ou equipamentos pessoais, levando em consideração várias coortes, como idade ou estado de saúde atual.*
>
> *A tarefa representa uma instância de um problema de otimização baseado em simulação computacional caro e de alta dimensionalidade. As simulações serão executadas por meio de uma interface e hospedadas em um de nossos servidores (similar ao nosso desafio do ano passado).*
>
> *Seu objetivo é encontrar uma configuração ótima de parâmetros para o simulador BaBSim.Hospital com um orçamento muito limitado de avaliações da função objetivo. Os participantes terão a liberdade de aplicar um ou vários algoritmos de otimização de sua escolha." (Tradução livre)*

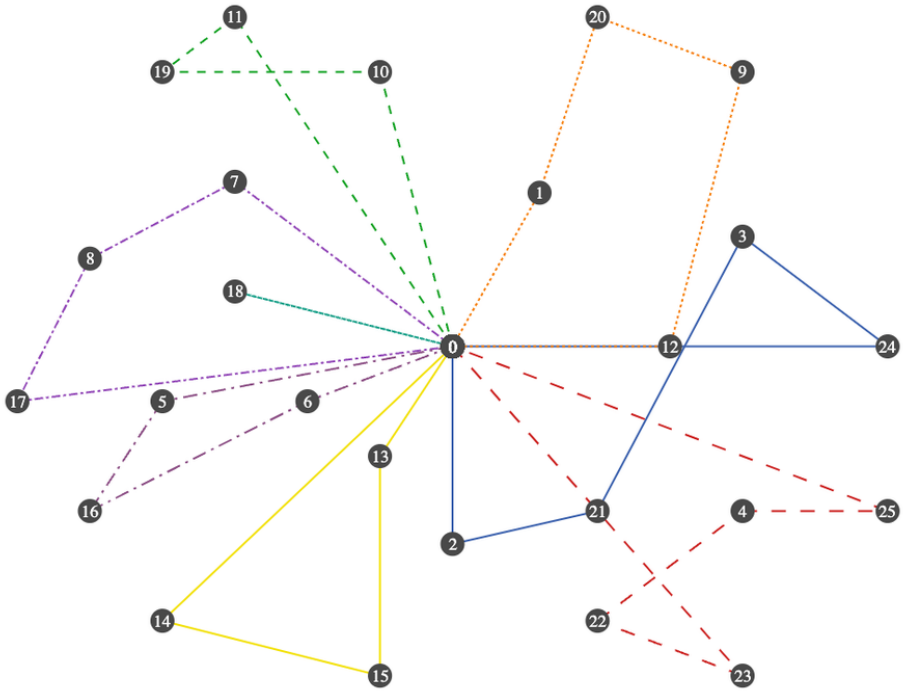


[1] https://gecco-2021.sigevo.org/Competition-Awards

APÊNDICE B – Apresentação topológica dos resultados obtidos pela heurística *VPDTR*

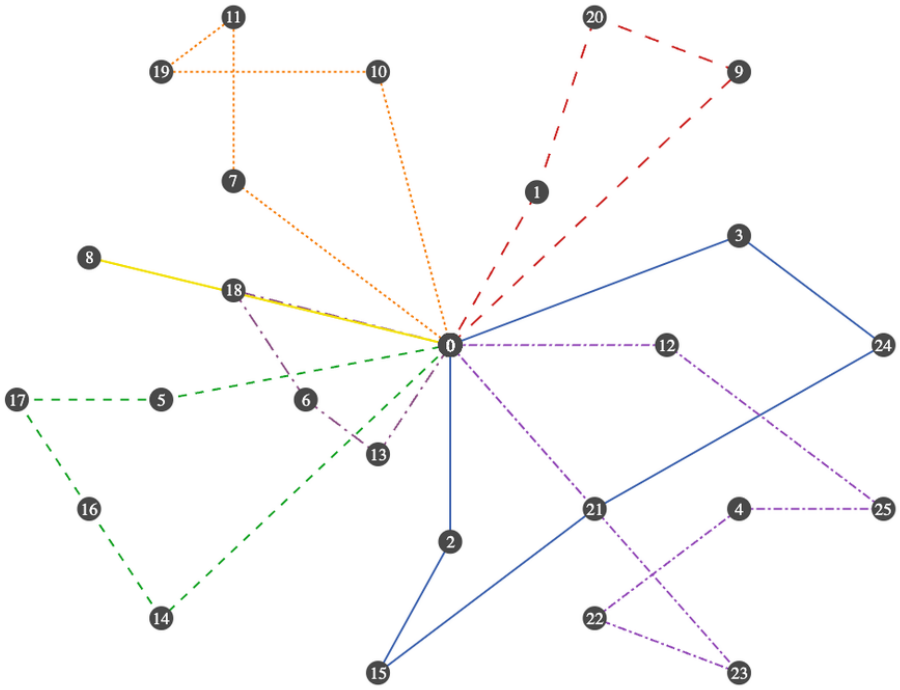
B.1 Instância R1

Figura 41 – R101



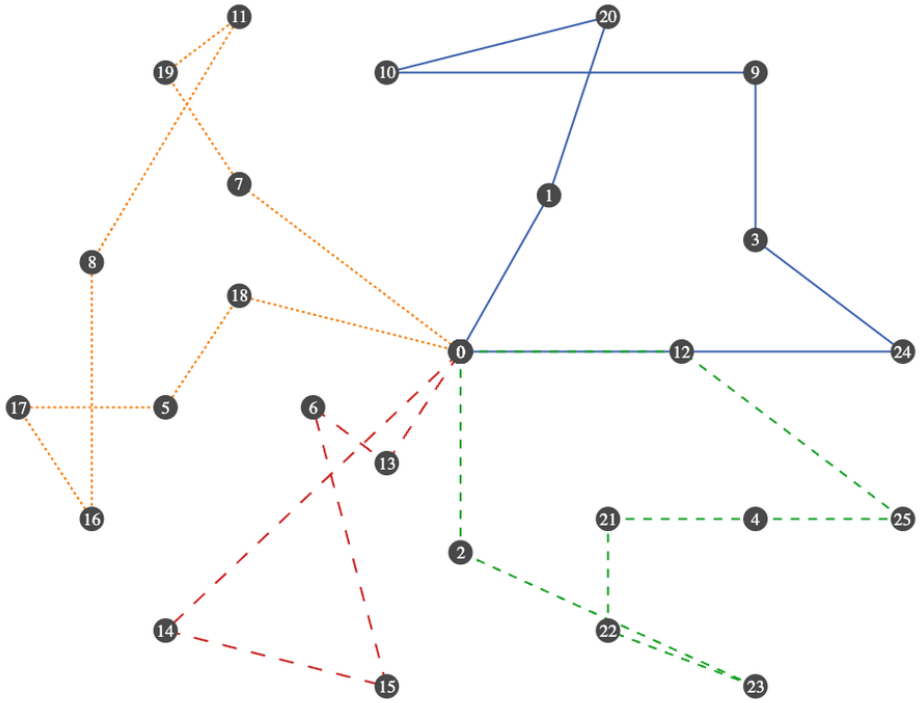
Elaborado pelo autor

Figura 42 – R102



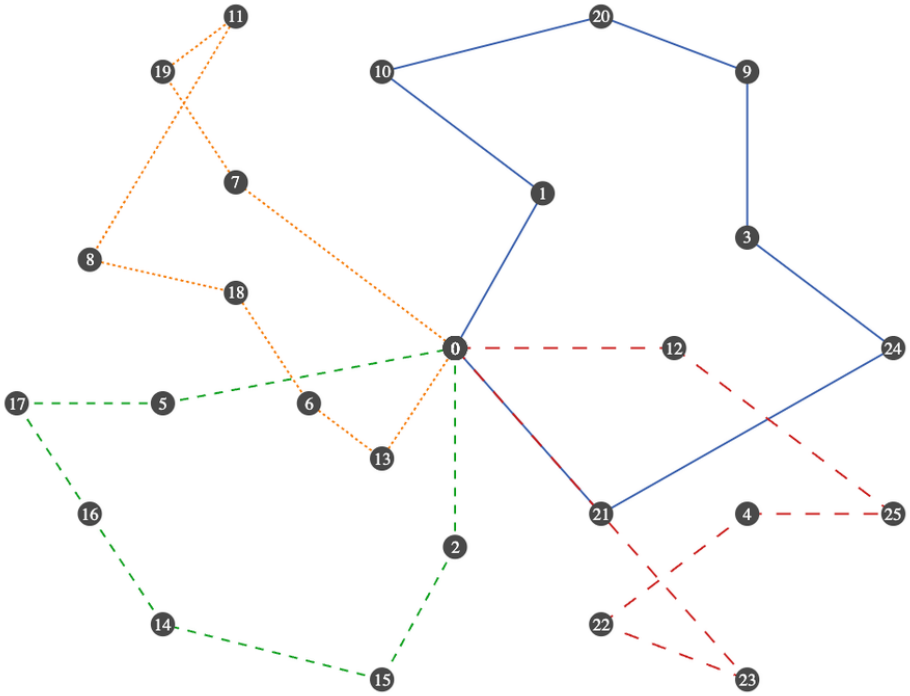
Elaborado pelo autor

Figura 43 – R103



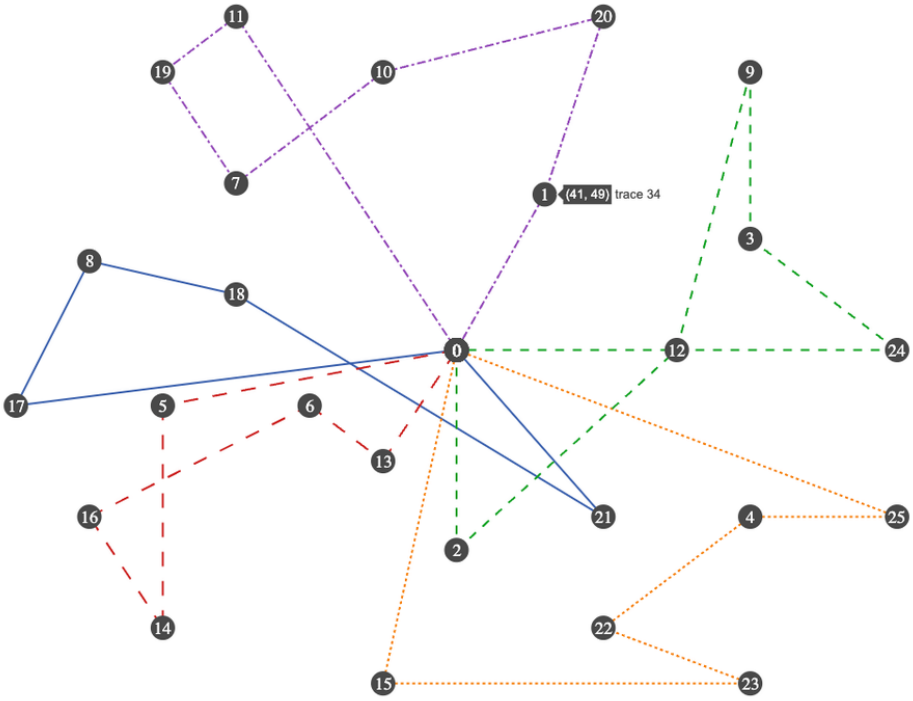
gr103

Figura 44 – R104



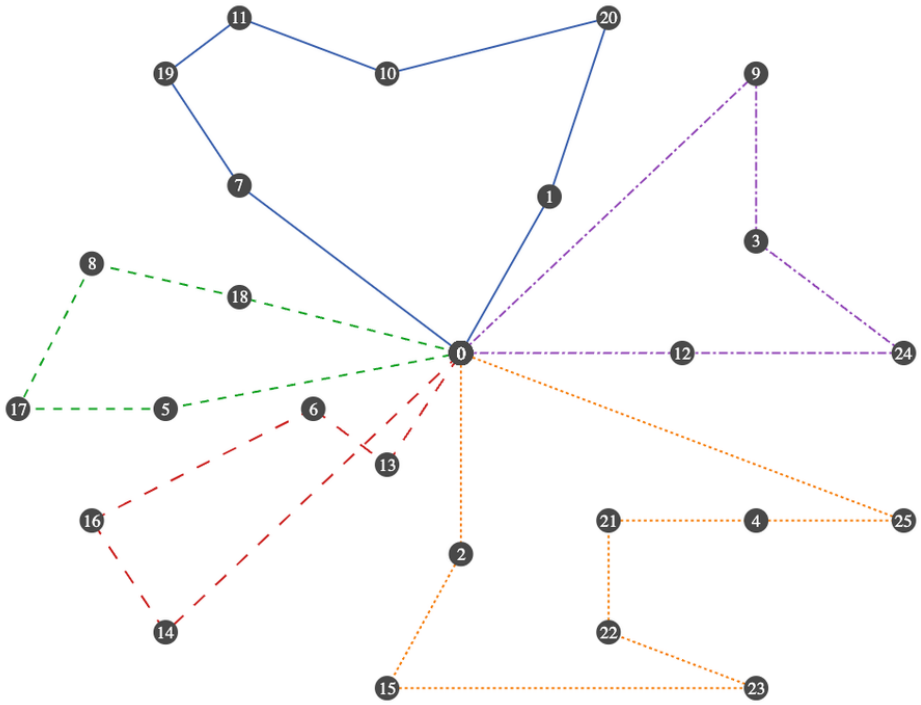
Elaborado pelo autor

Figura 45 – R105



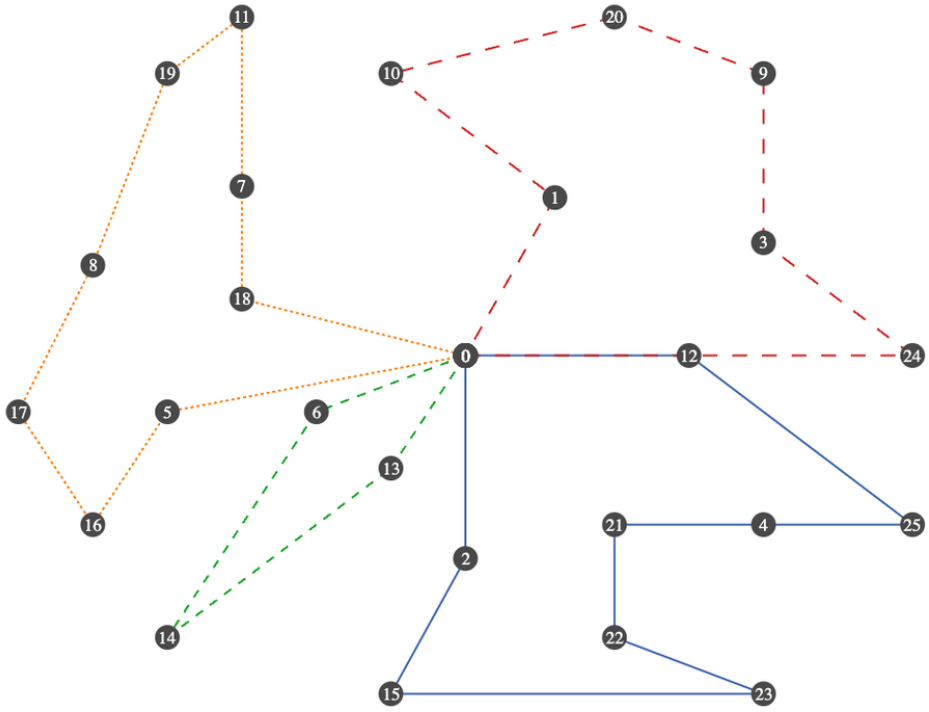
Elaborado pelo autor

Figura 46 – R106



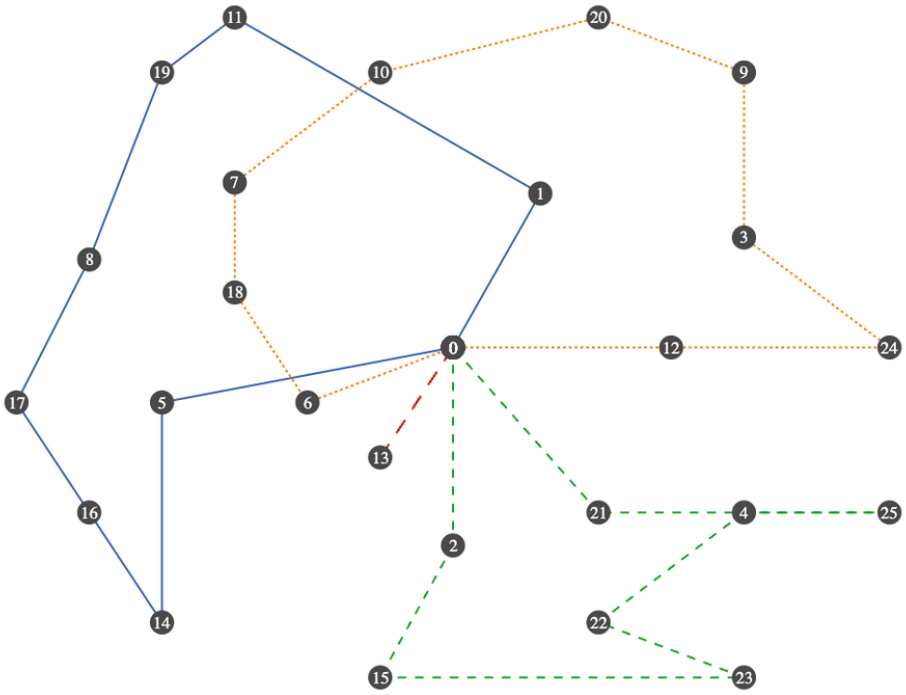
Elaborado pelo autor

Figura 47 – R107



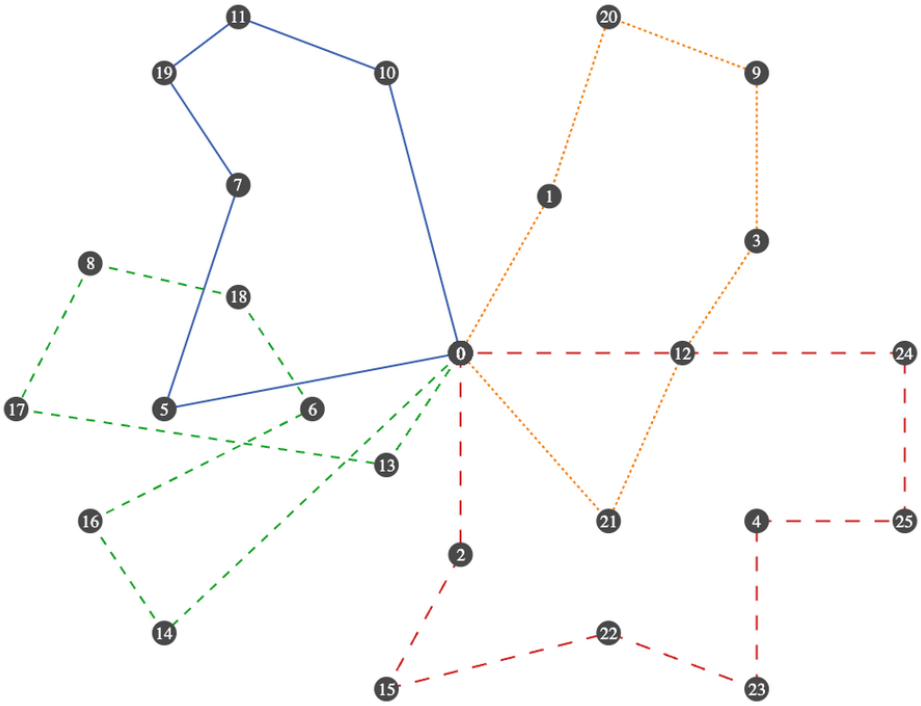
Elaborado pelo autor

Figura 48 – R108



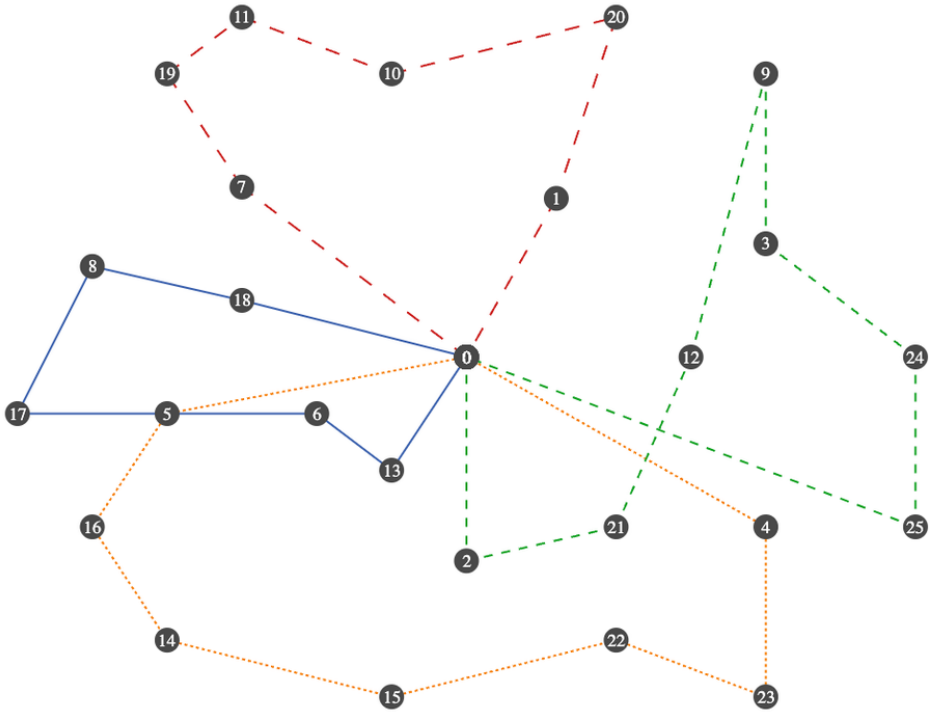
Elaborado pelo autor

Figura 49 – R109



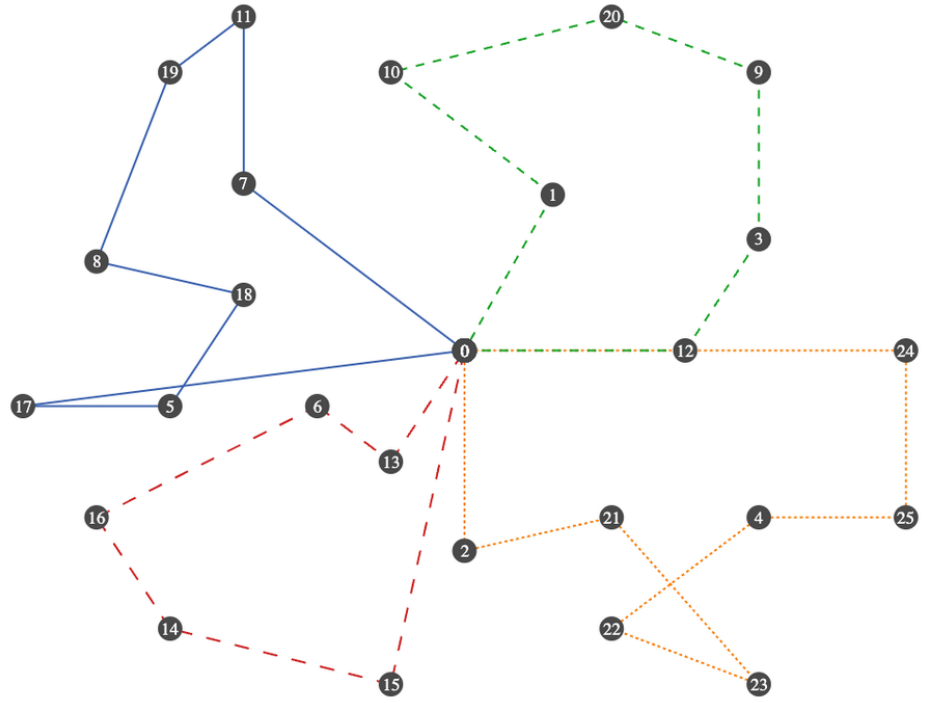
Elaborado pelo autor

Figura 50 – R110



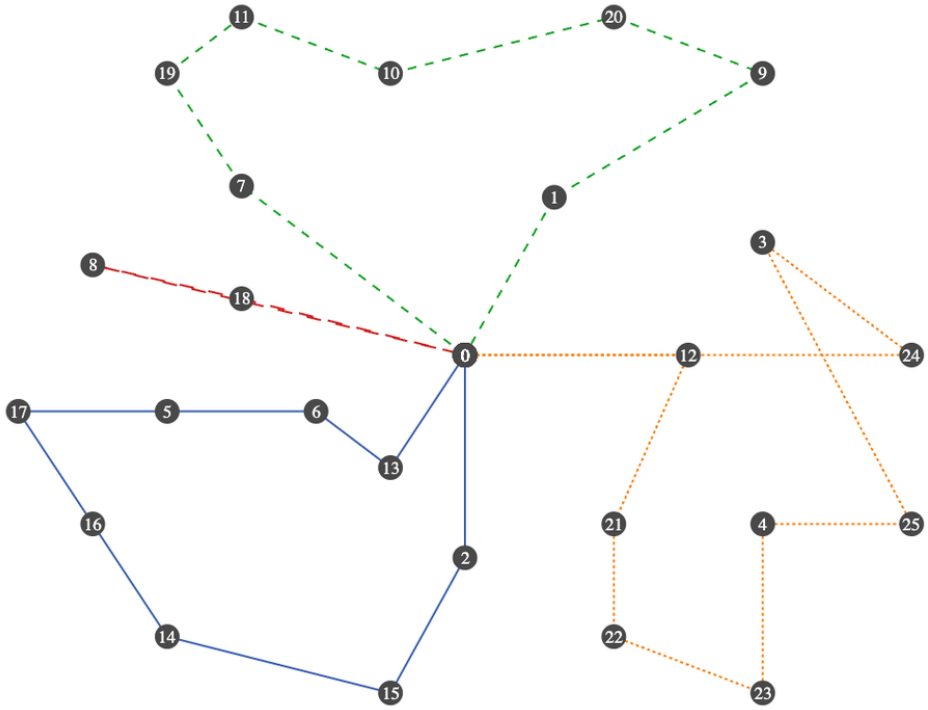
Elaborado pelo autor

Figura 51 – R111



Elaborado pelo autor

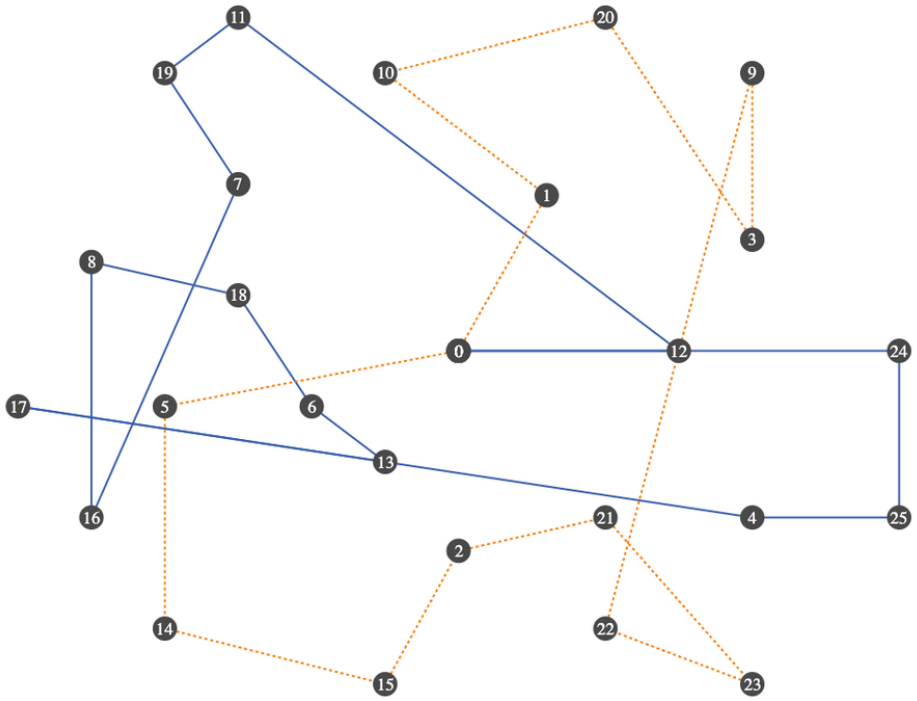
Figura 52 – R112



Elaborado pelo autor

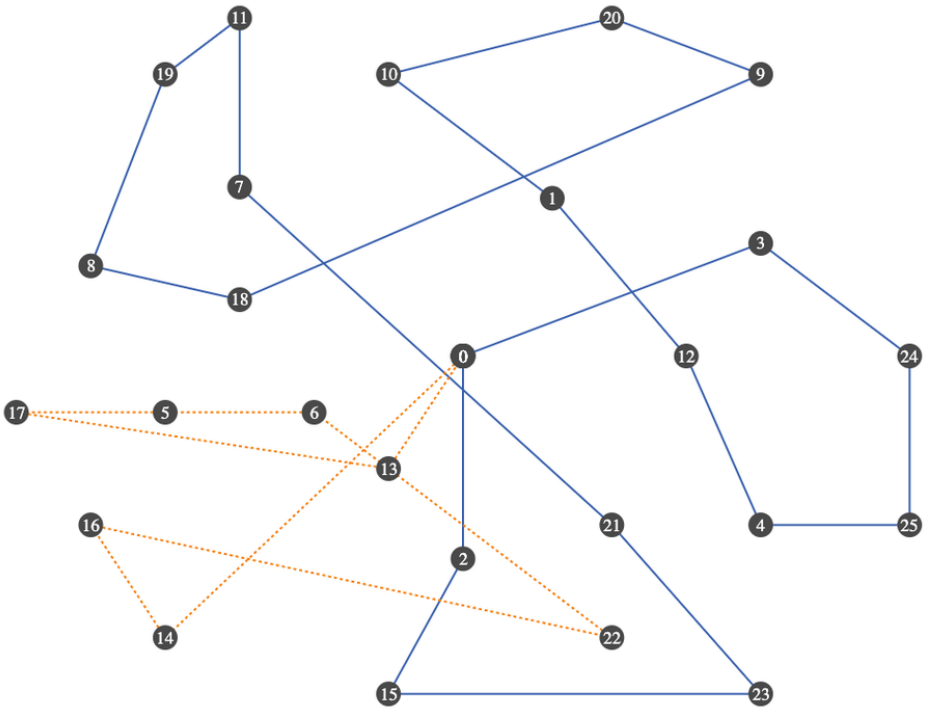
B.2 Instâncias R2

Figura 53 – R201



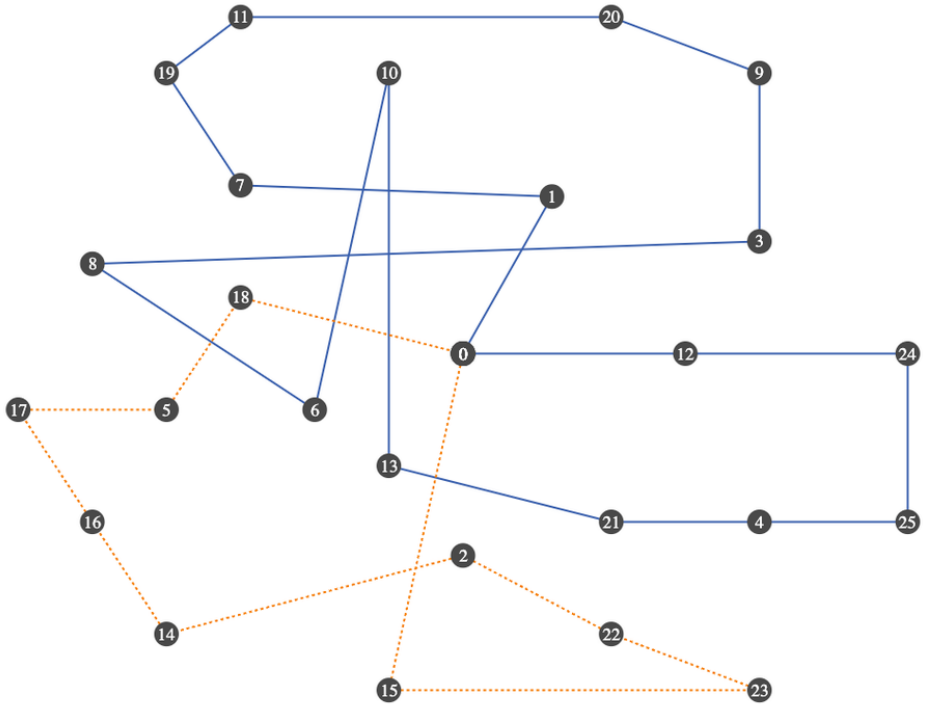
Elaborado pelo autor

Figura 54 – R202



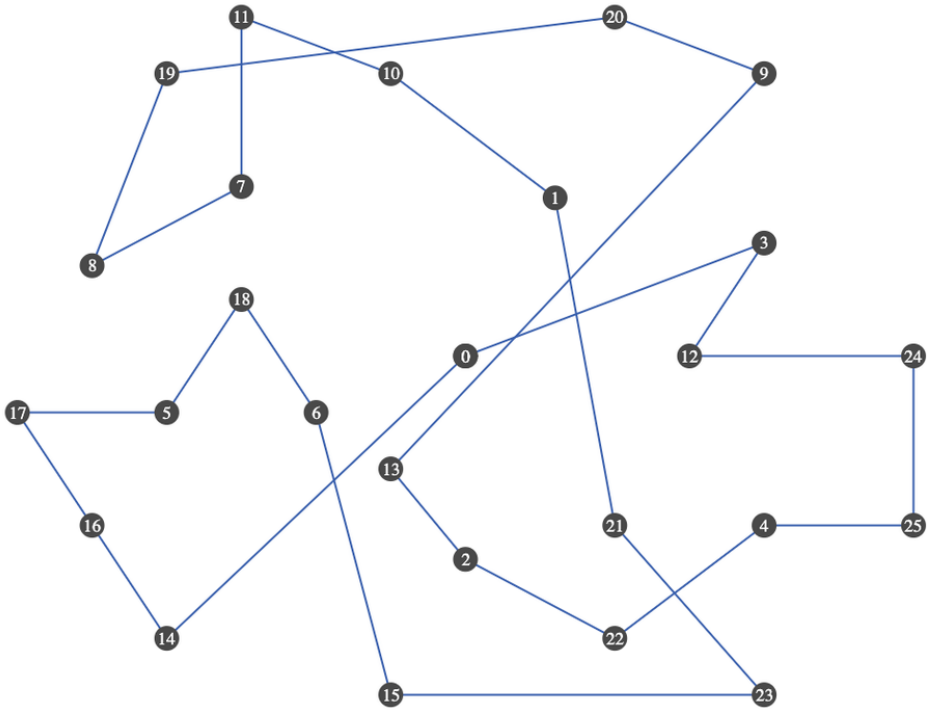
Elaborado pelo autor

Figura 55 – R203



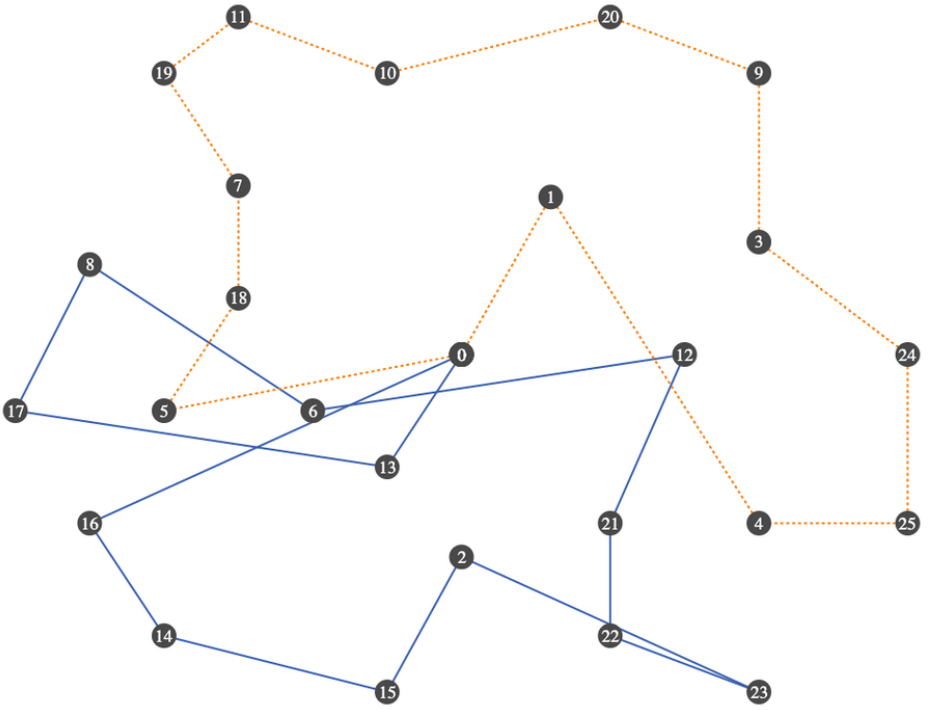
Elaborado pelo autor

Figura 56 – R204



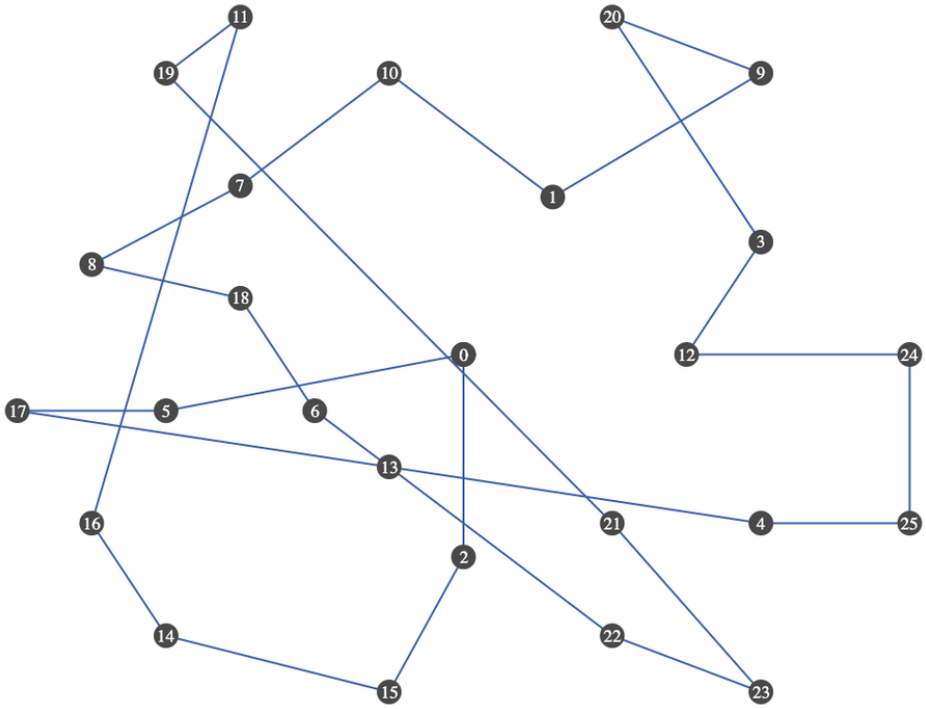
Elaborado pelo autor

Figura 57 – R205



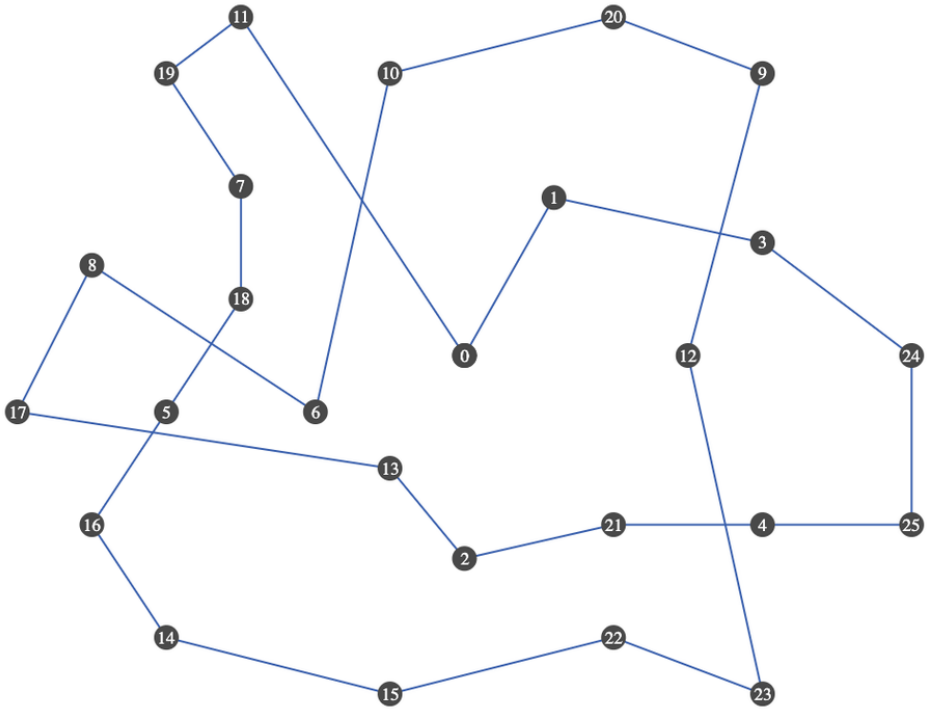
Elaborado pelo autor

Figura 58 – R206



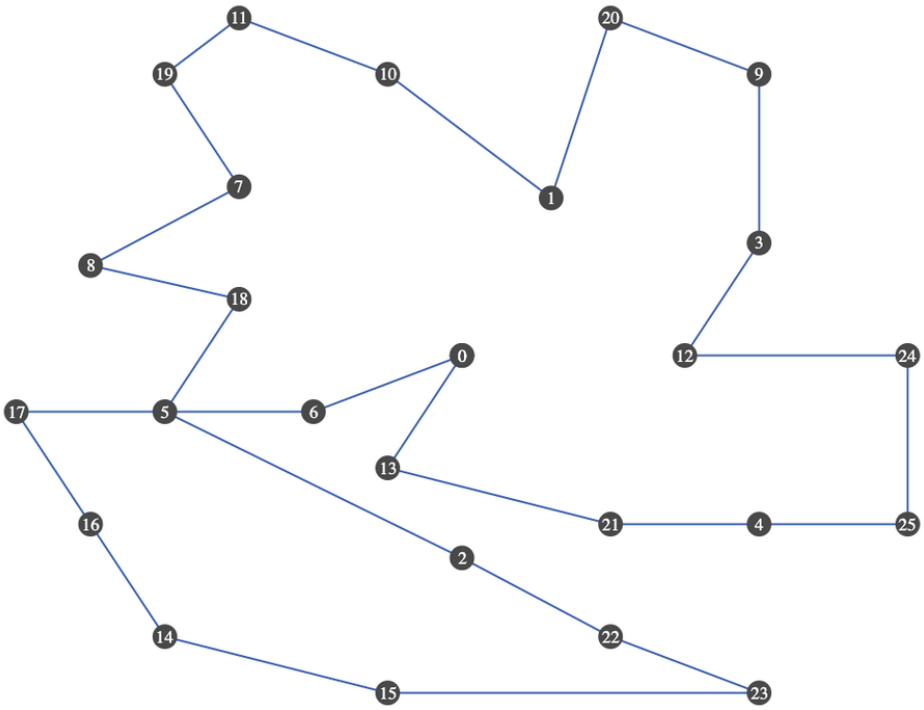
Elaborado pelo autor

Figura 59 – R207



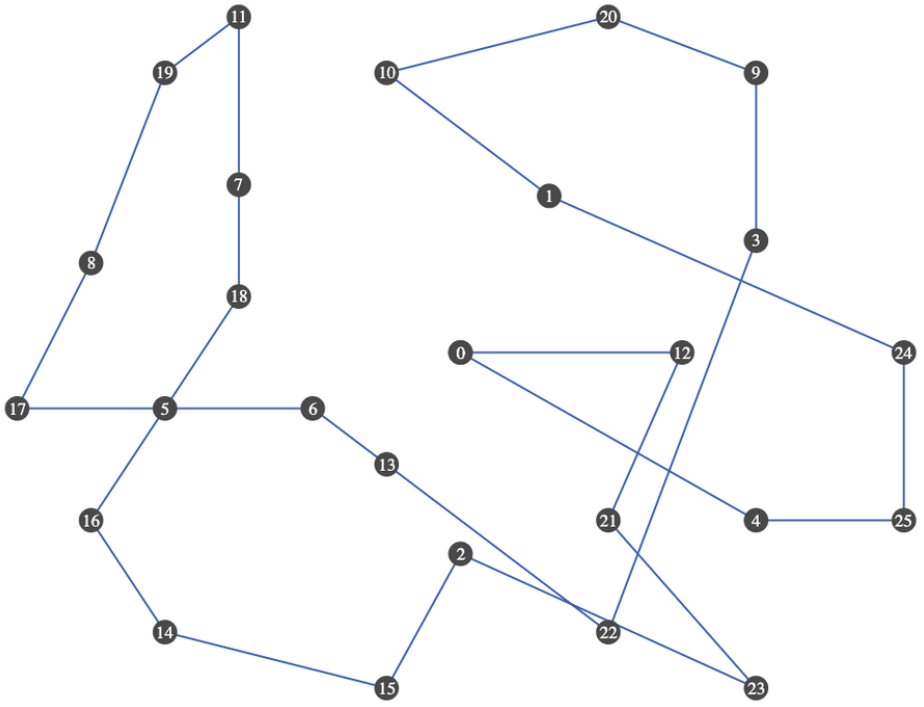
Elaborado pelo autor

Figura 60 – R208



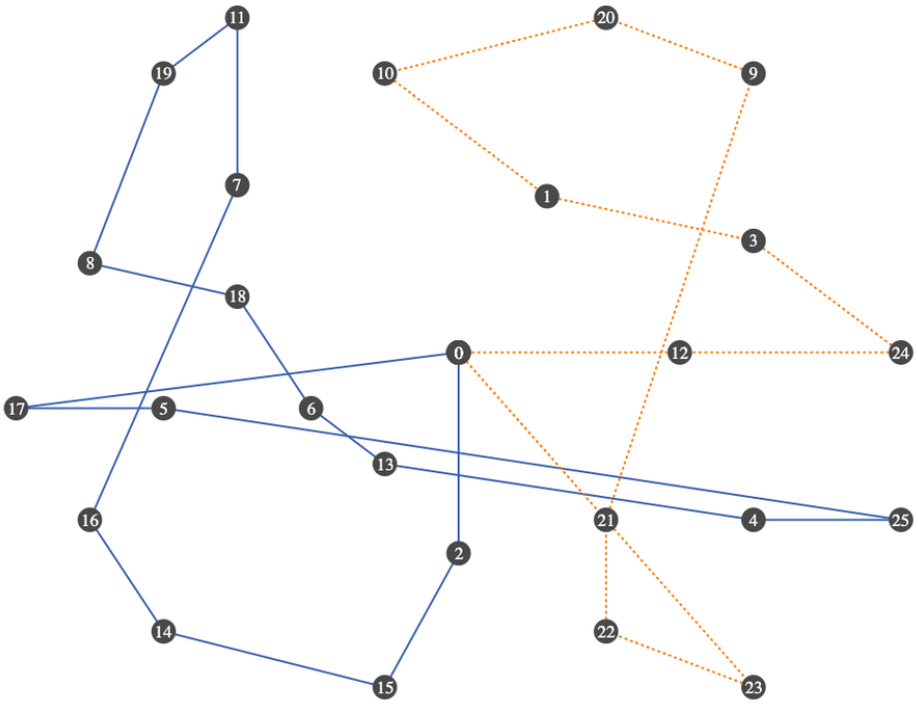
Elaborado pelo autor

Figura 61 – R209



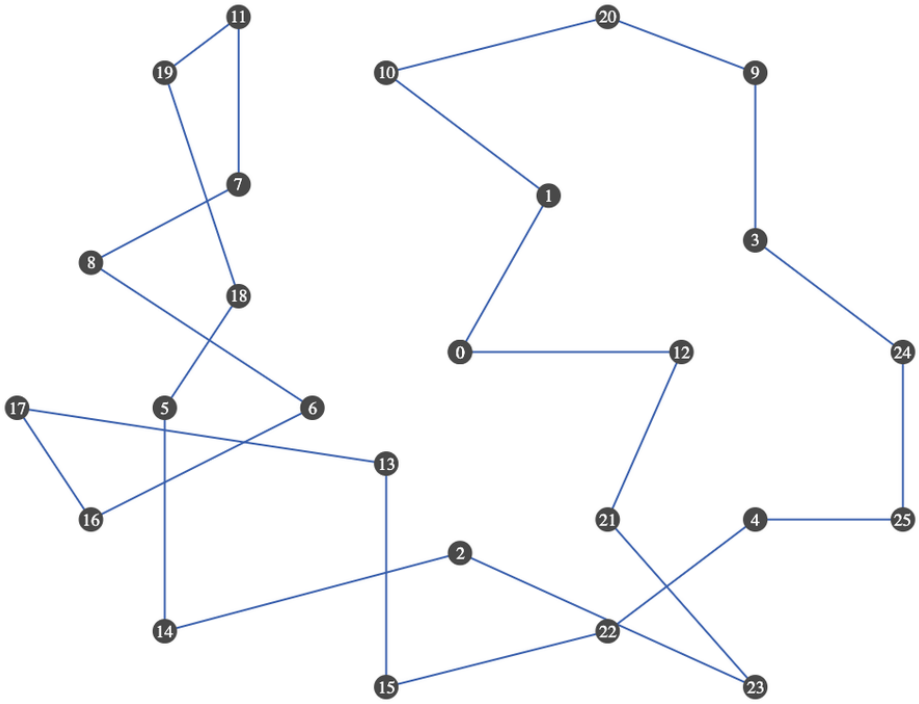
Elaborado pelo autor

Figura 62 – R210



Elaborado pelo autor

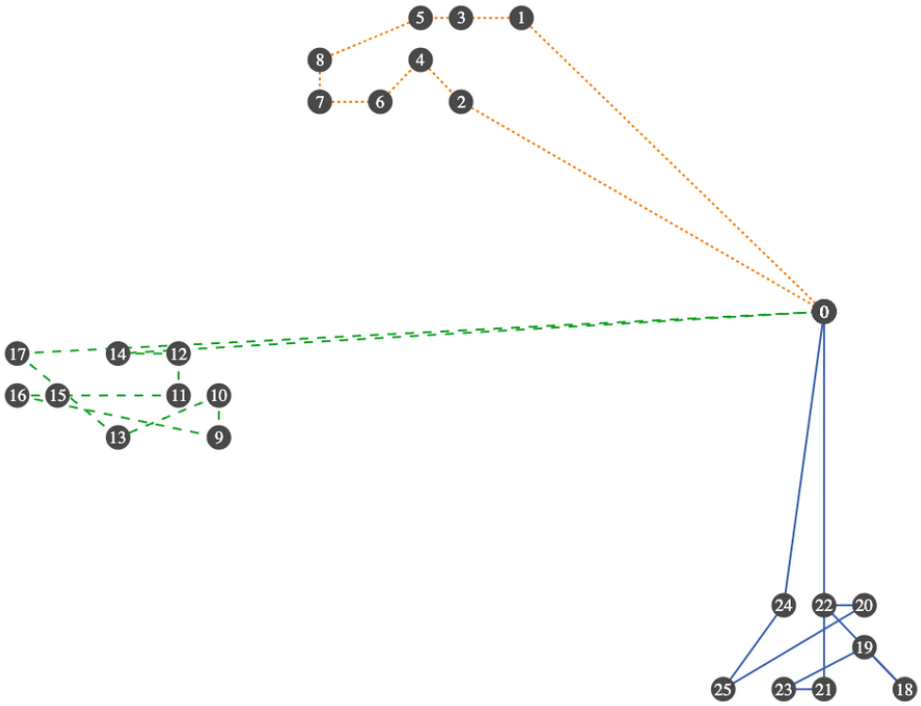
Figura 63 – R211



Elaborado pelo autor

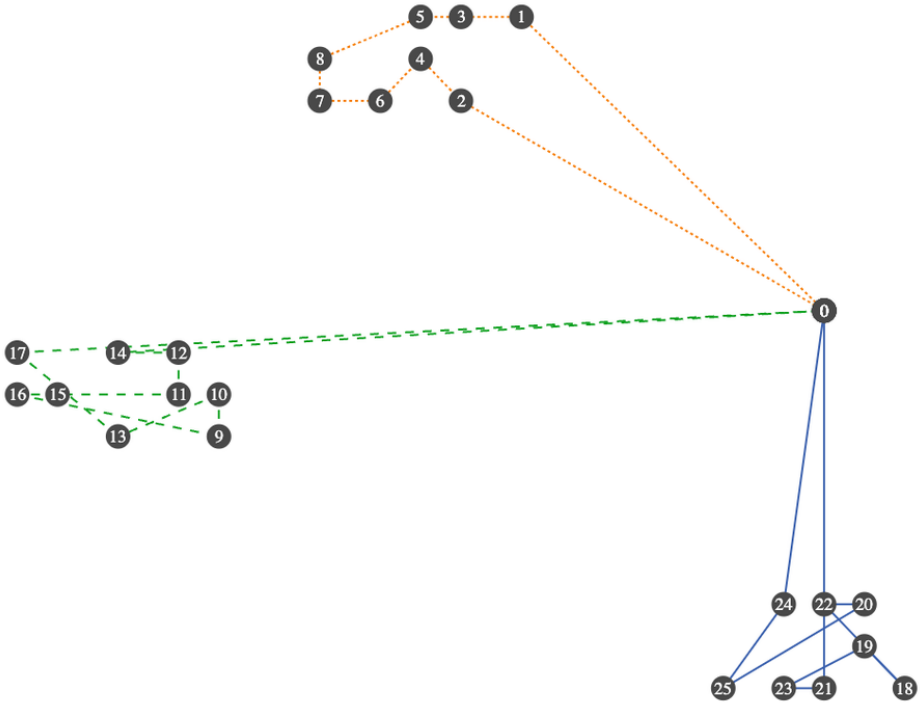
B.3 Instâncias RC1

Figura 64 – RC101



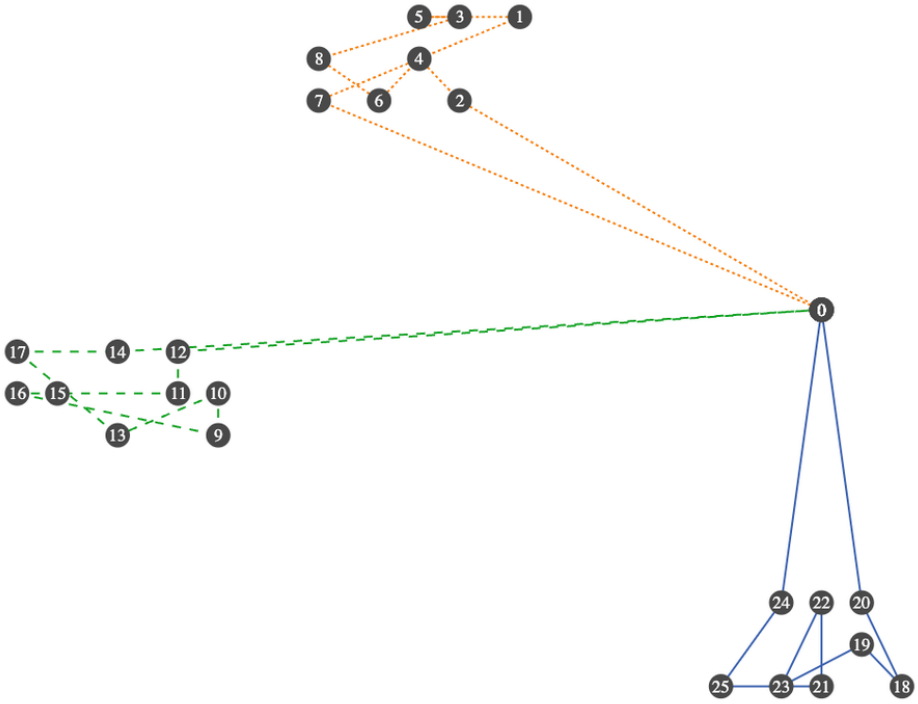
Elaborado pelo autor

Figura 65 – RC102



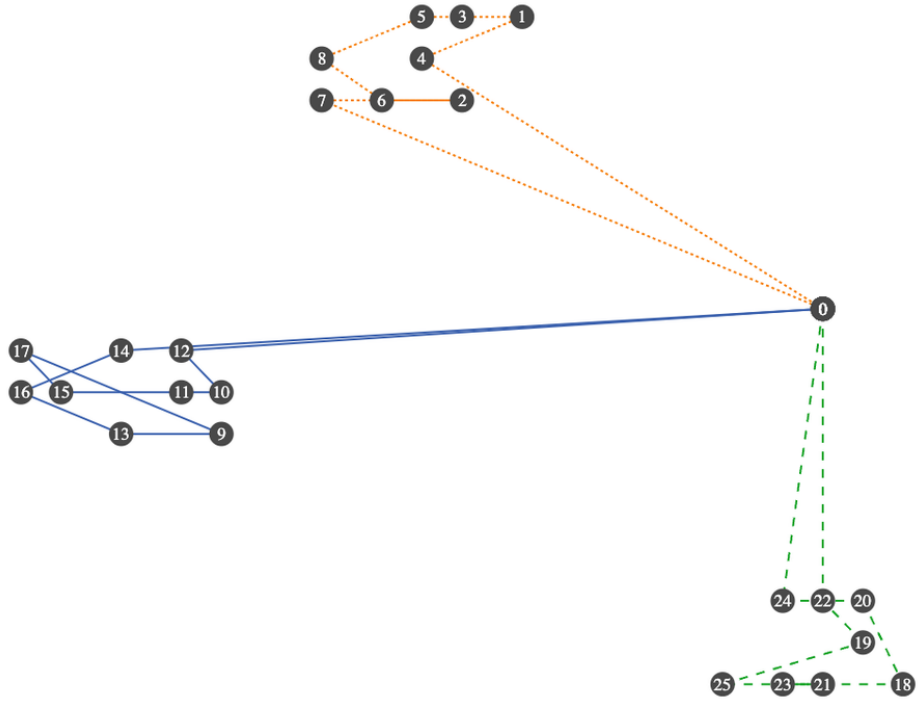
Elaborado pelo autor

Figura 66 – RC103



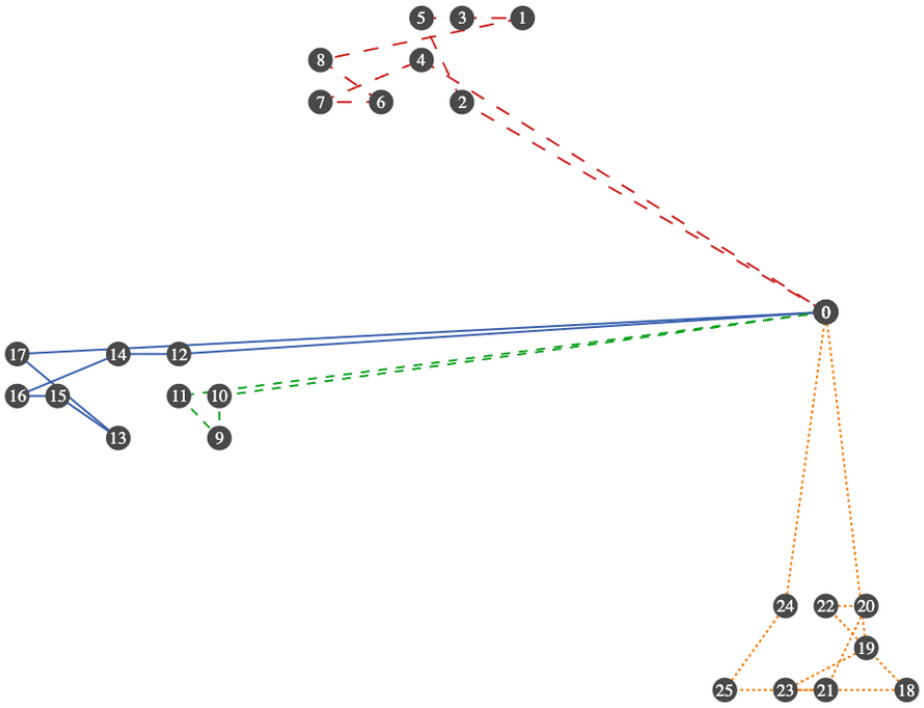
Elaborado pelo autor

Figura 67 – RC104



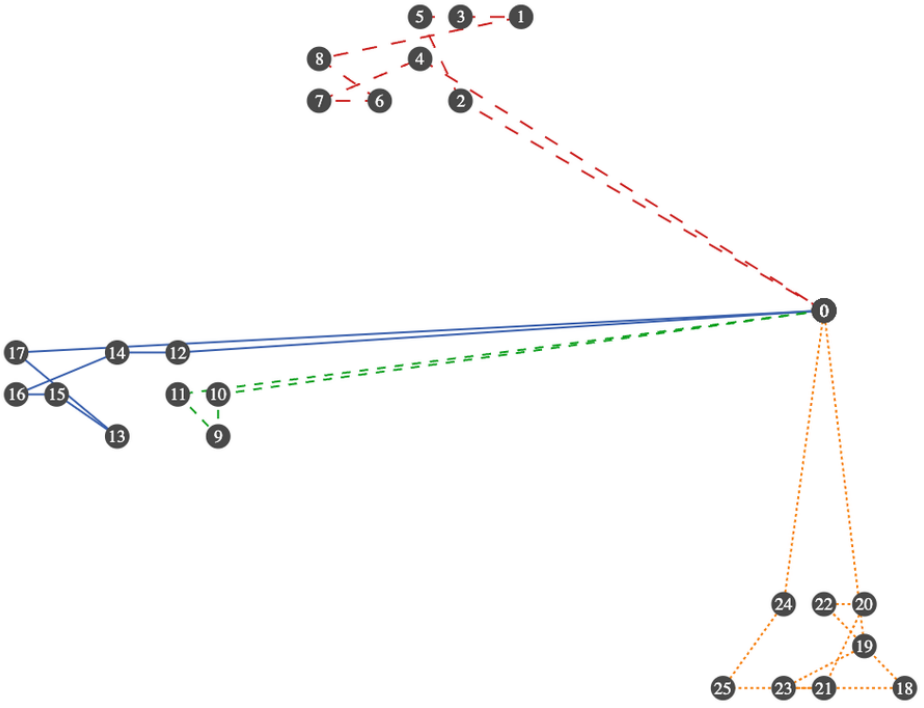
Elaborado pelo autor

Figura 68 – RC105



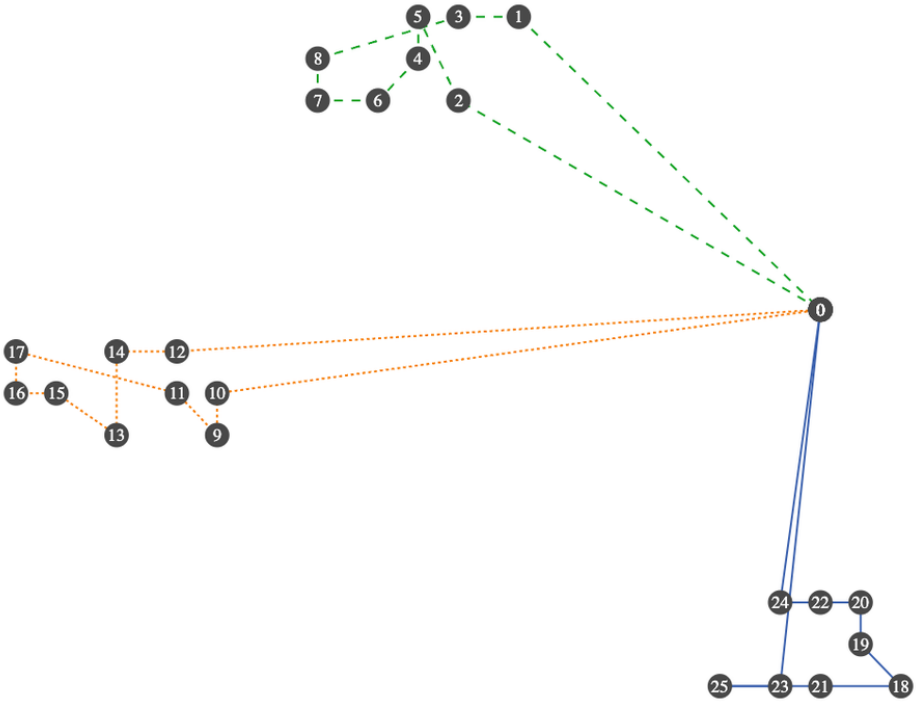
Elaborado pelo autor

Figura 69 – RC106



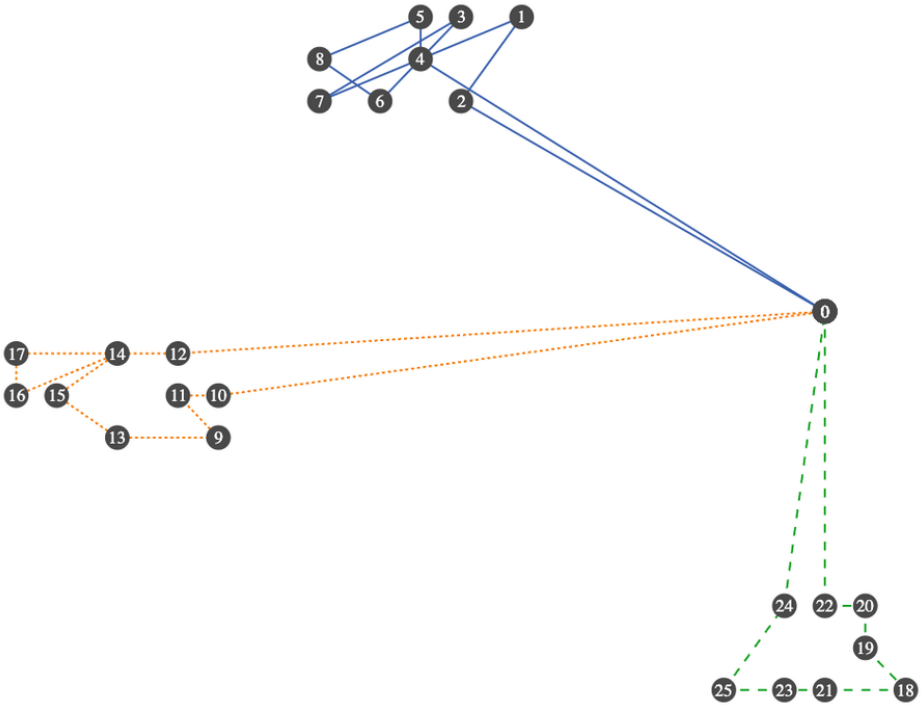
Elaborado pelo autor

Figura 70 – RC107



Elaborado pelo autor

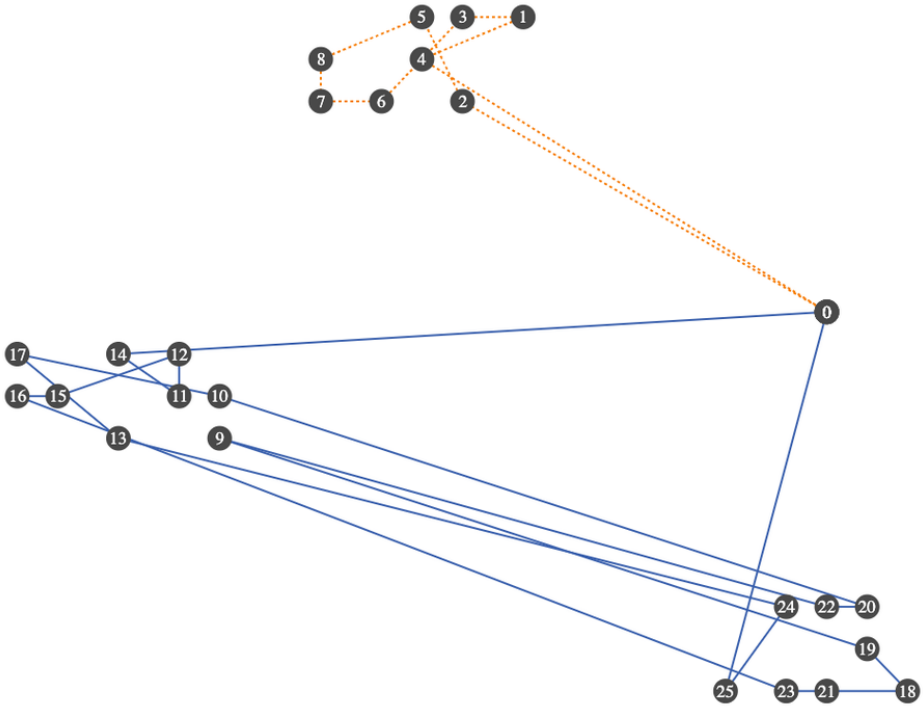
Figura 71 – RC108



Elaborado pelo autor

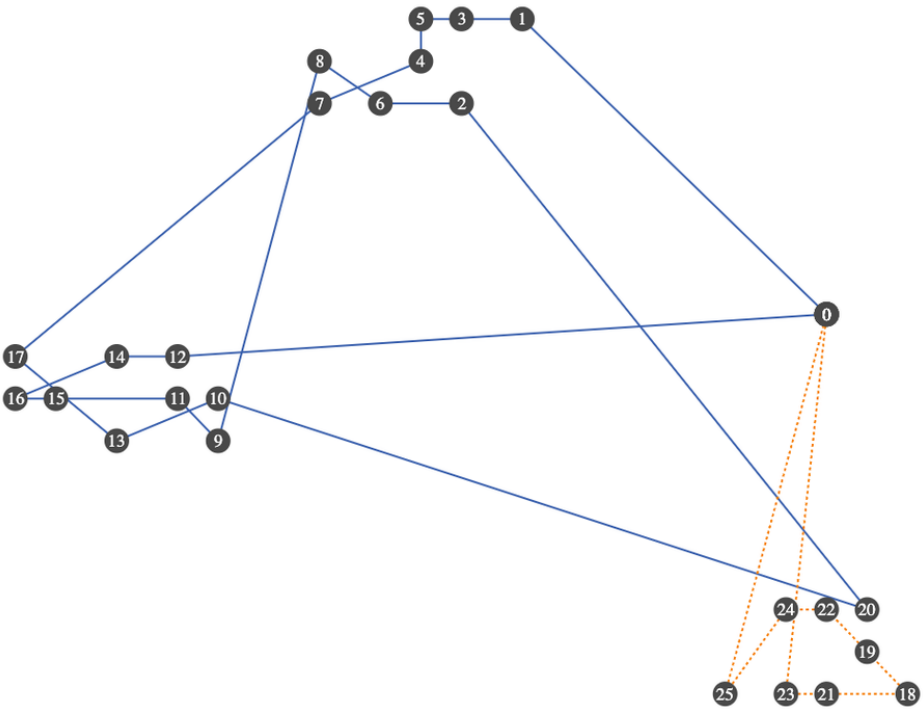
B.4 Instâncias RC2

Figura 72 – RC201



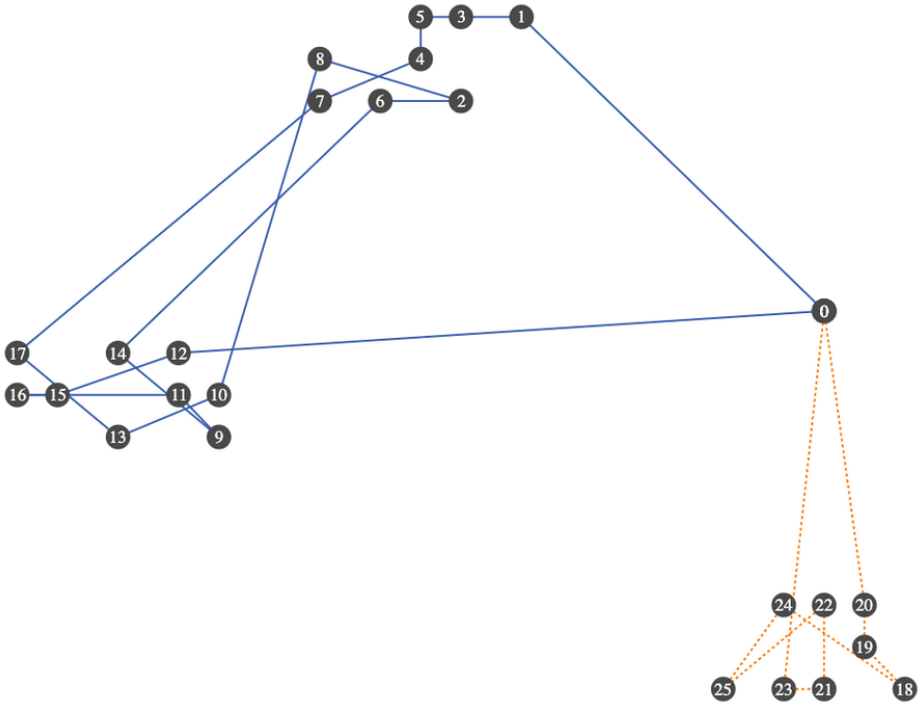
Elaborado pelo autor

Figura 73 – RC202



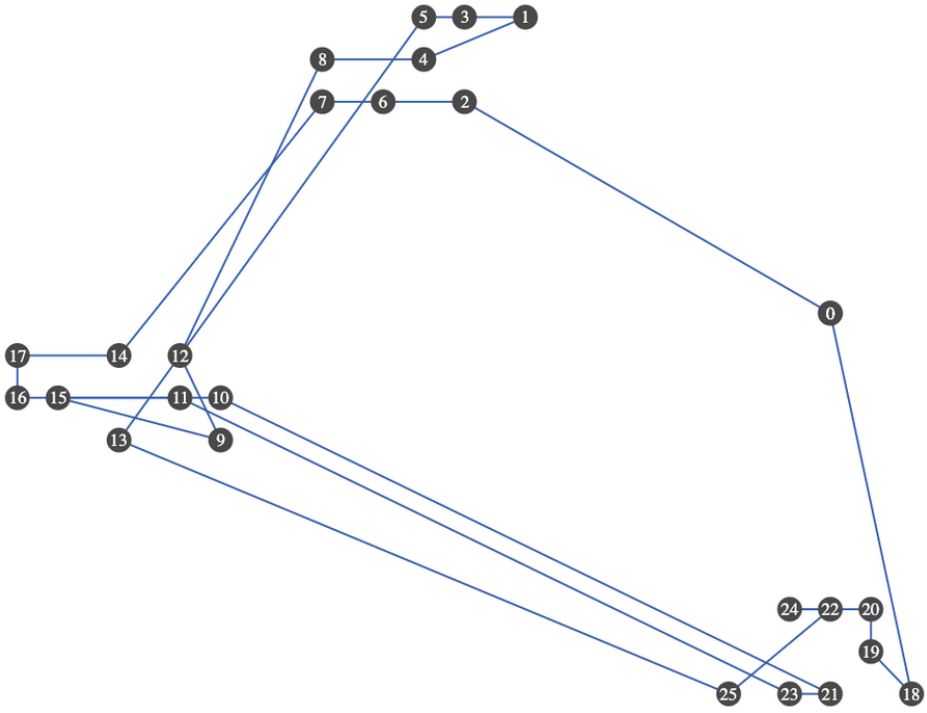
Elaborado pelo autor

Figura 74 – RC203



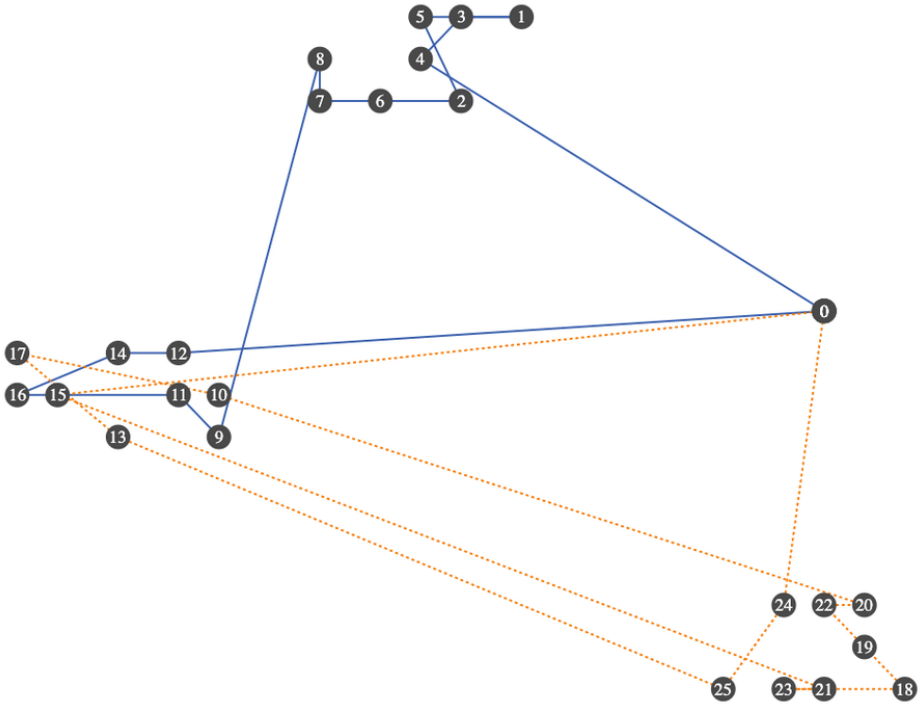
Elaborado pelo autor

Figura 75 – RC204



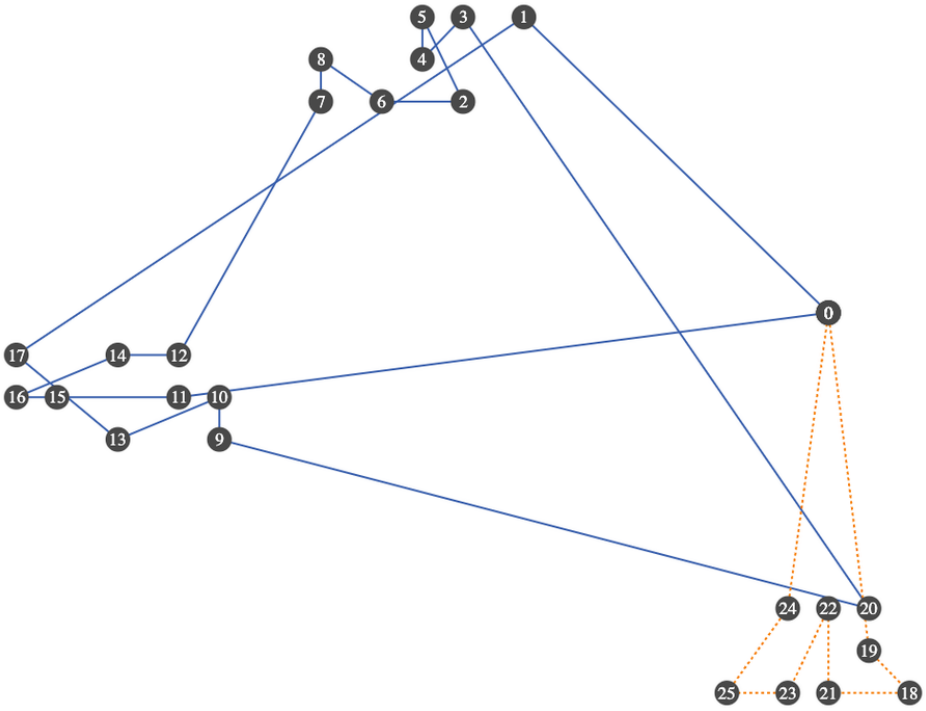
Elaborado pelo autor

Figura 76 – RC205



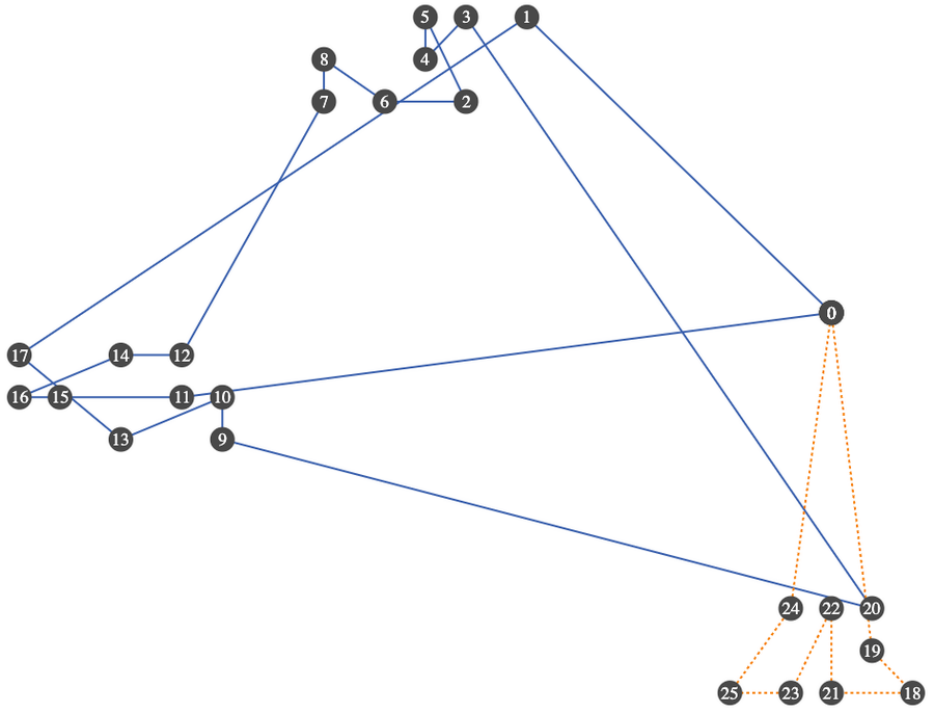
Elaborado pelo autor

Figura 77 – RC206



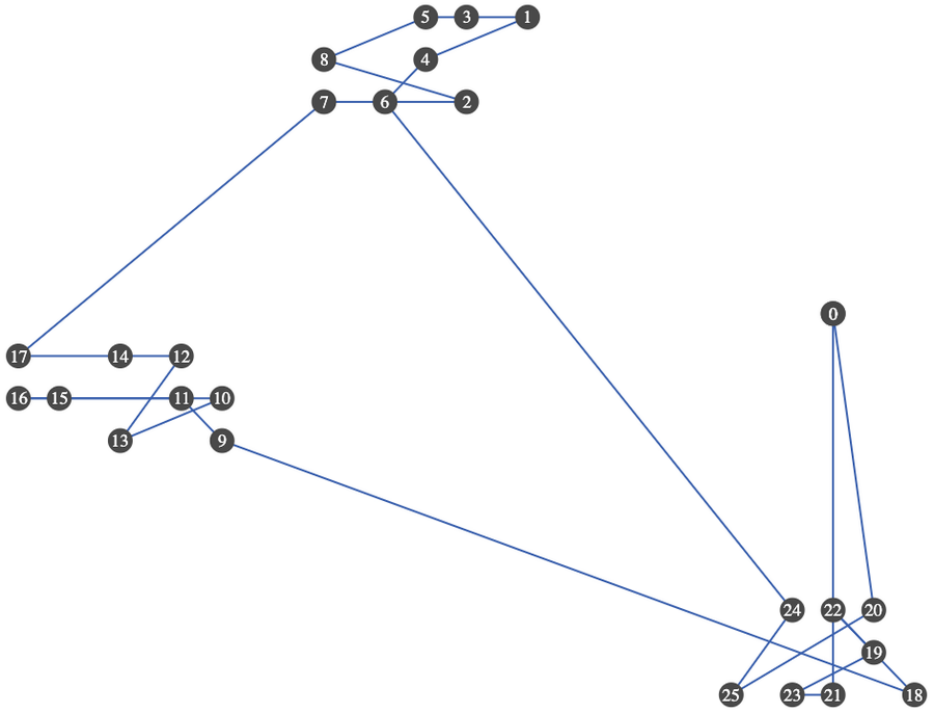
Elaborado pelo autor

Figura 78 – RC207



Elaborado pelo autor

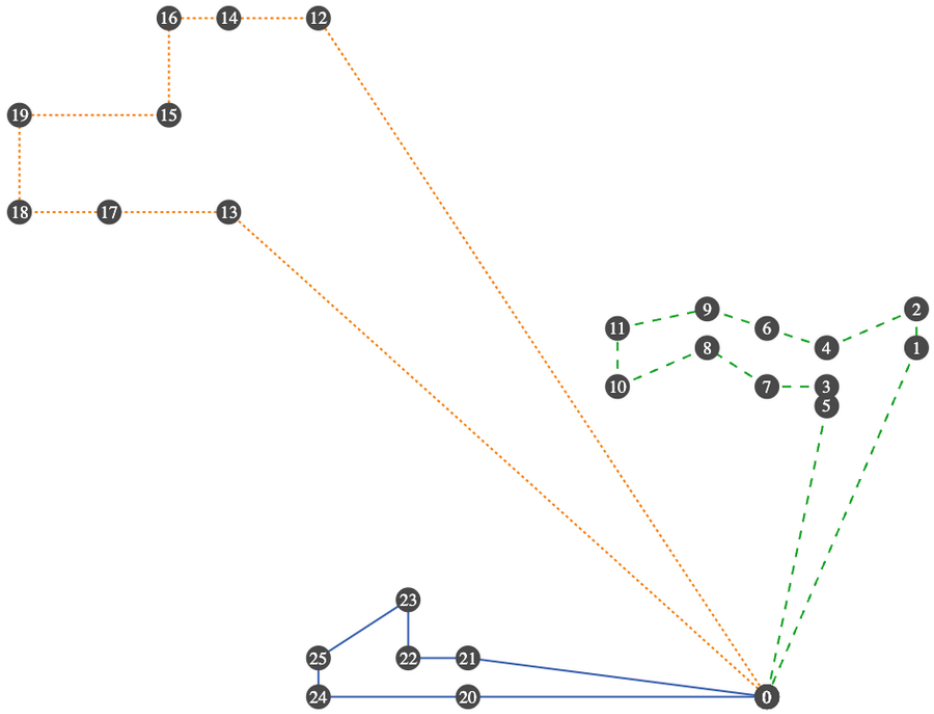
Figura 79 – RC208



Elaborado pelo autor

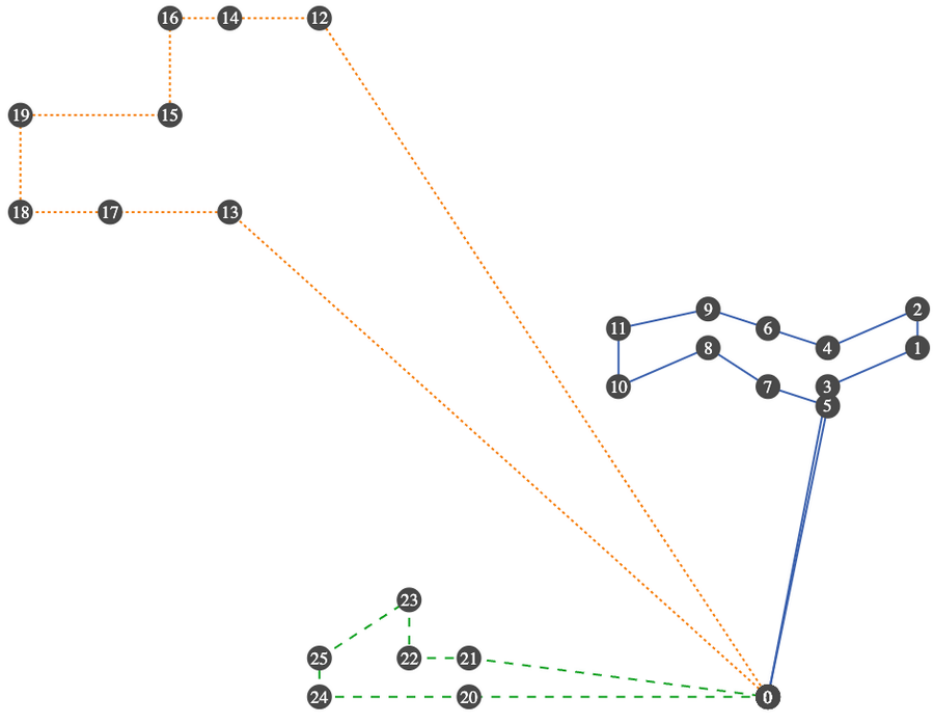
B.5 Instâncias C1

Figura 80 – C101



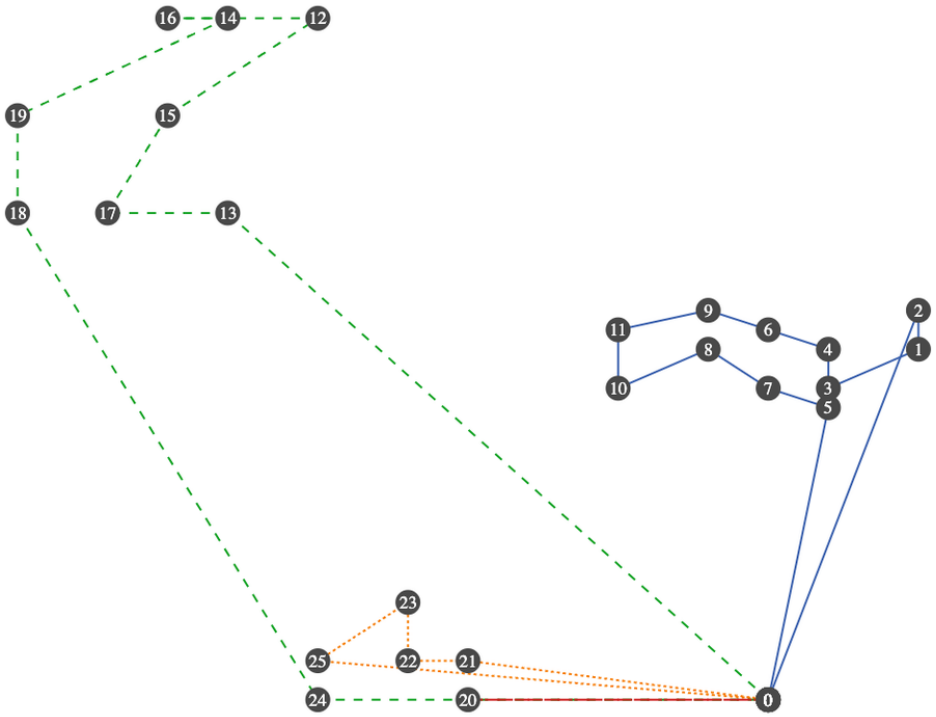
Elaborado pelo autor

Figura 81 – C102



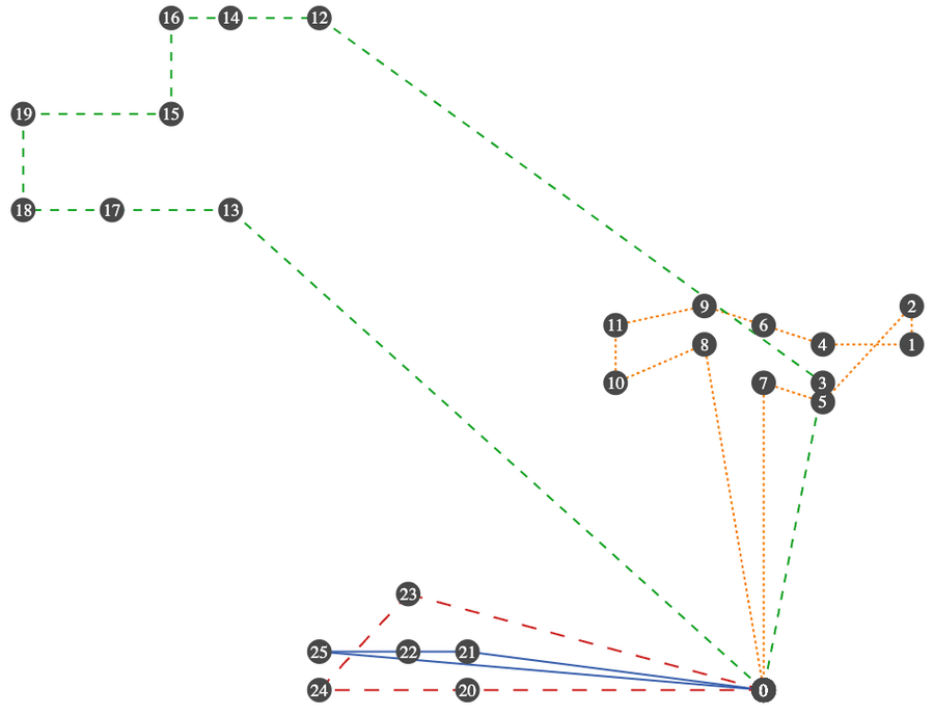
Elaborado pelo autor

Figura 82 – C103



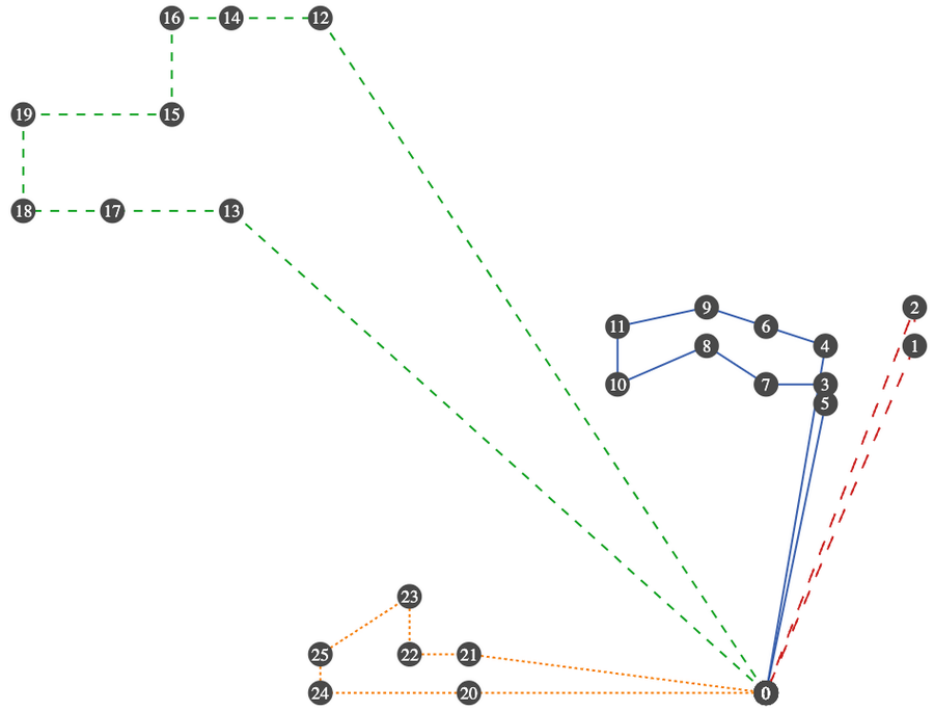
Elaborado pelo autor

Figura 83 – C104



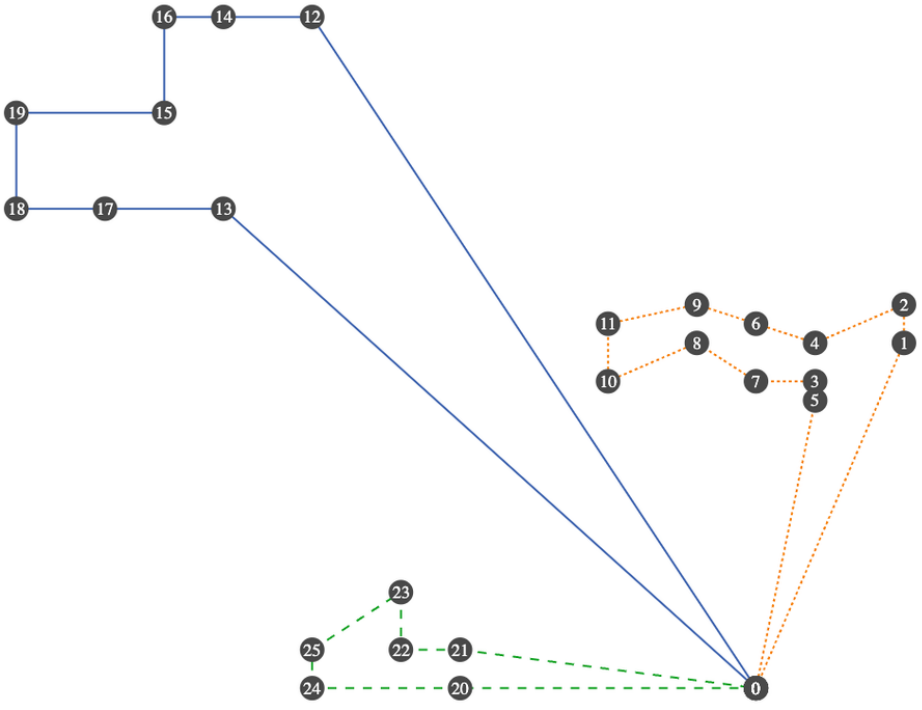
Elaborado pelo autor

Figura 84 – C105



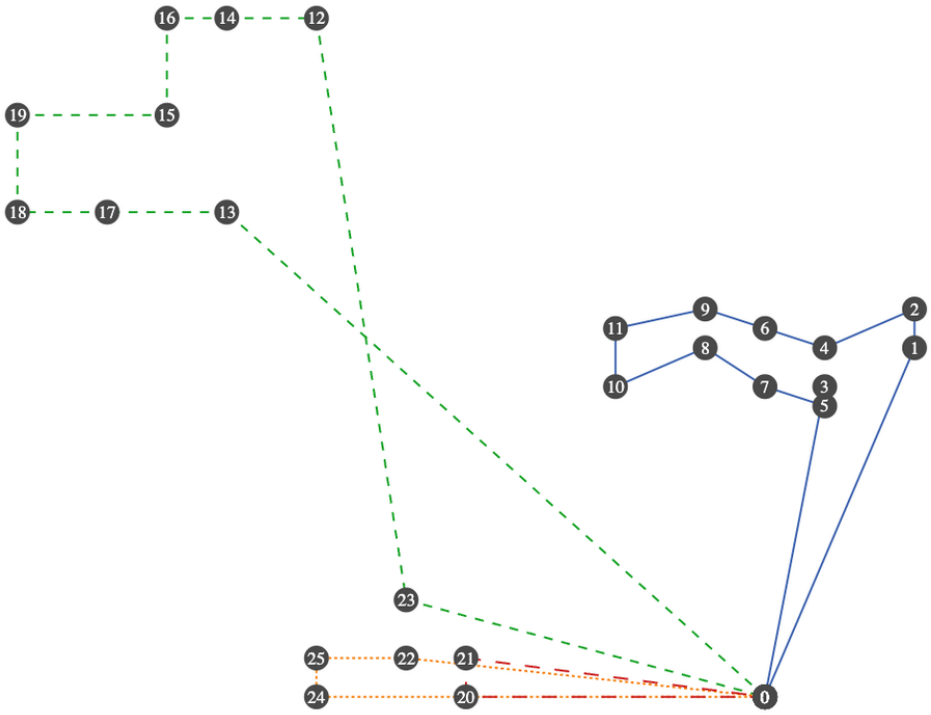
Elaborado pelo autor

Figura 85 – C106



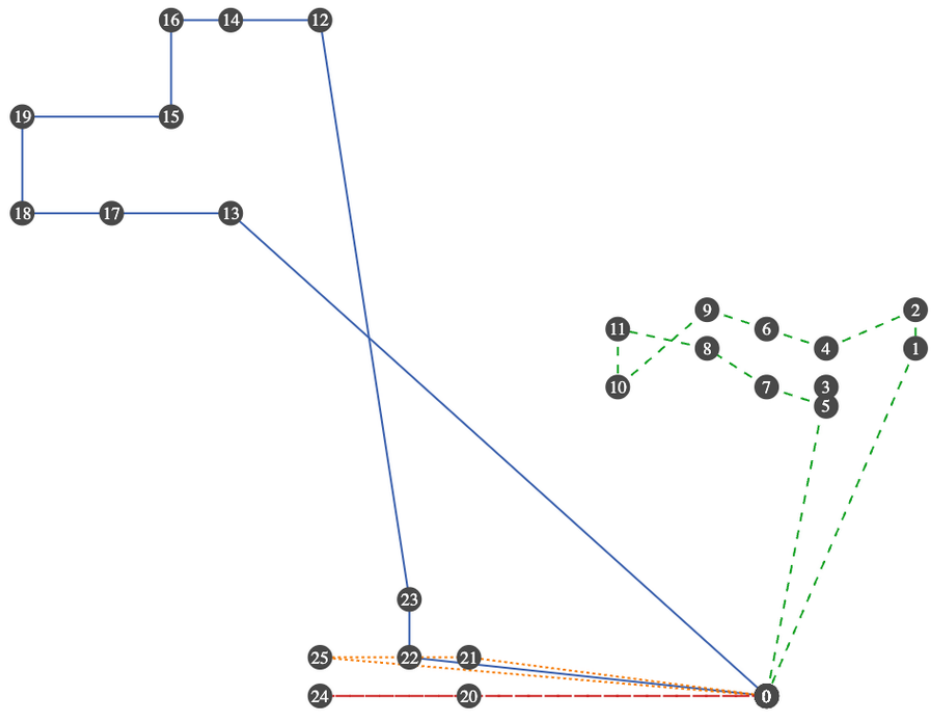
Elaborado pelo autor

Figura 86 – C107



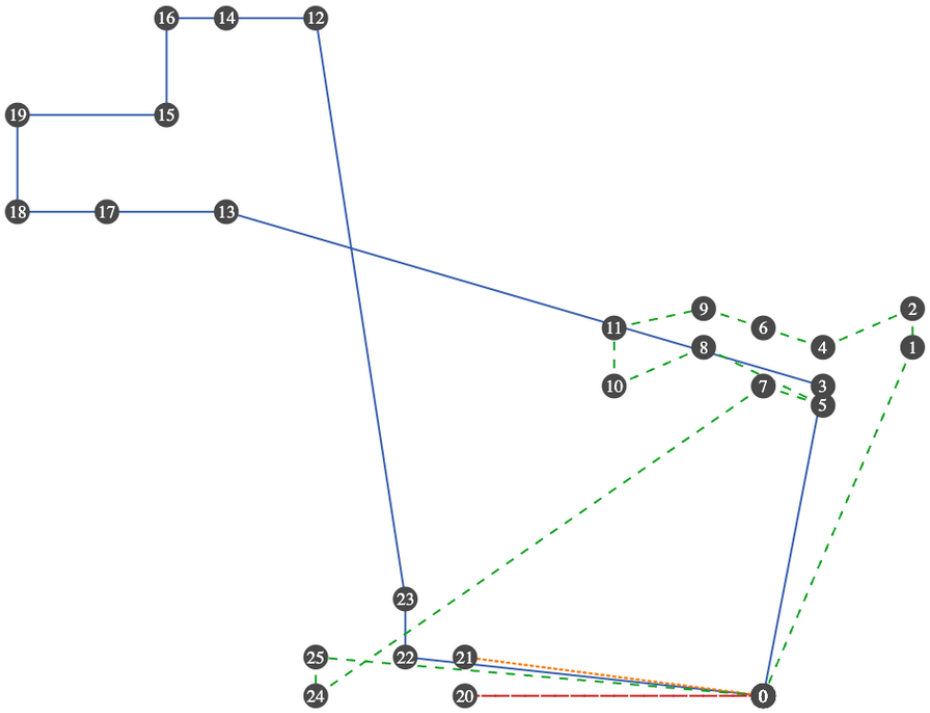
Elaborado pelo autor

Figura 87 – C108



Elaborado pelo autor

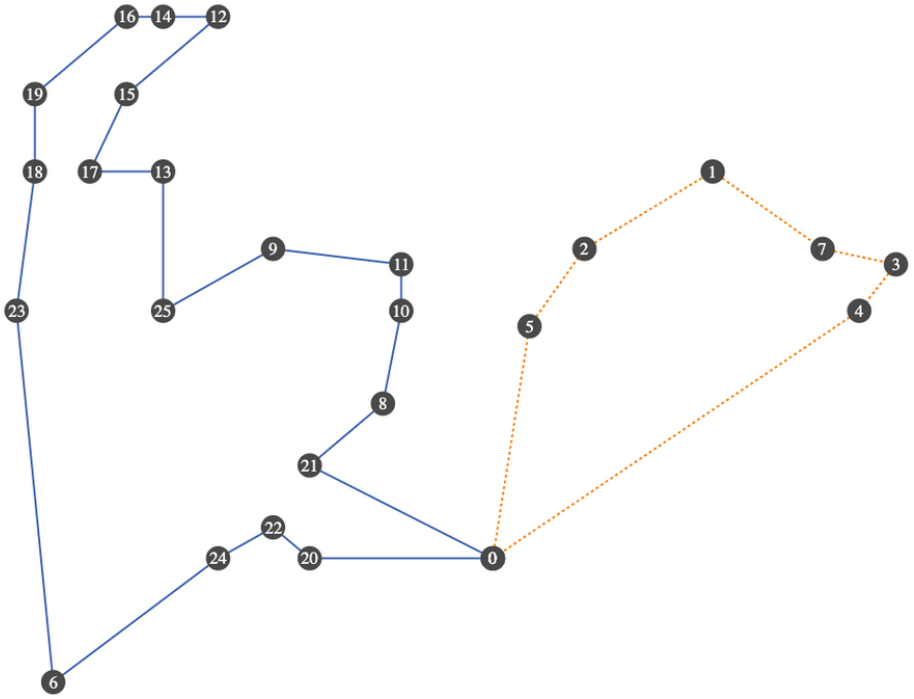
Figura 88 – C109



Elaborado pelo autor

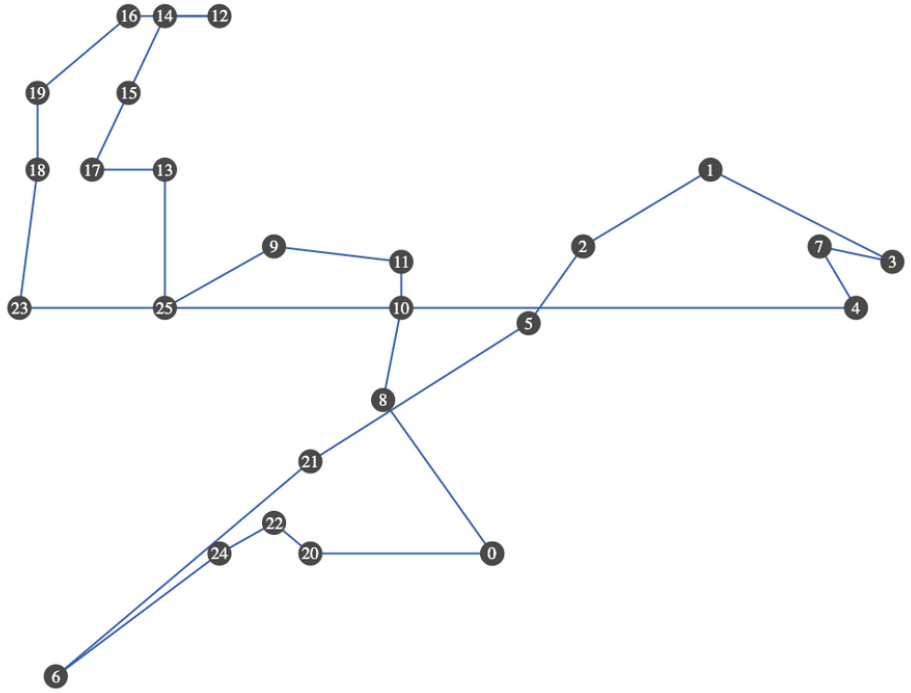
B.6 Instâncias C2

Figura 89 – C201



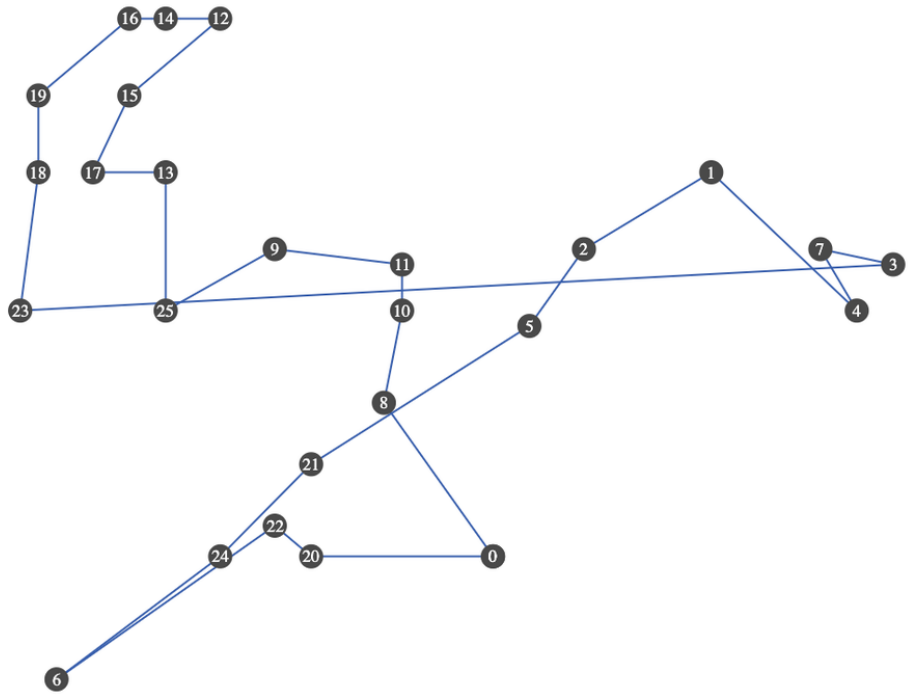
Elaborado pelo autor

Figura 90 – C202



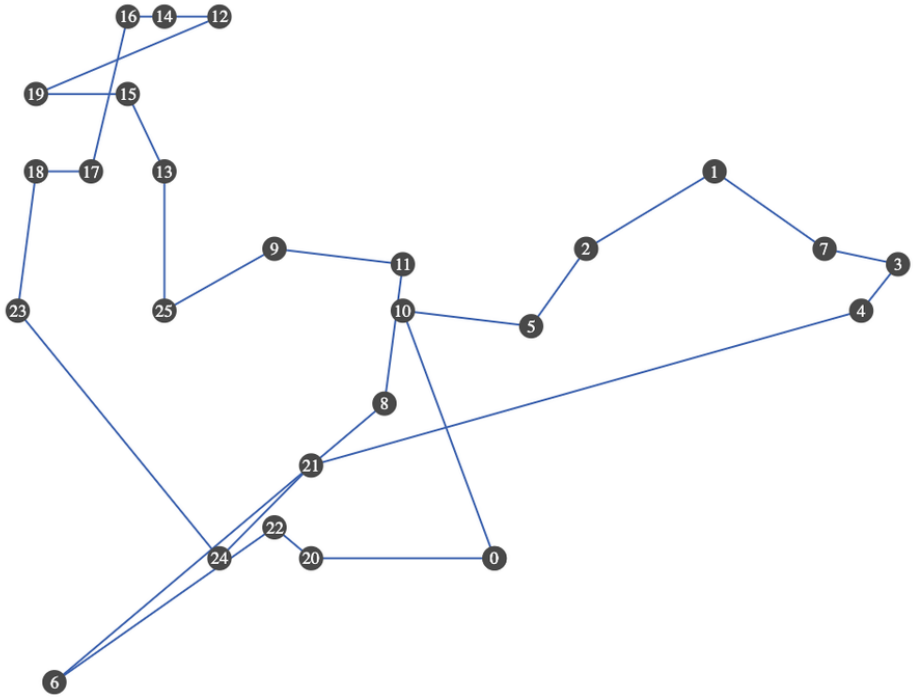
Elaborado pelo autor

Figura 91 – C203



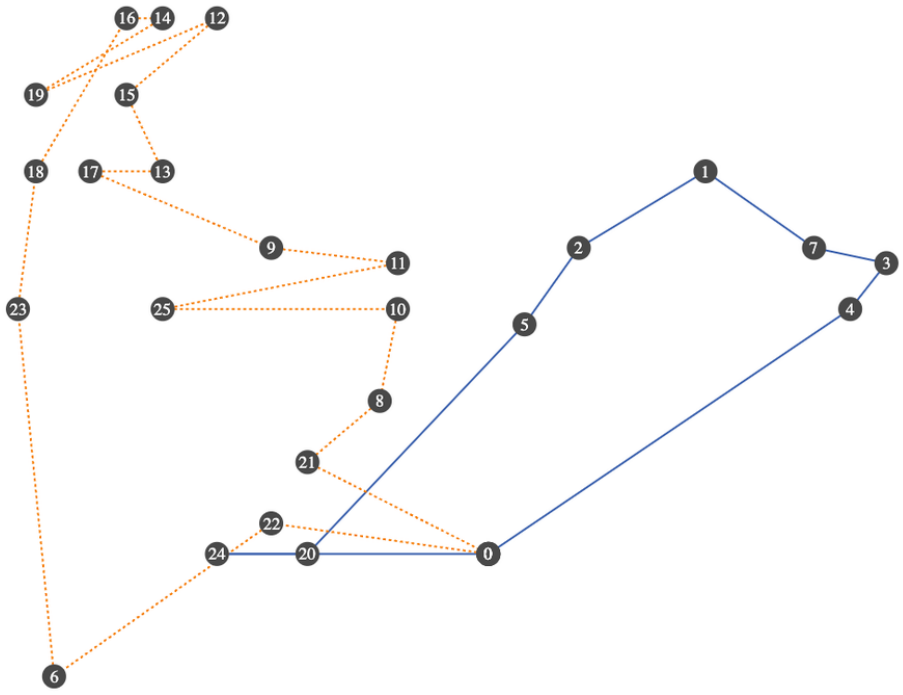
Elaborado pelo autor

Figura 92 – C204



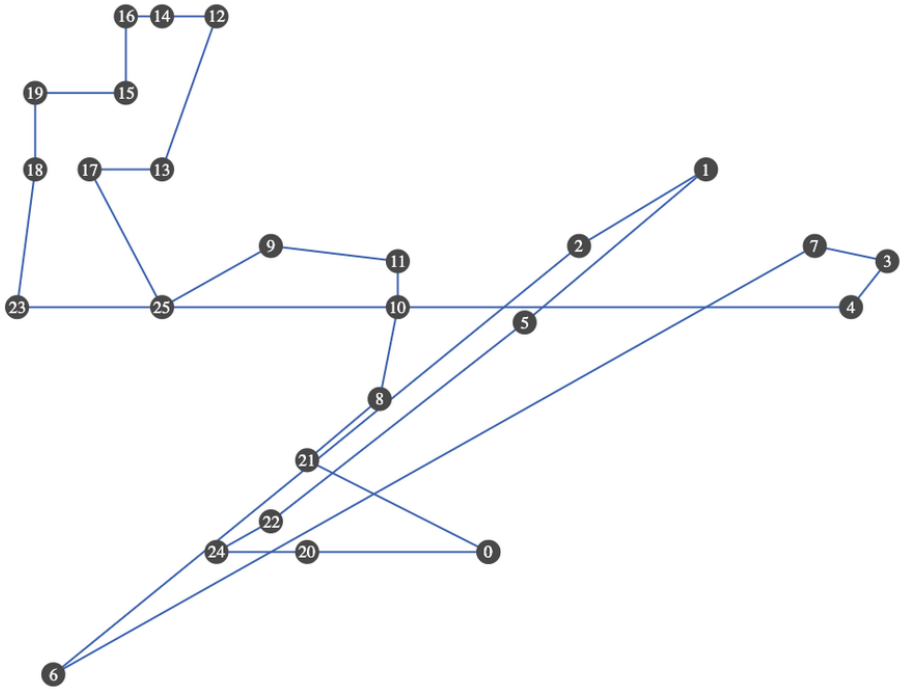
Elaborado pelo autor

Figura 93 – C205



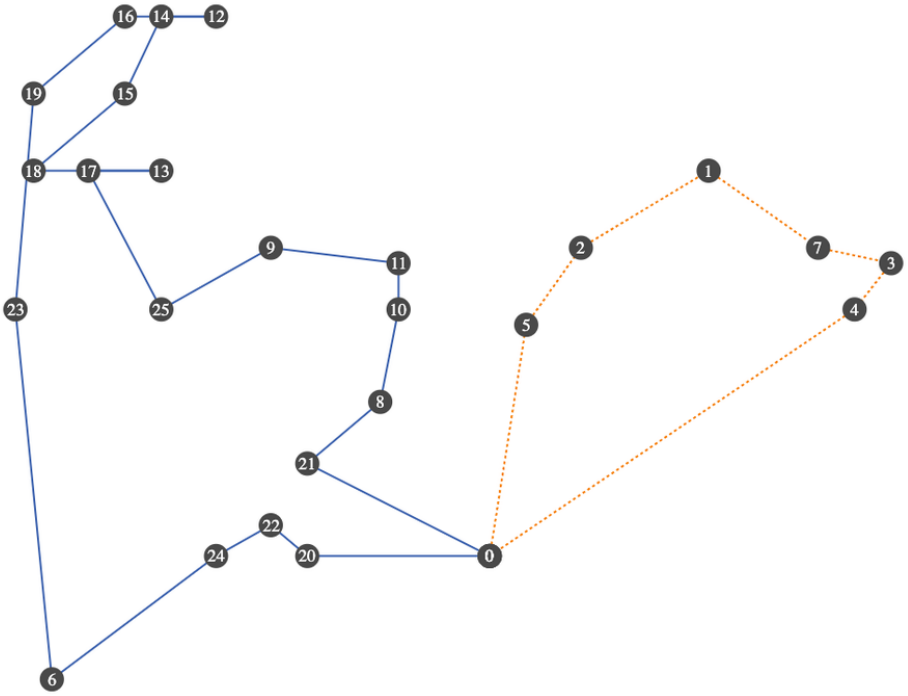
Elaborado pelo autor

Figura 94 – C206



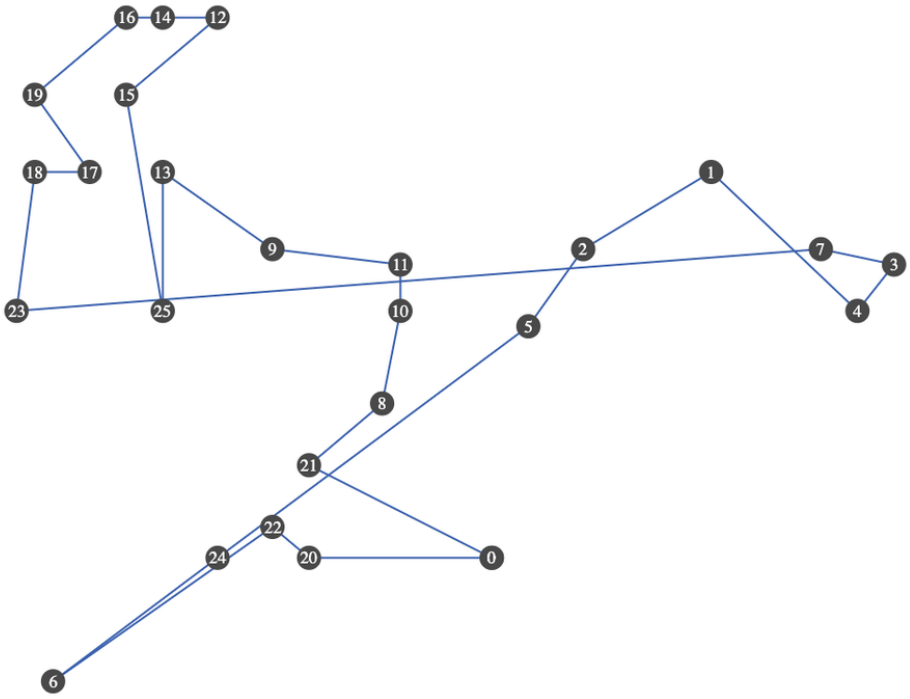
Elaborado pelo autor

Figura 95 – C207



Elaborado pelo autor

Figura 96 – C208

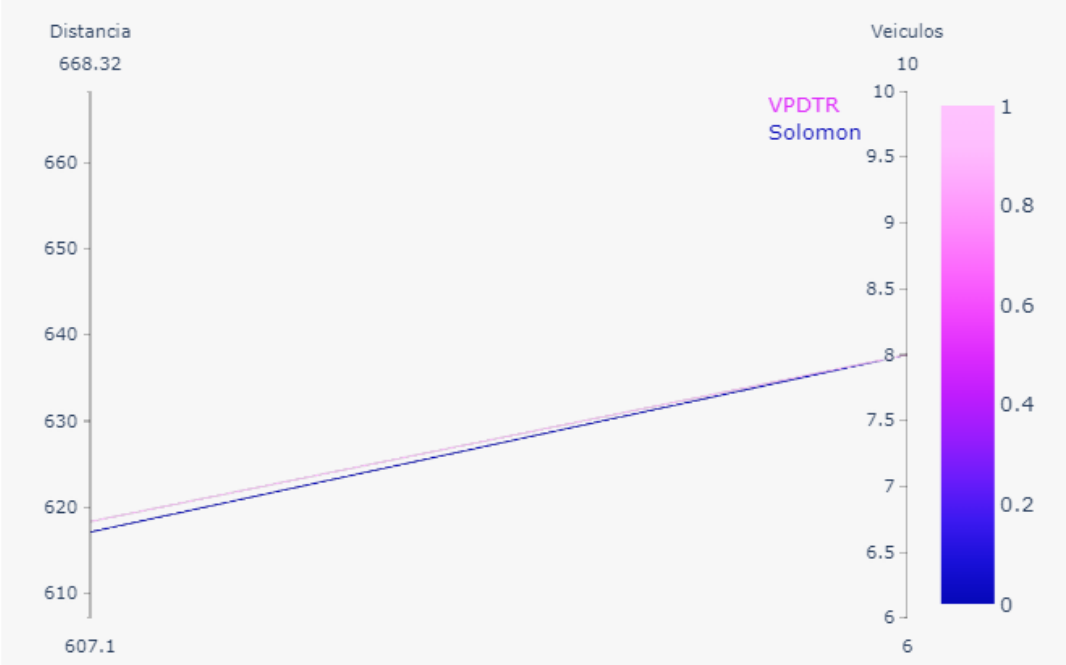


Elaborado pelo autor

APÊNDICE C – Comparação dos resultados para os problemas da Instância R

C.1 Análises dos problemas R1

Figura 97 – R101

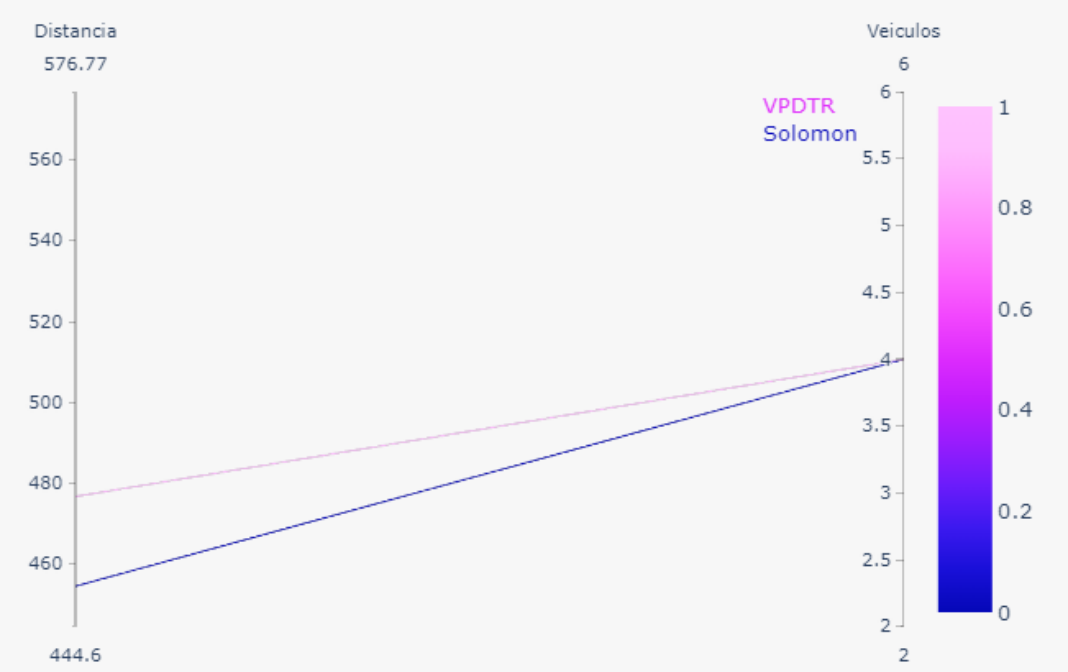


Elaborado pelo autor

Figura 98 – R102

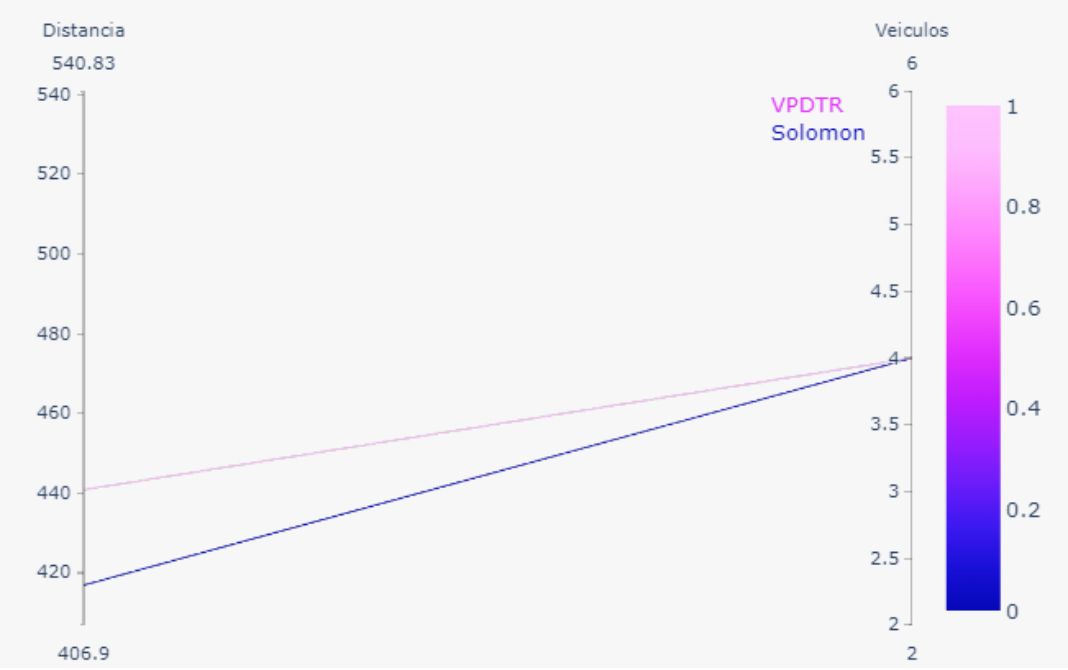
Elaborado pelo autor

Figura 99 – R103



Elaborado pelo autor

Figura 100 – R104

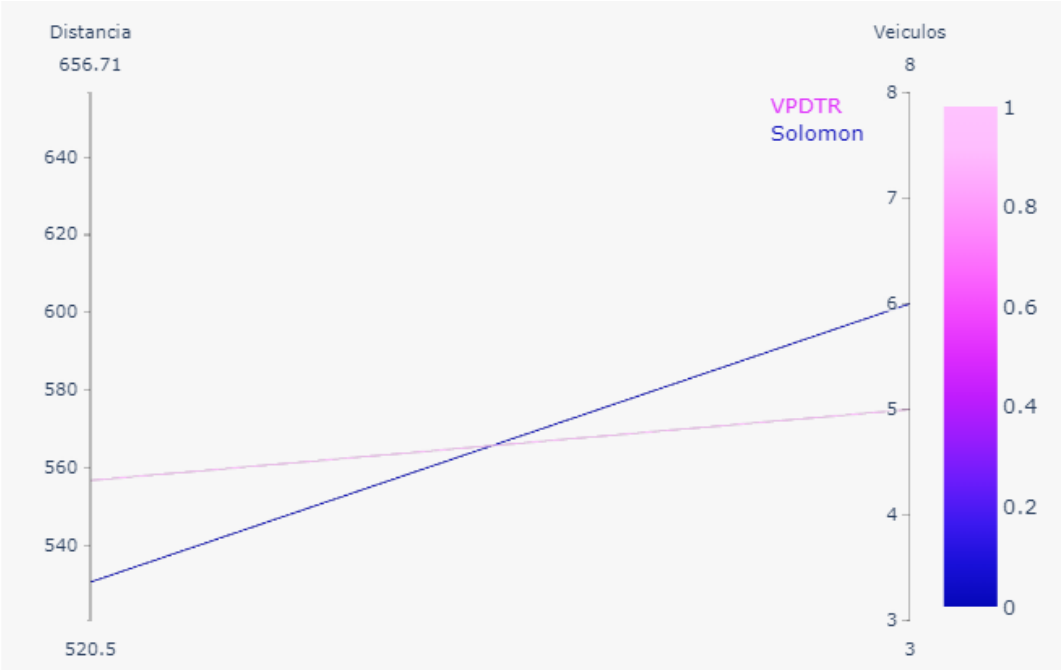


Elaborado pelo autor

Figura 101 – R105a

Elaborado pelo autor

Figura 102 – R105b



Elaborado pelo autor

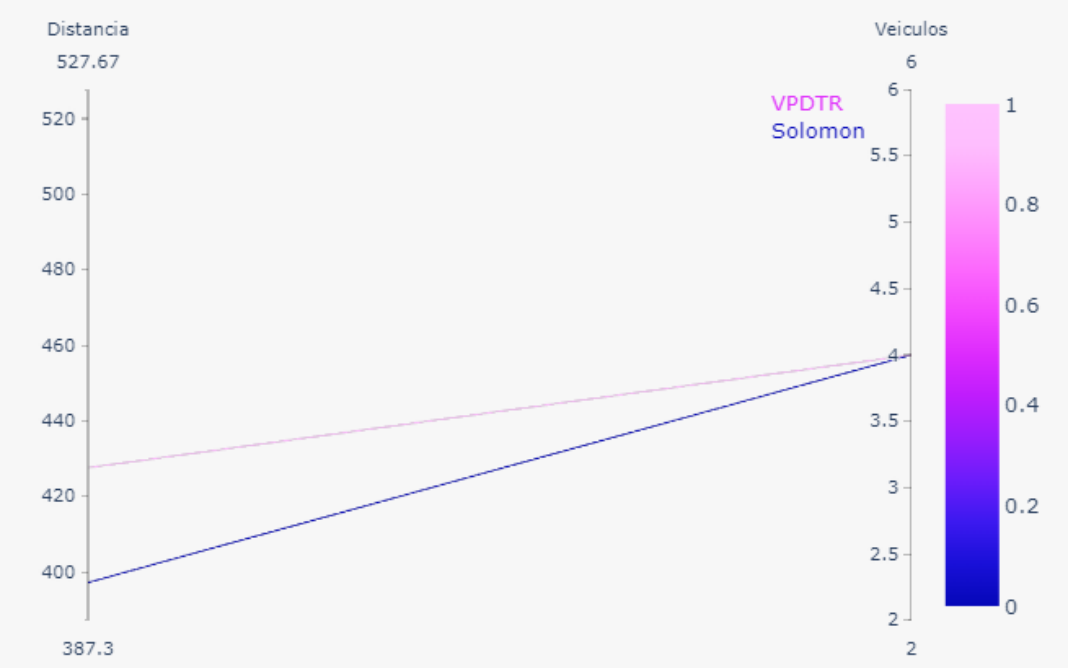
Figura 103 – R106

Elaborado pelo autor

Figura 104 – R107

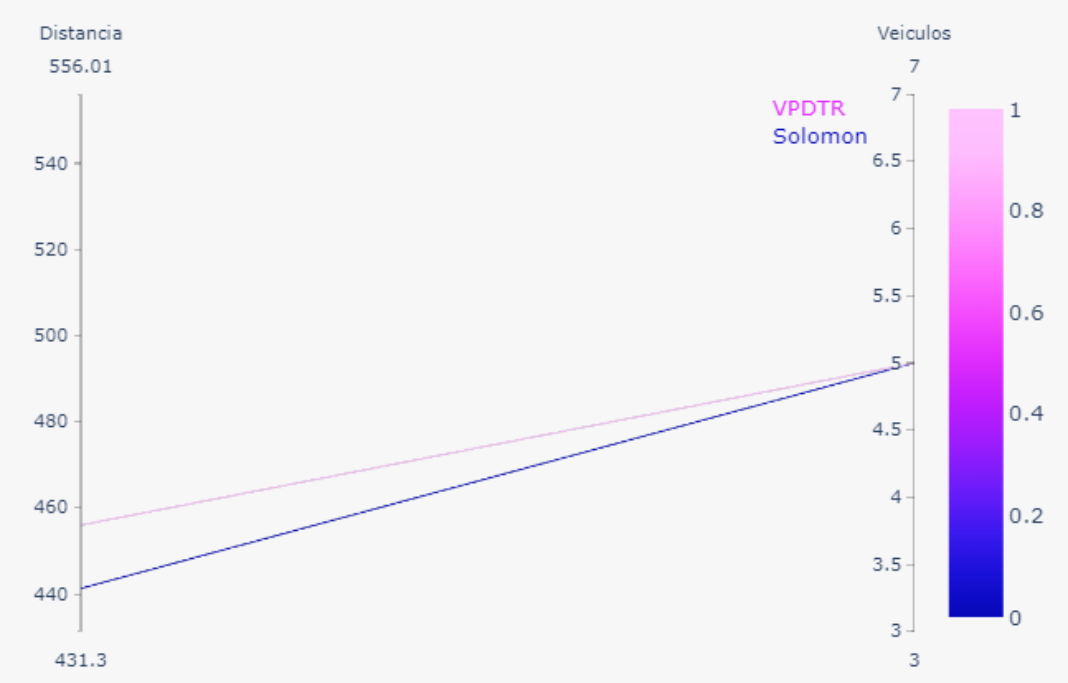
Elaborado pelo autor

Figura 105 – R108



Elaborado pelo autor

Figura 106 – R109a



Elaborado pelo autor

Figura 107 – R109b

Elaborado pelo autor

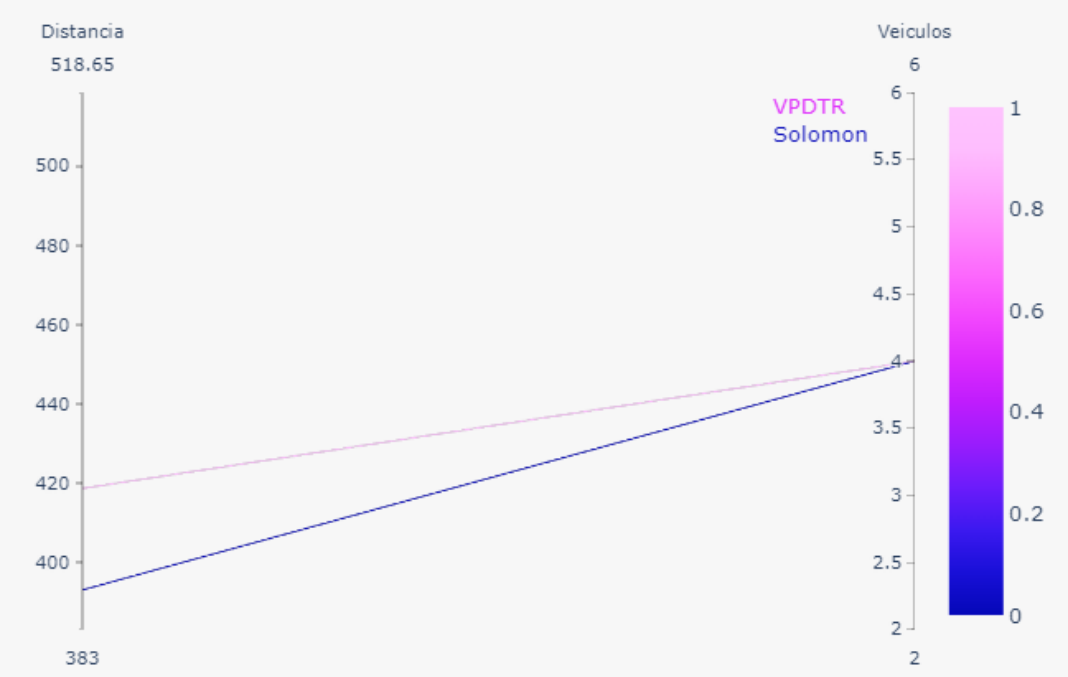
Figura 108 – R110

Elaborado pelo autor

Figura 109 – R111

Elaborado pelo autor

Figura 110 – R112



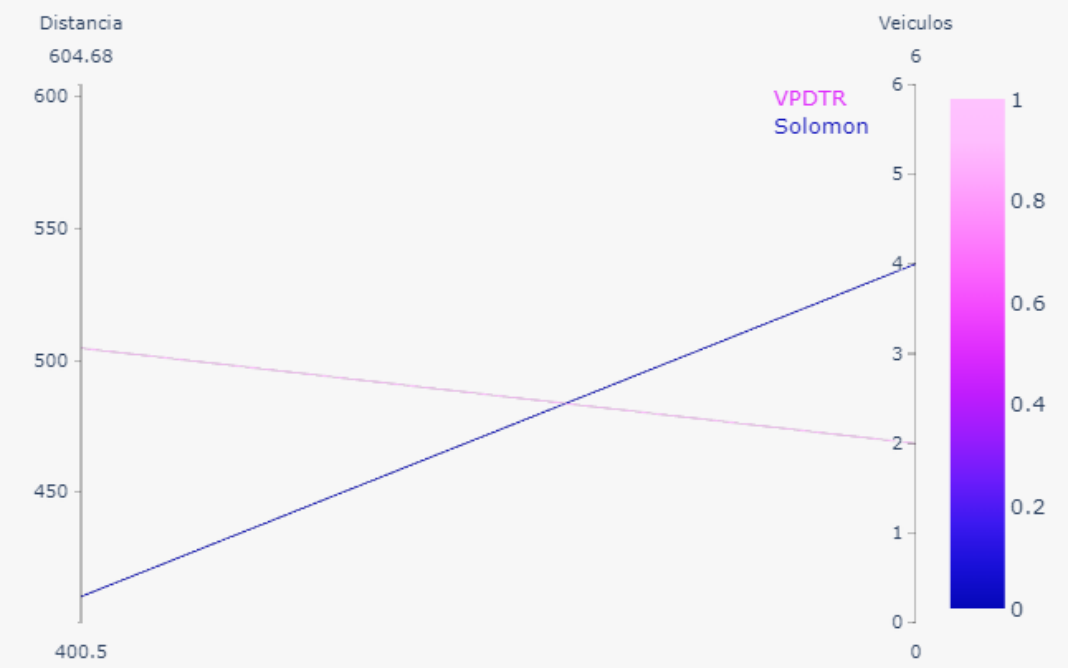
Elaborado pelo autor

C.2 Análises dos problemas R2

Figura 111 – R201

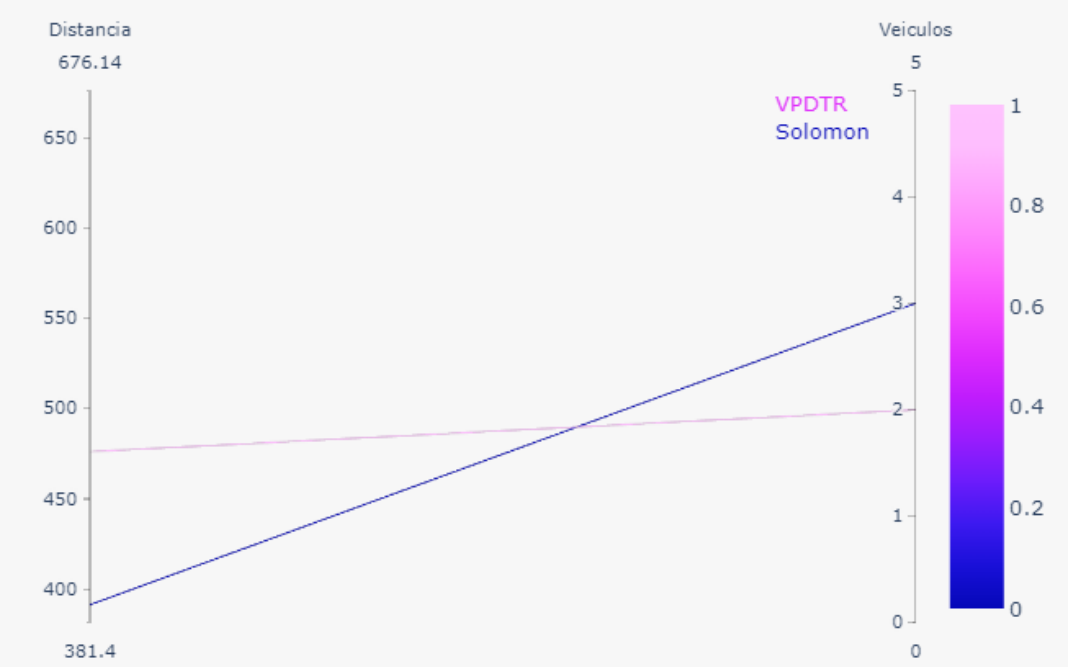
Elaborado pelo autor

Figura 112 – R202



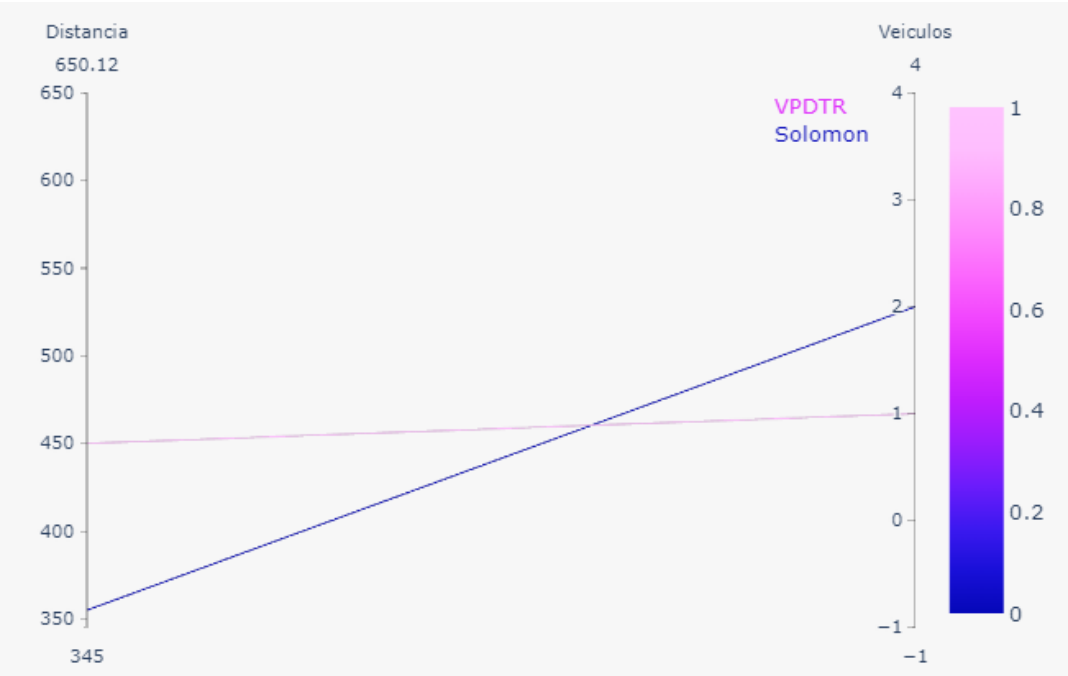
Elaborado pelo autor

Figura 113 – R203



Elaborado pelo autor

Figura 114 – R204



Elaborado pelo autor

Figura 115 – R205

Elaborado pelo autor

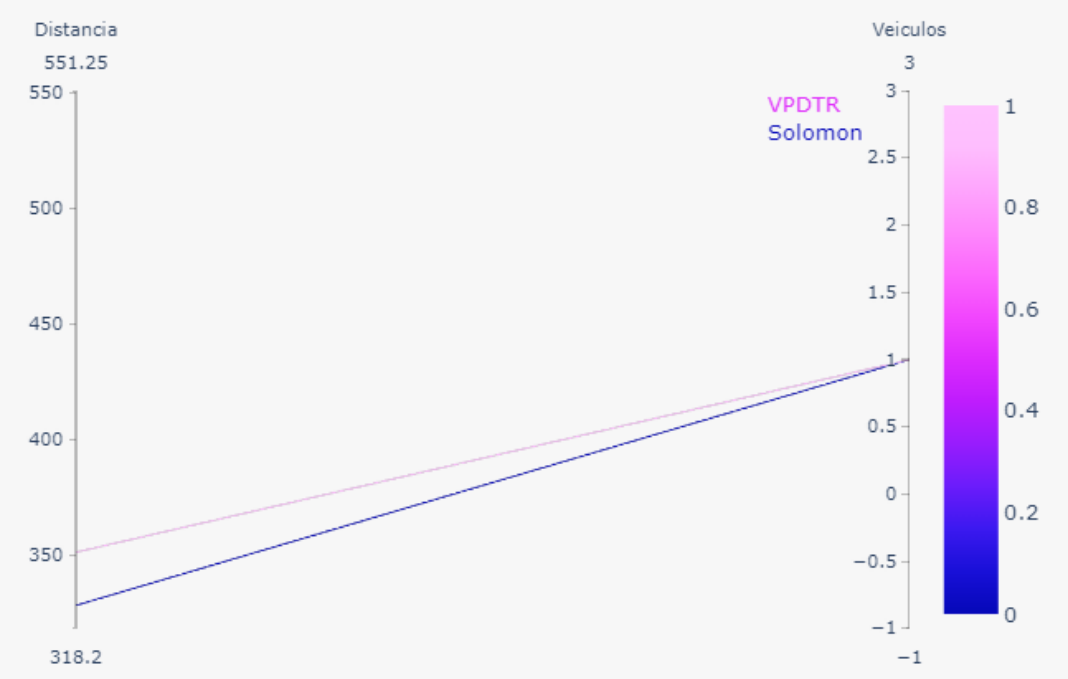
Figura 116 – R206

Elaborado pelo autor

Figura 117 – R207

Elaborado pelo autor

Figura 118 – R208



Elaborado pelo autor

Figura 119 – R209

Elaborado pelo autor

Figura 120 – R210

Elaborado pelo autor

Figura 121 – R211

Elaborado pelo autor

APÊNDICE D – Comparação dos resultados para os problemas da Instância RC

D.1 Análises dos problemas RC1

Figura 122 – RC101

Elaborado pelo autor

Figura 123 – RC102

Elaborado pelo autor

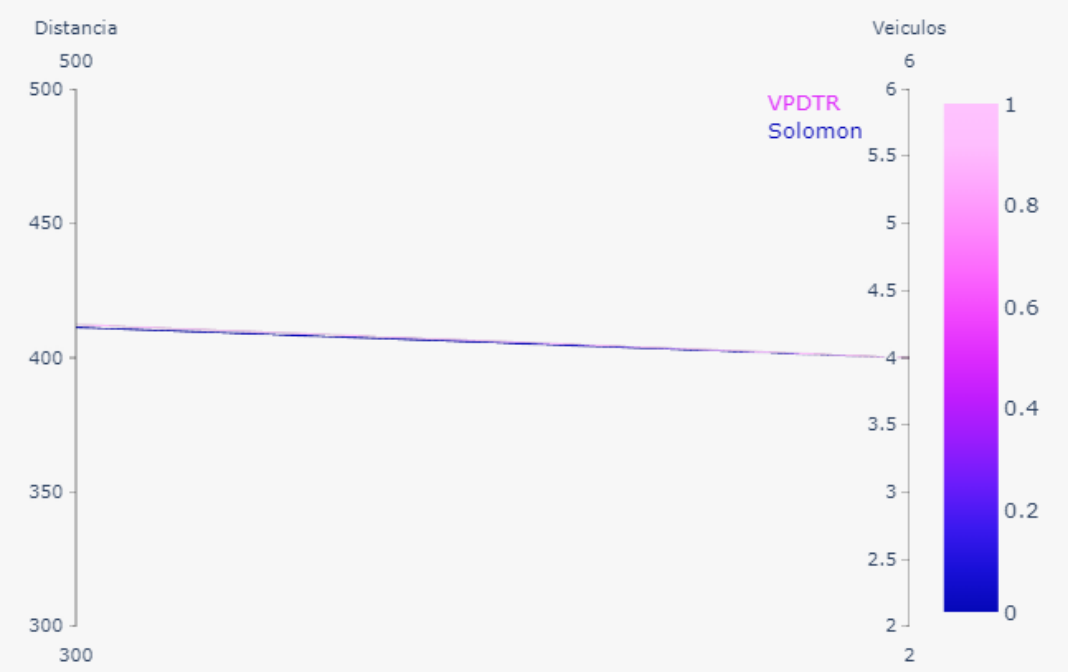
Figura 124 – RC103

Elaborado pelo autor

Figura 125 – RC104

Elaborado pelo autor

Figura 126 – RC105



Elaborado pelo autor

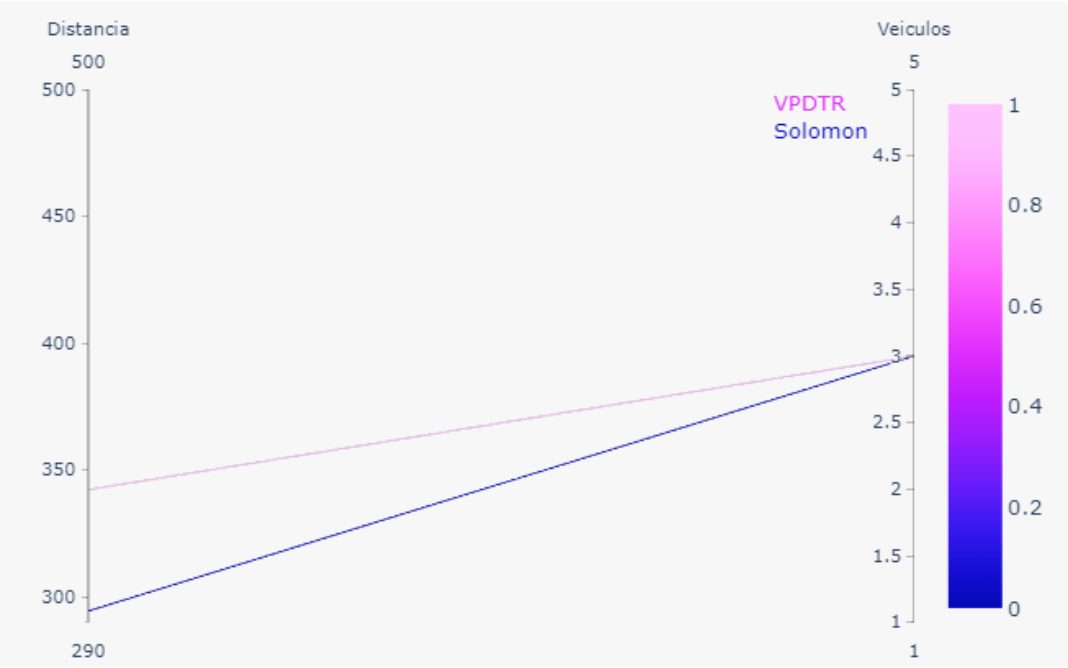
Figura 127 – RC106

Elaborado pelo autor

Figura 128 – RC107

Elaborado pelo autor

Figura 129 – RC108



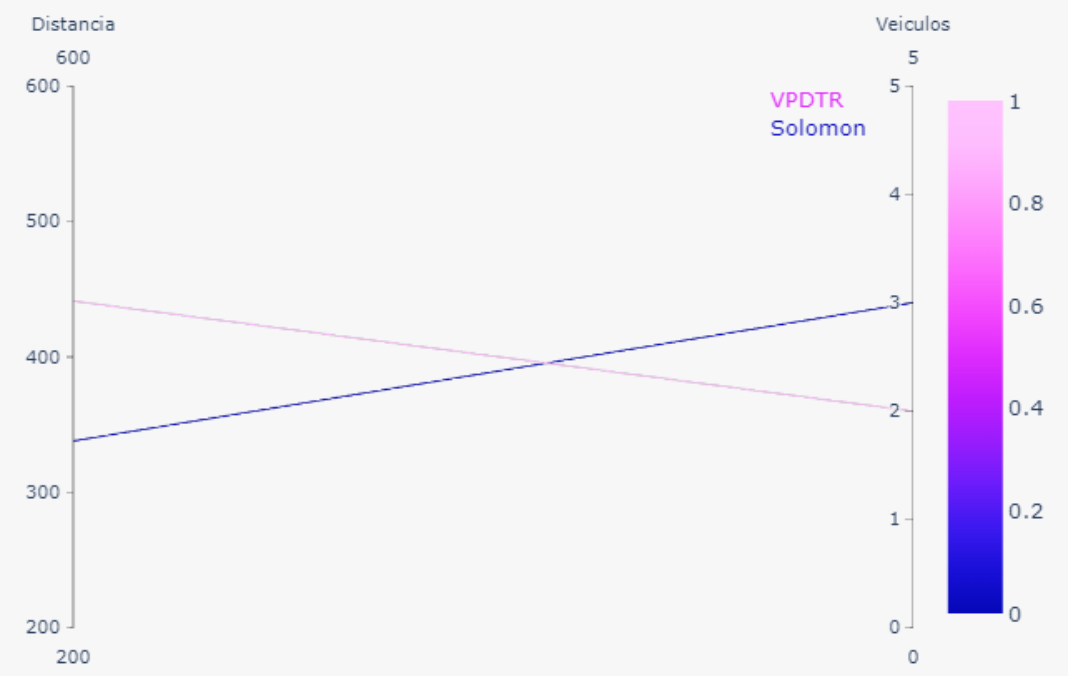
Elaborado pelo autor

D.2 Análises dos problemas RC2

Figura 130 – RC201

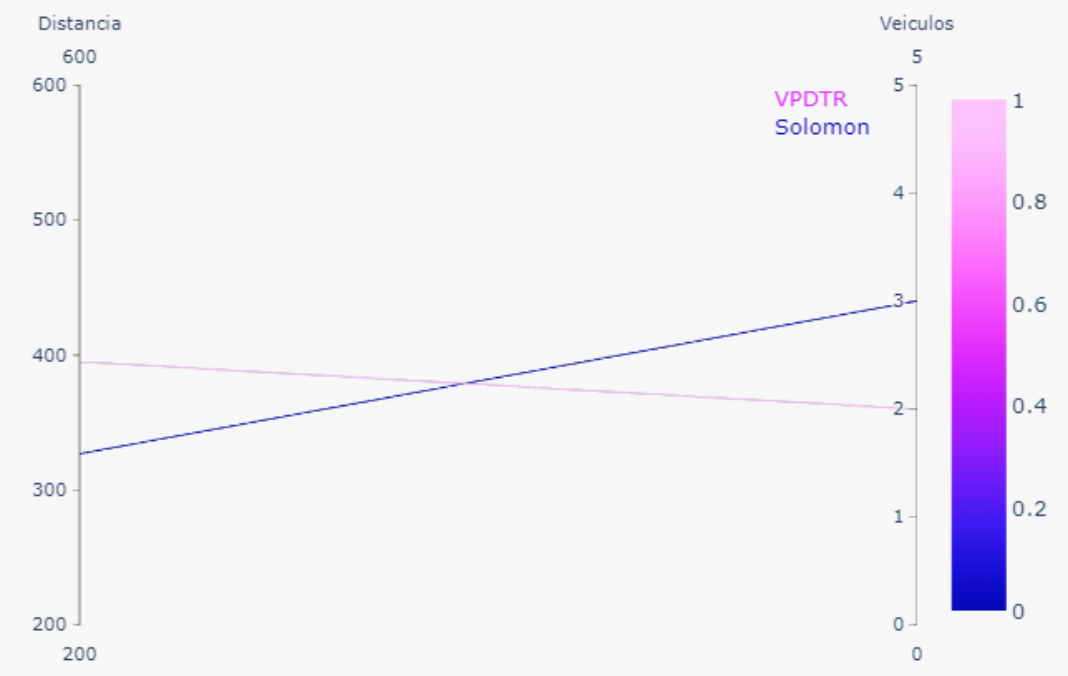
Elaborado pelo autor

Figura 131 – RC202



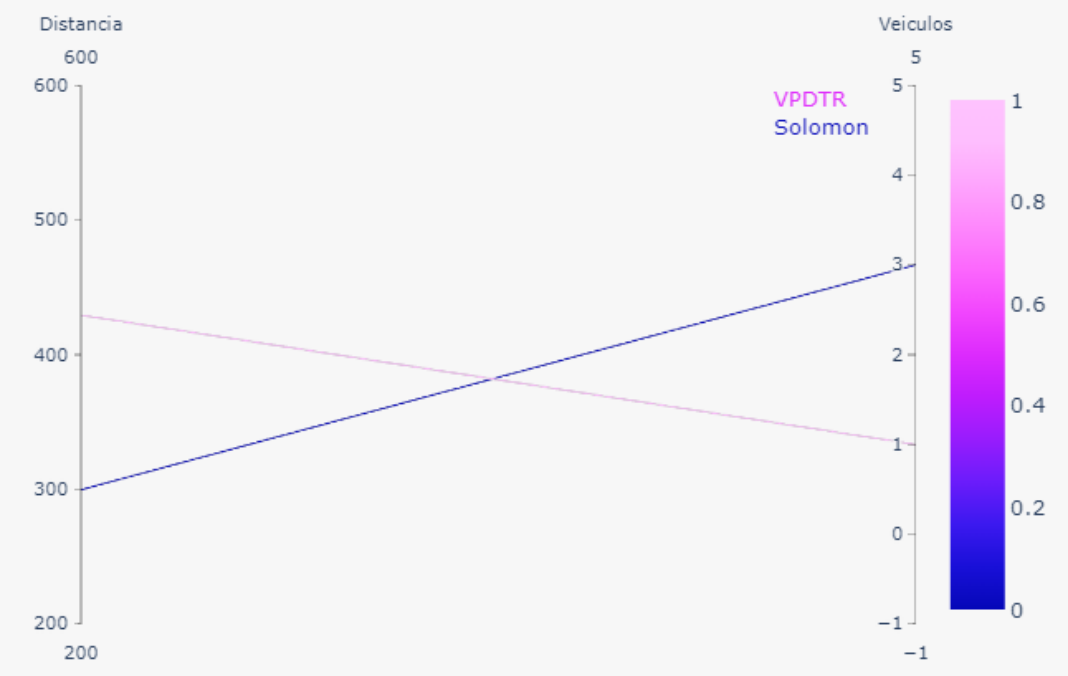
Elaborado pelo autor

Figura 132 – RC203



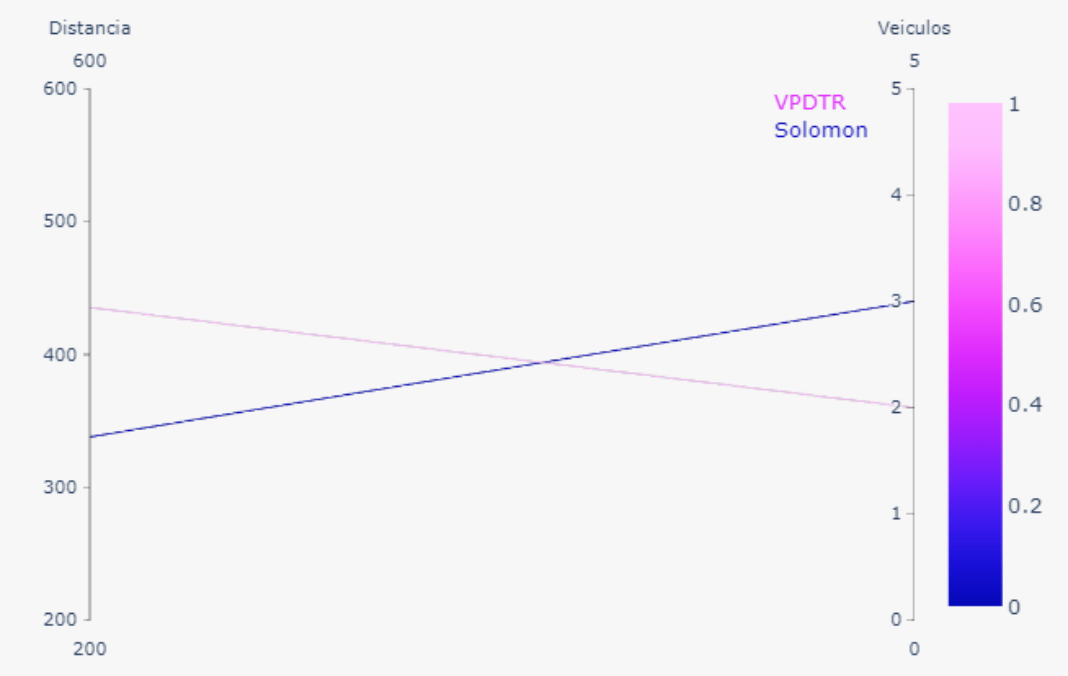
Elaborado pelo autor

Figura 133 – RC204



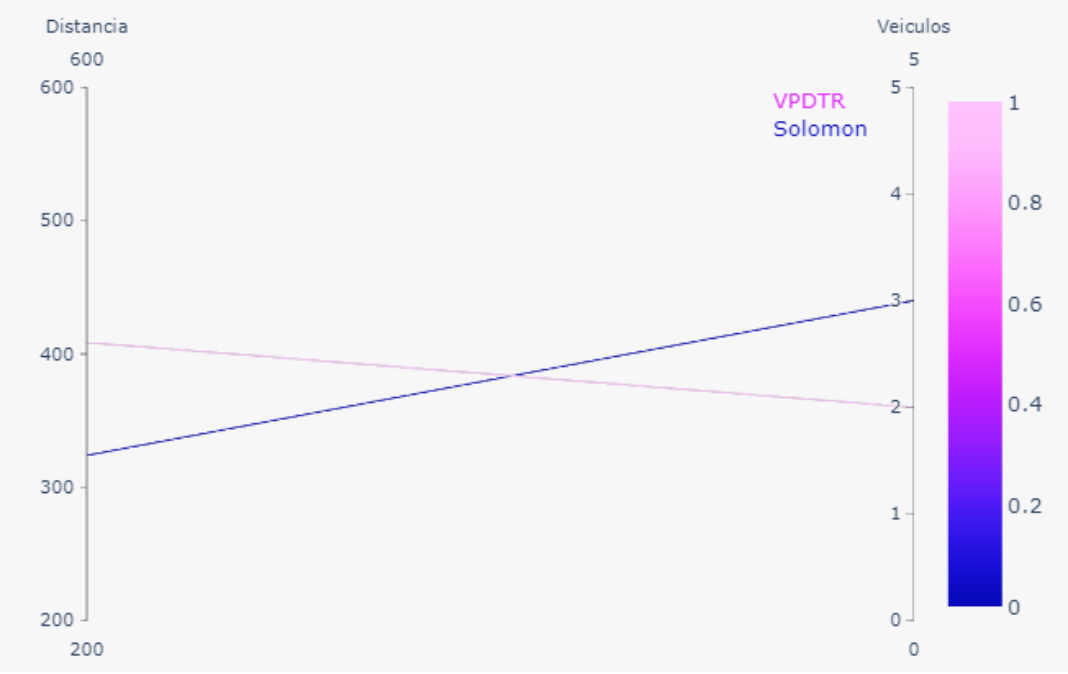
Elaborado pelo autor

Figura 134 – RC205



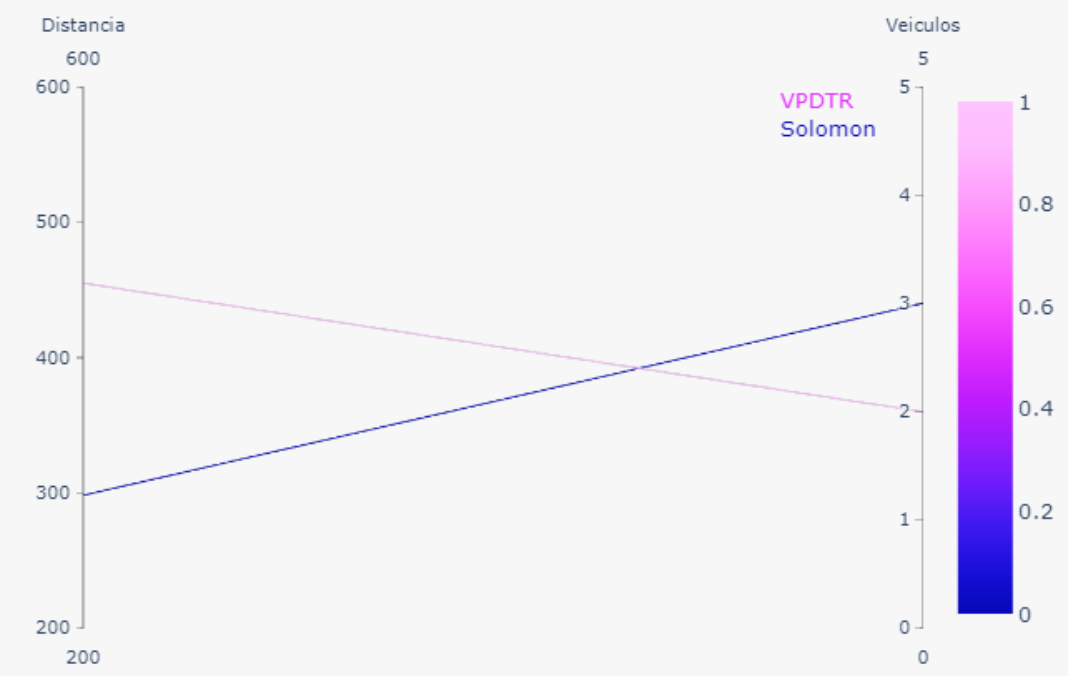
Elaborado pelo autor

Figura 135 – RC206



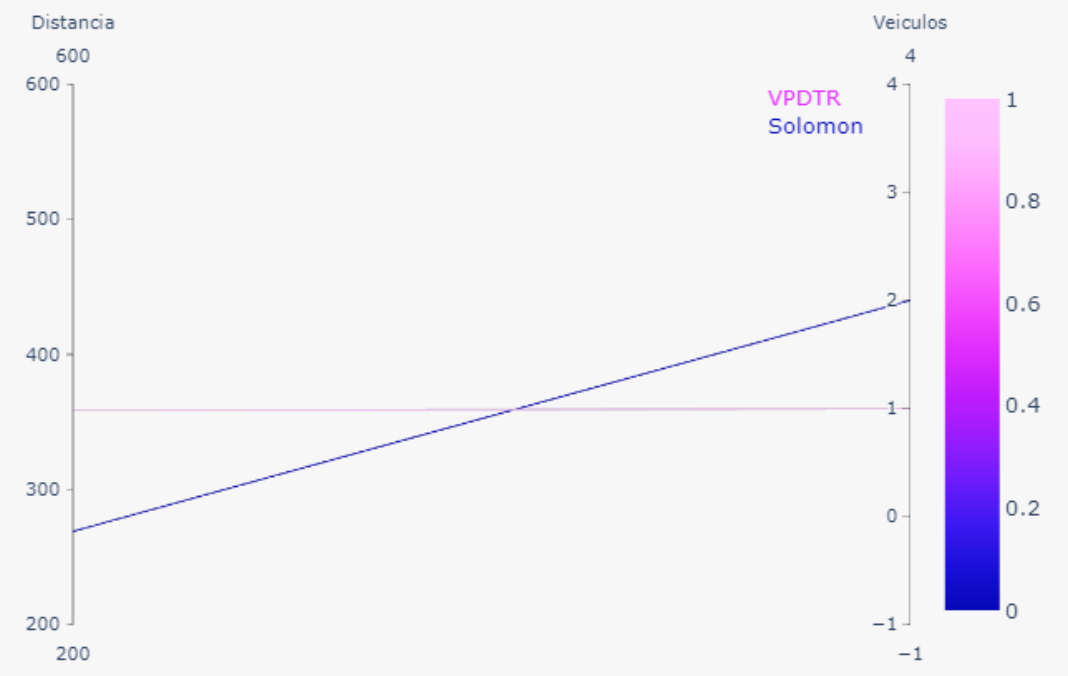
Elaborado pelo autor

Figura 136 – RC207



Elaborado pelo autor

Figura 137 – RC208

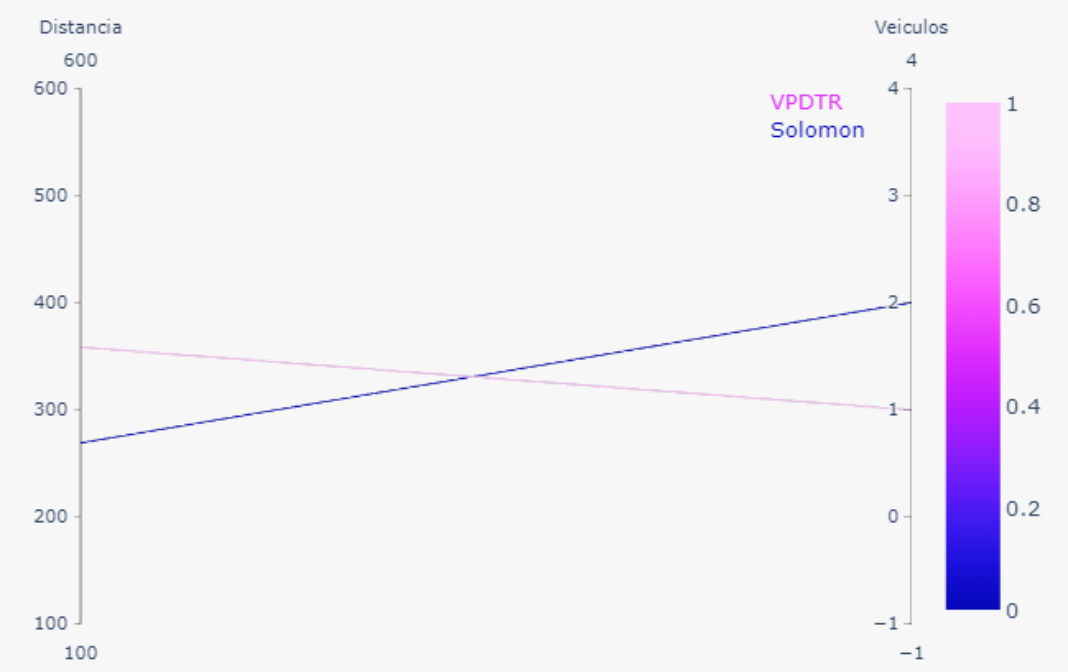


Elaborado pelo autor

APÊNDICE E – Comparação dos resultados para os problemas da Instância C

E.1 Análises dos problemas C1

Figura 138 – C101

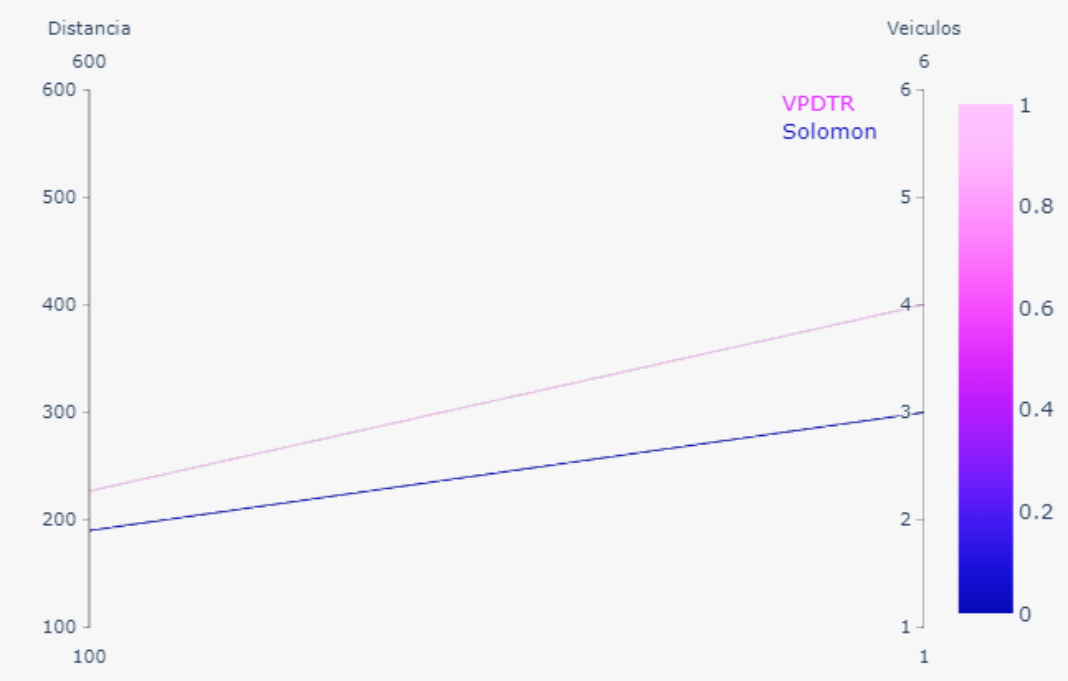


Elaborado pelo autor

Figura 139 – C102

Elaborado pelo autor

Figura 140 – C103

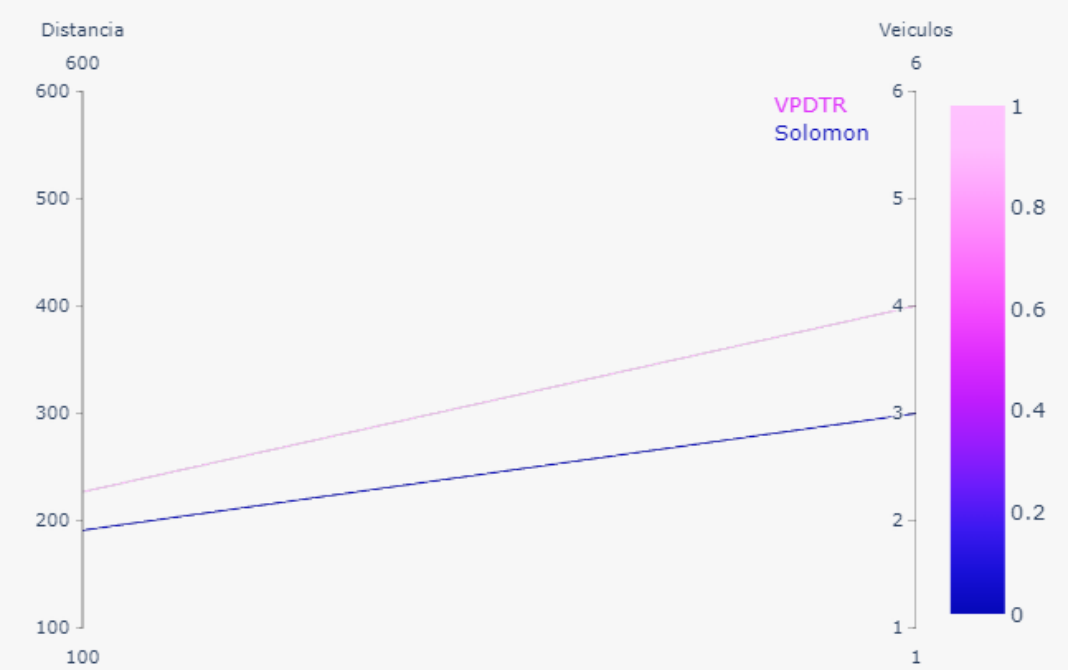


Elaborado pelo autor

Figura 141 – C104

Elaborado pelo autor

Figura 142 – C105

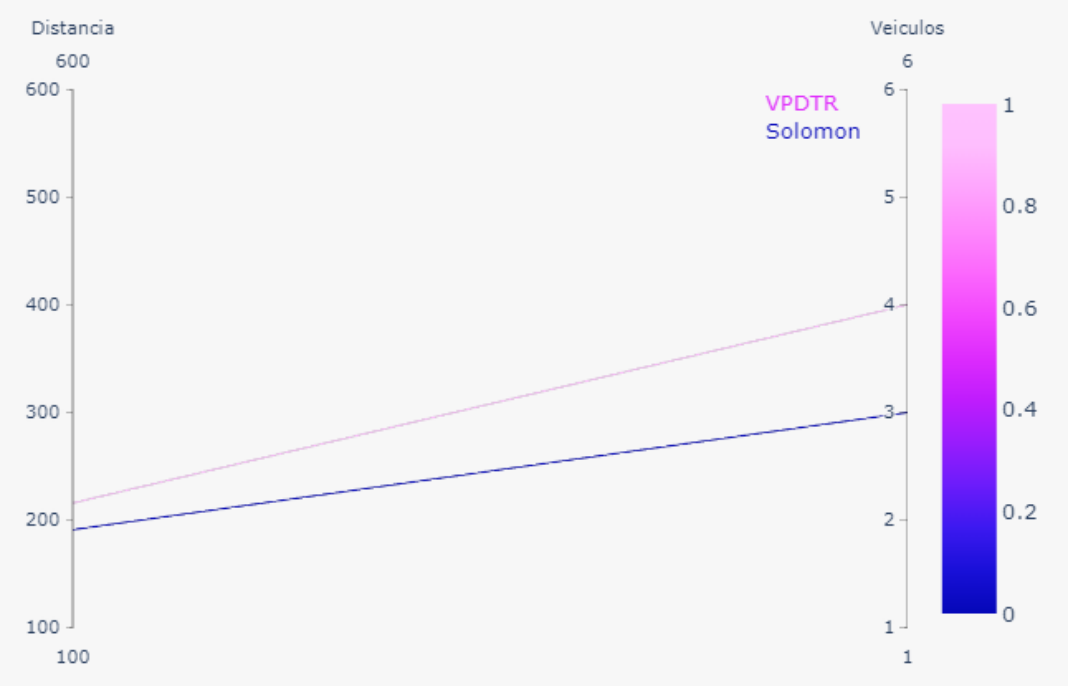


Elaborado pelo autor

Figura 143 – C106

Elaborado pelo autor

Figura 144 – C107



Elaborado pelo autor

Figura 145 – C108

Elaborado pelo autor

Figura 146 – C109

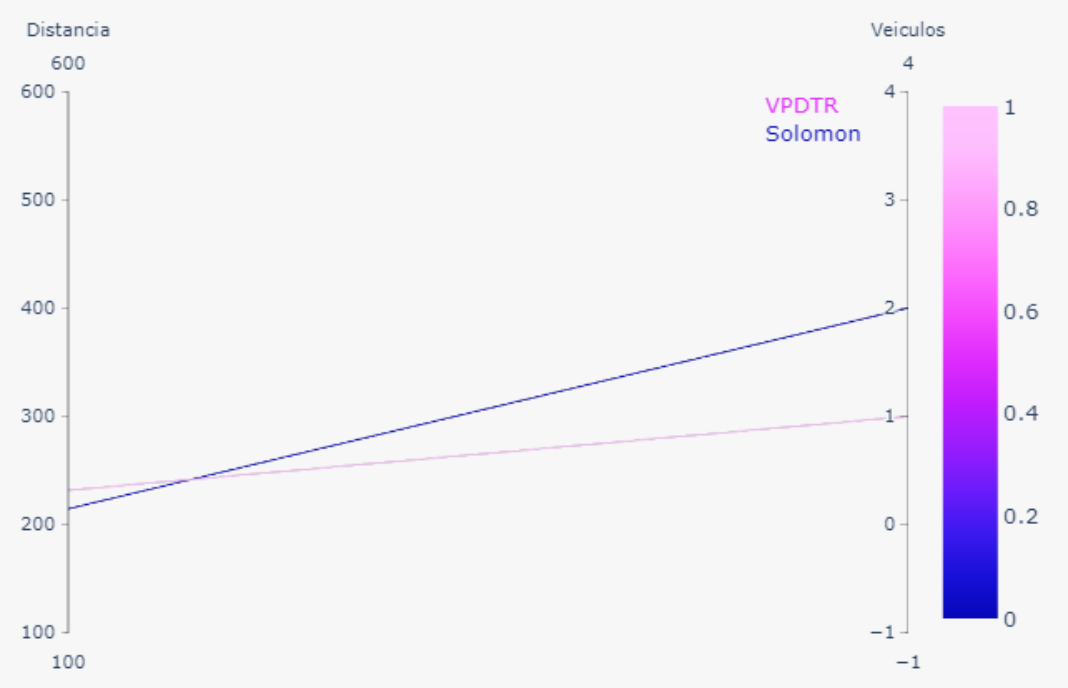
Elaborado pelo autor

E.2 Análises dos problemas C2

Figura 147 – C201

Elaborado pelo autor

Figura 148 – C202



Elaborado pelo autor

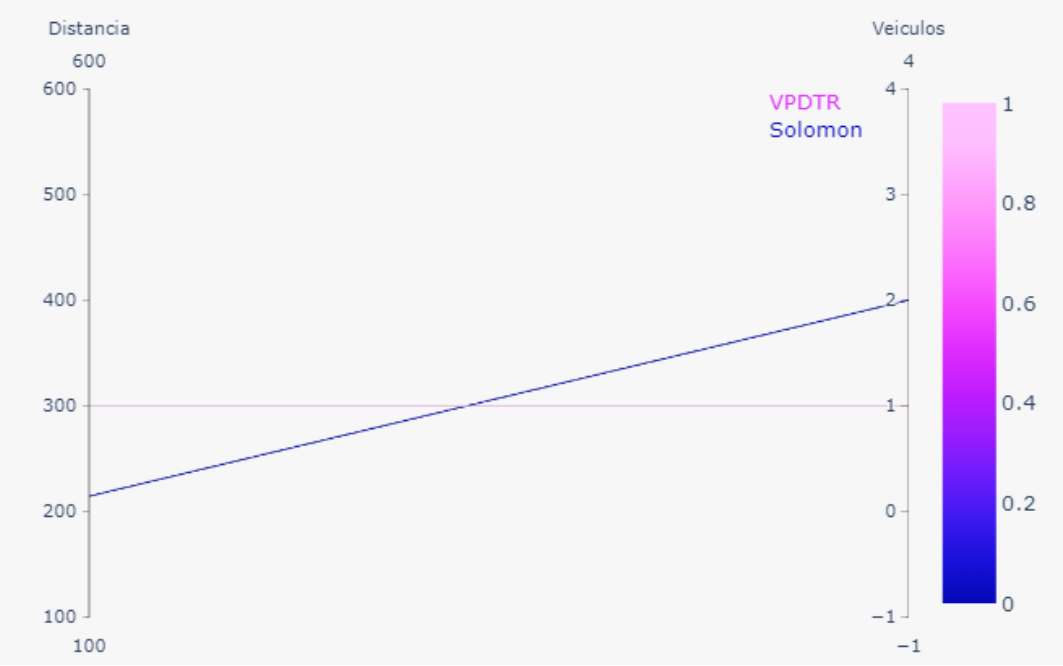
Figura 149 – C203

Elaborado pelo autor

Figura 150 – C204

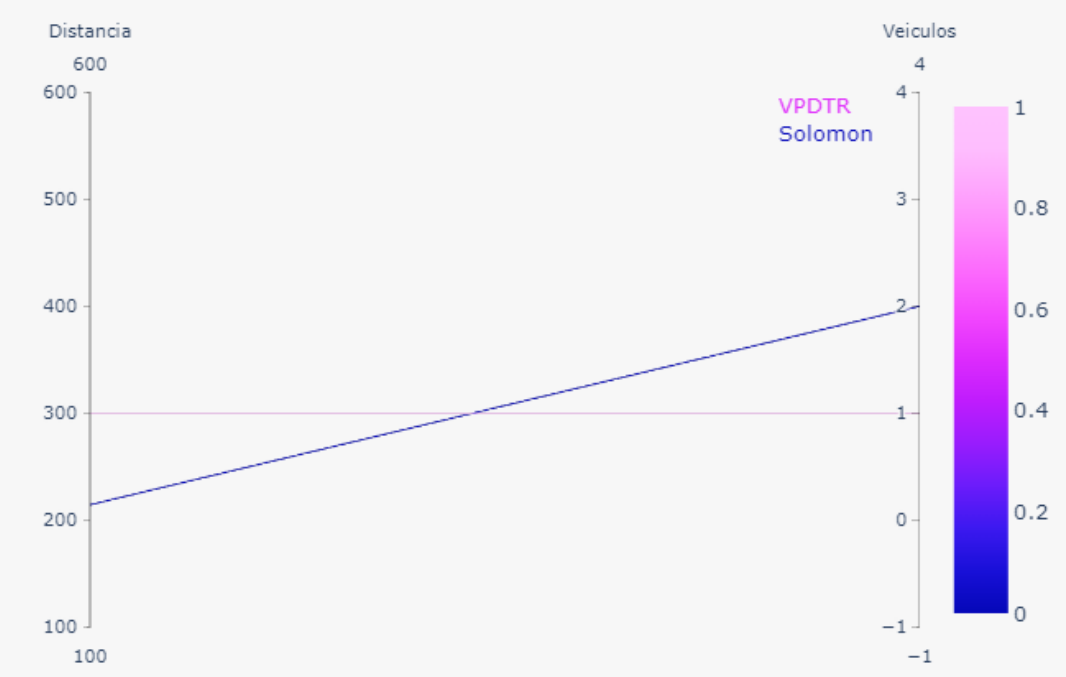
Elaborado pelo autor

Figura 151 – C205



Elaborado pelo autor

Figura 152 – C206



Elaborado pelo autor

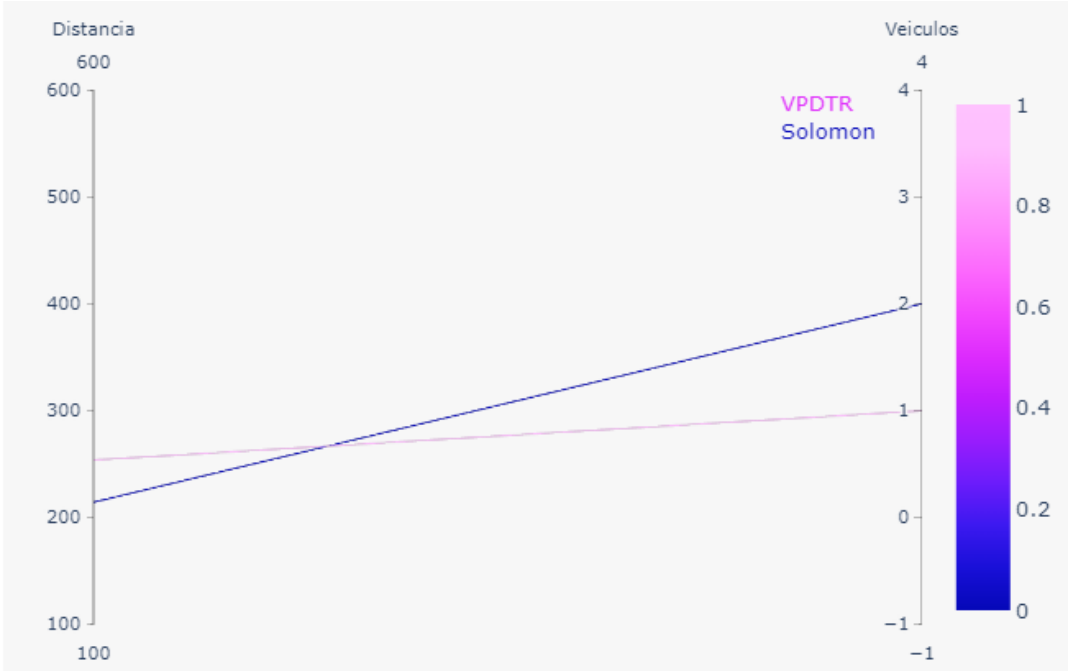
Figura 153 – C207a

Elaborado pelo autor

Figura 154 – C207b

Elaborado pelo autor

Figura 155 – C208



Elaborado pelo autor