Universidade Federal da Bahia
Instituto de Computação
MATA53 - Teoria dos grafos

Relatório: Problema do caixeiro viajante com bônus e passageiros

Antoniel Magalhães
João Leahy
Luis Felipe

Salvador - Bahia

8 de janeiro de 2025

Relatório: Problema do caixeiro viajante com bônus e passageiros

Antoniel Magalhães
João Leahy
Luis Felipe

Estudo dirigido entregue ao professor Islame Felipe da Costa Fernandes como método avaliativo da disciplina MATA53 - Teoria dos grafos

Salvador - Bahia

8 de janeiro de 2025

Sumário

Introdução

O **Problema do Caixeiro Viajante com Coleta Opcional de Bônus e Passageiros (TSP-OBP)**, também referenciado na literatura como **Problema do Caixeiro Viajante com Coleta de Prêmios (PCVCP)**, é uma extensão do clássico Problema do Caixeiro Viajante (PCV). Este problema combina a necessidade de coletar **bônus ou prêmios** em determinados locais, o transporte de **passageiros** e a otimização de rotas, excluindo o tempo de coleta, o que o torna relevante para aplicações como **logística, sistemas de entrega e serviços de transporte compartilhado**. Estudos como o de **Lopes Filho** [**3**] e **Carvalho** [**2**] destacam a importância do TSP-OBP e suas variantes na otimização de recursos e melhoria da eficiência operacional em sistemas complexos. Embora a tese de Lopes Filho [3] inclua o tempo de coleta, o presente trabalho foca na variante sem esta restrição.

A complexidade inerente ao TSP-OBP advém da necessidade de equilibrar objetivos conflitantes: **maximizar a coleta de bônus ou prêmios, minimizar a distância total percorrida e satisfazer as restrições de transporte de passageiros**, tais como a capacidade do veículo. Esta natureza multiobjetivo, juntamente com as restrições combinatórias, torna o problema **NP-difícil**, requerendo abordagens computacionais sofisticadas para sua resolução.

No contexto atual, onde a otimização de recursos e a eficiência operacional são cruciais, o TSP-OBP emerge como uma ferramenta essencial para modelar e resolver problemas complexos de roteamento. As aplicações do TSP-OBP abrangem desde sistemas de entrega com incentivos até serviços de transporte sob demanda, onde a flexibilidade na coleta de bônus e a gestão eficiente de passageiros são fundamentais. Esta modelagem é relevante em situações de logística, transporte de pessoas, e outros sistemas de roteamento que necessitem de coleta seletiva com passageiros.

Este relatório tem como objetivo apresentar uma análise do problema, incluindo suas variantes, a modelagem matemática, os métodos de resolução (heurísticas, metaheurísticas, algoritmos híbridos) e suas aplicações práticas. Além disso, exploraremos as contribuições teóricas e práticas que o TSP-OBP oferece para a pesquisa em otimização combinatória, com foco na variante sem tempo de coleta. A estrutura do trabalho está organizada para proporcionar uma compreensão progressiva do tema, desde os fundamentos teóricos até as aplicações e perspectivas futuras.

0.1 Contextualização e Relevância

O TSP-OBP surge como uma resposta natural à evolução dos sistemas de transporte e logística modernos, onde a simples otimização de rotas já não é suficiente para atender às demandas complexas do mercado. A incorporação de bônus e passageiros ao problema clássico do caixeiro viajante reflete a necessidade de modelos mais sofisticados, capazes de capturar a realidade multifacetada dos sistemas de transporte contemporâneos.

Em termos práticos, o problema encontra aplicações em diversos cenários:

- Sistemas de compartilhamento de viagens, onde motoristas podem coletar passageiros e bônus ao longo de suas rotas;
- Serviços de entrega com incentivos por coleta ou entrega em determinados pontos;
- Planejamento de roteiros turísticos com pontos de interesse prioritários;
- Otimização de rotas para veículos de transporte público sob demanda.

0.2 Objetivos e Escopo

O presente trabalho tem como objetivos específicos:

- Analisar a estrutura matemática do TSP-OBP e suas propriedades fundamentais;
- Investigar os métodos de resolução existentes e suas aplicabilidades;
- Avaliar o impacto das diferentes abordagens na qualidade das soluções obtidas;
- Discutir as implicações práticas e teóricas das variantes do problema;

O escopo do trabalho abrange desde a fundamentação teórica até as aplicações práticas. São consideradas tanto as abordagens exatas quanto as heurísticas, bem como as implicações de diferentes estruturas de dados e algoritmos na eficiência das soluções propostas.

Capítulo 1

Fundamentação Teórica

1.1 Problema do Caixeiro Viajante (TSP)

O **Problema do Caixeiro Viajante (PCV)**, ou Traveling Salesman Problem (TSP), consiste em encontrar o ciclo hamiltoniano de menor custo em um grafo ponderado, onde o ciclo visita cada vértice (cidade) exatamente uma vez e retorna ao vértice de origem [4]. Este problema é um dos clássicos da otimização combinatória e tem sido amplamente estudado na teoria da computação devido à sua complexidade e aplicabilidade. Como apontam [4, 3], a introdução de variáveis adicionais como **bônus** e **passageiros**, no **Problema do Caixeiro Viajante com Coleta Opcional de Bônus e Passageiros (TSP-OBP)**, aumenta significativamente a complexidade do problema original, exigindo novas abordagens de modelagem e solução.

Matematicamente, o TSP pode ser representado como um grafo completo $G = (N, M)$, onde N é o conjunto de vértices (cidades) e M é o conjunto de arestas (conexões entre cidades). Cada aresta $(i, j) \in M$ possui um custo associado c_{ij}, que representa a distância ou o tempo de viagem entre as cidades i e j [4]. O objetivo é encontrar um **ciclo hamiltoniano** de custo mínimo, ou seja, um caminho que visite todos os vértices exatamente uma vez e retorne ao ponto inicial.

A natureza **NP-difícil** do TSP implica que não existe um algoritmo conhecido que possa resolver o problema em tempo polinomial para todas as instâncias [4, 1]. Essa característica fundamental do problema base apresenta desafios significativos para sua resolução, que são amplificados quando consideramos extensões como o TSP-OBP, onde a coleta de bônus e o transporte de passageiros introduzem novas dimensões de complexidade combinatória e decisões a serem tomadas [3, 2]. O livro de Goldbarg detalha vários algoritmos para o PCV, incluindo heurísticas e métodos de aproximação, que podem servir como ponto de partida para a compreensão de problemas mais complexos, como o TSP-OBP.

1.2 Extensões do TSP

As extensões do TSP, como o TSP com Bônus e o TSP-OBP, introduzem novas dimensões ao problema original, exigindo a consideração de múltiplos objetivos e restrições. Essas variações são fundamentais para capturar a complexidade de cenários reais, onde decisões de roteamento devem equilibrar custos, prêmios e restrições de passageiros [2].

Entre as principais extensões, podemos destacar:

- **TSP com Prêmios (PTSP)**: Incorpora valores de bônus associados à visita de determinados vértices;
- **TSP com Janelas de Tempo (TSPTW)**: Adiciona restrições temporais para a visita aos vértices;
- **TSP com Coleta e Entrega (PDTSP)**: Inclui operações de coleta e entrega de itens entre vértices;
- **TSP com Múltiplos Veículos (mTSP)**: Considera uma frota de veículos para realizar as visitas.

O TSP-OBP combina elementos dessas diferentes variantes, incorporando tanto aspectos de coleta de bônus quanto o gerenciamento de passageiros, o que resulta em um problema significativamente mais complexo.

1.3 Definição Formal do Problema

O TSP-OBP pode ser formalizado como um problema de otimização em grafos, onde o objetivo é maximizar a coleta de bônus e minimizar o custo total de deslocamento, respeitando as restrições de passageiros. Esta definição formal permite a aplicação de técnicas avançadas de otimização para encontrar soluções eficientes.

Formalmente, o TSP-OBP pode ser definido como:

- **Grafo**: $G = (V, E)$, onde V é o conjunto de vértices e E o conjunto de arestas;
- **Custos**: c_{ij} representa o custo de viagem entre os vértices i e j;
- **Bônus**: b_i é o valor do bônus associado ao vértice $i \in V$;
- **Passageiros**: P é o conjunto de passageiros, onde cada p $\in$ P possui:
  - origem o_p e destino d_p
  - janela de tempo $[t_{p\_inicio}, t_{p\_fim}]$ para embarque

– tempo máximo de viagem t_{max_p}

A função objetivo do TSP-OBP pode ser expressa como uma combinação ponderada entre a maximização dos bônus coletados e a minimização dos custos de viagem:

$$\max \sum_{i \in V} b_i x_i - \alpha \sum_{(i,j) \in E} c_{ij} y_{ij} \tag{1.1}$$

onde x_i é uma variável binária que indica se o vértice i é visitado, y_{ij} indica se a aresta (i, j) é utilizada na solução, e α é um parâmetro de balanceamento entre os objetivos [3].

1.4 Complexidade Computacional

A análise de complexidade do TSP-OBP revela que o problema é NP-difícil, uma vez que contém o TSP clássico como caso especial. Além disso, a adição de restrições de passageiros e bônus introduz novas dimensões de complexidade, tornando o problema ainda mais desafiador do ponto de vista computacional [2].

Alguns aspectos que contribuem para a complexidade do problema incluem:

- Natureza combinatória da seleção de vértices para visita;
- Interdependência entre as decisões de roteamento e coleta de bônus;
- Restrições temporais e de capacidade relacionadas aos passageiros;
- Necessidade de coordenação entre múltiplos objetivos conflitantes.

Esta complexidade inerente motiva o desenvolvimento de diferentes abordagens de solução, desde métodos exatos para instâncias pequenas até heurísticas e metaheurísticas para problemas de maior escala.

1.5 Prova de NP-Dificuldade do TSP-OBP

Demonstra-se que o problema **TSP-OBP (Traveling Salesman Problem with Onboard Passengers)** é NP-difícil por meio de uma redução polinomial a partir do **Problema do Caixeiro Viajante (PCV)**, que já é conhecido por ser NP-difícil.

1.5.1 Instância do PCV

Considere uma instância qualquer do PCV com um grafo $G = (V, A)$, onde V é o conjunto de vértices e A é o conjunto de arestas. Suponha que H é uma solução ótima para esta instância do PCV.

1.5.2 Transformação para o TSP-OBP

Cria-se uma nova instância do TSP-OBP, $G' = (V, A, P)$, baseada em G, da seguinte forma:

- Cada vértice em G', exceto o vértice de origem em H, recebe um bônus duas vezes maior do que o custo da maior aresta em G.
- O tempo de coleta em cada vértice é zero.
- O tempo para percorrer qualquer aresta e a capacidade do veículo são zero.
- Não existem passageiros disponíveis para embarque, ou seja, o conjunto P é vazio.

1.6 Análise da Redução

1.6.1 Tempo Polinomial

A transformação de G para G' pode ser feita em tempo polinomial.

1.6.2 Relação entre Soluções

- Se existe uma solução ótima H para o PCV em G, então esta mesma solução também é uma solução válida para o TSP-OBP em G', uma vez que a solução ótima do PCV é um ciclo hamiltoniano que visita todos os vértices, e neste caso específico os custos de coleta e transporte de passageiros não impactam o resultado.
- Por outro lado, se existisse uma solução para o TSP-OBP em G' melhor do que a solução do PCV em G, isso significaria que o PCV não era NP-difícil.

1.6.3 Complexidade

Dado que o PCV é NP-difícil e que o TSP-OBP pode ser reduzido polinomialmente ao PCV , o TSP-OBP também é NP-difícil. Esta prova estabelece que o TSP-OBP herda a complexidade do PCV, o que significa que encontrar uma solução ótima para o TSP-OBP em tempo polinomial é improvável.

Capítulo 2

Modelagem Matemática

2.1 Variáveis de Decisão

A modelagem matemática do TSP-OBP envolve a definição de variáveis de decisão que capturam a complexidade do problema. Além das variáveis tradicionais do TSP, o TSP-OBP requer a consideração de variáveis que representam a coleta de bônus e o transporte de passageiros. Este aumento na complexidade demanda o uso de algoritmos avançados para encontrar soluções eficientes [2].

As principais variáveis de decisão do modelo são:

- x_{ij}: Variável binária que indica se a aresta (i,j) é utilizada na solução

$$x_{ij} = \begin{cases} 1, & \text{se o arco (i,j) é utilizado na rota} \\ 0, & \text{caso contrário} \end{cases} \tag{2.1}$$

- y_i: Variável binária que indica se o vértice i é visitado

$$y_i = \begin{cases} 1, & \text{se o vértice i é visitado} \\ 0, & \text{caso contrário} \end{cases} \tag{2.2}$$

- t_i: Variável contínua que representa o instante de chegada no vértice i
- p_{ik}: Variável binária que indica se o passageiro k está no veículo ao visitar o vértice i

2.2 Função Objetivo

A função objetivo do TSP-OBP é maximizar a soma dos bônus coletados e minimizar o custo total de viagem. Esta função deve ser cuidadosamente balanceada para garantir que as soluções propostas sejam viáveis e otimizadas em termos de custo-benefício.

A formulação matemática da função objetivo pode ser expressa como:

$$\max \sum_{i \in V} b_i y_i - \alpha \sum_{(i,j) \in E} c_{ij} x_{ij} \tag{2.3}$$

onde:

- b_i é o valor do bônus associado ao vértice i
- c_{ij} é o custo de viagem entre os vértices i e j
- α é o parâmetro de balanceamento entre bônus e custos

2.3 Restrições de Fluxo

$$\sum_{j \in V} x_{ij} = y_i \qquad \forall i \in V \tag{2.4}$$

$$\sum_{i \in V} x_{ij} = y_j \qquad \forall j \in V \tag{2.5}$$

A **Equação** (2.4) garante que, para cada vértice i, se ele está na solução ($y_i = 1$), então existe uma aresta saindo de i. Já a **Equação** (2.5) assegura que, se o vértice j está na solução ($y_j = 1$), então há uma aresta chegando em j. Essas restrições são fundamentais para garantir a continuidade da rota.

Essas restrições são chamadas de **restrições de atribuição** [3, 2]. As variáveis x_{ij} são **binárias**, indicando se a aresta (i, j) faz parte da rota. As variáveis y_i também são **binárias**, representando se o vértice i está na rota ou não.

2.4 Restrições de Conectividade

Eliminação de Subciclos com Miller-Tucker-Zemlin (MTZ)

Para garantir que a solução seja um único ciclo Hamiltoniano, e não múltiplos subciclos desconexos, utilizamos as restrições propostas por Miller, Tucker e Zemlin (MTZ). Estas restrições são formuladas como:

$$u_i - u_j + n x_{ij} \leq n - 1 \quad \forall i, j \in V, i \neq j \tag{2.6}$$

onde:

- u_i é uma variável auxiliar **inteira** que representa a ordem de visita do vértice i na rota.

- n representa o número total de vértices no grafo.
- x_{ij} é uma variável **binária** que indica se a aresta que conecta diretamente o vértice i ao vértice j faz parte da solução. Se $x_{ij} = 1$, a aresta está na rota; caso contrário, $x_{ij} = 0$.
- V representa o conjunto de todos os vértices no grafo.

Explicação Detalhada

O objetivo principal dessas restrições é **eliminar a formação de subciclos** que não incluem todos os vértices do grafo. A ideia central é que, se um ciclo Hamiltoniano é formado, os valores de u_i devem representar uma sequência coerente de visitas aos vértices.

2.4.1 Restrições de Passageiros

As restrições de passageiros visam garantir a viabilidade e a conformidade do roteiro em relação aos passageiros e suas demandas.

- **Restrição de Atendimento Único:**

$$\sum_{i \in V} p_{ik} \leq 1 \quad \forall k \in P \tag{2.7}$$

  Essa restrição garante que cada passageiro k no conjunto de passageiros P seja atendido no máximo uma vez. A variável binária p_{ik} indica se o passageiro k foi embarcado no vértice i. A soma de todas as localizações de embarque de um passageiro deve ser menor ou igual a 1, garantindo que um passageiro não será embarcado em mais de um vértice.

- **Restrição de Tempo de Chegada (Sequência Temporal):**

$$t_i + s_i + c_{ij} - M(1 - x_{ij}) \leq t_j \quad \forall i, j \in V \tag{2.8}$$

  Essa restrição assegura a **sequência temporal** da rota, onde:

  - t_i é o tempo de chegada no vértice i.
  - s_i é o tempo de serviço ou coleta no vértice i.
  - c_{ij} é o tempo de deslocamento da aresta entre o vértice i e o vértice j.
  - x_{ij} é uma variável binária que indica se a aresta (i, j) faz parte da rota.
  - M é uma constante suficientemente grande.

Se a aresta (i, j) é utilizada ($x_{ij} = 1$), a restrição garante que o tempo de chegada no vértice j (t_j) seja maior ou igual ao tempo de chegada no vértice i (t_i), somado ao tempo de serviço em i (s_i) e ao tempo de deslocamento de i para j (c_{ij}). O termo $M(1 - x_{ij})$ desativa a restrição caso a aresta (i, j) não seja utilizada ($x_{ij} = 0$). Essa restrição é similar à utilizada para eliminar subciclos.

- **Restrição de Janela de Tempo - Limite Inferior:**

$$t_i \geq e_i \quad \forall i \in V \tag{2.9}$$

Essa restrição garante que o tempo de chegada no vértice i (t_i) deve ser maior ou igual ao limite inferior da janela de tempo do vértice i (e_i). Ela assegura que o serviço em um vértice ou embarque de um passageiro ocorra depois de um determinado tempo limite.

- **Restrição de Janela de Tempo - Limite Superior:**

$$t_i \leq l_i \quad \forall i \in V \tag{2.10}$$

Essa restrição garante que o tempo de chegada no vértice i (t_i) não deve ultrapassar o limite superior da janela de tempo do vértice i (l_i). Ela assegura que o serviço em um vértice ou embarque de um passageiro em um vértice ocorra antes de um determinado tempo limite. Combinadas, as restrições de janela de tempo (limite inferior e superior) garantem que o serviço ou embarque ocorram dentro de um intervalo de tempo aceitável para o vértice i.

Variáveis e Parâmetros:

- s_i: Tempo de serviço/coleta no vértice i.
- e_i: Limite inferior da janela de tempo para o vértice i.
- l_i: Limite superior da janela de tempo para o vértice i.
- M: Constante suficientemente grande.
- p_{ik}: Variável binária que indica se o passageiro k foi embarcado no vértice i.
- t_i: Tempo de chegada no vértice i.
- c_{ij}: Custo ou tempo de deslocamento entre os vértices i e j.
- x_{ij}: Variável binária que indica se a aresta (i, j) faz parte da rota.
- P: Conjunto de passageiros.
- V: Conjunto de vértices.

Capítulo 3

Métodos de Resolução

3.1 Introdução

Para abordar o problema de forma acessível e eficaz, optou-se por utilizar dois métodos principais: **GRASP (Greedy Randomized Adaptive Search Procedure)** e **Atribuição Heurística de Passageiros**. Estes métodos foram escolhidos por sua simplicidade conceitual e capacidade de gerar soluções de alta qualidade em tempo computacional reduzido.

3.2 Métodos Utilizados

GRASP (Greedy Randomized Adaptive Search Procedure)

O GRASP é uma meta-heurística com duas fases principais: uma fase de construção semi-gulosa e uma fase de busca local. Seu objetivo é equilibrar a **exploração** (buscar soluções diversas) e a **intensificação** (refinar as soluções mais promissoras), de modo a produzir boas soluções para problemas de otimização combinatória em um tempo razoável.

Fase de Construção Semi-Gulosa

A fase de construção do GRASP gera uma solução inicial de forma rápida, misturando elementos gulosos e aleatoriedade controlada. No contexto do problema de roteamento com bônus (PCVP-BoTc), isso envolve decidir em cada passo como construir uma rota que considere (ou não) passageiros em função do ganho (bônus) e do custo (distância e tempo) para inseri-los.

Lista de Candidatos (LCD)

- **Composição**: A LCD é formada por todos os vértices que podem ser adicionados à rota sem violar restrições de capacidade e tempo, incluindo passageiros (para coleta/entrega) e pontos obrigatórios (por exemplo, ponto inicial).
- **Avaliação de cada vértice candidato**: Para cada vértice i que pode seguir o vértice atual j, definem-se duas opções de custo:

  $$D1_{ij} = c_{ij} + c_{j1} - b_j \qquad \text{(incluindo o bônus)}, \;\; D2_{ij} = c_{ij} + c_{j1} \qquad \text{(sem considerar o bônus)}.$$

  Aqui, c_{ij} é o custo de ir de i a j, c_{j1} o custo de ir de j de volta ao ponto inicial, e b_j o bônus do vértice j.

Lista Restrita de Candidatos (LRC)

- **Seleção de Melhores Candidatos**: A LRC é formada pelos vértices mais promissores da LCD, filtrados por um parâmetro α. Define-se um valor-limite

  $$e_{\text{limite}} = e_{\max} - \alpha(e_{\max} - e_{\min}), \tag{3.1}$$

  onde $e_{\max}$ e $e_{\min}$ são os valores de ”economia”(ou potencial de melhoria) do melhor e do pior candidato, respectivamente.
- **Seleção Aleatória**: Dentro dessa LRC, escolhe-se um vértice de forma aleatória. Essa etapa adiciona diversidade ao processo, pois nem sempre o ”melhor”candidato é selecionado.

Construção Iterativa da Solução

- **Inserção de Vértices**: Em cada iteração, um vértice é escolhido da LRC e adicionado à solução, decidindo também se a coleta do bônus será aproveitada (opção 1) ou não (opção 2).
- **Viabilidade da Rota**: A função `embarcarPass()` atualiza a rota, garantindo que as restrições de capacidade e tempo sejam respeitadas.
- **Atualização das Listas**: Tanto a LCD quanto a LRC são atualizadas a cada inserção, até que a rota não possa receber mais vértices ou seja concluída.

Fase de Busca Local com VND

Após construída a solução inicial, o GRASP faz a **intensificação** por meio de uma busca local, aqui realizada pelo *Variable Neighborhood Descent* (VND). O foco dessa fase é refinar a solução, explorando diferentes vizinhanças.

Variable Neighborhood Descent (VND)

- **Exploração Gradual**: O VND utiliza diferentes estruturas de vizinhança, ordenadas por complexidade (ou custo computacional). Ele inicia na vizinhança mais simples e só avança para a seguinte caso não encontre melhoria.
- **Movimentos na Vizinhança**: Exemplos de movimentos incluem:
  - **Pass1**: Seleciona um passageiro sem origem/destino de maior bônus e tenta reinseri-lo em outro local da rota.
  - **Pass2**: Remove um passageiro sem destino/origem de maior bônus, se isso puder gerar economia de custo.
  - **Pass3**: Tenta trocar a posição de dois passageiros, buscando melhorar o custo total (ou aumentar a coleta de bônus).
- **Reinício da Busca**: Sempre que ocorre melhoria em alguma vizinhança, o processo retorna à primeira vizinhança. Isso permite uma "exploração fina" ao redor da nova solução encontrada.

Critérios de Parada

A busca local termina quando:

- **Número máximo de avaliações da função objetivo** é atingido ($maxFc$).
- **Número máximo de iterações sem melhoria** é atingido ($maxConv$).

Integração com o GVND

O GRASP pode ser combinado com o *Greedy Variable Neighborhood Descent* (GVND), uma versão adaptada do VND que explora múltiplas vizinhanças de forma gulosa. Essa integração alia a **diversidade** do GRASP à **intensificação** do GVND, resultando em soluções potencialmente mais robustas.

Algorithm 1 GRASP Genérico

Require: nomeInstancia, maxFc, maxConv

$G \leftarrow leituraInst(nomeInstancia)$

$S \leftarrow \emptyset$ ▷ Solução inicial vazia

$f(S) \leftarrow \infty$ ▷ Valor da solução inicial

$numCh \leftarrow 0$ ▷ Contador de chamadas à função objetivo

$conv \leftarrow 0$ ▷ Contador de iterações sem melhoria

while $(numCh \leq maxFc)$ **e** $(conv \leq maxConv)$ **do**

  $S^{*} \leftarrow constrSolucao()$ ▷ Constrói solução semi-gulosa

  $S^{*} \leftarrow VND(S^{)}$ ▷ Aplica busca local VND

  if $f(S^{)} < f(S)$ **then**

    $S \leftarrow S^{*}$ ▷ Atualiza melhor solução

    $conv \leftarrow -1$ ▷ Reseta contador sem melhoria

  end if

  $conv \leftarrow conv + 1$ ▷ Incrementa contador sem melhoria

end while

return S ▷ Retorna melhor solução encontrada

Capítulo 4

Resultados e Discussão

4.1 Resultados Computacionais

A partir da implementação do *GRASP* com busca local via *VND* (Variable Neighborhood Descent), foi possível gerar soluções de forma relativamente rápida e com boa qualidade para instâncias de pequeno e médio porte do TSP-OBP. Vale ressaltar, entretanto, que, no âmbito deste trabalho, **não foi realizada uma análise comparativa detalhada com outros métodos**, em função de o foco ter permanecido na implementação e experimentação de *apenas uma abordagem meta-heurística*.

O código-fonte referente à implementação deste método encontra-se disponível em:

`https://github.com/antoniel/mata53-seminario`

Neste repositório, pode-se observar todo o processo de construção da solução, abrangendo a fase de construção gulosa (com elementos de aleatoriedade controlada), a etapa de busca local e o modo como passageiros e bônus foram tratados no contexto do TSP-OBP.

4.1.1 Processo de Construção da Solução

A construção inicial segue o paradigma do **GRASP**, compondo:

1. **Fase de Construção Semi-Gulosa**: Gera uma rota inicial ao escolher iterativamente vértices e passageiros segundo um critério de ganho líquido (bônus menos custo), com pequena aleatoriedade para aumentar a diversidade das soluções.

2. **Fase de Busca Local (VND)**: Explora vizinhanças específicas, como *remoção*, *reinserção* e *troca* de passageiros na rota, visando reduzir a função objetivo (distância total subtraída do bônus coletado).

Embora essa abordagem seja efetiva em *práticas* de prototipação rápida, **não realizamos** a calibração aprofundada de parâmetros nem a comparação rigorosa com outros métodos (por exemplo, *Branch & Cut*, *Heurísticas Construtivas* ou variações de *GRASP* e *VND*). Ainda assim, os experimentos iniciais evidenciam que a meta-heurística implementada consegue resolver instâncias de tamanho moderado em tempo razoável, produzindo soluções com qualidade satisfatória em termos de distância percorrida e bônus coletado.

4.2 Discussão

A análise dos resultados indica que a implementação do *GRASP + VND* funciona como **prova de conceito** para aplicação de meta-heurísticas ao TSP-OBP. Em particular, pudemos verificar:

- **Simplicidade de Extensão**: A inclusão de novos passageiros ou bônus é acomodada pela fase de construção gulosa, ajustando o critério de seleção de vértices.
- **Limitação na Exploração de Vizinhanças**: Como não há uma etapa de reordenação completa da rota (por exemplo, via *2-opt* ou *3-opt*), certas instâncias podem estagnar em soluções subótimas.
- **Potencial de Melhoria**: A adoção de estruturas de vizinhança mais complexas (inserção de pares origem–destino em diferentes posições, inversões de subcaminhos etc.) pode aumentar a capacidade de explorar o espaço de soluções.

Futuramente, **uma análise comparativa** com heurísticas construtivas básicas ou com métodos exatos para instâncias pequenas possibilitará quantificar o desempenho do algoritmo. Além disso, **parâmetros** como a intensidade da aleatoriedade na construção ou o número de vizinhanças no *VND* podem ser ajustados para buscar melhores equilíbrios entre tempo de execução e qualidade da solução.

4.2.1 Limitações e Desafios

Em linha com as considerações gerais do problema, destacam-se:

- **Escalabilidade**: Apesar de a heurística lidar bem com instâncias moderadas, problemas de maior escala requerem estratégias mais elaboradas para a busca local.
- **Qualidade das Soluções**: Sem uma etapa de reordenação mais agressiva (*2-opt*, *3-opt*, *K-opt*), a rota resultante pode permanecer distante do ótimo.
- **Integração de Passageiros e Janelas de Tempo**: Em cenários mais complexos, a checagem e a simulação do horário de embarque/desembarque podem demandar modificações adicionais no modelo.

Em suma, embora o método desenvolvido cumpra o papel de apresentar uma meta-heurística funcional, **não constitui** uma comparação exaustiva entre diferentes algoritmos, ficando evidente a necessidade de testes adicionais e melhorias de vizinhanças para ampliar o espectro de soluções viáveis.

Capítulo 5

Conclusão

Este trabalho apresentou uma análise abrangente do Problema do Caixeiro Viajante com Coleta de Bônus e Passageiros (TSP-OBP), evidenciando sua relevância prática em cenários de transporte e logística. A incorporação de restrições como a coleta de bônus e o transporte de passageiros ampliou significativamente a complexidade do problema em relação ao TSP clássico, exigindo abordagens mais robustas para produzir soluções de qualidade em tempo factível.

Ao longo do texto, foram discutidos desde os aspectos teóricos e a modelagem matemática até os desafios de implementação e a escolha de métodos de resolução. A explanação de heurísticas e metaheurísticas, em particular a abordagem baseada em *GRASP* e *VND*, demonstrou como é possível combinar fases de construção gulosa com busca local para lidar com esse tipo de problema de maneira eficiente.

5.1 Considerações Finais

O TSP-OBP representa um avanço significativo na modelagem de problemas complexos de roteamento, incorporando aspectos práticos essenciais como a coleta de bônus e o transporte de passageiros. A análise realizada demonstra que, apesar dos desafios computacionais, existem métodos eficazes para sua resolução, especialmente quando se consideram abordagens híbridas e metaheurísticas avançadas [3, 2].

Em termos de aplicações, a variedade de cenários em que o TSP-OBP pode ser empregado — como logística urbana ou roteirização de veículos com incentivos — ressalta a importância de investir em técnicas de otimização capazes de equilibrar múltiplos objetivos, como minimização de custo, maximização de bônus e respeito a restrições de capacidade e tempo.

Por fim, vale ressaltar que, apesar de a formulação geral do TSP-OBP possuir aplicações práticas, a riqueza do problema também está em sua relação profunda com conceitos fundamentais de grafos. Da perspectiva teórica, a complexidade e as dificuldades de resolução se tornam campos férteis para pesquisa, ligando-se de forma intrínseca aos estudos sobre ciclos hamiltonianos. Pesquisas futuras podem investigar melhorias nas estruturas de vizinhança, heurísticas híbridas e a aplicação de técnicas de aprendizado de máquina para aprimorar a seleção e o ajuste de parâmetros em meta-heurísticas, tornando o TSP-OBP cada vez mais relevante no cenário de otimização combinatória e operações.

Referências Bibliográficas

[1] W. Carnielli and R. Epstein. *Computabilidade e Funções Computáveis*. UNESP, 2017.

[2] M. R. Carvalho. Métodos heurísticos para o tsp-obp. *Journal of Combinatorial Optimization*, 2022.

[3] José Gomes Lopes Filho. *Problema do Caixeiro Viajante com Coleta Opcional de Bônus, Tempo de Coleta e Passageiros*. Tese de doutorado, Universidade Federal do Rio Grande do Norte, Natal-RN, 2019.

[4] Marco Goldbarg and Elizabeth Goldbarg. *Grafos: Conceitos, algoritmos e aplicações*. Elsevier, Rio de Janeiro, 2012.