Universidade Federal do Rio Grande do Norte
Centro de Ciências Exatas e da Terra
Departmento de Informática e Matemática Aplicada
Programa de Pós-Graduação em Sistemas e Computação
Doutorado Acadêmico em Sistemas e Computação

O Problema do Caixeiro Viajante com Cota, Múltiplos Passageiros, Transporte Incompleto e Tempo de Coleta

Bruno de Castro Honorato Silva

Natal-RN

Setembro de 2020

Bruno de Castro Honorato Silva

O Problema do Caixeiro Viajante com Cota, Múltiplos Passageiros, Transporte Incompleto e Tempo de Coleta

Tese de doutorado apresentada ao Programa de Pós-Graduação em Sistemas e Computação do Departamento de Informática e Matemática Aplicada da Universidade Federal do Rio Grande do Norte.

Linha de pesquisa:
Algoritmos Experimentais

Orientador

Prof. Dr. Marco Cesar Goldbarg

PPgSC – Programa de Pós-Graduação em Sistemas e Computação
DIMAp – Departamento de Informática e Matemática Aplicada
CCET – Centro de Ciências Exatas e da Terra
UFRN – Universidade Federal do Rio Grande do Norte

Natal-RN

Setembro de 2020

Universidade Federal do Rio Grande do Norte - UFRN
Sistema de Bibliotecas - SISBI
Catalogação de Publicação na Fonte. UFRN - Biblioteca Setorial Prof. Ronaldo Xavier de Arruda - CCET

Silva, Bruno de Castro Honorato.
O problema do caixeiro viajante com cota, múltiplos passageiros, transporte incompleto e tempo de coleta / Bruno de Castro Honorato Silva. - 2020.
139f.: il.

Tese (Doutorado) - Universidade Federal do Rio Grande do Norte, Centro de Ciências Exatas e da Terra, Programa de Pós-Graduação em Sistemas e Computação. Natal, 2020.
Orientador: Marco César Goldbarg.

1. Computação - Tese. 2. Problema do caixeiro viajante - Tese. 3. Mobilidade sob demanda - Tese. 4. Programação matemática - Tese. I. Goldbarg, Marco César. II. Título.

RN/UF/CCET CDU 004

Elaborado por Joseneide Ferreira Dantas - CRB-15/324

O Problema do Caixeiro Viajante com Cota, Múltiplos Passageiros, Transporte Incompleto e Tempo de Coleta

Autor: Bruno de Castro Honorato Silva
Orientador(a): Prof. Dr. Marco Cesar Goldbarg

Resumo

O Problema do Caixeiro Viajante com Cota, Múltiplos Passageiros, Transporte Incompleto e Tempo de Coleta é uma variante do Problema do Caixeiro Viajante com Cota em que o vendedor usa um sistema de compartilhamento de viagens flexível para minimizar os custos de viagem enquanto visita alguns vértices para satisfazer uma cota pré-estabelecida. É apresentado um modelo matemático em que se consideram restrições operacionais relacionadas à capacidade do veículo, tempo de viagem, limitações de passageiros e penalidades por viagens que não atendam aos requisitos dos passageiros. O modelo matemático é implementado em dois *solvers* de otimização. Por se tratar de um problema inédito, um banco de 144 instâncias foi gerado. Heurísticas *naive* foram implementadas para produzir os resultados preliminares do banco de instâncias. Abordagens meta-heurísticas com aplicações bem sucedidas em problemas correlatos são adaptadas ao problema proposto. Também é apresentado uma heurística de carregamento para dar suporte às meta-heurísticas e uma heurística de busca local baseada em múltiplos operadores de vizinhança.

Palavras-chave: Problema do Caixeiro Viajante, Mobilidade sob Demanda, Programação Matemática.

The Quota Travelling Salesman Problem with Passengers, Incomplete Ride and Collection Time

Author: Bruno de Castro Honorato Silva
Supervisor: PhD. Prof. Marco Cesar Goldbarg

Abstract

The Quota Travelling Salesman Problem with Passengers, Incomplete Ride, and Collection Time is a new version of the Quota Travelling Salesman Problem. In this problem, the salesman uses a flexible ridesharing system to minimize travel costs while visiting some vertices to satisfy a pre-established quota. Operational constraints regarding vehicle capacity, travel time, passenger limitations, and penalties for rides that do not meet passenger requirements are considered. A mathematical programming model for the proposed problem is presented and submited to optimization solvers. Since it is a new problem, a benchmark set of 144 instances is created. Meta-heuristic approaches with successful applications in related problems are adapted to the proposed problem. We implemented and compared the results of naive heuristics with those produced by the meta-heuristics. We also present a ride-matching heuristic to support the meta-heuristics and a local search based on multiple neighborhood operators.

Keywords: Travelling Salesman, Mobility on Demand, Mathematical Programming.

MINISTÉRIO DA EDUCAÇÃO E DESPORTO
UNIVERSIDADE FEDERAL DO RIO GRANDE DO NORTE
CENTRO DE CIÊNCIAS EXATAS E DA TERRA
DEPARTAMENTO DE INFORMÁTICA E MATEMÁTICA APLICADA
PROGRAMA DE PÓS-GRADUAÇÃO EM SISTEMAS E COMPUTAÇÃO

Ata nº. 75

ATA DA SESSÃO DE AVALIAÇÃO DE TESE DE DOUTORADO DO PROGRAMA DE PÓS-GRADUAÇÃO EM SISTEMAS E COMPUTAÇÃO.

Aos dezoito dias do mês de setembro de dois mil e vinte (18/09/2020), às 08h30, por videoconferência, foi instalada a Comissão Examinadora responsável pela avaliação da tese de doutorado intitulada: "***O Problema do Caixeiro Viajante com Cota, Múltiplos Passageiros, Transporte Incompleto e Tempo de Coleta***", como trabalho final apresentado pelo doutorando **BRUNO DE CASTRO HONORATO SILVA**, ao Programa de Pós-Graduação em Sistemas e Computação, da Universidade Federal do Rio Grande do Norte, como parte dos requisitos para obtenção do título de **DOUTOR EM CIÊNCIA DA COMPUTAÇÃO**. A Comissão Examinadora foi presidida pelo professor **Dr. MARCO CÉSAR GOLDBARG** (Orientador - UFRN), e contou com a participação dos professores: **Dra. ELIZABETH FERREIRA GOUVÊA GOLDBARG** (UFRN), **Dr. LUCÍDIO DOS ANJOS FORMIGA CABRAL** (UFPB), **Dr. MATHEUS DA SILVA MENEZES** (UFERSA), **Dr. PAULO HENRIQUE ASCONAVIETA DA SILVA** (IFRS) e **Dra. SILVIA MARIA DINIZ MONTEIRO MAIA** (UFRN) na qualidade de examinadores. A sessão teve a duração de 04 (quatro) horas e a Comissão emitiu o seguinte parecer: o trabalho e desempenho do candidato atenderam aos requisitos necessários a uma Tese de Doutorado, tendo a Comissão Examinadora, portanto APROVADO o trabalho. (APROVADO / REPROVADO)

Dra. ELIZABETH FERREIRA GOUVÊA GOLDBARG
Examinadora Interna

Dr. LUCÍDIO DOS ANJOS FORMIGA CABRAL
Examinador Externo

Dr. MARCO CÉSAR GOLDBARG
Orientador / Presidente

MATHEUS DA SILVA MENEZES:03329300418
Assinado de forma digital por MATHEUS DA SILVA MENEZES:03329300418
Dados: 2020.09.30 18:29:06 -03'00'

Dr. MATHEUS DA SILVA MENEZES
Examinador Externo

Dr. PAULO HENRIQUE ASCONAVIETA DA SILVA
Examinador Externo

Dra. SILVIA MARIA DINIZ MONTEIRO MAIA
Examinadora Interna

Discente: **BRUNO DE CASTRO HONORATO SILVA**

BRUNO DE CASTRO HONORATO SILVA

"O Problema do Caixeiro Viajante com Cota, Múltiplos Passageiros, Transporte Incompleto e Tempo de Coleta"

Esta Tese foi julgada adequada para a obtenção do título de doutor em Ciência da Computação e aprovada em sua forma final pelo Programa de Pós-Graduação em Sistemas e Computação do Departamento de Informática e Matemática Aplicada da Universidade Federal do Rio Grande do Norte.

Dr. MARCO CÉSAR GOLDBARG (UFRN)
Presidente – Orientador

Profa. Dra. ANNE MAGALY DE PAULA CANUTO
Coordenadora do PPgSC

Banca Examinadora

Dra. ELIZABETH FERREIRA GOUVÊA GOLDBARG
Examinadora Interna

Dr. LUCÍDIO DOS ANJOS FORMIGA CABRAL
Examinador Externo

MATHEUS DA SILVA MENEZES:03329300418
Assinado de forma digital por MATHEUS DA SILVA MENEZES:03329300418
Dados: 2020.09.30 18:31:33 -03'00'

Dr. MATHEUS DA SILVA MENEZES
Examinador Externo

Dr. PAULO HENRIQUE ASCONAVIETA DA SILVA
Examinador Externo

Dra. SILVIA MARIA DINIZ MONTEIRO MAIA
Examinadora Interna

Discente: **BRUNO DE CASTRO HONORATO SILVA**

Setembro, 2020

O simples pode ser mais difícil que complexo. Você precisa trabalhar duro para deixar o seu pensamento limpo e manter a simplicidade.

Steve Jobs

À minha esposa Elaine, pelo amor, paciência e apoio nos momentos difíceis ao longo desta jornada. Aos meus filhos, Eduardo e Manuela, pelo que representam na minha vida.

Aos meus pais, Lassance e Hayda, meu irmão, Breno, e a toda a minha família que, com muito carinho e apoio, não mediram esforços para que eu chegasse até esta etapa de minha vida e vibraram comigo em cada conquista desta jornada.

Agradecimentos

Ao meu orientador, o professor Marco Goldbarg, pela orientação, oportunidade e motivação nos momentos de dificuldade. Á professora Elizabeth Goldbarg pela co-orientação e correções precisas para melhoria da pesquisa.

À minha esposa Elaine Padilha pelo suporte, carinho e amor a mim dedicados. Aos meus pequenos Eduardo e Manuela, pelo que representam na minha vida. Aos meus pais Hayda e Lassance, por serem exemplos de pessoas honestas, regradas e batalhadoras, e por serem responsáveis por boa parte da plenitude que tive para chegar até aqui. A criação e educação que vocês me proveram foram essências na minha vida. Ao meu irmão Breno pelo grande amigo que é.

À toda minha família, no Ceará e no Paraná, por sempre me apoiarem e torcerem muito por mim.

À todos os amigos do LAE, especialmente os colegas que participaram diretamente comigo desta jornada: Islame Felipe, Gustavo Sabry, Sidemar Fideles, Thiago Soares e Ygor Medeiros.

Aos meus amigos de Natal-RN que sempre torceram por mim: Janduí Egito, Paulo Gonçalves, Carla Jéssica e Lane Alves.

Aos professores da banca de defesa: Silvia Maia, Matheus Menezes, Lucidio Cabral e Paulo Asconavieta, por terem lido o meu trabalho e pelas valiosas correções e sugestões. À professora da banca de qualificação Thatiana Navarro pelas sugestões de melhorias.

À todos os colegas que prestigiaram minha defesa.

À todos os professores e funcionários do PPgSC/UFRN.

Lista de tabelas

Lista de abreviaturas e siglas

MoD – Mobilidade sob Demanda

DARP – *Dial-a-Ride Problem*

PDP – *Pickup and Delivery Problem*

RP – *Ridesharing Problem*

PCV-CPTIT – Problema do Caixeiro Viajante com Cota, Múltiplos Passageiros, Transporte Incompleto e Tempo de Coleta

PCV-C – Problema do Caixeiro Viajante com Cota

ACO – *Ant Colony Optimization*

AS – *Ant System*

ILS – *Iterated Local Search*

AM – Algorimto Memético

PCV-P – Problema do Caixeiro Viajante com Passageiros

PCV-CP – Problema do Caixeiro Viajante com Cota e Passageiros

PCV – Problema do caixeiro Viajante

HC – Heurística de Carregamento

HBL-Mo – Heurística de Busca Local com Múltiplos-operadores

HN – Heurística *Naive*

GRASP – *Greedy Randomized Adaptive Search Procedure*

LRC – Lista Restrita de Candidatos

PR – *Path Relinking*

AG – Algoritmo Genético

AM-VC – *Algoritmo Memético com Variação Cultural*

IRACE – *Iterated Racing for Automatic Algorithm Configuration*

Lista de algoritmos

Sumário

1 Introdução

O consumo colaborativo é um dos principais paradigmas que influenciam o mundo comercial moderno. De acordo com (HEIKKILA, 2014), pelo menos cinco dinâmicas contribuem para a consolidação do consumo colaborativo: sustentabilidade, mudanças sociais, recessão econômica, aprimoramentos tecnológicos e oportunidades para novos modelos de negócios. No segmento de transporte, esse paradigma tornou os sistemas de Mobilidade sob Demanda (MoD), um dos atores mais poderosos na transformação da mobilidade urbana. Esse modelo de transporte incentiva a mobilidade compartilhada, em vez dos serviços de transporte tradicionais que seriam subutilizados.

Um modelo de otimização bem conhecido para o gerenciamento de sistemas MoD é o *Dial-a-Ride Problem* (DARP). O conceito do DARP veio de casos práticos nos quais os clientes solicitavam serviços de transporte por telefone. Considerando uma frota de veículos, o objetivo desse problema é programar rotas para transportar vários usuários que especificam pontos de embarque e desembarque (CORDEAU; LAPORTE, 2007). O DARP é considerado uma generalização do *Pickup and Delivery Problem* (PDP), no qual as pessoas são transportadas em vez de produtos (SAVELSBERGH; SOL, 1995). Uma revisão das generalizações do DARP é apresentada por (HO et al., 2018).

Outro problema relacionado ao gerenciamento de sistemas MoD é o *Ridesharing Problem* (RP). Esse problema consiste em encontrar o melhor arranjo de passageiros para compartilhar os custos de uma única viagem em um veículo, sem levar em consideração um acordo prévio ou um histórico de cooperação (NOURINEJAD; ROORDA, 2014). Uma visão geral sobre o RP, suas variantes e métodos de resolução é apresentada em (SIMONIN; O'SULLIVAN, 2014). Vale ressaltar que um fator comum nas versões tradicionais do DARP, RP e suas variantes (HO et al., 2018; SIMONIN; O'SULLIVAN, 2014) é a maximização da satisfação do usuário que exige transporte.

Como o DARP e o RP, os problemas mais estudados da literatura relacionados aos sistemas MoD, se concentram na satisfação do usuário, propõe-se o Problema do Caixeiro

Viajante com Cota, Múltiplos Passageiros, Transporte Incompleto e Tempo de Coleta (PCV-CPTIT). Esse problema é uma variante do Problema do Caixeiro Viajante com Cota (PCV-C)(AWERBUCH et al., 1998) contextualizado no segmento de mobilidade compartilhada para representar situações práticas em que os interesses do motorista são o foco. No PCV-C tradicional, o caixeiro viajante não é obrigado a visitar todas as cidades em sua *tour*. Um bônus é associado a cada cidade, e o caixeiro deve atingir uma determinada cota de bônus visitando um número suficiente de cidades. Esse problema consiste em encontrar um *tour* de custo mínimo tal que a restrição de cota seja atendida.

No PCV-CPTIT, o caixeiro é o motorista de um veículo e pode reduzir os custos de deslocamento compartilhando as despesas com os passageiros enquanto visita um conjunto de localidades para atingir uma determinada cota de bônus. Ele deve respeitar as limitações orçamentárias e o tempo máximo de viagem de cada passageiro. Cada passageiro pode ser transportado diretamente para o destino desejado, modalidade esta chamada de transporte completo, ou para um destino alternativo, resultando no transporte incompleto. Como apontado por (ZHAO et al., 2018), em alguns casos, os locais de embarque ou desembarque de passageiros podem estar espacialmente próximos, mas topologicamente inacessíveis ou até mesmo temporariamente inacessíveis para veículos. A ideia de desembarque em destino alternativo sugere que, quando houver a possibilidade de compartilhar uma viagem, preocupações pró-ambientais ou de economia de dinheiro podem induzir os passageiros a concordar em atender às suas necessidades em um destino semelhante (LIRA et al., 2018). Diferentemente do PCV-C, há uma penalidade de tempo coleta no caso de caixeiro optar por coletar o bônus de uma cidade por ele visitada.

O problema proposto aqui combina elementos do sistema estático de compartilhamento de viagens, em que cada solicitação de viagem tem uma origem e um destino conhecidos antes da execução de um processo de alocação de passageiros (AGATZ et al., 2012), com os conceitos de coleta e entrega seletiva (SCHÖNBERGER; KOPFER; MATTFELD, 2003) e o viés de desembarque em destinos alternativos (LIRA et al., 2018). Conforme abordado em (ALONSO-MORA et al., 2017), muitos sistemas de MoD operando nas cidades concentram-se em carros e só são operacionais para este tipo de veículo. O modelo proposto não tem essa limitação, isto é, pode ser aplicado para gerenciar casos de mobilidade compartilhada, independentemente do tipo de veículo.

Grande parte da literatura recente sobre mobilidade como serviço considera: planejar itinerários ideais para uma frota de veículos de tamanho fixo que atenda ao maior número possível de solicitações de viagem (RIEDLER; RAIDL, 2018; HO et al., 2018), gerenciamento

de sistemas MoD com base em veículos autônomos (LEVIN et al., 2017; FAGNANT; KOCKELMAN, 2018; FARHAN; CHEN, 2018), alocação ótima de um determinado conjunto de solicitações de viagens a uma única viagem (ALONSO-MORA et al., 2017), restrições operacionais para maximizar o atendimento de solicitações de viagem feitas de forma *on-line* e dinâmicas (SIMONIN; O'SULLIVAN, 2014; DONG et al., 2018; ZHANG et al., 2018; CHEN; LIU; WEI, 2018; HOU; LI; ZHANG, 2018) e melhorias de mobilidade urbana em cidades específicas (MASSON et al., 2017; LOKHANDWALA; CAI, 2018; WANG; ZHENG; LIM, 2018). Estes estudos tratam de problemas orientados para maximizar o número de solicitações de viagem atendidas. Nenhum desconsidera as penalidades por solicitações de viagens não atendidas ou suporta o transporte incompleto, i. e., o desembarque de passageiros em destinos alternativos.

O PCV-CPTIT requer que o motorista conheça as solicitações de viagens com antecedência. É uma suposição realista, pois uma parte significativa da demanda de transporte de um usuário pode ser conhecida com antecedência. Isto viabiliza a combinação das preferências de passageiros com as do motorista. Também ajuda a reduzir efeitos indesejados que podem resultar de períodos não alocado, ou seja, o período em que o veículo está disponível para ser atribuído a alguém e viaja sem passageiro.

A aplicação prática do PCV-CPTIT ocorre em situações em que uma pessoa que viaja em um veículo particular utiliza um serviço de compartilhamento de viagens para oferecer caronas e reduzir as despesas de transporte enquanto visita algumas localidades para satisfazer uma cota pessoal. Por exemplo, considere um *courier* local e independente, que deve atingir uma cota mínima de trabalho. O PCV-CPTIT pode ser aplicado para auxiliar esse individuo no planejamento de suas entregas, minimizando os custos de transporte com a concessão dos respectivos assentos de veículos para potenciais passageiros. Existem restrições quanto ao limite de orçamento de cada passageiro e ao tempo máximo de permanência de cada passageiro dentro do veículo. Os passageiros podem ser desembarcados no destino final ou nos pontos de transbordo. O modelo não permite que o *courier* lucre com as viagens. Visa apenas a redução de custos. O processo de otimização é realizado exclusivamente do ponto de vista do *courier*. Ele não é obrigado a atender às solicitações de transporte feitas pelos potenciais passageiros, se assim o preferir.

Casos práticos relacionados a vendas e turismo também são pertinentes. No primeiro, o caixeiro deve escolher quais locais visitar para atingir uma cota mínima de vendas e em que ordem visitá-los para aproveitar as melhores solicitações de viagem. No segundo caso, o caixeiro pode ser considerado um turista que dirige um carro e deve escolher as

melhores atrações turísticas para visitar durante sua viagem de férias. Este turista usa um sistema de compartilhamento de viagens para compartilhar os custos da viagem. Nos dois casos, o motorista negocia descontos com passageiros que concordam em ir para destinos alternativos.

Esses exemplos práticos mostram o potencial do modelo proposto para auxiliar o planejamento de rotas no contexto de mobilidade compartilhada. Seja em situações sociais, como indicado no exemplo do turismo, ou em novas possibilidades de negócios, graças ao surgimento das tecnologias mobilidade compartilhada, conforme descrito no caso do *courier*.

1.1 Objetivos da Pesquisa

O objetivo geral da pesquisa realizada é propor e analisar o problema de otimização inédito na literatura, denominado Problema do Caixeiro Viajante com Cota, Múltiplos Passageiros, Transporte Incompleto e Tempo de Coleta. Tal estudo contém a descrição formal do problema, seguido de seu modelo matemático e aplicação de abordagens exatas, heurísticas e meta-heurísticas na resolução do problema.

Os objetivos específicos desdobram-se em:

- Descrever o problema e propor um modelo matemático para o mesmo;
- Propor e testar o banco de instâncias;
- Analisar a dificuldade do problema;
- Implementar e validar abordagens heurísticas de resolução do problema;
- Realizar experimentos computacionais a cerca das abordagens heurísticas propostas.

O experimento heurístico desta pesquisa compreende a adaptação de algumas meta-heurísticas da família de algoritmos *Ant Colony Optimization*(ACO)(DORIGO; STÜTZLE, 2003). O primeiro algoritmo ACO, conhecido como Ant System(AS), foi introduzido por (DORIGO; MANIEZZO; COLORNI, 1996) e aplicado ao Problema do Caixeiro Viajante (DORIGO; STÜTZLE, 2003). Optou-se por implementar esses algoritmos, uma vez que eles foram bem-sucedidos em outros problemas de roteamento semelhantes ao PCV-CPTIT (LIAW; CHANG; TING, 2017; TRIPATHY et al., 2018; HERBAWI; WEBER, 2011; HUANG; JIAU; LIU, 2018; ZUFFEREY; FARRES; GLARDON, 2015).

A execução do experimento heurístico compreende também as meta-heurísticas *Iterated Local Search* (ILS)(BAUM, 1986a), Greedy Randomized Adaptive Search Procedure (GRASP)(FEO; RESENDE, 1989) e Algoritmo Memético (AM)(MOSCATO; PLATA; NORMAN, 1999). Assim como os algoritmos de otimização por colônia de formigas, estas meta-heurísticas foram adaptadas por possuírem boas aplicações recentes em problemas correlatos ao em estudo (MORAES et al., 2018; KOCZY; FOLDESI; TUU-SZABO, 2018; BAO; LE; NGUYEN, 2018; SANTOS; CHAGAS, 2018; CHAMI et al., 2018).

Para suportar o experimento heurístico, as seguintes abordagens auxiliares foram implementadas:

- Heurística *Naive*: método concebida sobre uma abordagem ingênua de resolução do problema para se obter a ancoragem de resultados, e, com base nesta nestes resultados, validar a eficiência de outras abordagens meta-heurísticas propostas neste estudo;
- Heurística de Carregamento: método que fornece um esquema de alocação de passageiros para uma rota. Este algoritmo foi concebido para apoiar as outras abordagens heurísticas de resolução do problema propostas neste estudo;
- Heurística de Busca Local: método de busca baseado em múltiplos operadores de vizinhança projetado para ser acoplado em outra abordagens heurística e melhorar seus resultados.

1.2 Organização do Trabalho

A estrutura deste trabalho é a seguinte: problemas tidos como arcabouço do PCV-CPTIT são descritos no capítulo 2. O capítulo 3 apresenta o problema e sua formulação. As abordagens heurísticas propostas são introduzidas no capítulo 4. As abordagens meta-heurísticas são apresentadas no capítulo 5. Discutem-se os experimentos computacionais no capítulo 6. As conclusões e as direções futuras da pesquisa são apresentadas no capítulo 7.

2 Trabalhos correlatos

Neste capítulo são apresentados o Problema do Caixeiro Viajante com Passageiros (*PCV-P*) e o Problema do Caixeiro Viajante com Cota e Passageiros(*PCV-CP*), descritos nas seções 2.1 e 2.2, respectivamente. Estes são os problemas dos quais o *PCV-CPTIT* se deriva. Este capítulo tem por finalidade apresentar os problemas que forneceram a base teórica para o desenvolvimento do estudo conduzido neste trabalho.

2.1 Problema do Caixeiro Viajante com Passageiros

Nesta seção são apresentados detalhes sobre o *PCV-P*, tais como sua descrição (seção 2.1.1) e modelagem matemática (seção 2.1.2).

2.1.1 Descrição do problema

Descrito de forma pioneira por (CALHEIROS, 2017), o *PCV-P* é uma variante do Problema do Caixeiro Viajante (*PCV*). Neste problema, considera-se um grafo $G = (N, M)$, onde N é o conjunto de nós (cidades), $|N| = n$, e M é um conjunto de arestas (estradas), $|M| = m$. O conjunto de potencias passageiros solicitando caronas é indicado por L. Cada elemento $l \in L$, possui localidades de origem e destino, e um limite financeiro. No *PCV-P*, o caixeiro deve avaliar as solicitações de viagens a medida em que visita as cidades de seu *tour*. A origem de um passageiro l corresponde ao vértice em que este deve ser embarcado. Portanto, o destino de l corresponde ao vértice em que o mesmo deve ser desembarcado. Qualquer passageiro l possui o destino diferente de sua origem. O custo de cada trecho que compõe o *tour* é igualitariamente dividido entre os ocupantes do veículo. O caixeiro deve respeitar o limite orçamentário que dispõe cada passageiro para o rateio das despesas de viagem. As solicitações de viagem que forem atendidas devem ser concluídas em seus respectivos pontos de desembarque. Todas as solicitações de viagem possuem o mesmo limite orçamentário. O veículo utilizado durante o percurso possui uma

capacidade k de acentos disponíveis para carona que deve ser respeitada. O *tour* começa e termina no vértice 1, que é considerado como uma base para o caixeiro. O objetivo do problema é encontrar o circuito de custo mínimo, representado pelo somatório do custo de cada aresta trafegada dividido pelo número de passageiros a bordo naquele trecho, respeitando as restrições previamente descritas.

2.1.2 Modelagem matemática

Esta formulação matemática foi proposta em (CALHEIROS, 2017). Ela incorpora algumas restrições relativas ao embarque de passageiros à formulação desenvolvida em (MILLER; TUCKER; ZEMLIN, 1960). O problema é formulado em (2.1)–(2.11). Os parâmetros e variáveis são definidos da seguinte forma.

Parâmetros

L: conjunto de índices dos passageiros;
org(l): cidade de origem do passageiro l, $l \in L$;
dst(l): cidade de destino do passageiro l, $l \in L$;
bud(l): limite financeiro do passageiro l, $l \in L$;
k: capacidade do carro;
d_{ij}: custo operacional do carro c do vértice i para o vértice j, $(i,j) \in M$.

Variáveis

f_{ij}: variável binária que indica se o veículo atravessa a aresta (i, j) de i para j ($f_{ij} = 1$) ou não ($f_{ij} = 0$);
v_{lij}: variável binária que indica se passageiro l está embarcado no veículo durante a travessia da aresta (i, j) de i para j ($v_{lij} = 1$) ou não ($v_{lij} = 0$);
u_i: ordem em que o vértice i é visitado no *tour*.

$$\min \sum_{i,j \in N} \frac{d_{ij} f_{ij}}{1 + \sum_{l \in L} v_{lij}} \tag{2.1}$$

Sujeito a

$$\sum_{j \in N} f_{ij} = 1 \ , \quad \forall i \in N \tag{2.2}$$

$$\sum_{i \in N} f_{ij} = 1 \ , \quad \forall j \in N \tag{2.3}$$

$$u_i - u_j + 1 \leq n\,(1 - f_{ij})\,, \quad \forall i, j \in N \backslash \{1\} \tag{2.4}$$

$$\sum_{l \in L} v_{lij} \leq k f_{ij}, \quad \forall i, j \in N \tag{2.5}$$

$$\sum_{i \in N} v_{lir} + \sum_{i \in N} v_{l1i} = 0, \quad \forall l \in L | r = org(l), r \neq 1 \tag{2.6}$$

$$\sum_{i \in N} v_{lsi} + \sum_{i \in N} v_{li1} = 0, \quad \forall l \in L | s = dst(l), s \neq 1 \tag{2.7}$$

$$\sum_{j \in N} v_{lij} + \sum_{j \in N} v_{lji} = 0, \quad \forall l \in L, i \in N | i \neq org(l), i \neq dst(l) \tag{2.8}$$

$$\sum_{i,j \in N} \frac{d_{ij} f_{ij}}{1 + \sum_{l \in L} v_{lij}} \leq bud(l) \quad \forall l \in L \tag{2.9}$$

$$f_{ij}, v_{lij} \in \{0, 1\}, \quad \forall i, j \in N, \forall l \in L \tag{2.10}$$

$$u_i \in \Re_{\geq 0}, \quad 2 \leq i \leq n \tag{2.11}$$

A função objetivo (2.1) soma o custo das arestas trafegadas e divide pelo número de pessoas que estão no carro naquele trecho, incluindo o caixeiro. As restrições (2.2) e (2.3) garantem que cada vértice deverão ter somente duas arestas adjacentes, uma de entrada e uma de saída. A restrição (2.4) é a contribuição de *Miller-Tucker-Zemlin* (1960) no modelo tradicional do *PCV*, por meio da eliminação de *subtours*. Ou seja, as restrições (2.2)–(2.4) asseguram um ciclo Hamiltoniano para o motorista, representado pela variável f_{ij}. A restrição (2.5) certifica que os passageiros irão percorrer o mesmo caminho que o caixeiro e que a capacidade do veículo não seja excedida. A restrição (2.6) inviabiliza o retorno de qualquer passageiro à sua origem e que passageiros cuja origem não seja a cidade inicial estejam a bordo partindo da cidade 1. A restrição (2.7) assegura que nenhum passageiro irá passar de seu destino e também que passageiros cujo destino não seja a cidade inicial estejam a bordo a caminho da cidade 1. As restrições (2.6) e (2.7) impedem o ciclo de passageiros durante o *tour*, ou seja, garante que no primeiro trecho só poderão estar embarcados passageiros cuja origem seja a cidade inicial e que no último trecho só estarão a bordo passageiros cujo destino seja a cidade 1. Para garantir que as arestas ativas dos passageiros formem um caminho, é introduzida a restrição (2.8). Por fim, a restrição (2.9) incorpora os limites financeiros de cada passageiro. As equações (2.10) e (2.11) garantem a integridade e não-negatividade das variáveis de decisão.

2.2 Problema do Caixeiro Viajante com Cota e Passageiros

Nesta seção são apresentados detalhes sobre *PCV-CP*, tais como sua descrição(seção 2.2.1) e modelagem matemática (seção 2.2.2).

2.2.1 Descrição do problema

Proposto em (SILVA, 2017), este problema inclui no tradicional PCV-C o compartilhamento do veículo com passageiros para rateio de eventuais despesas. Segundo (SILVA, 2017), o PCV-CP é uma variação do PCV-C com o problema de *ridesharing*(NOURINEJAD; ROORDA, 2014) em que o caixeiro pode reduzir o custo total da viagem compartilhando os custos de deslocamento entre trechos com eventuais passageiros que solicitam carona. O problema considera um grafo $G = (N, A, Q)$, onde $N = 1, ..., n$ é o conjunto de cidades, $A = 1, ..., m$ é o conjunto de estradas que ligam as cidades de N e $Q = q_1, q_2, ..., q_n$ é o conjunto de bônus associados às cidades, onde $q_1 = 0$. Os bônus são coletados a cada cidade visitada e a soma total de bônus coletado deve ser no mínimo K. Custos c_{ij} para viajar de uma cidade i para a j, com $(i, j) \in A$, estão associados às estradas que as ligam. A viagem é iniciada na cidade 1. Existe um conjunto de solicitações de viagem, $L = 1, ..., l$, onde cada pessoa se encontra em uma cidade de N e deseja se deslocar para uma das cidades que o caixeiro pode visitar. Contudo, cada passageiro está disposto a pagar no máximo w_l, com $l \in L$, unidades monetárias pela viagem. Essa restrição de contribuição máxima garante uma vantagem econômica para o passageiro assim como impossibilita uma possível extensão da viagem para aumentar a contribuição do passageiro e consequentemente diminuir o custo do motorista. Cada passageiro l tem como cidade de origem $org(l)$ e destino $dst(l)$, $org(l) \neq dst(l)$. O custo c_{ij} de deslocamento entre as cidades i e j, é divido igualmente entre os ocupantes do veículo, incluindo o motorista. O caixeiro pode transportar no máximo R passageiros em seu veículo. Os passageiros e o motorista não levam em consideração a duração da viagem e o tempo de coleta do bônus.

2.2.2 Modelagem matemática

Uma formulação não linear inteira para o PCV-CP é apresentada em (2.12) - (2.28). Os parâmetros e variáveis estão listados a seguir.

Parâmetros

n: número de vértices

c_{ij}: custo para se percorres um arco (i, j)

q_i: bônus associado ao vértice i

$org(l)$: localidade de embarque da l-enésima solicitação de viagem

$dst(l)$: localidade de desembarque da l-enésima solicitação de viagem

w_l: orçamento máximo da l-enésima solicitação de viagem

K: cota mínima de bônus a ser coletado

R: capacidade máxima de ocupantes veículo

Varáveis:

x_{ij}: indica os arcos percorridos pelo caixeiro (i, j), $x_{ij} = 1$, caso contrário, $x_{ij} = 0$

p_i: indica a coleta de bônus no vértice i, $p_i = 1$, caso contrário, $p_i = 0$

v_{ij}^l: indica quais arcos (i, j) o passageiro l percorre, $v_{ij}^l = 1$, caso contrário, $v_{ij}^l = 0$

$$\min \sum_{(i,j)\in E} \frac{x_{ij}c_{ij}}{1 + \sum_{l\in L} v_{ij}^l} \tag{2.12}$$

Sujeito a:

$$\sum_{i\in N\setminus\{j\}} x_{ij} \leq 1 \qquad \forall j \in N\setminus\{s\} \tag{2.13}$$

$$\sum_{i\in N\setminus\{j\}} x_{ji} \leq 1 \qquad \forall j \in N\setminus\{s\} \tag{2.14}$$

$$\sum_{i\in N\setminus\{s\}} x_{si} = 1 \tag{2.15}$$

$$\sum_{i\in N\setminus\{s\}} x_{is} = 1 \tag{2.16}$$

$$\sum_{i\in N\setminus\{j\}} x_{ij} - \sum_{i\in N\setminus\{j\}} x_{ji} = 0 \qquad \forall j \in N\setminus\{s\} \tag{2.17}$$

$$u_i - u_j + (n-1)x_{ij} \leq n-2 \qquad 2 \leq i, j \leq n, i \neq j \tag{2.18}$$

$$\sum_{(i,j)\in A} p_i x_{ij} \geq K \tag{2.19}$$

$$\sum_{l\in L} v_{ij}^l \leq R x_{ij} \qquad \forall (i,j) \in A \tag{2.20}$$

$$\sum_{(i,j)\in A} \frac{v_{ij}^{l} c_{ij}}{1+\sum_{u\in L} v_{ij}^{u}} \leq w_l \qquad \forall l \in L \tag{2.21}$$

$$\sum_{i\in N\setminus\{j\}} v_{ij}^{l} - \sum_{i\in N\setminus\{j\}} v_{ji}^{l} = f_{lj} \qquad \forall l \in L\ ,\ j \in N\setminus\{org(l)\} \tag{2.22}$$

$$\sum_{i\in N\setminus\{org(l)\}} v_{iorg(l)}^{l} = 0 \qquad \forall l \in L \tag{2.23}$$

$$\sum_{i\in N\setminus\{dst(l)\}} v_{dst(l)i}^{l} = 0 \qquad \forall l \in L \tag{2.24}$$

$$\sum_{i\in N\setminus\{dst(l)\}} v_{dst(l)i}^{l} = 0 \qquad \forall l \in L \tag{2.25}$$

$$p_l = 0 \tag{2.26}$$

$$x_{ij} \in \{0,1\} \qquad \forall (i,j) \in A \tag{2.27}$$

$$p_i \in \{0,1\} \qquad \forall i \in N \tag{2.28}$$

A função objetivo, expressa em 2.12, visa minimizar o custo total da viagem do caixeiro fazendo o rateio dos custos de deslocamento dentre os ocupantes do carro, incluindo o motorista. As restrições 2.13 e 2.14 permitem que o percurso do caixeiro possa ser composto por um subconjunto de vértices de N. As restrições 2.15 e 2.16 garantem que o percurso feito pelo caixeiro inicia e termina na cidade inicial s. A restrição 2.17 garante que ao entrar em uma cidade, diferente de sua origem, o caixeiro tem que desembarcar. A restrição 2.18 garante que só haverá um subciclo na solução (MILLER; TUCKER; ZEMLIN, 1960). A restrição 2.19 garante a coleta mínima de bônus K durante o percurso. A restrição 2.20 garante a capacidade do carro, no máximo R. A restrição 2.21 garante que cada passageiro $l \in L$ pagará no máximo w_l unidades monetárias pela viagem. A restrição 2.22 garante que se o passageiro embarcar, ele deve ser conduzido até seu destino. As restrições 2.23 e 2.24 garantem que o passageiro $l \in L$ só trafegará da sua origem $org(l)$ ao seu destino $dst(l)$. A restrição 2.25 garante o retorno de um passageiro somente para origem, caso em que sua cidade de origem seja diferente de s e seu destino seja a cidade em que o caixeiro iniciou o percurso, caso contrário o passageiro não está apto para embarcar, uma vez que sua cidade destino já passou. E por último, a restrição 2.26 garante que não há coleta de bônus na cidade de origem do caixeiro.

3 O Problema do Caixeiro Viajante com Cota, Múltiplos Passageiros, Transporte Incompleto e Tempo de Coleta

Neste capítulo, é apresentada um descrição formal para o problema proposto junto a seção 3.1. Na seção 3.2, é descrito o modelo matemático proposto. A representação da solução do problema é exposta na 3.3. Por fim, uma análise da dificuldade do problema é realizada junto a seção 3.4.

3.1 Descrição formal

O Problema do Caixeiro Viajante (PCV) pode ser modelado como um grafo direcionado ponderado completo $G = (N, A)$ onde N é o conjunto de vértices, $A = \{(i, j) \mid i, j \in N\}$ é o conjunto de arcos e $C = [c_{ij}]$ é a matriz de pesos dos arcos definida de forma que c_{ij} seja o custo associado com o arco (i, j). O objetivo é determinar o menor ciclo hamiltoniano em G. O PCV é NP-difícil (KARP, 1975) e também um dos problemas mais investigados na Otimização Combinatória. A Figura 1 ilustra o PCV.

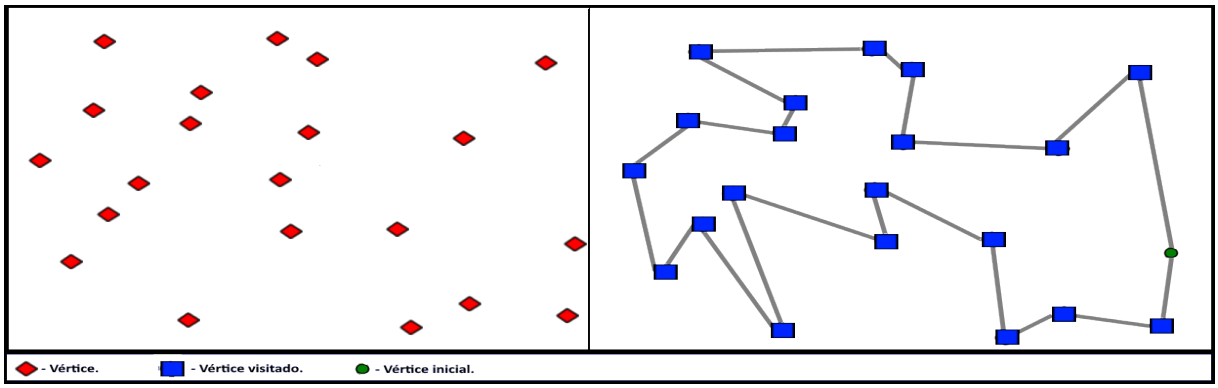

Figura 1: Ilustração da resolução do PCV.

No PCV-C, há um bônus associado a cada vértice no grafo. O caixeiro deve coletar uma cota mínima de bônus nos vértices visitados. Assim, o caixeiro precisa descobrir quais cidades visitar para atingir a cota mínima. O objetivo é encontrar um *tour* de custo mínimo, de modo que a soma dos bônus coletados nos vértices visitados seja pelo menos a cota mínima. Esse problema foi introduzido em (AWERBUCH et al., 1998), onde também foi apresentado outros problemas de roteamento orientados à coleta de prêmios. A Figura 2 ilustra o PCV-C.

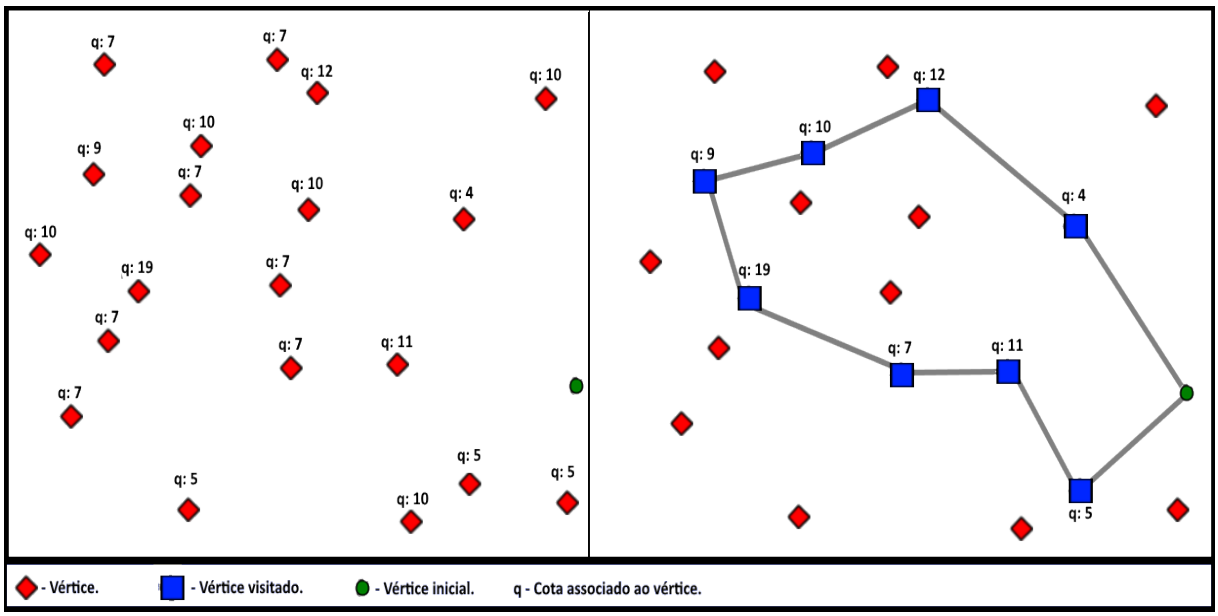

Figura 2: Ilustração da resolução do PCV-C.

O PCV-CPTIT é uma variante do PCV-C em que o caixeiro é o motorista de um veículo e pode reduzir os custos de viagem compartilhando as despesas com os passageiros. Existe uma solicitação de viagem, associada a cada pessoa que exige uma viagem, que consiste em um ponto de embarque e desembarque, um limite de orçamento, um limite para a duração da viagem e penalidades associadas a pontos de desembarque alternativos. Uma penalidade é associada a cada ponto diferente do ponto de entrega exigido pela pessoa que exige uma carona. O caixeiro pode aceitar ou recusar os pedidos de viagem. Este modelo combina elementos do sistema de compartilhamento de caronas (AGATZ et al., 2012) com destinos alternativos (LIRA et al., 2018) e o problema de coleta e entrega seletiva (SCHÖNBERGER; KOPFER; MATTFELD, 2003).

Seja $G(N, A)$ um grafo ponderado completo, onde $N = \{1, ..., n\}$ seja o conjunto de vértices e $A = \{(i, j) \mid i, j \in N\}$ o conjunto de arcos. O caixeiro começa na cidade $s = 1$ e visita cada nó de um ciclo $\Psi = (N', A')$, $N' \subseteq N$, $A' \subseteq A$, no máximo uma vez. Um bônus, q_i, é associado a cada vértice $i \in N$. O caixeiro pode escolher entre coletar ou não

o bônus de um vértice visitado. A coleção de bônus no vértice i requer unidades de tempo g_i. O caixeiro deve coletar, no mínimo, K unidades de bônus.

O custo e o tempo necessários para percorrer o arco $(i, j) \in A$ são c_{ij} e t_{ij}, respectivamente. O veículo tem capacidade para, no máximo, R passageiros, além do caixeiro. O caixeiro divide o custo de se percorrer o arco $(i, j) \in A$ com os passageiros que também atravessam (i, j). O custo é dividido igualmente entre os ocupantes do veículo. Tome L como o conjunto de solicitações de viagem, cada solicitação associada a um cliente. Denota-se por $L_i \subseteq L$ o subconjunto de solicitações de viagem para o qual $i \in N$ é a cidade de origem. Seja l uma pessoa que solicita uma carona. Denota-se por $org(l)$ e $dst(l)$, os pontos de embarque e desembarque exigidos por l. O orçamento máximo que uma pessoa l para ratear custos de uma viagem é indicada por w_l, e a duração máxima que l está disposto a permanecer embarcado no veículo é indicada por b_l. O caixeiro pode levar o passageiro para um local diferente do destino exigido, ou seja, um destino alternativo. Nesse caso, há uma penalidade a ser paga ao respectivo passageiro pelo seu transporte incompleto. A penalidade de levar l para a cidade j, $j \neq dst(l)$, é indicada por h_{lj}. O PCV-CPTIT consiste em encontrar um ciclo, Ψ, de modo que o custo da rota e as penalidades sejam minimizados, e a restrição mínima de cota seja atendida. A Figura 3 ilustra o PCV-CPTIT. Nesta Figura, valores nulos para q indicam que não houve coleta do bônus no respectivo vértice visitado.

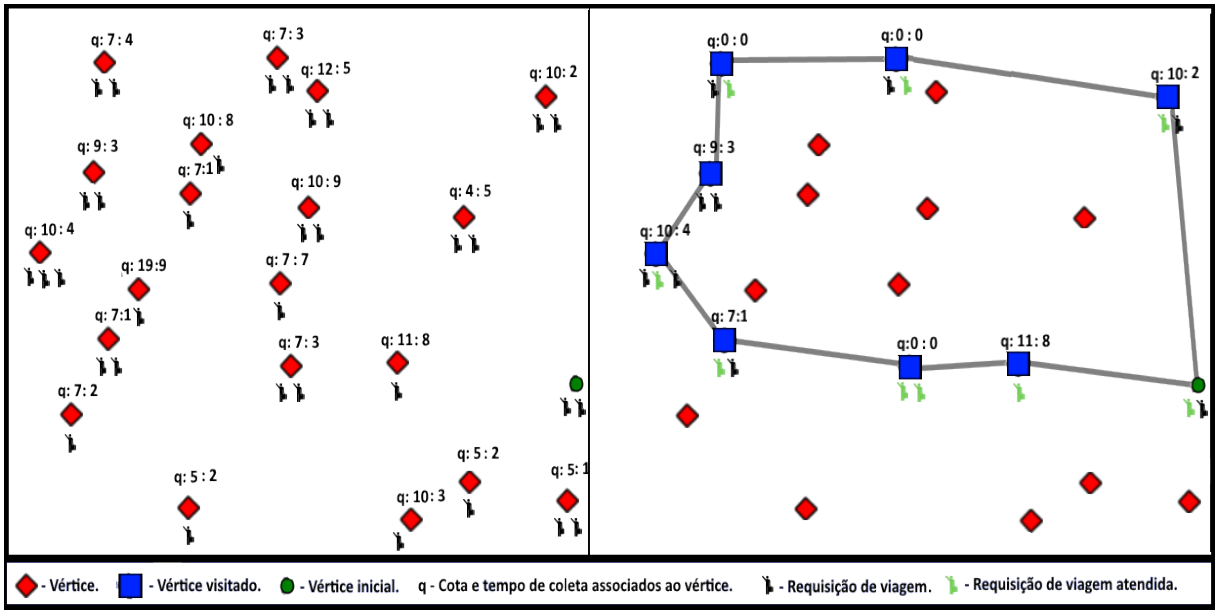

Figura 3: Ilustração da resolução do PCV-CPTIT.

O PCV-C é um problema NP-difícil (AWERBUCH et al., 1998) e um caso especial do PCV-CPTIT quando a lista de potenciais passageiros exigindo uma carona está vazia e o

tempo gasto para coletar o bônus em cada vértice é zero. Assim, o PCV-CPTIT também está na classe NP-difícil. A base do PCV-CPTIT são os estudos pioneiros dispostos em (CALHEIROS, 2017) e (SILVA, 2017).

Há na literatura, outros problemas clássicos que compartilham características com o PCV-CPTIT. No Traveling Salesman With Pickup and Delivery (PARRAGH; DOERNER; HARTL, 2008), embarques e desembarques devem ser realizados. Há também variantes dos problemas de roteamento com coleta onde a satisfação das metas é variável em função da rota adotada. Destes, destacam-se o *Traveling Salesman with Profits* (FEILLET; DEJAX; GENDREAU, 2005), *Prize Collecting Traveling Salesman* (BALAS, 2006), *Attractive Traveling Salesman* (ERDOGAN; CORDEAU; LAPORTE, 2010), *Selective Travelling Salesman* (GENDREAU; LAPORTE; SEMET, 1998) e *Orienteering Problems* (GUNAWAN; LAU; VANSTEENWEGEN, 2016).

3.2 Formulação matemática

Uma formulação não linear inteira para o PCV-CPTIT é apresentada em (3.1) - (3.22). Os parâmetros e variáveis estão listados a seguir.

Parâmetros

n: número de vértices
c_{ij}: custo para se percorrer um arco (i, j)
t_{ij}: tempo para se percorrer um arco (i, j)
q_i: bônus associado ao vértice i
g_i: tempo requerido para coletar o bônus em i
$org(l)$: localidade de embarque da l-enésima solicitação de viagem
$dst(l)$: localidade de desembarque da l-enésima solicitação de viagem
b_l: duração máxima da l-enésima solicitação de viagem
w_l: orçamento máximo da l-enésima solicitação de viagem
h_{lj}: penalidade associada ao desembarque de l na localidade j
K: cota mínima de bônus a ser coletado
R: capacidade máxima de ocupantes veículo

Variáveis

x_{ij}: indica os arcos percorridos pelo caixeiro (i, j), $x_{ij} = 1$, caso contrário, $x_{ij} = 0$

p_i: indica a coleta de bônus no vértice i, $p_i = 1$, caso contrário, $p_i = 0$

v_{ij}^l: indica quais arcos (i, j) o passageiro l percorre, $v_{ij}^l = 1$, caso contrário, $v_{ij}^l = 0$

f_{lj}: indica se o passageiro l desembarcou na localidade j, $f_{lj} = 1$, caso contrário, $f_{lj} = 0$

d_l: indica se o passageiro l foi embarcado, $d_l = 1$, caso contrário, $d_l = 0$

u_i: denota a ordem de visita do vértice i no *tour* do caixeiro. É válido ressaltar que $u_1 = 1$

$$\min \sum_{(i,j)\in E} \frac{x_{ij}c_{ij}}{1 + \sum_{l\in L} v_{ij}^l} + \sum_{l\in L}\sum_{j=1}^{n} h_{lj}f_{lj} \tag{3.1}$$

Sujeito a:

$$\sum_{i\in N\setminus\{j\}} x_{ij} \leq 1 \qquad \forall j \in N\setminus\{s\} \tag{3.2}$$

$$\sum_{i\in N\setminus\{j\}} x_{ji} \leq 1 \qquad \forall j \in N\setminus\{s\} \tag{3.3}$$

$$\sum_{i\in N\setminus\{s\}} x_{si} = 1 \tag{3.4}$$

$$\sum_{i\in N\setminus\{s\}} x_{is} = 1 \tag{3.5}$$

$$\sum_{i\in N\setminus\{j\}} x_{ij} - \sum_{i\in N\setminus\{j\}} x_{ji} = 0 \qquad \forall j \in N\setminus\{s\} \tag{3.6}$$

$$1 \leq u_i \leq n-1 \qquad \forall i \in N \tag{3.7}$$

$$u_i - u_j + (n-1)x_{ij} \leq n-2 \qquad 2 \leq i, j \leq n, i \neq j \tag{3.8}$$

$$\sum_{(i,j)\in A} p_i q_i x_{ij} \geq K \tag{3.9}$$

$$\sum_{(i,j)\in A} v_{ij}^l t_{ij} + \sum_{(i,j)\in A} p_i g_i v_{ij}^l \leq b_l \qquad \forall l \in L \tag{3.10}$$

$$\sum_{l\in L} v_{ij}^l \leq R x_{ij} \qquad \forall (i,j) \in A \tag{3.11}$$

$$\sum_{j \in N} f_{lj} \leq 1 \qquad \forall l \in L \tag{3.12}$$

$$\sum_{(i,j) \in A} \frac{v_{ij}^{l} c_{ij}}{1 + \sum_{u \in L} v_{ij}^{u}} \leq w_l \qquad \forall l \in L \tag{3.13}$$

$$f_{li} = 0 \qquad \forall l \in L \ , \ i = org(l) \tag{3.14}$$

$$v_{ij}^{l} \leq d_l \qquad \forall l \in L \ , \ (i,j) \in A \tag{3.15}$$

$$\sum_{i \in N \setminus \{j\}} v_{ij}^{l} - \sum_{i \in N \setminus \{j\}} v_{ji}^{l} = f_{lj} \qquad \forall l \in L \ , \ j \in N \setminus \{org(l)\} \tag{3.16}$$

$$\sum_{j \in N} f_{lj} = d_l \qquad \forall l \in L \tag{3.17}$$

$$\sum_{j \in N} v_{org(l)j}^{l} = d_l \qquad \forall l \in L \tag{3.18}$$

$$f_{li} + \sum_{j \in N} v_{ij}^{l} \leq 1 \qquad \forall l \in L \ , \ \forall i \in N \tag{3.19}$$

$$\sum_{i \in N} v_{iorg(l)}^{l} = 0 \qquad \forall l \in L \tag{3.20}$$

$$\sum_{j \in N \setminus \{s\}} v_{js}^{l} = f_{ls} \qquad \forall l \in L \tag{3.21}$$

$$\sum_{j \in N} v_{dst(l)j}^{l} = 0 \qquad \forall l \in L \tag{3.22}$$

$$x_{ij} \in \{0,1\} \qquad \forall (i,j) \in A \tag{3.23}$$

$$p_i \in \{0,1\} \qquad \forall i \in N \tag{3.24}$$

$$v_{ij}^{l} \in \{0,1\} \qquad \forall l \in L, (i,j) \in A \tag{3.25}$$

$$f_{lj} \in \{0,1\} \qquad \forall l \in L, \forall j \in N \tag{3.26}$$

$$d_l \in \{0,1\} \qquad \forall l \in L \tag{3.27}$$

A função objetivo, equação (3.1), visa minimizar o custo da rota, sob o ponto de vista do caixeiro, e as penalidades referentes aos passageiros transportados para destinos alternativos. O caixeiro divide o custo da viagem com os passageiros. As restrições (3.2) e (3.3) garantem que o caixeiro visite cada cidade de um subconjunto de vértices $N' \subseteq N$ exatamente uma vez. As restrições (3.4) e (3.5) garantem que o *tour* comece e termine no vértice $s \in N$, $s = 1$. A restrição (3.6) garante a continuidade do *tour*, implicando que, se o caixeiro chegar a um vértice, ele deverá seguir em diante. As restrições (3.7) e (3.8) proíbem *sub-tours* (MILLER; TUCKER; ZEMLIN, 1960), em que as variáveis de nível u_i denotam a ordem de visita dos vértices no *tour* do caixeiro. A restrição (3.8) garante que se o vértice j for visitado imediatamente após o vértice i, o posição de i na ordem de

visitas da viagem corrente será uma unidade menor que a posição de j. Quando $x_{ij} = 0$, a desigualdade da restrição (3.8) é trivialmente satisfeita. A restrição (3.9) garante que o caixeiro colete a cota mínima K. A restrição (3.10) garante que a duração máxima de uma viagem exigida por cada passageiro seja satisfeita. A restrição (3.11) garante que a capacidade do veículo não seja excedida. A restrição (3.12) garante que nenhum passageiro deixe o veículo em mais de uma localidade. A restrição (3.13) garante que os limites orçamentários de todos os passageiros não sejam excedidos. A restrição (3.14) garante que os pontos de embarque e desembarque de cada passageiro sejam diferentes. A restrição (3.15) garante que o passageiro $l \in L$ percorra o arco (i, j) somente se l tiver sido embarcado. A restrição (3.16) garante que, se o passageiro l chegar a $j \in N \backslash \{org(l)\}$ e não continuar em outra cidade, l será desembarcado em j. Essa restrição também garante que, se l chegar a j e for para outra cidade, l não deixe o veículo em j. A restrição (3.17) garante que o passageiro l desembarque do veículo na cidade j se e somente se l tiver sido embarcado. A restrição (3.18) garante que um passageiro embarcado no vértice i deve atravessar pelo menos um arco (i, j). Se o passageiro l desembarcar do veículo no vértice i, a restrição (3.19) garante que l não continue no veículo após i. A restrição (3.20) garante que o passageiro l, embarcado em $j = org(l)$, não atravesse nenhum arco (i, j). A restrição (3.21) garante que se o passageiro l chegar ao vértice s, l desembarque do veículo em s, pois s é o ponto inicial e final do caixeiro. A restrição (3.22) garante que o passageiro l não atravesse nenhum arco $(dst(l), j)$, ou seja, l deixa o veículo no vértice $dst(l)$ se l está no veículo em algum arco $(i, dst(l))$. Finalmente, restrições (3.23) - (3.27) definem o escopo das variáveis binárias.

3.3 Representação da solução do problema

Uma solução para o PCV-CPTIT é definida como $S = (N', Q', L', H')$, onde N' é uma lista de vértices que representam um ciclo Ψ que satisfaz a restrição de cota mínima K; Q' é uma lista binária na qual o i-ésimo elemento é 1 se o caixeiro coletar o bônus da cidade i, $i \in N'$; L' é uma lista binária na qual o l-ésimo elemento é 1 se o caixeiro aceitar a l-ésima solicitação de viagem; e, H' é uma lista de números inteiros nos quais o l-ésimo elemento é o índice da cidade onde o l-ésimo passageiro desembarcou do carro. Se $L'[l] = 0$, então $H'[l] = 0$.

O custo da solução S, indicado por $S.cost$, é calculado com a equação (3.28).

$$S.cost = \sum_{(i,j)\in N'} \frac{c_{ij}}{1 + \sum_{l\in L'} v_{ij}^{l}} + \sum_{l\in L'} h_{lH_l'} \tag{3.28}$$

3.4 Análise do problema

A tomada de decisão inerente ao processo de otimização do *PCV-CPTIT* associa a ordem de cidades visitadas com as solicitações de viagem a serem atendidas. No sentido de avaliar a influência de cada um destes fatores na complexidade do problema, foi realizada uma breve análise acerca do espaço de soluções.

Para o *PCV-CPTIT*, o tamanho do espaço de soluções é afetado por dois fatores preponderantes: número de cidades ($|N|$) e número de solicitações de viagem ($|L|$). Há então dois subproblemas. Desta forma, se faz necessário analisar as características de cada um destes subproblemas separadamente para mensurar o tamanho do espaço de buscas do problema proposto.

Visto que uma solução viável para o problema compreende um ciclo hamiltoniano e que este sempre inicia na cidade 1, para avaliar a quantidade de soluções possíveis (E_{cid}) basta utilizar a equação (3.29).

$$E_{cid} = (|N| - 1)! \tag{3.29}$$

Considerando apenas o número solicitações de viagem (E_{pas}), para se mensurar o número de soluções viáveis, deve-se ponderar que cada $l \in L$ possuí apenas duas possibilidades: atendida ou o oposto. Assim sendo, a equação (3.30) estima a ordem de grandeza do espaço de busca deste subproblema.

$$E_{pas} = 2^{|L|} \tag{3.30}$$

Assim, para se obter o tamanho total do espaço de soluções (E_{sol}), é necessário multiplicar E_{cid} e E_{pas}, conforme a equação (3.31).

$$E_{sol} = E_{cid} \times E_{pas} \tag{3.31}$$

A menor instância do problema possuí 10 cidades e 30 solicitações de viagem. Desta forma, toma-se $|N| = 10$ e $|L| = 30$. Logo, o cálculo será:

$$E_{sol} = 362880 \times 1073741824 = 3,8963943e14$$

Observa-se que de $|L|$ advém o maior impacto no espaço de soluções. Fica evidente também que até mesmo o menor caso de teste do problema apresenta um espaço de busca demasiado extenso.

4 Abordagens Heurísticas

Neste capítulo, apresentamos três métodos usados em nossos experimentos. A seção 4.1 apresenta a Heurística de Carregamento (HC). A HC define quais requisições de viagens serão atendidas pelo motorista. A seção 4.2 apresenta a Heurística de Busca Local com Múltiplos-operadores (HBL-Mo) usada para refinar as soluções encontradas pelos métodos propostos. Finalmente, a seção 4.3 apresenta o conceito da Heurística *Naive* (HN) que fornece resultados iniciais para as instâncias dos experimentos computacionais.

4.1 Heurística de Carregamento

A HC é um método projetado para decidir quais solicitações de viagens serão aceitas pelo motorista enquanto este viaja em seu veículo para satisfazer a restrição de quota miníma. A descrição formal desta heurística é a seguinte: seja S uma solução e Ψ um ciclo em S. Ψ consiste em uma lista de vértices, indicada por N', e uma lista de arcos, indicada por A'. A HC define quais solicitações de viagens serão atendidas dado o ciclo Ψ. O algoritmo 1 apresenta o pseudo-código da HC. Nesta heurística, existem cinco parâmetros de entrada:

- S: uma solução inicial sem quaisquer solicitações de viagem atribuídas, ou seja, $L'[i] = 0$, $i = 1, ..., |L|$;
- L: o conjunto de solicitações de passeio;
- R: a capacidade do veículo;
- b: a lista da duração máxima de uma jornada para cada pessoa;
- w: a lista do valor máximo que cada pessoa concorda em pagar por uma viagem;
- h: a lista de penalidades para transportar cada passageiro a um destino diferente do exigido.

Como, inicialmente, não há qualquer solicitação de viagem atribuída a Ψ, o custo de S é $S.cost = \sum_{(i,j)\in A'} c_{ij}$. A saída da HC é S', uma solução com solicitações de viagens atribuídas ao seu ciclo Ψ.

Algoritmo 1 Heurística de Carregamento

Parâmetros de entrada: S, L, R, b, w, h
1. $S' \leftarrow$ copiar(S)
2. $E \leftarrow L$ classificada em ordem decrescente de w_l
3. Para cada $j \in E$
4. Se $org(L[E[j]]) \in S.N'$
5. $\alpha \leftarrow compute\_drop(S.N', h, org(L[E[j]]), b_{L[E[j]]}, w_{L[E[j]]}, R)$
6. Atribuí $L[E[j]]$ para S tal que $L[E[j]]$ desembarque em α
7. $S.cost \leftarrow avaliar(S)$
8. Se $S.cost < S'.cost$
9. $S' \leftarrow S$
10. Senão
11. Remova $L[E[j]]$ de S
12. Retorna S'

A primeira instrução da HC é classificar a lista de requisições de viagem L em ordem decrescente dos orçamentos w_l associados as requisições de viagens dispostas na lista $L[E[j]]$. Desta forma, são avaliadas por primeiro aquelas requisições de viagem que tem o maior potencial de contribuição para o rateio das despesas de deslocamento inerentes a viagem do motorista. A lista ordenada é definida como E (linha 2). O *loop* principal (linhas 3-11) verifica a possibilidade de atender às solicitações de viagem. Se o vendedor visitar a cidade de origem $org(L[E[j]])$ de uma requisição de viagem $E[j]$, o algoritmo calcula o ponto de desembarque α para $E[j]$ junto ao procedimento *compute_drop*() (etapa 5). O procedimento *compute_drop*() retorna um vértice de $S.N'$, ou seja, uma cidade visitada pelo motorista. Seja v o vértice retornado por esse procedimento, v deve ser posterior a $org(L[E[j]])$ na ordem de visita da rota do motorista e a penalidade $h_{L[E[j]]v}$ por desembarcar o passageiro $E[j]$ no vértice v é mínima com relação aos outros vértices que compõem $S.N'$ e as restrições relacionadas ao orçamento máximo e tempo máxima da viagem ($b_{L[E[j]]}$, $w_{L[E[j]]}$) do passageiro $E[j]$ são satisfeitas. A capacidade R do veículo deve ser considerada neste processo e não pode ser excedida. O potencial passageiro $L[E[j]]$ é atribuído a S com $org(L[E[j]])$ e α como pontos de embarque e desembarque, respectivamente (linha 6). Após alocar um novo passageiro a rota, o custo da solução S é calculado pela equação (3.28) (linha 7). A solução S substitui S', se o custo do primeiro for menor que o do segundo (linhas 8 a 9). Caso contrário, o algoritmo remove a atribuição do potencial passageiro $E[j]$ (linha 11).

4.2 Heurística de Busca Local com Múltiplos-operadores

O algoritmo de busca local proposto neste estudo possuí diferentes operadores de vizinhança. Este algoritmo é denominado Heurística de Busca Local com Múltiplos-operadores. O conceito de múltiplos operadores foi projetado para contemplar: a coleta de bônus; e, o fato de que as soluções do PCV-CPTIT possuem rotas de tamanho variado. O cerne deste algoritmo é receber uma solução e definir um operador de vizinhança aleatoriamente com distribuição uniforme, promover um sequência de modificações na rota desta solução conforme o operador de vizinhança sorteado, e, com isso, testar novos esquemas de alocação de passageiros possíveis após a execução das modificações na rota da solução de entrada. O algoritmo 2 apresenta o pseudocódigo da HBL-Mo. A entrada, S, é uma solução do PCV-CPTIT. A saída é S', a melhor solução encontrada pelo algoritmo.

Neste algoritmo, depois que S é copiado para S' (linha 1), um operador de vizinhança é selecionado aleatoriamente e atribuído a θ (linha 2). Existem cinco estruturas de vizinhança: *swap*, *flip*, *push*, *pop-vertex* e *pop-quota*.

Seja $\Psi = (v_0, ..., v_i, ..., v_j, ..., v_0)$ o ciclo em S. O bônus de cada vértice de Ψ é marcado como coletado. A ideia é que S satisfaça a restrição de coleta de bônus K. Em cada iteração, um vértice de Ψ é avaliado. Seja $v_i \in \ Psi$ o vértice avaliado na i-ésima iteração. O operador de vizinhança de *swap* substitui v_i por v_j, $v_j \notin \Psi$. O vértice v_j é selecionado aleatoriamente com distribuição uniforme. A cota associada ao vértice v_j é marcada como coletada.

Seja $P = (v_i, v_{i+1}, ..., v_{k-1}, v_k)$ um caminho em Ψ e v_k determinado aleatoriamente com distribuição uniforme, o operador *flip* inverte a ordem dos vértices que compõem P produzindo $\Psi = (v_0, ..., v_k, v_{k-1}, ..., v_{i+1}, v_i, ..., v_0)$. Em outras palavras, pode-se dizer que o operador *flip* inverte a ordem de visita de um caminho P em Ψ entre os vértices v_i e v_k, $v_k \in \Psi$, $i < k$.

O operador *push* adiciona um vértice não visitado, escolhido aleatoriamente com distribuição uniforme, a Ψ. O vértice não visitado é inserido após um vértice $v_i \in \Psi$. O operador *pop-vertex* seleciona, aleatoriamente com distribuição uniforme, um vértice $v_i \in \Psi$ e o remove de Ψ. O operador *pop-quota* desmarca a coleta de de bônus no vértice v_i. Assim, o motorista não contabiliza o bônus associado a v_i no seu calculo para satisfazer a restrição de coleta de bônus K e nem o tempo para coletar o bônus de v_i. É possível que esta economia de tempo de coleta viabilize a alocação de mais passageiros em Ψ. Desta forma, com a inclusão de mais passageiros, o rateio das despesas de viagem aumenta e o

custo final de S diminuí. A Figura 4 ilustra a execução dos operadores da vizinhança da HBL-Mo.

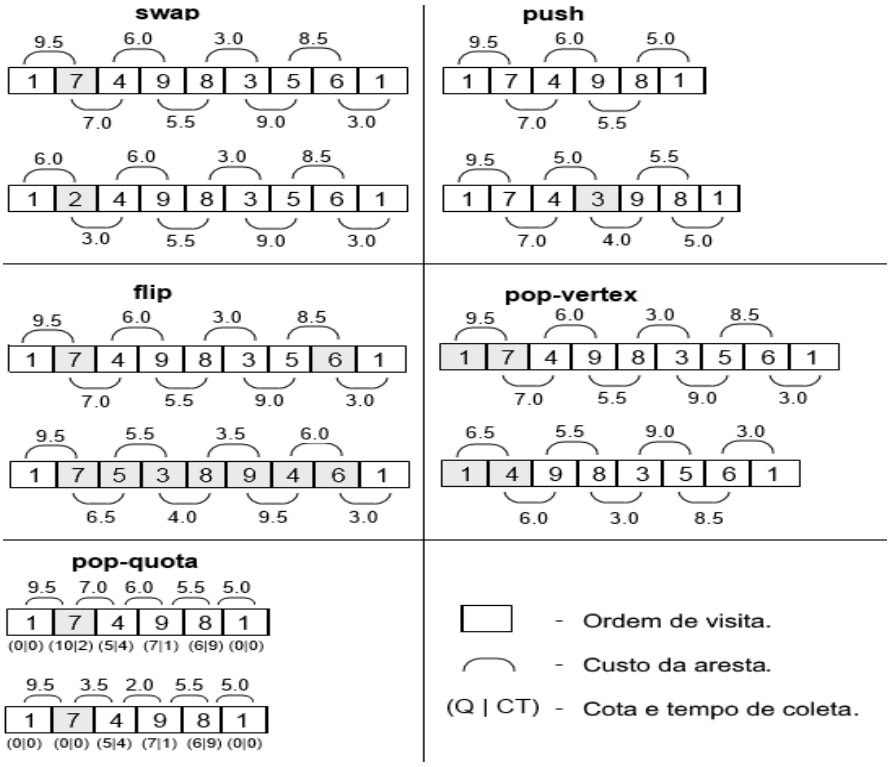

Figura 4: Demonstração gráfica da execução dos operadores da vizinhança da HBL-Mo.

Algoritmo 2 Heurística de Busca Local com Múltiplos-operadores

Parâmetros de entrada: S

1. $S' \leftarrow copy(S)$
2. $\theta \leftarrow$ defina aleatoriamente o operador
3. Para cada $i \in S'.N' - \{v_0\}$ faça
4. $\quad S'' \leftarrow$ modificação(S', i, θ)
5. $\quad$ Enquanto $S''.quota < K$
6. $\quad\quad S'' \leftarrow$ modificação$(S', i, push)$
7. $\quad S'' \leftarrow HC(S'', L)$
8. $\quad$ Se $S''.cost < S'.cost$
9. $\quad\quad S' \leftarrow S''$
10. Retorna S'

No *loop* principal (linhas 3 a 9), o algoritmo 2 segue promovendo modificações, com base no operador θ, que levem a geração de novas soluções. Os operadores *flip* e *push* não violam a restrição de coleta de bônus K. No entanto, os outros operadores podem violá-la. Se isto vier a ocorrer, o algoritmo executará o operador *push* (linha 6), de modo que um vértice $v_j \notin \Psi$ é escolhido aleatoriamente com distribuição uniforme e inserido na posição que resulta no menor incremento no custo de Ψ. A coleta de bônus de v_j é ativada afim de se incrementar o calculo do bônus total coletado em Ψ. Enquanto o bônus coletado em

Ψ for menor que o exigido pela restrição de coleta de bônus K, o operador *push* continua a ser executado. Finalmente, o algoritmo 2 retorna a melhor solução gerada (linha 10).

4.3 Heurística *Naive*

Dado que o PCV-CPTIT é um problema inédito, não há resultados previstos na literatura. Desta forma, implementamos uma heurística simples, a qual chamamos de *Naive*, para fornecer os resultados preliminares para o PCV-CPTIT. Nos experimentos computacionais, comparamos esses resultados com os produzidos pelas abordagens propostas neste estudo.

O algoritmo 3 apresenta o pseudo-código da HN. Este pseudo-código consiste em dois procedimentos: *roteamento*() e *carregamento*(). Existem três parâmetros de entrada: $INST$, uma instância do PCV-CPTIT, ξ e κ. Os dois últimos parâmetros estão relacionados aos procedimentos *roteamento*() e *carregamento*() e são explicados mais detalhadamente.

Algoritmo 3 Heurística *Naive*

Parâmetros de entrada: $INST, \xi, \kappa$
1. $S \leftarrow roteamento(INST, \xi)$
2. $S' \leftarrow carregamento(S, \kappa)$
3. Retorna S'

O procedimento *roteamento* gera um ciclo que satisfaz a restrição de cota mínima, ou seja, uma solução satisfatória para o PCV-C. O parâmetro ξ representa o tipo de abordagem usado dentro do procedimento: *exato* ou *heurístico*. Se $\xi = exato$, a instância é submetida a um solver de otimização para executar o modelo matemático do PCV-C apresentado em (AWERBUCH et al., 1998). Caso contrário, a instância será submetida a uma adaptação da meta-heurística GRASP (FEO; RESENDE, 1989).

A meta-heurística GRASP possui duas fases: construção e busca local. A primeira fase constrói uma solução viável. O procedimento executado durante a fase construtiva do GRASP pode ser descrito da seguinte forma: inicialize o conjunto C com todos os elementos $e \in E$, onde E é o conjunto de todos os elementos que podem compor uma solução válida para o problema de otimização em questão. Em seguida, avalie o custo incremental $c(e)$ de cada elemento $e \in C$. Por custo incremental $c(e)$, compreenda-se como o beneficio de se adicionar o elemento e a solução corrente. Feito o calculo dos custos incrementais para todo $e \in C$, a solução S de retorno deste procedimento será gerada, iterativamente, a partir de uma lista de elementos ordenados segundo uma função

gulosa. Durante este processo de ordenação, os elementos com melhor classificação formam a Lista Restrita de Candidatos (LRC). É da LRC de onde os elementos são selecionados aleatoriamente. A cada iteração, o tamanho da LRC é ajustado por um limite $c(e) \in [c^{min}, c^{min} + \alpha(c^{max} - c^{min})]$, em que $\alpha \in [0, 1]$ é um parâmetro, $c(e)$ é o custo incremental de e, c^{min} e c^{max} são os custos para que se adicione o melhor e o pior elemento de C à S, respectivamente. Depois que um elemento e selecionado é incorporado à S, C é atualizado e os custos incrementais são reavaliados. Se $C \neq \emptyset$, cessam as iterações e S é retornada. O algoritmo 4 apresenta o pseudocódigo deste procedimento construtivo.

Algoritmo 4 Procedimento Construtivo

Parâmetros de entrada: α
1. $S \leftarrow \emptyset$
2. Inicializar a lista de candidatos: $C \leftarrow E$
3. Avaliar o custo incremental $c(e)$ para todo $e \in C$
4. Enquanto $C \neq \emptyset$ faça
5. $\quad c^{min} \leftarrow min[c(e)|e \in C]$
6. $\quad c^{max} \leftarrow max[c(e)|e \in C]$
7. $\quad LRC \leftarrow \{e \in C \ \ |c(e) \leq c^{min} + \alpha(c^{max} - c^{min})\}$
8. $\quad$ Selecionar aleatoriamente um elemento e da LRC
9. $\quad S \leftarrow S \cup e$
10. $\quad$ Atualizar a lista de candidatos C
11. $\quad$ Reavaliar o custo incremental $c(e)$ para todo $e \in C$
12. Retornar S

No procedimento construtivo do algoritmo GRASP empregado na primeira fase da HN, a rota começa em s, onde s é o vértice de partida segundo a instância corrente do PCV-CPTIT. Em cada iteração, uma nova localidade é adicionada à rota. A LRC contém as localidades mais próximas da última adicionada. Assim como no procedimento clássico, o tamanho da LRC é ajustado por um limite $c(e) \in [c^{min}, c^{min} + \alpha(c^{max} - c^{min})]$, em que $\alpha \in [0, 1]$ é um parâmetro, $c(e)$ é o custo para visitar a localidade e, c^{min} e c^{max} são o custo para que se visite a localidade mais próxima e a mais distante, respectivamente. Uma localidade é escolhida aleatoriamente na LRC com distribuição uniforme. As cidades são adicionadas iterativamente à solução até atingir a cota mínima K. O algoritmo 5 apresenta o pseudocódigo desta adaptação de procedimento construtivo do GRASP.

È comum que uma heurística seja incorporada ao GRASP para desempenhar a fase de busca de local. Nesta adaptação, a segunda fase emprega heurística *2-opt* (CROES, 1958) à rota criada na primeira fase. Esta heurística consiste basicamente em avaliar, a cada iteração, se reconexões entre arestas dispostas em um ciclo de entrada Ψ podem resultar na diminuição do custo total deste ciclo. O algoritmo 6 apresenta o pseudo-código desta heurística.

Algoritmo 5 Procedimento Construtivo *Naive*

Parâmetros de entrada: α, K

1. $S \leftarrow \{s\}$
2. Inicializar a lista de candidatos: $C \leftarrow E$
3. Inicializar contador de bônus coletado: $\gamma \leftarrow 0$
4. Avaliar o custo incremental $c(e)$ para todo $e \in C$
5. Enquanto $\gamma < K$ faça
6. $\quad c^{min} \leftarrow min[c(e)|e \in C]$
7. $\quad c^{max} \leftarrow max[c(e)|e \in C]$
8. $\quad LRC \leftarrow \{e \in C \ \ |c(e) \leq c^{min} + \alpha(c^{max} - c^{min})\}$
9. $\quad$ Selecionar aleatoriamente um elemento e da LRC
10. $\quad S \leftarrow S \cup e$
11. $\quad$ Atualizar a lista de candidatos C
12. $\quad$ Incrementar γ com o bônus coletado em e
13. $\quad$ Reavaliar o custo incremental $c(e)$ para todo $e \in C$
14. Retornar S

Algoritmo 6 Heurística *2-opt*

Parâmetros de entrada: $\Psi = (v_i, v_{i+1}, ..., v_{n-1}, v_n)$

1. Para $i = 1$ até $n - 1$
2. $\quad$ Para $j = i + 1$ até nx
3. $\quad\quad$ Se $j < n$
4. $\quad\quad\quad$ Se $(c(v_i, v_{i+1}) + c(v_j, v_{i+1}) < c(v_j, v_{i+1}) + c(v_j, v_{j+1}))$
5. $\quad\quad\quad\quad$ Permuta v_i por v_j em Ψ
6. $\quad\quad\quad$ Senão se $(c(v_i, v_1) + c(v_n, v_{i+1}) < c(v_i, v_{i+1}) + c(v_n, v_1))$
7. $\quad\quad\quad\quad$ Permuta v_i por v_n em Ψ
8. Retorna Ψ

A Figura 5 esboça um movimento da heurística *2-Opt* na qual arestas a e b seriam permutadas pelas arestas c e d.

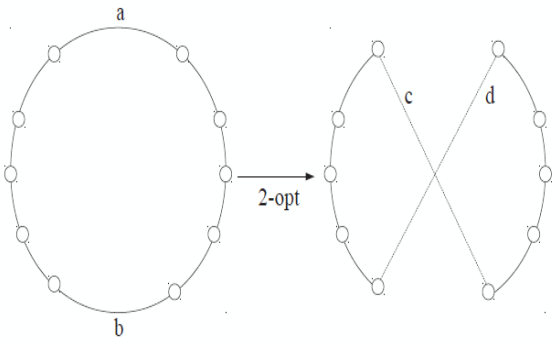

Figura 5: Substituição de arestas executada pela heurística *2-opt*.

O critério de parada para a implementação do GRASP proposto para ser incorporado a HN é um número máximo de iterações denotado por $maxIter$(FEO; RESENDE, 1989). O pseudocódigo desta adaptação é apresentado no algoritmo 7.

Algoritmo 7 GRASP *Naive*

Parâmetros de entrada: $maxIter$, α, K
1. $S \leftarrow \emptyset$
2. Para $i = 1$ até $maxIter$
3. $\quad S' \leftarrow$ Procedimento Construtivo $Naive(\alpha, K)$
4. $\quad S^h \leftarrow$ *2-opt*(S')
5. $\quad$ Se $(S^h.cost < S.cost)$ então
6. $\quad\quad S \leftarrow S^h$
7. Retorne S

O procedimento *carregamento* da HN recebe o parâmetro κ e a solução S criada pelo procedimento *roteamento*. Seguindo a mesma lógica do parâmetro ξ, κ representa o tipo de método usado para atribuir passageiros à rota: *exato* ou *heurístico*. Se $\kappa = exato$, o modelo apresentado em (3.1) - (3.22) será submetido a um solver de otimização com as variáveis x_{ij} prefixadas de acordo com S, ou seja, a rota criada pelo o procedimento *roteamento*. Caso contrário, a instância é submetida à heurística HC.

5 Abordagens Meta-heurísticas

Neste capítulo são apresentadas as abordagens meta-heurísticas propostas no estudo. Na seção 5.1, é descrita a adaptação da meta-heurística ILS. Uma adaptação da meta-heurística GRASP é apresentada na seção 5.2. Na seção 5.3, algoritmos evolucionários implementados para o PCV-CPTIT são é apresentados. Os algoritmos da família ACO aplicados ao PCV-CPTIT são apresentados na seção 5.4.

5.1 *Iterated Local Search*

Nesta seção, é apresentada a meta-heurística ILS adaptada ao PCV-CPTIT. Na sub-seção 5.1.1, é feita uma descrição deste algoritmo. Discorre-se sobre a adaptação da ILS ao PCV-CPTIT na sub-seção 5.1.2.

5.1.1 Descrição

O funcionamento da ILS pode ser descrito da seguinte forma: Seja Z o conjunto de soluções viáveis de um problema de otimização. Dada uma solução de partida $S^0 \in Z$, aplica-se uma heurística de busca local. A respectiva heurística de busca local produzirá $S^* \in Z$. A solução S^* pode ser interpretada como o ótimo local corrente mais próximo de S^0. Esta meta-heurística repete este processo durante m iterações, porém, uma solução $S^{'}$, obtida através da aplicação de uma estrategia de modificação em S^*, é submetida ao algoritmo de busca local em vez de S^0. O procedimento de busca local retornará uma solução $S^{''}$. Na iteração corrente, se $S^{''}$ é melhor avaliado do que S^*, então atribui-se $S^{''}$ a S^*. A solução retornada pela ILS para um problema de otimização é então o melhor S^* produzido durante m iterações.

Sendo assim, o projeto desta meta-heurística contempla basicamente uma estrategia de modificação e um método de busca local. O algoritmo de busca local a ser utilizado nesta meta-heurística deve ser responsável pela intensificação e melhoria da qualidade das

soluções encontradas. A estrategia de modificação para gerar $S^{'}$ a partir de S^{*} desempenha o papel da diversificação da busca em Z. O algoritmo 8 apresenta o pseudocódigo da ILS para um problema de minimização.

Algoritmo 8 ILS

Parâmetros de entrada: m, $INST$
1. $S^0 \leftarrow GerarSolucaoDePartida(INST)$
2. $S^* \leftarrow$ BuscaLocal(S^0)
3. Para $i = 1$ até m
4. $\quad S^{'} \leftarrow$ modificar S^*
5. $\quad S^{''} \leftarrow$ BuscaLocal($S^{'}$)
6. $\quad$ Se ($S^{''}.cost < S^*.cost$) então
7. $\quad\quad S^* \leftarrow S^{''}$
9. Retorne S^*

O principal aspecto de um operador de modificação é a sua intensidade, isto é, a quantidade de alterações realizadas em S^{*} para se obter $S^{'}$. O mecanismo de intensidade pode ser variável ou constante, todavia, empiricamente ficou evidenciado que a estratégia variável apresenta melhores resultados (BLUM, 2012). Por mecanismo de modificação com intensidade constante entende-se como aquele que, a cada iteração, utiliza sempre a mesma quantidade de operações para alterar a estrutura de uma solução. Já um mecanismo de modificação com intensidade variável tende a variar a quantidade de operações realizadas para modificar a estrutura de uma solução durante as iterações desta meta-heurística. O ideal é que o mecanismo de modificação tenha uma intensidade balanceada, i. e., seja forte o suficiente para que a busca possa explorar diferentes regiões de Z, e ao mesmo tempo, seja fraco a ponto de guardar características do ótimo local corrente.

Para favorecer o viés de intensificação da busca desta meta-heurística, é indicado que apenas a melhor solução encontrada até então seja modificada e submetida ao algoritmo de busca local (BRANDAO, 2018). Desta forma, preserva-se as melhores decisões realizadas entre as iterações.

Segundo (STÜTZLE; RUIZ, 2018), os primeiros estudos a abordarem o conceito explorado no projeto da meta-heurística ILS foram (BAUM, 1986a, 1986b; JOHNSON, 1990). Segundo (LOURENÇO; MARTIN; STÜTZLE, 2019), as variações mais conhecidas desta meta-heurística são: *Depth Search Heuristic* (BALAS; VAZACOPOULOS, 1998), cadeias de Markov de grande porte (HONG; KAHNG; MOON, 1997), Lin-Kernighan iterada (APPLEGATE; COOK; ROHE, 2003) e *Chained Local Search* (MARTIN; OTTO, 1999).

A meta-heurística ILS é paralelizada e aplicada ao PCV em (ZHOU; HE; QIU, 2016). Neste estudo, técnicas de paralelismo são empregadas para se contornar o consumo demasi-

ado de tempo desta meta-heurística decorrente da resolução de instâncias de grande porte do problema em questão. Em (CACCHIANI et al., 2018), uma variação do ILS é aplicado ao *Pollution Traveling Salesman Problem*. Neste estudo, uma variação do PCV que visa minimizar o consumo de combustível juntamente com os custos de deslocamento do motorista é otimizada pela ILS. O operador de modificação denominado *k-exchange* é citado por (LOURENÇO; MARTIN; STÜTZLE, 2019) como um bom mecanismo de diversificação para o ILS. O funcionamento deste operador é simples: seja $\Psi = (v_0, ..., v_i, ..., v_j, ..., v_0)$ o ciclo em S^*, o operador *k-exchange* seleciona k pares de vértices e os inverte na ordem de visita em Ψ. Desta forma, se um deste pares é composto pelos vértice v_i e v_j, teremos $\Psi = (v_0, ..., v_j, ..., v_i, ..., v_0)$. Segundo (LOURENÇO; MARTIN; STÜTZLE, 2019), o valor de k deve ser uma parâmetro de entrada para o ILS. Estudos gerais da meta-heurística ILS são apresentados em (STÜTZLE; RUIZ, 2018; MCMENEMY; VEERAPEN; OCHOA, 2018).

5.1.2 *Iterated Local Search* proposto para o PCV-CPTIT

A essência desta meta-heurística é simples: construir soluções a partir da iteração de uma heurística voltada especificamente para o problema em questão e dispor de um mecanismo de diversificação de busca. O algoritmo 9 apresenta o pseudocódigo do ILS adaptado ao PCV-CPTIT.

Algoritmo 9 ILS Adaptado

Parâmetros de entrada: $maxIter$, $INST$
1. $S^0 \leftarrow HN(INST, \textit{operador heurístico}, \textit{operador heurístico})$
2. $S^* \leftarrow$ HBL-Mo(S)
3. Para $i = 1$ até $maxIter$
4. $\quad S^{'} \leftarrow$ modificar S^* com operador *k-exchange*
5. $\quad S^{''} \leftarrow$ HBL-Mo($S^{'}$)
6. $\quad$ Se ($S^{''}.cost < S^*.cost$) então
7. $\quad\quad S^* \leftarrow S^{''}$
9. Retorne S^*

Na linha 1 do algoritmo 9, aplica-se a heurística *naive* para obtenção de uma solução inicial S^0. O pseudocódigo da heurística *naive* é descrito em 3. Na linha 2, aplica-se a heurística HBL-Mo para se refinar a solução inicial S^0. Após a execução das instruções da linha 2, obtém-se S^*. Depois a ILS modifica S^* com o operador *k-exchange*. Diferente do operador *k-exchange* descrito (LOURENÇO; MARTIN; STÜTZLE, 2019), nesta adaptação, o valor de k é sorteado com distribuição uniforme no intervalo $[2, n/2]$ para variar a intensidade da modificação, com n denotando o tamanho do ciclo Ψ de S^*. A ideia é balancear a intensidade do mecanismo de modificação para favorecer o viés de diversifica-

ção desta meta-heurística (BLUM, 2012). Após a modificar S^* com o operador *k-exchange*, esta solução é submetida ao HBL-Mo. Uma nova solução S'' é produzida. Se a solução S'' é melhor avaliada que S^*, então atribui-se S'' para S^* e esta passa a ser a melhor solução encontrada até então. Este processo é repetido até que o número máximo de iterações $maxIter$ seja alcançado. A melhor solução é retornada ao fim do algoritmo.

5.2 *Greddy Randomized Adaptative Search Procedure*

Nesta seção, é apresentada a meta-heurística GRASP adaptada ao PCV-CPTIT. Na sub-seção 5.2.1, é feito uma descrição desta meta-heurística. Discorre-se sobre a adaptação desta meta-heurística ao PCV-CPTIT na sub-seção 5.2.2.

5.2.1 Descrição

O GRASP é uma meta-heurística proposta para encontrar soluções aproximadas para problemas de otimização combinatória. Seu projeto original se divide em duas fases: construção; busca local. Foi introduzida por Feo e Resende em um estudo no qual abordavam uma heurística probabilística para o Problema de Cobertura de Conjuntos(FEO; RESENDE, 1989). O algoritmo 10 apresenta o pseudocódigo clássico desta meta-heurística.

Algoritmo 10 GRASP

Parâmetros de entrada: $maxIter$,α
1. $S \leftarrow \emptyset$
2. Para $i = 1$ até $maxIter$
3. $\quad S' \leftarrow$ Procedimento Construtivo(α)
4. $\quad S'' \leftarrow$ BuscaLocal(S')
5. $\quad$ Se ($f(S'') < f(S)$) então
6. $\quad\quad S \leftarrow S''$
7. Retorne S

Na fase de construção, uma solução de partida $S' \in Z$ é construída para a fase de busca local, onde Z é o conjunto de soluções viáveis do problema de otimização em questão. Na fase de busca local, um algoritmo de busca local é empregado para vasculhar junto a vizinhança $N(S')$ de S' por uma solução $S'' \in N(S')$ tal que $f(S'')$ seja melhor que $f(S')$, onde f é uma função de avaliação da qualidade de uma solução. Na iteração corrente, se a qualidade da melhor solução encontrada até então S é inferior a de S'', então S'' passa a ser a melhor encontrada por esta meta-heurística. É válido ressaltar que o procedimento da fase construtiva já foi previamente descrito na seção 4.3.

Segundo (RESENDE; RIBEIRO, 2019), a utilização de uma estrategia puramente gulosa na fase construtiva diminui o poder de diversificação do GRASP. Caso seja empregado uma estrategia puramente aleatória na fase construtiva, será obtido um alto nível de diversificação na geração de soluções, porém a qualidade média destas soluções tende a ser inferior se comparada com a de soluções produzidas por um estrategia gulosa e o resultado final do processo de otimização do GRASP é negativamente afetado. Neste sentido, é comum observar que em várias estudos a respeito do GRASP dispostos na literatura, os autores empregam uma LRC para se balancear diversificação e a qualidade média das soluções produzidas ao longo do processo de otimização desta meta-heurística. É digno de nota citar que o funcionamento da LRC foi previamente descrito na seção 4.3.

Uma forma de melhorar a eficiência do GRASP é por meio da implementação de mecanismos de memória adaptativa para se evitar trabalho redundante (RESENDE; RIBEIRO, 2019). Em (RESENDE; RIBEIRO, 2005), é descrito um mecanismo para armazenar em uma tabela de *hash* todas as soluções construídas e usadas como soluções de partida para a fase de busca local. Sempre que uma nova solução for construída, ela será usada apenas como uma solução de partida na fase de busca local se não estiver presente na respectiva tabela *hash*. Estrategias de filtragem de soluções a serem submetidas a fase de busca local evitam a redundância de processamento e também que soluções de baixa qualidade sejam submetidas a esta fase (RESENDE; RIBEIRO, 2005).

Em (PRAIS; RIBEIRO, 2000), é proposto o GRASP Reativo. Nesta versão do GRASP, um mecanismo de modificação do parâmetro α é introduzido para ajustar as probabilidades de seleção de elementos que serão inseridos na LRC à medida que as iterações prosseguem, favorecendo assim valores de α que levaram a melhores soluções nas iterações anteriores.

Em (RESENDE; RIBEIRO, 2003), é descrita uma estratégia para o uso de memória de longo prazo que consiste em combinar, a cada iteração, a solução produzida pela fase de busca de local com uma solução elite selecionada aleatoriamente a partir de uma lista de soluções deste tipo. Neste estudo, a combinação da solução corrente com uma solução de elite aleatoriamente selecionada é feita pelo procedimento conhecido como *Path-Relinking* (PR) .

O procedimento PR busca balancear o poder intensificação e diversificação quando aplicado em uma meta-heurística. Considerando duas soluções, denominadas solução de origem e solução de destino, o objetivo do PR é propor melhores soluções por meio da identificação de um caminho da solução de origem até a solução destino. Este caminho é

definido por meio de movimentos que empregam atributos da solução destino na solução de origem. As melhores soluções encontradas (ou soluções de elite) junto ao processo de busca são armazenadas em uma lista com dimensão pré-fixada denominada lista de elite. Em um cenário em que a lista de elite está vazia, a melhor solução encontrada até então será inserida nela. No cenário em que a lista de elite não está vazia, uma nova solução só sera inserida se duas restrições forem satisfeitas: a nova solução é incluída na lista se ela for melhor do que a melhor solução da lista; ou se a nova solução for avaliada como melhor do que a pior solução da lista, mas que satisfaça a um critério de distinção para com as outras soluções presentes na lista de elite. A ideia da segunda restrição é de evitar que se tenha uma lista de elite com soluções semelhantes e, desta forma, favorecer o poder de diversificação. No cenário em que a lista de elite está cheia e uma das respectivas restrições é satisfeita, a pior solução da lista deve ser excluída para que a nova solução seja inserida. O algoritmo 11 apresenta o pseudocódigo do procedimento PR.

Algoritmo 11 *Path-Relinking*

Parâmetros de entrada: S_o,S_d, f
1. Computar a diferença simétrica $\Delta(S_o, S_d)$
2. $S^* \leftarrow min\{f(S_o), f(S_d)\}$
3. $S \leftarrow S_o$
4. Enquanto $\Delta(S, S_d) \neq \emptyset$ faça
5. $\quad k^* \leftarrow minf(S \oplus k) : k \in \Delta(S, S_d)$
6. $\quad \Delta(S \oplus k^*, S_d) \leftarrow \Delta(S, S_d) \backslash \{k^*\}$
7. $\quad S \leftarrow S \oplus k^*$
8. $\quad$ Se $f(S) < f(S^*)$
9. $\quad\quad S^* \leftarrow S$
10. Retorne S^*

O algoritmo 11 recebe como entrada a solução de origem S_o, a solução destino S_d e uma função de avaliação de custo f. O procedimento inicia computando a diferença simétrica $\Delta(S_o, S_d)$, i. e., gera-se um arranjo Δ que contenha os elementos de S_o que sejam distintos daqueles presentes em S_d. Em cada iteração, este procedimento avalia todos os movimentos $k \in \Delta(S, S_d)$ da solução corrente S e seleciona aquele que representar o melhor beneficio conforme $f(S)$, i. e., é selecionado aquele movimento que minimiza $f(S \oplus k)$, onde $S \oplus k$ é representa a solução gerada após a aplicação do movimento k na solução S. O melhor movimento k^* é então executado, produzindo a solução $S \oplus k^*$. O conjunto de movimentos disponíveis é atualizado de tal forma que K^* não poderá mais ser realizado. Em seguida, é avaliado se $S \leftarrow S \oplus k^*$, realizando a operação $S^* \leftarrow S$ caso esta condição seja verdadeira. As iterações encerram quando não há mais movimentos $k \in \Delta(S, S_d)$ a serem testados, i. e., $\Delta(S, S_d) = \emptyset$. Ao final do procedimento, a melhor solução gerada S^* é retornada.

O procedimento PR pode ser aplicado no GRASP como uma estratégia de pós-otimização. Neste caso, aplica-se o PR em todos os pares de soluções de uma lista de soluções de elite construída durante o processo de otimização desta meta-heurística. Outra forma de se aplicar o procedimento PR no GRASP é como uma estratégia de intensificação em toda solução obtida após a fase de busca local. Segundo (ROSSETI, 2003), a aplicação do PR como uma estratégia de intensificação a cada ótimo local é mais eficaz do que se empregado como uma estratégia de pós-otimização. Como é exposto no algoritmo 12, no GRASP com PR como estratégia de intensificação, cada iteração passa a ter três etapas principais: construção; busca local; e, *Path-Relinking*.

Algoritmo 12 GRASP com *Path-Relinking*

Parâmetros de entrada: $maxIter$,α
1. $S.cost \leftarrow \infty$
2. Para $i = 1$ até $maxIter$
3. $\quad S^{'} \leftarrow$ Construção(α)
4. $\quad S^{''} \leftarrow$ BuscaLocal($S^{'}$)
5. $\quad S^{k} \leftarrow$ *Path-Relinking*($S^{''}$)
6. $\quad$ Se ($f(S^{k}) < f(S)$) então
7. $\quad\quad S \leftarrow S^{k}$
8. Retorne S

O primeiro livro a tratar exclusivamente do GRASP foi publicado em 2016 por Resende e Ribeiro (RESENDE; RIBEIRO, 2016). Em (RESENDE; RIBEIRO, 2005), é descrito que fase construtiva do GRASP pode produzir uma solução inviável para o problema que está sendo abordado. Neste estudo, alguns procedimentos de reparo para garantir a viabilidade das soluções são descritos. Uma revisão recente desta meta-heurística é apresentada em (RESENDE; RIBEIRO, 2019).

5.2.2 GRASP Proposto para o PCV-CPTIT

A adaptação do GRASP descrito nesta seção tem propósito de otimizar o PCV-CPTIT de uma forma geral. É diferente do GRASP descrito na seção 4.3 o qual desconsidera as solicitações de viagens e limita-se apenas a obter uma rota para satisfazer a restrição de cota mínima pré-definida no escopo do problema. Nesta adaptação, aplica-se o procedimento PR como estrategia de intensificação.

Apesar de ter um proposito mais abrangente do que a adaptação apresentada na seção 4.3, há semelhanças no projeto da fase construtiva de ambas as adaptações. Na primeira fase desta adaptação, a rota começa em s. A cada iteração, uma localidade é adicionada à rota. Utiliza-se uma LRC para especificar as localidades mais próximas da última adicio-

nada a rota em construção. O método de ajuste do comprimento da LRC e o mecanismo de inserção de elementos nesta lista seguem a mesma lógica daqueles descritos na seção 4.3. Nesta versão, as localidades também são adicionadas iterativamente à solução até atingir a cota mínima K. Ao final da primeira fase, aplica-se a heurística HC a solução construída. O algoritmo 13 apresenta o pseudocódigo deste procedimento construtivo. A segunda fase aplica a heurística HBL-Mo na solução previamente construída. O procedimento de *Path-Relinking* implementado nesta adaptação tem como único diferencial para o algoritmo 11 a aplicação da heurística HC logo após a testagem de cada movimento. O critério de parada desta adaptação do GRASP é um limite para o número de iterações denotado por $maxIter$(FEO; RESENDE, 1989). O algoritmo 14 apresenta o pseudocódigo da adaptação proposta para o PCV-CPTIT.

Algoritmo 13 Procedimento Construtivo Adaptado

Parâmetros de entrada: α, K
1. $S \leftarrow \{s\}$
2. Inicializar a lista de candidatos: $C \leftarrow E$
3. Inicializar contador de bônus coletado: $\gamma \leftarrow 0$
4. Avaliar o custo incremental $c(e)$ para todo $e \in C$
5. Enquanto $\gamma < K$ faça
6. $\quad c^{min} \leftarrow min[c(e)|e \in C]$
7. $\quad c^{max} \leftarrow max[c(e)|e \in C]$
8. $\quad LRC \leftarrow \{e \in C \ \ |c(e) \leq c^{min} + \alpha(c^{max} - c^{min})\}$
9. $\quad$ Selecionar aleatoriamente um elemento e da LRC
10. $\quad S \leftarrow S \cup e$
11. $\quad$ Atualizar a lista de candidatos C
12. $\quad$ Incrementar γ com o bônus coletado em e
13. $\quad$ Reavaliar o custo incremental $c(e)$ para todo $e \in C$
14. Aplicar HC em S
15. Retornar S

Algoritmo 14 GRASP Adaptado

Parâmetros de entrada: $maxIter$,α, K, $INST$
1. $S \leftarrow HN(INST$, *operador heurístico*, *operador heurístico*$)$
2. Para $i = 1$ até $maxIter$
3. $\quad S^{'} \leftarrow$ Procedimento Construtivo Adaptado(α, K)
4. $\quad S^{''} \leftarrow$ HBL-Mo($S^{'}$)
5. $\quad S^{k} \leftarrow$ *Path-Relinking*($S^{''}$)
6. $\quad$ Se ($S^{k}.cost < S.cost$) então
7. $\quad\quad S \leftarrow S^{k}$
8. Retorne S

5.3 Algoritmos Evolucionários

Computação Evolucionária é o nome coletivo de uma classe de técnicas de resolução de problemas baseadas em princípios da evolução biológica, como seleção natural e herança genética. Na sub-seção 5.3.1, é feita uma descrição geral sobre algoritmos evolucionários. Discorre-se sobre as adaptações de meta-heurísticas evolucionárias ao PCV-CPTIT na sub-seção 5.3.2.

5.3.1 Descrição

Algoritmos evolucionários compreendem uma classe de heurísticas computacionais inspiradas na evolução *Darwiniana* e são amplamente usados para resolver problemas de otimização combinatória. A Figura 6 ilustra os conceitos comuns que fazem parte do projeto de um algoritmo evolucionário.

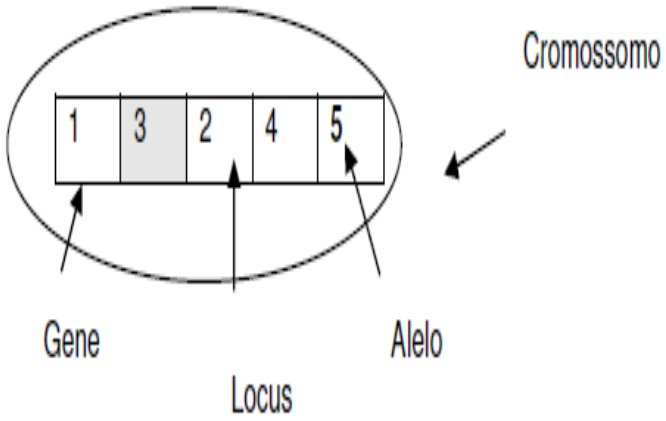

Figura 6: Ilustração de componentes comuns de uma solução segundo a classe de algoritmos evolucionários.

Estes conceitos são elencados como se segue:

- Indivíduo: conhecido também como cromossomo, é uma entidade da população constituída por uma estrutura de dados a qual abstraí uma solução para o problema em questão;
- Gene: especifica uma posição na ordem em que está organizada a estrutura de dados do individuo;
- Alelo: valoração de um gene;
- Locus: posição ocupada por um gene no indivíduo;

- Genótipo: arranjo de genes que representa um individuo;
- Fenótipo: interpretação do arranjo de genes;
- População: é o conjunto dos indivíduos;
- Geração: iteração completa de uma algoritmo evolucionário na qual se produz uma nova população.

Os algoritmos evolucionários são diferentes da maioria das outras meta-heurísticas porque exploram quatro ideias principais:

- Seleção: processo no qual uma estrategia de seleção define quais cromossomos serão selecionados para o procedimento de cruzamento. Seja $P(x)$ a probabilidade de um individuo x na população atual ser selecionado para reproduzir a população descendente, há na literatura várias estratégias de seleção, dos quais destacam-se:
  - Seleção aleatória: a seleção dos indivíduos é realizada com distribuição uniforme, desta forma, cada indivíduo tem a mesma chance de ser selecionado;
  - Seleção por classificação: na seleção por classificação, os indivíduos são classificados por sua aptidão e ordenados numa lista conforme esta classificação. A probabilidade do individuo $P(x)$ ser selecionado é inversamente proporcional à sua posição nesta lista ordenada. Isto implica que um indivíduo posicionado no início da lista tenha mais chances de ser selecionado do que aquele posicionado mais ao fim da lista. Este processo também é conhecido como seleção por *ranking*;
  - Seleção por roleta: também é conhecida como seleção proporcional à aptidão. Na seleção por roleta, cada indivíduo é selecionado de acordo com sua aptidão. Quanto mais adequados os cromossomos, maiores são as chances de serem selecionados e sobreviverem até a próxima geração. Em particular, a probabilidade $P(x)$ de selecionar um cromossomo ou individuo x para reprodução pode ser calculada conforme a equação:

    $$P(x) = \frac{f(x)}{\sum_{x=1}^{n} f(x)} \tag{5.1}$$

    onde $f(x)$ denota o valor de aptidão do indivíduo x;
  - Seleção por torneio: neste processo um subconjunto de indivíduos é selecionado aleatoriamente e somente o indivíduo mais apto é selecionado para reprodução;

  - Seleção por torneio proporcional: similar ao processo de seleção por torneio, contudo utiliza-se o método de roleta para obtenção do subconjunto de indivíduos;

- Cruzamento: procedimento que combina dois ou mais cromossomos, chamados de ascendentes, para produzir um ou mais cromossomos, denominados descendentes. Este procedimento é análogo a reprodução de genes em células. Neste processo genético, trechos das características de um indivíduo são trocados pelo trecho equivalente do outro. O resultado esperado desta operação é um indivíduo que potencialmente combine os melhores atributos dos indivíduos usados como base. Assim como no procedimento de seleção, há diversas estratégias de cruzamento. Destas estratégias, as mais conhecidas são:

  - Cruzamento de ponto único: um ponto de corte ι é definido. Em seguida, os genes posicionados na porção anterior a ι são copiados do primeiro cromossomo ascendente para o cromossomo descendente. O restante da carga genética do cromossomo descendente é copiado da porção posterior a ι do segundo cromossomo ascendente. A Figura 7 ilustra o funcionamento desta estratégia;

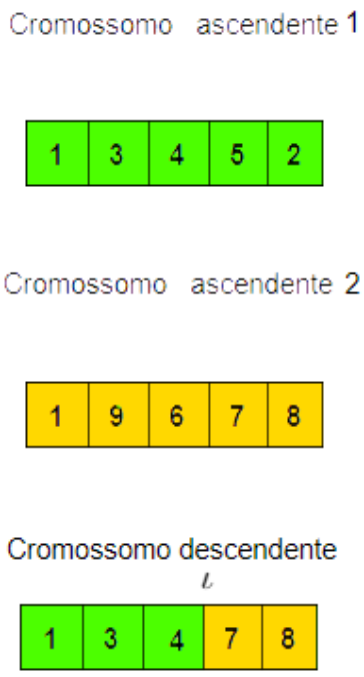

Figura 7: Ilustração do operador de cruzamento de ponto único.

  - Cruzamento de dois pontos: dois pontos de corte ι_0 e ι_d são definidos. Em seguida, os genes posicionados na porção anterior a ι_0 e posterior a ι_d são copiados do primeiro cromossomo ascendente para o cromossomo descendente.

O restante da carga genética do cromossomo descendente é copiado da porção situada entre ι_0 e ι_d do segundo cromossomo ascendente. A Figura 8 ilustra o funcionamento desta estratégia;

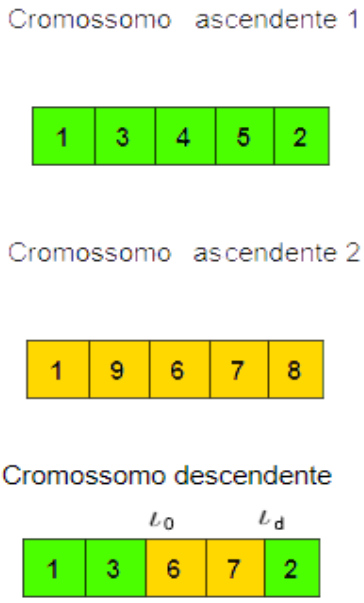

Figura 8: Ilustração do operador de cruzamento de dois pontos.

– Cruzamento uniforme: sorteia-se, com distribuição uniforme, os genes que passarão dos dois cromossomos ascendentes para o cromossomo descendente. A Figura 9 ilustra o funcionamento desta estratégia.

Figura 9: Ilustração do operador de cruzamento uniforme.

- Mutação: procedimento que visa modificar a informação genética contida em um cromossomo. Utiliza uma estratégia probabilística que define se um cromossomo vai ou não passar por mutação. A ideia deste procedimento é aumentar o poder de diversificação do algoritmo. Estratégias de mutação comuns são:

  - Mutação binária: também conhecida como mutação *flip bit*, esta estrategia consiste em inverter o valor dos genes do cromossomo selecionado, i. e., se um gene possuí valor 1, este passará a ser 0. A Figura 10 ilustra o funcionamento desta estratégia;

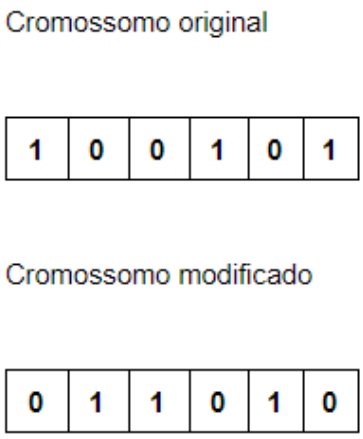

Figura 10: Ilustração do operador de mutação binária.

  - Mutação uniforme: o gene a ser modificado recebe um valor definido aleatoriamente com distribuição uniforme em um intervalo definido por limites máximos e mínimos. Em outras palavras, seja $E = (e_1, ..., e_k, ...e_n)$ um cromossomo, a mutação uniforme gera um novo cromossomo $E^{'}$ a partir de E, tal que $E^{'} = (e_1, ..., e_k^{'}, ..., x_n)$ onde $e_k^{'} = x$, sendo e_k um gene selecionado para mutação e x um valor aleatório selecionado dentro de um limite superior e outro inferior.

- Renovação: procedimento que visa substituir um arranjo de cromossomos da população corrente por novos cromossomos e, assim, produzir a próxima geração. Estratégias de renovação comuns são:

  - Renovação total: substituição de todos os cromossomos da população corrente;
  - Renovação elitista: são preservados apenas os melhores cromossomos entre as gerações;

– Renovação incremental: substituição apenas de uma pequena parcela de cromossomos ascendentes por novos cromossomos recém-gerados.

O ramo mais conhecido da computação evolucionária é aquele que trata do estudo dos Algoritmos Genéticos (AG). Segundo (YANG, 2014), este ramo de estudo foi proposto pelo professor *John Holland*, da Universidade de Michigan, nos Estados Unidos, e posteriormente, aprimorado por ele e seus alunos. *Holland* provavelmente foi o primeiro a empregar os conceitos de seleção, cruzamento, reprodução e mutação no estudo de sistemas artificiais e adaptativos (YANG, 2014). A popularidade dos AG's veio no fim da decada de 1980 junto da publicação do livro *Genetic algorithms in search, optimization, and machine learning* (GOLDBERG, 1989). A publicação desse livro marca o reconhecimento dos AG's como método de otimização. Antes deste marco, o foco dos estudos sobre AG's era sobre a aplicações destes algoritmos em modelos de aprendizado automático (WHITLEY, 2001). Aplicações recentes desta meta-heurística em problemas correlatos ao em estudo podem ser encontradas em (SILVA, 2017; CALHEIROS, 2017; WANG et al., 2016; BANIAMERIAN; BASHIRI; TAVAKKOLI-MOGHADDAM, 2019). A Figura 11 apresenta o fluxograma padrão de um AG.

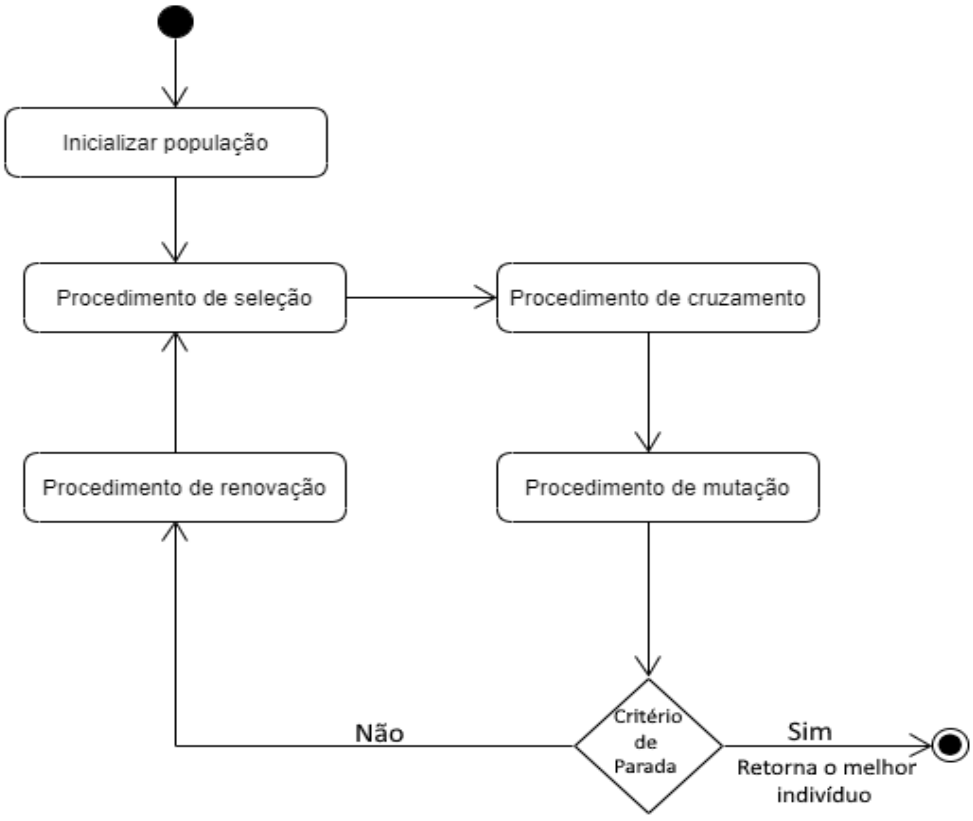

Figura 11: Fluxograma de um Algoritmo Genético.

Outro tópico bastante estudado dentro da computação evolucionária são os Algoritmos Meméticos. Apesar de ter projeto de implementação similar ao de um AG, um AM se baseia na evolução cultural introduzido por *Richard Dawkins* (DAWKINS, 2016). *Dawkins* classifica a cultura humana em unidades culturais chamadas de memes que podem ser replicadas, alteradas e combinadas na mente humana (NERI; COTTA, 2012). Os memes podem se extinguir ou se multiplicar dentro da cultura humana. Quanto melhor for a qualidade da informação cultural de um meme, maior suas chances de sobrevivência. A estrutura geral do AM segue os mesmos padrões do algoritmo genético com a incorporação de um procedimento de melhoramento após a criação de novos memes. Normalmente, a técnica de melhoramento de memes empregada consiste em um algoritmo de busca local. Aplicações recentes desta meta-heurística em problemas correlatos ao em estudo podem ser encontradas em (SILVA, 2017; CALHEIROS, 2017; KOCZY; FOLDESI; TUU-SZABO, 2018; WOOD, 2016). A Figura 12 apresenta o fluxograma padrão de um AM.

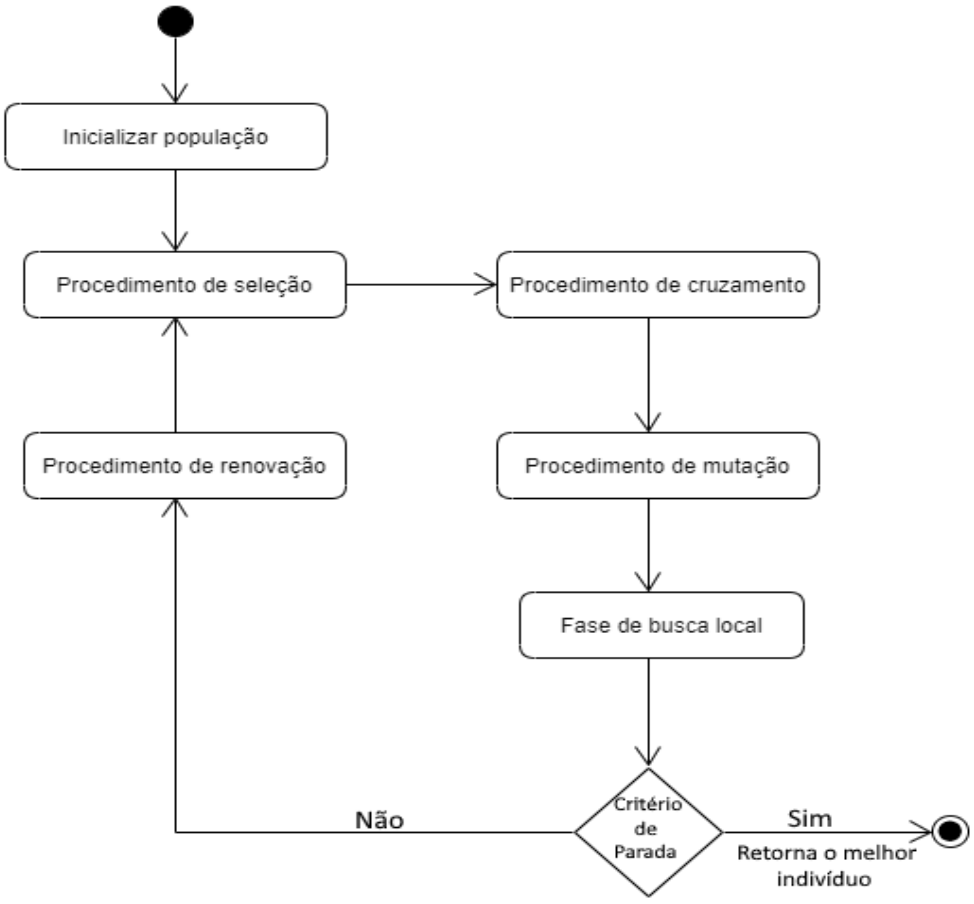

Figura 12: Fluxograma de um Algoritmo Memético.

5.3.2 Algoritmos Evolucionários Propostos para o PCV-CPTIT

Nesta seção são descritos os algoritmo evolucionários aplicados no estudo. As seções 5.3.2.1 e 5.3.2.2 apresentam as adaptações do AG e do AM, respectivamente. A seção 5.3.2.3 apresenta o Algoritmo Memético com Variação Cultural(AM-VC).

5.3.2.1 Algoritmo Genético

Nesta seção é descrita a adaptação de um Algoritmo Genético (GOLDBERG, 1989) para o PCV-CPTIT. O algoritmo 15 apresenta o pseudocódigo da adaptação desta metaheurística ao PCV-CPTIT. Os parâmetros de entrada são: o número de gerações ($nGer$), o tamanho da população ($tamPopl$), a taxa de cruzamento ($txRec$) que representa o número de indivíduos que reproduzem em cada geração, a taxa de mutação ($txMut$) e a taxa de renovação da população ($txRen$).

Algoritmo 15 Algoritmo Genético

Parâmetros de entrada: $INST$, $nGer$, $tamPopl$, $txRec$, $txMut$, $txRen$
1. $L_{popl} \leftarrow$ gerarPopulacaoInicial($tamPopl$)
2. faseDeBuscaLocal(L_{popl})
3. Para i de 1 até $nGer$
4. para j de 1 até $tamPopl \times txRec$
5. $parente1$,$parente2 \leftarrow$ selecaoDeParentes()
6. $herdeiro1$, $herdeiro2 \leftarrow$ realizaCruzamento($parente1$,$parente2$)
7. $herdeiro1$, $herdeiro2 \leftarrow$ realizaMutacao($herdeiro1$,$herdeiro2$,$txMut$)
8. se $herdeiro1$, $herdeiro2 < L_{popl}[parente1]$,$L_{popl}[parente2]$
9. $L_{popl}[parente1] \leftarrow herdeiro1$, $L_{popl}[parente2] \leftarrow herdeiro2$
10. geracaoDeNovosIndividuos($tamPopl \times txRen$)
11. Retorne o melhor individuo de L_{popl}

No AG proposto, o cromossomo é representado simplesmente pela sequência de vértices que compõem do trajeto do caixeiro. Os cromossomos podem ter tamanhos variados. No procedimento de geração da população inicial, cada individuo é obtido com base no *Procedimento Construtivo Adaptado* do GRASP descrito no algoritmo 13. Ainda nessa fase, aplica-se HC em cada indivíduo gerado. Estas operações fazem parte do procedimento $gerarPopulacaoInicial()$. Os indivíduos da população inicial são registrados na lista L_{popl}. Cada individuo de L_{popl} é refinado com o algoritmo HBL-Mo. As iterações que compreendem os passos 4 ao 10 do algoritmo implementam o processo de evolução da população. Durante este processo, um percentual de indivíduos da população corrente será selecionado para o experimento evolucionário. Este percentual é dado pelo parâmetro $txRen$.

Em toda geração, dois indivíduos são escolhidos no procedimento $selecaoDeParentes()$. Utiliza-se o método de roleta durante a escolha com base na avaliação do aptidão de cada individuo. Considerando que uma solução viável é abstraída como um individuo, sua aptidão corresponde ao custo desta solução. Ambos os indivíduos selecionados ($parente1$ e $parente2$) são submetidos a um procedimento de cruzamento chamado $realizaCruzamento()$, gerando dois novos indivíduos ($herdeiro1$ e $herdeiro2$). Para favorecer a diversificação do AG, foi utilizada a estrategia de cruzamento uniforme.

Os herdeiros previamente gerados durante o experimento de cruzamento são conduzidos ao processo de mutação. Este processo é abstraído no procedimento $realizaMutacao()$. Foi utilizado o operador de mutação proposto em (WANG et al., 2016). Este operador é descrito como se segue: seja o cromossomo $E = \{1, 5, 8, 7, 3, 2, 1\}$ aquele selecionado para mutação, sorteiam-se com distribuição uniforme dois pontos de mutação em E, denotados por $|$, tal que $E = \{1, |5, 8, 7, 3|, 2, 1\}$. O trecho delimitado pelos pontos de mutação $|$ é invertido e obtém-se $E^{'} = \{1, 3, 7, 8, 5, 2, 1\}$.

Em toda geração nova, um subconjunto de indivíduos da população corrente é trocada por novos indivíduos. Esta operação de renovação é abstraída no procedimento $geracaoDeNovosIndividuos()$. Os novos cromossomos são gerados com o mesmo método construtivo adotado em $gerarPopulacaoInicial()$. O parâmetro $txRen$ define a percentual de indivíduos que serão renovados. Assim, o simples cálculo $tamPopl \times txRen$ resulta na quantidade de novos indivíduos a serem criados para substituir um mesmo número de indivíduos menos adaptados da população corrente. Este processo de renovação visa diversificar e evitar uma convergência prematura no processo de busca no espaço de soluções viáveis do problema.

5.3.2.2 Algoritmo Memético

Nesta seção é descrita a adaptação de um Algoritmo Memético(MOSCATO; PLATA; NORMAN, 1999) para o PCV-CPTIT. O algoritmo 16 apresenta o pseudocódigo da adaptação desta meta-heurística. Nesta adaptação, um meme é representado pela sequência de vértices que compõem do trajeto do caixeiro e possuem tamanhos variados. Enquanto que em um AG os genes são submetidos ao processo de transmissão de informação onde os descendentes herdam habilidades e características dos seus progenitores, os memes (analogia ao gene, porém no contexto cultural) são adaptados pelo indivíduo que o recebe com base no seu conhecimento e para atender suas necessidades (ALMEIDA, 2015). Essa distinção conceitual se deve ao fato de que um AG visa reproduzir a evolução biológica,

enquanto que um AM simula a evolução cultural.

Semelhante ao AG descrito na seção anterior, os parâmetros de entrada do AM proposto são: o número de gerações ($nGer$), o tamanho da população ($tamPopl$), a taxa de cruzamento ($txRec$) que representa o número de indivíduos que reproduzem em cada geração, a taxa de mutação ($txMut$) e a taxa de renovação da população ($txRen$).

Algoritmo 16 Algoritmo Memético

Parâmetros de entrada: $INST$, $nGer$, $tamPopl$, $txRec$, $txMut$, $txRen$
1. $L_{popl} \leftarrow$ gerarPopulacaoInicial($tamPopl$)
2. faseDeBuscaLocal(L_{popl})
3. Para i de 1 até $nGer$
4. para j de 1 até $tamPopl \times txRec$
5. $parente1$,$parente2 \leftarrow$ selecaoDeParentes()
6. $herdeiro1$, $herdeiro2 \leftarrow$ realizaCruzamento($parente1$,$parente2$)
7. $herdeiro1$, $herdeiro2 \leftarrow$ realizaMutacao($herdeiro1$,$herdeiro2$,$txMut$)
8. HBL-Mo($herdeiro1$,$herdeiro2$)
9. se $herdeiro1$, $herdeiro2 < L_{popl}[parente1]$,$L_{popl}[parente2]$
10. $L_{popl}[parente1] \leftarrow herdeiro1$, $L_{popl}[parente2] \leftarrow herdeiro2$
11. geracaoDeNovosIndividuos($tamPopl \times txRen$)
12. Retorne o melhor individuo de L_{popl}

Como pode ser visto no algoritmo 16, após as operações de cruzamento e mutação, aos herdeiros, aplica-se o algoritmo HBL-Mo para promover o aperfeiçoamento da população. As aptidões dos indivíduos produzidos são comparadas com as de seus parentes e, dentre estes quatro, os dois mais bem adaptados sobrevivem. Esta é a única diferença desta adaptação de AM para o AG apresentado na seção anterior.

5.3.2.3 Algoritmo Memético com Variação Cultural

O AM-VC utiliza a mesma estrutura lógica do AM proposto na seção anterior, porém duas alterações são introduzidas. A primeira é aplicada no procedimento de geração da população inicial. Visto que a tomada de decisão necessária para tratar o PCV-CPTIT é complexa e exige a consideração de restrições de custo de deslocamento, tempo de deslocamento, coleta de bônus e satisfação dos passageiros, e o *Procedimento Construtivo Adaptado*, descrito no algoritmo 13 e utilizado também nas abordagens evolucionárias deste trabalho para gerar a população inicial, considera apenas o custo de deslocamento, foi projetado um mecanismo para variar o viés de construção de indivíduos e contemplar os outros aspectos da tomada de decisão do PCV-CPTIT.

Neste novo procedimento de inicialização da população, chamado de *Procedimento Construtivo Cultural*, um operador κ é sorteado com distribuição uniforme para definir

qual será a lógica da função de avaliação incremental. O pseudocódigo deste procedimento é apresentado no algoritmo 17.

Algoritmo 17 Procedimento Construtivo Cultural

Parâmetros de entrada: α, K
1. $S \leftarrow \{s\}$
2. Inicializar a lista de candidatos: $C \leftarrow E$
3. Inicializar contador de bônus coletado: $\gamma \leftarrow 0$
4. Definir aleatoriamente o operador de avaliação incremental: $c() \leftarrow \kappa$
5. Avaliar o custo incremental $c(e)$ para todo $e \in C$
6. Enquanto $\gamma < K$ faça
7. $\quad c^{min} \leftarrow min[c(e)|e \in C]$
8. $\quad c^{max} \leftarrow max[c(e)|e \in C]$
9. $\quad LRC \leftarrow \{e \in C \ \ |c(e) \leq c^{min} + \alpha(c^{max} - c^{min})\}$
10. $\quad$ Selecionar aleatoriamente um elemento e da LRC
11. $\quad S \leftarrow S \cup e$
12. $\quad$ Atualizar a lista de candidatos C
13. $\quad$ Incrementar γ com o bônus coletado em e
14. $\quad$ Reavaliar o custo incremental $c(e)$ para todo $e \in C$
15. Aplicar HC em S
16. Retornar S

Como pode ser visto no algoritmo 17, a única distinção na estrutura do algoritmo *Procedimento Construtivo Cultural* para o *Procedimento Construtivo Adaptado*, apresentado no algoritmo 13, é a definição do operador de avaliação incremental κ o qual deve variar a cada execução do procedimento. Quatro implementações foram fornecidas para o operador κ. A lista a seguir elenca estas implementações:

- Operador de custo: considera a variável c_{ij} para inserir vértices e ajustar o tamanho da LRC de tal forma que os vértices escolhidos para compor o indivíduo em construção sejam aqueles que representem o menor acréscimo na distância total percorrida;

- Operador de tempo: considera a variável t_{ij} para inserir vértices e ajustar o tamanho da LRC de tal forma que os vértices escolhidos para compor o indivíduo em construção sejam aqueles que representem o menor acréscimo no tempo de deslocamento da viagem;

- Operador de bônus: considera a variável q_j para inserir vértices e ajustar o tamanho da LRC de tal forma que os vértices escolhidos para compor o indivíduo em construção sejam aqueles que representem o maior acréscimo ao contador de bônus coletado γ;

- Operador de passageiros: considera a variável $|L_j|$ para inserir vértices e ajustar o tamanho da LRC de tal forma que os vértices escolhidos para compor o indivíduo em construção sejam aqueles sejam que possuem o maior número de solicitações de viagem;

O termo *Variação Cultural* faz referência ao fato de que a população gerada neste algoritmo é composta por indivíduos concebidos com perspectivas distintas sobre como resolver o problema em questão. A ideia é combinar estas perspectivas no procedimento de cruzamento para potencializar o poder de diversificação da busca. A única distinção estrutural do algoritmo *Procedimento Construtivo Cultural* para o *Procedimento Construtivo Adaptado*, apresentado no algoritmo 13, é a definição do operador de avaliação incremental o qual deve variar a cada chamada do procedimento de construção de novos indivíduos.

A segunda alteração é implementada no procedimento de mutação. No AM-VC, utilizam-se múltiplos operadores de mutação. Estes operadores foram projetados para contemplar as diferentes vertentes do PCV-CPTIT. A lista a seguir descreve estes operadores:

- Mutação sobre o bônus: consiste em selecionar aleatoriamente com distribuição uniforme um vértice com coleta de de bônus marcada e desmarcá-la, tal que a restrição do bônus mínimo K continue satisfeita. Em seguida, aplica-se a HC. O objetivo de eliminar o tempo de coleta do bônus de um vértice e tentar alocar mais passageiros no trajeto a ser percorrido. O cromossomo permanece com o mesmo tamanho;

- Mutação *push*: consiste em selecionar aleatoriamente com distribuição uniforme um vértice não visitado v_k e encontrar uma aresta (v_i, v_{i+1}) no ciclo Ψ tal que $(c_{i,k} + c_{k,i+1} - c_{i,i+1})$ seja mínimo. Em seguida, aplica-se a HC. O cromossomo aumenta de tamanho;

- Mutação *pop*: consiste em selecionar aleatoriamente com distribuição uniforme um vértice $v \in \Psi$ e removê-lo. Decorrente desta operação, se a restrição do bônus mínimo K não for mais satisfeita, executa o operador *push* até que a restrição K seja novamente satisfeita. Em seguida, aplica-se a HC. O cromossomo pode diminuir de tamanho;

- Mutação *flip*: aplica-se o operador de mutação proposto em (WANG et al., 2016) e descrito na sub-seção anterior. Em seguida, aplica-se a HC. O cromossomo permanece do mesmo tamanho.

5.4 Algoritmos de Formigas

Nesta seção são descritas noções importantes para o entendimento da classe de algoritmos *Ant Colony Optimization* e a aplicação destes algoritmos no problema em estudo. Na sub-seção 5.4.1, é feito uma descrição geral sobre esta classe de algoritmos. As adaptações de meta-heurísticas dessa classe ao PCV-CPTIT são apresentadas na sub-seção 5.4.2.

5.4.1 Descrição

Ant Colony Optimization é uma classe de meta-heurísticas inspiradas na capacidade das formigas encontrarem o caminho mais curto entre o ninho e a fonte de alimento. Essa capacidade é simulada em colônias artificiais de formigas para a resolução de problemas de otimização combinatória.

Tomando como exemplo o PCV clássico, se um algoritmo da classe ACO é aplicado em sua resolução, as formigas artificiais tendem a se deslocar entre os vértices do grafo do PCV. Neste contexto, o comportamento das formigas artificiais preserva quatro noções do comportamento natural das formigas:

- Deposito de feromônio na trilha percorrida;
- Predileção por trilhas com concentração de feromônio;
- Concentração da quantidade de feromônio em trilhas menos extensas;
- Comunicação entre as formigas por meio do deposito de feromônio.

O feromônio pode ser descrito como uma estrutura química de comunicação e sinalização (JACKSON; RATNIEKS, 2006). Segundo (DORIGO; BONABEAU; THERAULAZ, 2000), é um recurso que viabiliza o processo de estigmergia e auto-organização em que agentes simples são capazes de emergir comportamentos complexos e direcionado a um objetivo. O princípio da estigmergia é que o traço deixado no ambiente por uma ação estimula o desempenho de uma próxima ação, pelo mesmo ou por um agente diferente (HADELI et al., 2004). Nas formigas, por exemplo, o processo de estigmergia ocorre pela troca de informações através do estabelecimento de feromônio no caminho de volta para a colônia após elas terem descoberto alimentos (DORIGO; BONABEAU; THERAULAZ, 2000).

Considerando o contexto da aplicação de um algoritmo da formigas na resolução do PCV, ao se deslocarem pelo grafo, as formigas artificias tendem a seguir por caminhos com

maiores taxas de deposito de feromônio. Como as formigas tendem a depositar feromônio pelo caminho que seguem, a medida em que mais formigas optam pelo mesmo caminho, a taxa de feromônio tende a crescer nestes caminhos. Este mecanismo de cooperação induz as formigas a encontrarem boas soluções, pois funciona como uma forma de memória compartilhada constantemente atualizada e compartilhada para auxiliar na tomada de decisão (SKINDEROWICZ, 2016). A Figura 13 ilustra esse comportamento.

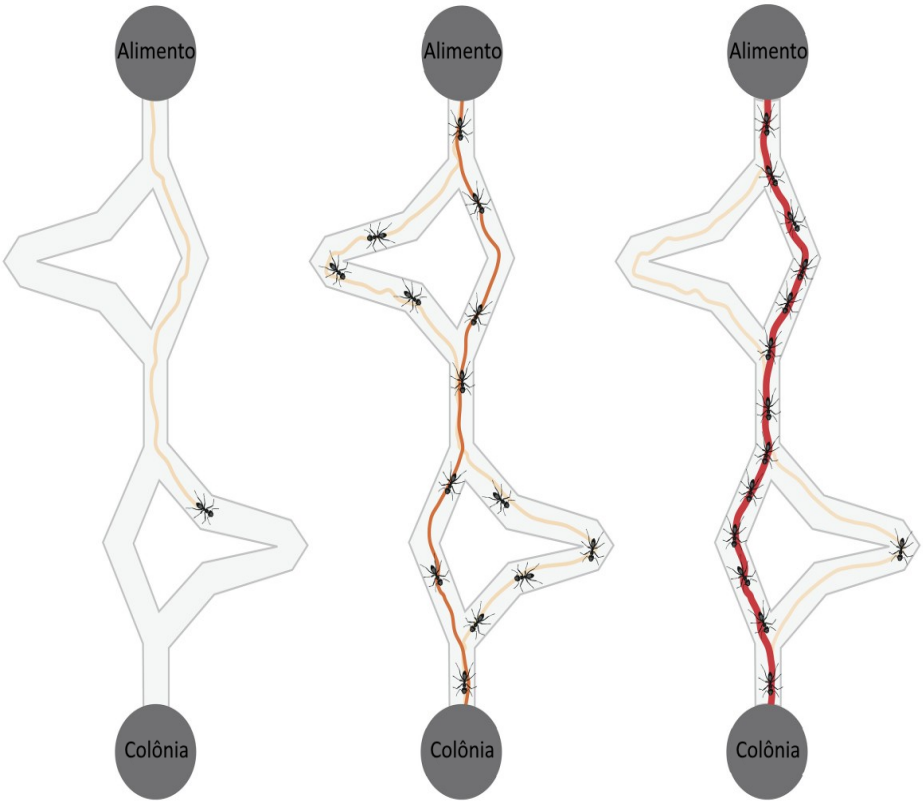

Figura 13: Comportamento de formigas para encontrar alimento.

O primeiro algoritmo proposto da classe ACO é o *Ant System* (DORIGO; MANIEZZO; COLORNI, 1996). Segundo (STÜTZLE; DORIGO et al., 1999), os algoritmos de formigas que foram propostos depois do AS são aprimoramentos deste. Todas essas versões aprimoradas do AS têm em comum estrategias elitistas de atualização de feromônio e uso de algoritmos de busca local para aprimorar as soluções encontradas. A variante mais conhecida do AS é o algoritmo *Ant Colony System*(DORIGO; STÜTZLE, 2003). Uma revisão recente da classe de algoritmos ACO pode ser consultada em (DORIGO; STÜTZLE, 2019). A estrutura geral dos algoritmos da classe ACO é ilustrado na Figura 14.

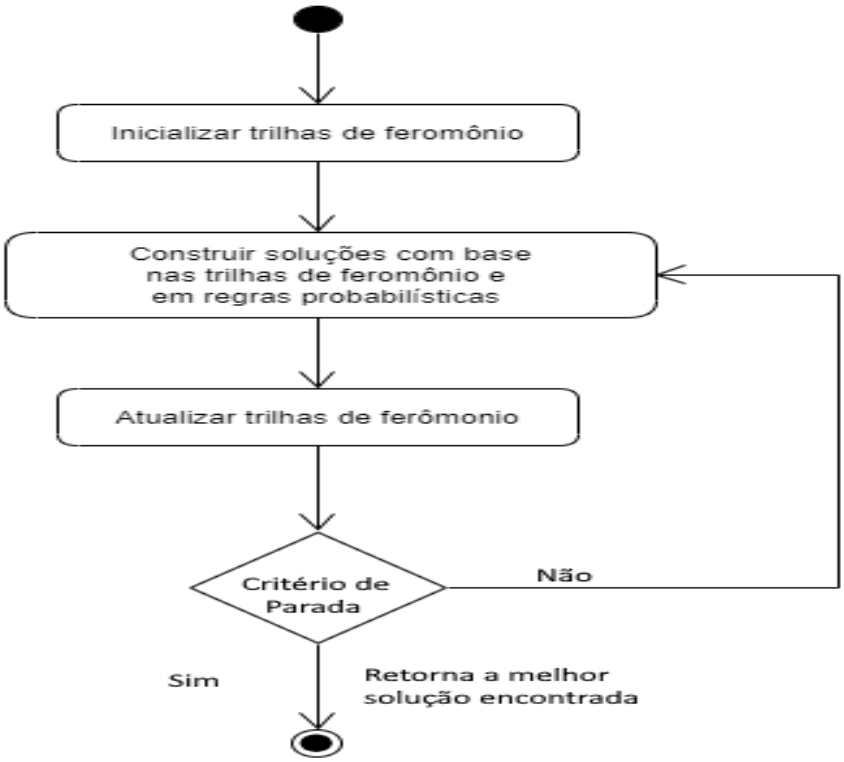

Figura 14: Estrutura geral dos algoritmos de formigas.

5.4.2 Algoritmos de Formigas Propostos para o PCV-CPTIT

A seguir, são descritos os algoritmo de formigas aplicados no estudo. As seções 5.4.2.1 e 5.4.2.2 apresentam as adaptações do AS e do ACS, respectivamente. A seção 5.4.2.3 apresenta o *Multi-Strategy Ant Colony System*.

5.4.2.1 *Ant System*

O algoritmo AS é o arcabouço da classe de algoritmos ACO(DORIGO; STÜTZLE, 2003) e possui os seguintes parâmetros: número total de formigas, $m \in \mathcal{Z}^*$, coeficiente de feromônio, $\alpha \in \mathcal{Z}$, coeficiente heurístico, $\beta \in \mathcal{Z}$ e fator de evaporação, $\rho \in [0, 1]$ (DORIGO; MANIEZZO; COLORNI, 1996). α e β são parâmetros para ponderar a influência das informações de feromônio e fator heurístico durante o processo de construção da rota executado por cada formiga. O algoritmo 18 apresenta o pseudocódigo do AS adaptado ao PCV-CPTIT. Cada formiga começa no vértice s. Todas seguem incluindo vértices na sua rota até atingir a cota mínima. Qualquer rota criada por uma formiga é submetida ao algoritmo HC.

Algoritmo 18 AS(m, α, β, ρ, τ_{ij}^0)

1. Inicializar as trilhas de feromônio
2. Para $k \leftarrow 1$ até m
3. $\quad W^k[1] \leftarrow s$; $W^k[2] \leftarrow$ sortear_cidade($N \setminus \{s\}$)
4. For $k \leftarrow 1$ to m
5. $\quad W^k \leftarrow$ construir_rota(α, β)
6. $\quad S^k \leftarrow$ HC(W^k)
7. $\quad$ Depositar_feromônio(W^k,ρ,τ_{ij}^0)
8. $\quad$ Atualizar(S^*)
9. Retornar S^*

O mecanismo utilizado para definição do feromônio é o mesmo proposto por (DORIGO; GAMBARDELLA, 1997). Inicialmente, os arcos têm a mesma quantidade de feromônio, τ_{ij}^0, calculada pela equação (5.3), em que n é o número de vértices e D é o custo de uma rota satisfatória para o PCV construída pela heurística do vizinho mais próximo(ROSENKRANTZ; STEARNS; LEWIS, 1977). Cada formiga começa no vértice s e o segundo vértice adicionado à rota é selecionado aleatoriamente com distribuição uniforme (etapas 2 e 3).

As formigas constroem suas rotas na etapa 5 usando a equação (5.2), que calcula a probabilidade de a k-ésima formiga se mover do vértice i para j na t-ésima iteração, em que $\eta_{ij} = \frac{1}{c_{ij}}$ é o fator heurístico, $\tau_{ij}(t)$ é o feromônio no arco (i, j) na t-ésima iteração e Λ^k é a lista de vértices não visitados pela k-ésima formiga. α e β são coeficientes que calibram a influência do feromônio e da informação heurística, respectivamente. Se $\alpha = 0$, a probabilidade calculada pela equação (5.2) é influenciada apenas pelas informações heurísticas. Portanto, o algoritmo de formigas tende a se comportar como uma heurística gulosa. Se $\beta = 0$, as formigas tendem a selecionar frequentemente caminhos com níveis mais altos de feromônio. Isso pode levar o algoritmo à estagnação precoce. Portanto, equilibrar os valores de α e β é fundamental para garantir uma estratégia de busca adequada.

$$\Upsilon_{ij}^k(t) = \frac{[\tau_{ij}(t)]^\alpha \cdot [\eta_{ij}]^\beta}{\sum_{w \in \Lambda^k} \left[[\tau_{iw}(t)]^\alpha \cdot [\eta_{iw}]^\beta \right]}, \qquad j \in \Lambda^k \tag{5.2}$$

A solução S^k, construída pela k-ésima formiga, é calculada atribuindo passageiros a W^k com o algoritmo HC (etapa 6). As equações (5.4) - (5.5) mostram as fórmulas usadas para atualizar trilhas de feromônios (etapa 7), em que $\tau_{ij}(0)$ denota o feromônio inicial no arco (i, j), ρ é o coeficiente de evaporação, $Custo(W^k)$ é o custo da rota W da k-ésima formiga e $\Delta\tau_{ij}^k$, calculado pela equação (5.6), é o feromônio depositado no arco (i, j) pela k-ésima formiga. A melhor solução encontrada até então é computada na etapa 8. Ao

final, o algoritmo retorna S^*.

$$\tau_{ij}^0 = (n \times Custo(D))^{-1} \tag{5.3}$$

$$\tau_{ij} = (1 - \rho) \times \tau_{ij} + \rho \times \Delta\tau_{ij}, \qquad \rho \in [0, 1] \tag{5.4}$$

$$\Delta\tau_{ij} = \sum_{k=1}^{m} \Delta\tau_{ij}^k \tag{5.5}$$

$$\Delta\tau_{ij} = \begin{cases} \frac{1}{Custo(W^k),} & \text{se o arco } (i,j) \in W^k \text{ .} \\ 0, & \text{caso contrário.} \end{cases} \tag{5.6}$$

A principal ideia por trás das equações (5.4)-(5.6) é criar um mecanismo de memória distribuída abstraído como feromônio, onde as informações não são armazenadas individualmente em cada formiga, mas distribuídas nos arcos do grafo (DORIGO; MANIEZZO; COLORNI, 1996). Como o custo c_{ij} de cada arco $(i, j) \in W$ é dividido igualmente entre o caixeiro e os passageiros, a quantidade de feromônio depositada no arco (i, j) é influenciada pelas requisições de viagem atendidas em W^k.

5.4.2.2 *Ant Colony System*

O algoritmo ACS também usa a equação (5.4) para calcular a probabilidade de uma formiga seguir do vértice i para j. No entanto, um viés guloso é introduzido pelo uso do coeficiente $q0$, um parâmetro ajustável. Seja ω um número aleatório do intervalo $[0, 1]$. Se $\omega < q0$, a formiga escolhe se mover para j de modo que $\eta_{ij} = \frac{1}{c_{ij}}$ seja máximo. Caso contrário, a regra de transição estabelecida pela equação (5.4) é aplicada. Além de m, α, β, ρ, τ_{ij}^0 e $q0$, o ACS possui o parâmetro $maxIter$, o número máximo de iterações e o parâmetro ϕ usado para controlar a evaporação do feromônio. A equação (5.7) apresenta a forma de cálculo da evaporação de feromônio proposta em (DORIGO; STÜTZLE, 2003). O pseudocódigo do algoritmo ACS adaptado ao PCV-CPTIT é apresentado no algoritmo 19.

$$\tau_{ij} = (1 - \phi) \times \tau_{ij} + \phi \times \tau_{ij}^0, \qquad \phi \in [0, 1] \tag{5.7}$$

Algoritmo 19 ACS($maxIter$, m, α, β, ρ, ϕ, τ_{ij}^0, $q0$)

1. Inicializar as trilhas de feromônio
2. For $k = 1$ to m
3. $W^k[1] \leftarrow s$; $W^k[2] \leftarrow$ sortear_cidade($N \setminus \{s\}$)
4. Para $i = 1$ até $maxIter$
5. Para $k = 1$ até m
6. $W^k \leftarrow$ constuir_rota(α, β, $q0$)
7. $S^k \leftarrow$ alocar_passageiros(W^k)
8. Evaporar_feromônio(W^k,ϕ,τ_{ij}^0)
9. Atualizar(W^*,S^*)
10. $S^* \leftarrow$ HBL-Mo(S^*)
11. Depositar_feromônio(S^*,ρ)
12. Retornar S^*

O ACS inicializa trilhas de feromônio, conforme descrito na seção 5.4.2.1 (etapa 1). Como todas as formigas começam no vértice s, a seleção do segundo vértice da rota construída por cada formiga é aleatória com distribuição uniforme (etapas 2 e 3). A k-ésima formiga constrói a rota na etapa 6. Em seguida, o algoritmo usa a heurística HC para atribuir passageiros a W^k, concluindo o processo de construção da solução da k-ésima formiga (etapa 7). Diferente do AS, o ACS possuí atualiza o feromônio localmente e globalmente. A equação (5.7) é usada para atualização local (etapa 8), onde ϕ é um fator de evaporação e τ_{ij}^0 é o feromônio inicial no arco (i, j). A melhor rota encontrada até então, W^*, é usada para a regra de intensificação global das trilhas de feromônio. O algoritmo HBL-Mo é aplicado à melhor solução encontrada até então (etapa 10). Após a busca local, a rota associada a S^*, W' é usada para a intensificar as taxas de deposito de feromônio, via equação (5.4). Ao final, o algoritmo retorna S^*.

5.4.2.3 *Multi-Strategy Ant Colony System*

Existem diferentes tipos de decisões a serem tomadas durante o processo de resolução do PCV-CPTIT, tais como: a satisfação da cota mínima; a aceitação ao máximo aproveitamento das solicitações de carona; e a minimização dos custos de viagem sob a ótica do caixeiro. Nesse contexto, a ideia do *Multi-Strategy Ant Colony System* é a de inserir o conceito de várias estratégias heurísticas na tomada de decisão do algoritmo ACS.

No ACS tradicional, toda formiga usa como fator heurístico o custo dos arcos. No MS-ACS, existem quatro estratégias heurísticas listadas a seguir:

- Orientado a custos: as informações heurísticas são as mesmas usadas no ACS, ou seja, $\eta_{ij} = \frac{1}{c_{ij}}$.

- Orientado para o tempo: as informações heurísticas são $\eta_{ij} = \frac{1}{t_{ij}}$. Essa estratégia visa economizar tempo de viagem.

- Orientado a cotas: as informações heurísticas são $\eta_{ij} = \frac{q_j}{c_{ij}}$. Essa estratégia visa escolher vértices j com as maiores taxas de bônus.

- Orientado ao passageiro: as informações heurísticas são $\eta_{ij} = \frac{|L_j|}{c_{ij}}$. Essa estratégia visa maximizar a possibilidade de satisfazer o maior número possível de solicitações de viagem. Então, vértices com boas taxas de número de solicitações de viagem têm maior probabilidade de serem visitados partindo-se de i.

A estratégia heurística de cada formiga é definida aleatoriamente com distribuição uniforme. As diferentes estratégias heurísticas foram projetadas para cobrir todos os aspectos do problema e evitar que o processo de busca fique preso prematuramente a ótimos locais.

6 Experimentos Computacionais

Os experimentos computacionais deste trabalho foram executados em um computador com processador Intel core i5, 2,6 GHz, Windows 10, 64 bits e 6 GB de memória RAM. Os algoritmos foram implementados em linguagem C++ e compilados com o GNU g ++ versão 4.8.4. A formulação matemática, descrita nas equações (3.1)-(3.22), foi implementada no *Gurobi optimizer* (GUROBI, 2018), versão 7.5, e no *C-Plex optimizer* (C-PLEX, 2020), versão 12.8.

Foi criado um conjunto de 144 instâncias (simétricas e assimétricas) usando o método descrito na seção 6.1. O porte dessas instâncias varia de 10 a 500 vértices. Nesta estudo, instâncias consideradas pequenas são aquelas cujo o número de vértices não ultrapassa 40. Instâncias de porte médio não possuem mais que 100 vértices. Já as instâncias de grande porte possuem mais que 100 vértices. Todas as instâncias geradas neste estudo estão disponíveis para download no `https://github.com/laegit/QTSP_PIC_Benchmark_Set`.

O ajuste dos parâmetros dos algoritmos heurísticos foi feito com o software *Iterated Racing for Automatic Algorithm Configuration*(IRACE)(LÓPEZ-IBÁÑEZ et al., 2016). O ajuste do parâmetro é detalhado na seção 6.3. As instâncias foram submetidas aos *solvers* para concluir sobre a dificuldade de resolver nosso modelo e obter os melhores resultados. Foi definido como 80.000 segundos o tempo máximo de processamento para ambos os *solvers*. A seção 6.2 apresenta os resultados deste experimento.

Nesta seção, também são relatados os melhores resultados e a média destes em 20 execuções independentes dos algoritmos heurísticos e os tempos médios de processamento em segundos. Foram implementadas cinco versões da HN. Estas implementações se distinguem pela definição dos procedimentos de roteamento e alocação de passageiros os quais podem ser exatos ou heurísticos. No procedimento de roteamento exato, a entrada para o *solver* Gurobi é a função objetivo (6.1) e as restrições (3.2)-(3.9) e (3.23). Neste procedimento, o *solver* é configurado para procurar uma solução ótima para o PCV-C enquanto o tempo máximo de processamento não é atingido. No procedimento exato de

alocação de passageiros, a entrada para o *solver* Gurobi é a formulação (3.1)- (3.22) com as variáveis x_{ij} prefixadas pelo procedimento de roteamento. O tempo máximo de processamento permitido para o *solver* foi de 80.000s. Durante a implementação das *Naives* que utilizam a HC, foi constatado que a HC conseguiu decrescer em média 42% do custo do ciclo produzido na fase de roteamento.

$$\min \sum_{(i,j)\in A} x_{ij}c_{ij} \tag{6.1}$$

As cinco implementações da HN fornecidas neste estudo são elencada como se segue:

- $HN1$: neste implementação, é combinado roteamento exato com alocação de passageiros exato;
- $HN2$: combina roteamento exato e procedimento de alocação de passageiros heurístico;
- $HN3$: consiste em roteamento heurístico e procedimento de alocação de passageiros exato;
- $HN4$: combina roteamento heurístico com e procedimento de alocação de passageiros heurístico.
- $HN5$: consiste em produzir uma solução com a $HN4$ e aperfeiçoar esta solução com a $HBL - Mo$.

Os métodos $HN1$ e $HN2$ foram executados apenas uma vez para cada instância. A ideia do experimento conduzido com as implementações da HN fornecer resultados iniciais para o conjunto de instâncias de *benchmarking*. Os resultados documentados neste experimento servem para validar a eficácia das abordagens meta-heurísticas propostas neste estudo. Tais abordagens meta-heurísticas também são comparadas entre si para estabelecer qual destas é a que melhor resolve o problema em questão e para concluir sobre a influência de estratégias heurísticas específicas. A seção 6.4 apresenta os resultados comparativos dos algoritmos de formigas entre si. A seção 6.5 apresenta a análise comparativa dos algoritmos de evolucionários. A seção 6.6 apresenta a comparação dos resultados obtidos pelas meta-heurísticas GRASP, ILS, o algoritmo de formiga e o evolucionário mais promissores.

Na última seção deste capítulo, enumerada como 6.7, comparamos os resultados das meta-heurísticas híbridas GRASP-ILS, MS-ACS-ILS e AM-VC-ILS. A ideia deste experi-

mento é tentar melhorar a qualidade das soluções para o banco de instâncias. A escolha das meta-heurísticas GRASP, MS-ACS e AM-VC para fazerem parte do experimento de hibridização se deve ao bom desempenho destas meta-heurísticas em comparação com as demais implementadas nesta pesquisa. Como o projeto de implementação destas três meta-heurísticas compreende uma fase de melhoramento de soluções, a meta-heurística ILS foi empregada para desempenhar esta função. A escolha por estas combinações é sustentada por estudos em que se aplicou com êxito cada uma destas hibridizações em problemas correlatos ao em estudo. (PRINS, 2009; LIU; JIANG; GENG, 2013; AL-BEHADILI; KU-MAHAMUD; SAGBAN, 2020). Visto que os algoritmos hibridizados podem demandar bastante tempo de processamento, foi definido como 80.000 segundos o tempo máximo de processamento.

Em todas as seções que tratam do desempenho das meta-heurísticas, foram realizados experimentos para relatar a distância entre as melhores soluções encontradas para cada instância e os melhores resultados fornecidos pelas abordagens meta-heurísticas. Também é calculado a variabilidade na qual as abordagens meta-heurísticas alcançam as melhores soluções conhecidas das instâncias de *benchmarking*. A partir dos experimentos de distância e variabilidade, é possível concluir se cada abordagem meta-heurística é capaz de encontrar as melhores soluções conhecidas e com que variabilidade isso acontece.

Aplicamos o teste de *Friedman* (FRIEDMAN, 1937) com o procedimento *post-hoc* de *Nemenyi* (NEMENYI, 1963), nível de significância 0,05, para concluir sobre diferenças significativas entre os resultados das abordagens meta-heurísticas propostos neste estudo. Neste experimento estatístico, agrupamos as instâncias de acordo com seus tamanhos (número de vértices) para o teste de *Friedman*. Deste agrupamento, obteve-se oito grupos de instâncias simétricas e mais oito de assimétricas, cada um destes contendo nove instâncias. Estes grupos são denotados por $g < n >$, onde $< n >$ representa o tamanho do respectivo grupo g.

6.1 Banco de Instâncias

A entrada para o algoritmo de geração de instância é o número de vértices, n, a capacidade do veículo, R e o tipo da instância, simétrica ou assimétrica. A implementação do intervalo para o número de vértices n e do mecanismo de geração da quantidade de solicitações de viagem é baseado nos estudos (CORDEAU; LAPORTE, 2003; UCHOA, 2017; REINELT, 1991), uma vez que esses estudos abordam problemas correlatos ao do

PCV-CPTIT. Foram criadas três classes de instâncias denominadas A, B e C. Os parâmetros definidos para o algoritmo de geração de instâncias para criar um banco de testes com dificuldade variada são baseados em estudos sobre problemas correlatos tais como o PCV-C(AWERBUCH et al., 1998), DARP(CORDEAU; LAPORTE, 2007) e RP(NOURINEJAD; ROORDA, 2014).

Os elementos das matrizes de custo (c_{ij}) das três classes de instâncias foram gerados uniformemente a partir do intervalo $[100, 999]$. Os elementos das matrizes de tempo (t_{ij}) das instâncias da classe A foram gerados uniformemente a partir do intervalo $[20, 99]$. A geração dos elementos das matrizes de tempo das instâncias das classes B e C depende de suas matrizes de custo. Seja av_c a média dos elementos da matriz de custo de uma instância de classe B ou C. Se $c_{ij} < av\_c$, t_{ij} é um valor gerado uniformemente a partir do intervalo $[40, 99]$, caso contrário, o valor vem do intervalo $[20, 39]$.

O número de solicitações de viagem em cada vértice depende do tamanho do grafo e da capacidade do automóvel. Se $n \leq 20$, o número de solicitações de viagem em cada vértice é gerado uniformemente a partir de $[0, 3R]$; para $20 < n \leq 100$, o intervalo é $[0, 9R]$; e para $n > 100$, o intervalo é $[0, 27R]$. O destino da l-ésima solicitação de viagem ($dst(l)$) é gerado uniformemente a partir de $[1, n]$, $dst(l) \neq org(l)$. A tarifa máxima que o potencial passageiro l concorda em pagar, w_l, é gerada uniformemente a partir de $[0.15\iota, 0.3\iota]$, onde ι é o comprimento da árvore geradora mínima de G. Os pesos considerados para a árvore geradora mínima foram os custos, c_{ij}. A duração máxima de uma viagem para a l-ésima solicitação de viagem, b_l, refere-se ao comprimento do caminho mais curto entre $org(l)$ e $dst(l)$ em G. O peso considerado para cada arco (i, j) de G para calcular o caminho mais curto foi t_{ij}.

A penalidade para transportar o l-ésimo potencial passageiro ao destino j, h_{lj}, é 0 se $j = org(l)$ ou $j = dst(l)$. Caso contrário, $h_{lj} = h0_{lj} \times z_{ij}$, onde $h0_{lj}$ é o caminho mais curto entre $org(l)$ e j e z_{lj} é gerado uniformemente a partir de $[0.90, 1.50]$ para instâncias das classes A e B e de $[0.10, 0.30]$ para instâncias da classe C. As penalidades das instâncias da classe C são moderadas em comparação com as outras classes. Penalidades moderadas tendem a aumentar o número de solicitações de viagem que podem ser atendidas. Isso torna essas instâncias mais difíceis de resolver.

Seja $e_i = \sum_{j=1}^{n} c_{ij}, \forall i \in N$ e $z_\alpha = \frac{\sum_{i=1}^{n} e_i}{n}$. O valor q_i, $1 \leq i \leq n$, é gerado uniformemente a partir de $[100, 299]$ se $e_i \leq z_\alpha$, caso contrário, é gerado uniformemente a partir de $[300, 700]$. O objetivo é tornar vértices de alto custo, ou seja, aqueles para os quais $e_i > z_\alpha$, mais atraentes para o caixeiro do que vértices de baixo custo. O tempo neces-

sário para coletar o prêmio disponível no vértice i, g_i, é gerado uniformemente a partir de $[1, 39]$ se $e_i \leq z_\alpha$, caso contrário, a partir de $[40, 79]$. O valor da cota é calculado por $K = \epsilon(\sum_{i=1}^{n} q_i)$, onde ϵ é gerado uniformemente a partir de $[0.4, 0.6]$.

O formato de nomeação das instâncias é X - Y - Z, onde X, Y e Z representam a classe, o número de vértices e a capacidade do veículo, respectivamente. A Tabela 1 sumariza os parâmetros adotados no algoritmo de geração de instância.

Parâmetro	A	B	C
c_{ij}	[100,999]	[100,999]	[100,999]
t_{ij}	[20,99]	[20,39] or [40,99]	[20,39] or [40,99]
w_l	[0.15ι,0.3ι]	[0.15ι,0.3ι]	[0.15ι,0.3ι]
z_{lj}	[0.9,1.5]	[0.9,1.5]	[0.10,0.30]
q_i	[100,299] or [300,700]	[100,299] or [300,700]	[100,299] or [300,700]
g_i	[1,39] or [40,79]	[1,39] or [40,79]	[1,39] or [40,79]
g_i	[1,39] or [40,79]	[1,39] or [40,79]	[1,39] or [40,79]
ϵ	[0.4,0.6]	[0.4,0.6]	[0.4,0.6]

Tabela 1: Valores utilizados para os parâmetros do algoritmo de geração de instâncias em cada classe.

Os parâmetros relatados na Tabela 1 foram definidos com base em experimentos *ad-hoc* realizados para testar as restrições operacionais com condições rígidas de contorno.

6.2 Resultados Exatos

As Tabelas 2 e 3 mostram as instâncias para as quais os *solvers* de otimização conseguiram produzir resultados dentro do limite de tempo definido. Tais Tabelas mostram o nome da instância, na coluna *Instância*, o tipo (simétrico ou assimétrico), o valor da melhor solução encontrada, na coluna *UB* e o desvio percentual da melhor solução do limite inferior calculado pelo solver, na coluna *GAP (%)*.

A equação (6.2) mostra a fórmula para calcular o desvio percentual, em que LB representa o limite inferior, e UB, o limite superior. Nas Tabelas 2 e 3, o símbolo $*$ significa que o solver não conseguiu prosseguir com o processo de otimização devido a um de dois motivos:

- Consumo excessivo de memória primária;

- Ficou preso na fase de relaxamento de programação linear e não produziu nenhuma solução.

Na incidência de uma destas situações, os *solvers* não conseguem calcular um *GAP*. Nestes casos, o *GAP* nas Tabelas 2 e 3 é denotado pelo símbolo −.

$$GAP = \frac{|(LB - UB)|}{|UB|} \tag{6.2}$$

	Assimétrico		Simétrico	
Instância	UB	GAP(%)	UB	GAP(%)
A-10-3	431,58	35,00	545,92	0,00
A-10-4	467,60	55,00	460,00	41,00
A-10-5	638,08	83,60	412,73	75,00
A-20-3	5515,00	100,00	6931,00	99,00
A-20-4	5941,00	100,00	3936,00	99,50
A-20-5	∗	–	18904,00	100,00
B-10-3	729,50	54,00	834,67	33,80
B-10-4	395,60	81,80	578,73	81,00
B-10-5	431,30	73,70	748,35	71,50
B-20-3	5549,00	100,00	1946,00	100,00
B-20-4	4504,00	100,00	∗	–
B-20-5	3733,00	100,00	∗	–
C-10-3	356,33	100,00	513,33	35,27
C-10-4	362,80	100,00	412,57	96,17
C-10-5	569,67	95,55	533,80	87,14
C-20-3	∗	–	2871,75	100,00
C-20-4	2867,67	100,00	∗	–
C-20-5	∗	–	2399,17	100,00

Tabela 2: Resultados do solver de otimização *Gurobi* para o PCV-CPTIT.

Observando as Tabelas 2 e 3 é possível constatar que ambos os *solvers* conseguiram encontrar o resultado ótimo apenas para a instância simétrica *A-10-3*. Para tal feito, o *Gurobi* gastou 22087s nesse caso. Já o *C-Plex* gastou 20921s. Ambos os *solvers* conseguiram produzir soluções para instâncias de até 20 vértices das três classes. O *solver C-PLEX* apresentou melhores resultados para as instâncias simétricas *A-10-5* e *B-10-4*.

Em relação às instâncias com 10 vértices, as instâncias da classe C foram as que apresentaram os maiores *GAPs*. Como as penalidades relacionadas a desembarques em destinos alternativos foram relaxadas para as instâncias da classe C, aumentou o número de requisições que poderiam ser atendidas, e, consequentemente, a quantidade de soluções viáveis. Com o aumento do espaço de busca, os *solvers* tiveram que avaliar mais combinações que resultassem em soluções viáveis.

	Assimétrico		Simétrico	
Instância	UB	GAP(%)	UB	GAP(%)
A-10-3	431,58	37,00	545,92	0,00
A-10-4	467,60	56,00	460,00	38,00
A-10-5	638,08	79,00	371,93	59,00
A-20-3	5515,00	99,00	6931,00	95,00
A-20-4	5941,00	99,00	3936,00	98,00
A-20-5	∗	–	18904,00	100,00
B-10-3	729,50	52,00	834,67	27,00
B-10-4	395,60	79,00	493,58	72,00
B-10-5	431,30	70,20	748,35	71,00
B-20-3	5549,00	99,00	1946,00	99,00
B-20-4	4504,00	99,00	1829	99,00
B-20-5	3733,00	99,00	1923	99,00
C-10-3	356,33	95,00	513,33	32,90
C-10-4	362,80	91,00	412,57	92,50
C-10-5	569,67	89,20	533,80	80,10
C-20-3	∗	–	2871,75	99,00
C-20-4	2867,67	100,00	∗	–
C-20-5	∗	–	2399,17	99,00

Tabela 3: Resultados do solver de otimização *C-Plex* para o PCV-CPTIT.

6.3 Parametrização das Heurísticas

Os parâmetros dos algoritmos de formigas foram ajustados pelo software IRACE, apresentado por (LÓPEZ-IBÁÑEZ et al., 2016). Foram submetidas 20 instâncias simétricas e 20 assimétricas para ajustar os parâmetros. Essas instâncias foram selecionadas aleatoriamente. O IRACE usa os parâmetros $maxExperiments$ e $maxTime$ como critérios de parada. Foi definido $maxExperiments = 10^3$ e $maxTime = \infty$.

As Tabelas 4 e 5 apresentam os parâmetros produzidos pelo IRACE para os algoritmos de formigas, ILS e GRASP. A Tabela 6 apresenta os parâmetros produzidos pelo IRACE para as abordagens evolucionárias propostas no estudo.

Para a abordagem mémetica proposta neste estudo, os valores definidos para os parâmetros foram: o número de gerações ($nGer = 19$), o tamanho da população ($tamPopl = 21$), a taxa de recombinação ($txRec = 0,77$) que representa o número de indivíduos que reproduzem em cada geração, a taxa de mutação ($txMut = 0,68$) e a taxa de renovação da população ($txRen = 0,41$).

	Assimétrico				
Parâmetro	AS	ACS	MS-ACS	ILS	GRASP
$MaxIter$	–	20	19	51	43
m	344	38	43	–	–
α	0,15	6,10	2,39	–	0,69
β	3,31	8,33	0,00	–	–
ρ	0,18	0,98	0,86	–	–
ϕ	–	0,97	0,59	–	–
$q0$	–	0,82	0,96	–	–

Tabela 4: Valores referentes a parametrização dos algoritmos de formigas, ILS e GRASP, para instâncias assimétricas.

	Simétrico				
Parâmetro	AS	ACS	MS-ACS	ILS	GRASP
$MaxIter$	–	13	14	67	51
m	409	36	66	–	–
α	0,21	9,60	3,16	–	0,57
β	3,11	6,64	9,21	–	–
ρ	0,12	0,47	0,30	–	–
ϕ	–	0,77	0,48	–	–
$q0$	–	0,95	0,74	–	–

Tabela 5: Valores referentes a parametrização dos algoritmos de formigas, ILS e GRASP, para instâncias simétricas.

	Assimétrico			Simétrico		
Parâmetro	AG	AM	AM-VC	AG	AM	AM-VC
$nGer$	92	19	17	83	21	25
$tamPopl$	59	21	25	69	27	32
$txRec$	0,84	0,77	0,51	0,57	0,63	0,69
$txMut$	0,77	0,68	0,57	0,75	0,52	0,49
$txRen$	0,19	0,48	0,33	0,48	0,35	0,22

Tabela 6: Valores referentes a parametrização dos algoritmos evolucionários.

6.4 Análises Comparativas entre os Algoritmos de Formigas

Nesta seção, relatamos os resultados dos algoritmos de formigas e comparamos seus resultados com os produzidos pelo solver de otimização e pelas cinco versões das heurísticas *naive*. Também comparamos os algoritmo de formigas entre si.

A Tabela 7 apresenta a comparação entre os resultados dos algoritmos de formigas e o solver para instâncias com 10 vértices. Os resultados mostram a distância, υ, entre o melhor resultado encontrado por um dos algoritmos de formigas e a solução obtida pelo solver. A equação (6.3) mostra a fórmula para calcular υ, onde χ_{algo} e χ^* denotam a melhor solução encontrada pelo algoritmo de formigas e a solução produzida pelo solver, respectivamente. Valores positivos mostram que a solução obtida pelo solver foi melhor do que a obtida pelo algoritmo de formigas. Valores negativos denotam o oposto.

$$\upsilon = \frac{\chi_{algo}}{\chi^*} - 1 \tag{6.3}$$

	Assimétrico			Simétrico		
Instância	AS	ACS	MS-ACS	AS	ACS	MS-ACS
A-10-3	0,11	0,11	0,11	0,11	0,11	0,00
A-10-4	0,29	0,12	0,12	0,00	0,36	0,00
A-10-5	-0,15	-0,15	-0,24	0,16	0,12	0,00
B-10-3	0,07	0,25	0,00	0,00	0,00	0,00
B-10-4	-0,22	0,00	-0,22	0,00	0,00	0,00
B-10-5	0,01	0,30	0,01	0,09	0,08	-0,02
C-10-3	0,01	0,32	0,01	0,09	0,19	0,09
C-10-4	-0,12	-0,15	-0,15	0,05	0,04	-0,01
C-10-5	0,04	0,21	-0,01	-0,17	-0,22	-0,23

Tabela 7: Distância entre os resultados obtidos pelos algoritmos de formigas e as melhores soluções encontradas no experimento exato.

A Tabela 7 demostra que os resultados dos algoritmos de formigas estão próximos dos obtidos pelo solver. O algoritmo MS-ACS foi o melhor para esse conjunto de instâncias. Este foi capaz de encontrar os mesmos resultados encontrados pelo solver para 6 instâncias e melhorar a solução encontrada pelo solver para outras 7 instâncias.

A Tabela 8 apresenta um resumo da comparação entre as heurísticas *naive* e os algoritmos de formigas. Os dados apresentados na Tabela 8 incluem 72 instâncias simétricas e 72 assimétricas. Comparamos os melhores resultados obtidos por cada método. Os re-

sultados estão no formato $X \times Y$, onde X e Y representam o número de instâncias nas quais a heurística *naive* encontrou a melhor solução e o número de instâncias nas quais o algoritmo de formigas encontrou a melhor solução, respectivamente. As Tabelas 36 - 39 (A) apresentam resultados detalhados das heurísticas *naive*. As Tabelas 42 - 41 (B) mostram os resultados dos algoritmos de formigas.

	Assimétrico			Simétrico		
	AS	ACS	MS-ACS	AS	ACS	MS-ACS
$HN1$	7 x 65	11 x 61	0 x 72	3 x 69	7 x 65	0 x 72
$HN2$	1 x 71	6 x 66	0 x 72	0 x 72	2 x 70	0 x 72
$HN3$	0 x 72	1 x 71	0 x 72	1 x 71	1 x 71	1 x 71
$HN4$	0 x 72	0 x 72	0 x 72	0 x 72	1 x 71	0 x 72
$HN5$	1 x 71	2 x 70	0 x 72	1 x 71	1 x 71	0 x 72

Tabela 8: Comparativo entre a heurística *naive* e os algoritmos de formigas.

As heurísticas HN criam soluções tentando encontrar o caminho mais curto para atingir a cota mínima. Os algoritmos de formigas se comportaram de maneira diferente destas heurísticas. Durante o processo de resolução do problema, as formigas frequentemente selecionavam os arcos que viabilizavam que o maior número de solicitações de viagem fossem atendidas. Isto ocorreu devido as formigas atualizarem as trilhas de feromônios com base no custo rateado de cada arco pelos ocupantes do veículo. De acordo com a equação (3.28), o custo de cada arco em uma rota é reduzido, pois é rateado pelo número de passageiros no veículo. A Tabela 8 mostra que os algoritmos de formigas se saíram melhores que as versões da heurística *naive*.

A Tabela 9 apresenta a comparação entre os algoritmos de formigas. Nesta Tabela, observa-se que o MS-ACS foi o melhor algoritmo deste experimento. A adequação do MS-ACS ás diferentes facetas do problema resultou na expansão do processo de tomada de decisão, que inclui informações heurísticas diferentes. Neste sentido, pode-se dizer que o projeto das múltiplas estrategias heurísticas mostrou-se útil para diversificar a busca por melhores soluções.

	Assimétrico			Simétrico		
	AS	ACS	MS-ACS	AS	ACS	MS-ACS
AS	–	39 x 31	2 x 66	–	28 x 41	4 x 64
ACS	31 x 39	–	0 x 68	41 x 28	–	1 x 69
MS-ACS	66 x 2	68 x 0	–	64 x 4	69 x 1	–

Tabela 9: Comparativo entre os algoritmos de formigas

Também podemos deduzir da Tabela 9 que o AS funciona melhor que o ACS para as instâncias assimétricas. Nos cenários simétricos, o ACS é superior ao AS. Essas conclusões também são suportadas pelos dados da Tabela 10 que classificam os algoritmos de formigas com base no teste de Friedman (FRIEDMAN, 1937) e post-hoc de Nemenyi (NEMENYI, 1963). Em relação ao nível de significância de 0,05, os *p-values* apresentados na Tabela 10 mostram que o desempenho dos algoritmos de formigas não foram semelhantes, ou seja, a hipótese nula (NEMENYI, 1963) foi rejeitada em todos os casos.

	Assimétrico				Simétrico			
	p-value	AS	ACS	MS-ACS	p-value	AS	ACS	MS-ACS
$g10$	0,019034	b	b	a	0,012778	b	b	a
$g20$	0,000611	b	c	a	0,003096	a	b	a
$g30$	0,000816	b	c	a	0,004320	b	b	a
$g40$	0,000585	b	c	a	0,000912	b	b	a
$g50$	0,004828	b	b	a	0,001139	b	b	a
$g100$	0,000912	b	b	a	0,000585	c	b	a
$g200$	0,000300	b	c	a	0,000123	c	b	a
$g500$	0,000123	c	b	a	0,000300	c	b	a

Tabela 10: Valoração dos *p-values* do teste Friedman e rankeamento segundo post-hoc de Nemenyi para os algoritmos de formigas.

Para analisar a variabilidade de cada algoritmo heurístico, adotamos duas métricas em relação aos resultados produzidos pelos experimentos. A primeira métrica, ν, denota a média das porcentagens em relação ao número de vezes que um algoritmo de formigas encontrou o melhor resultado em 20 execuções independentes. As Tabelas 44 e 45 (B) mostram os dados referentes a esses resultados. A segunda métrica, Φ, é a distância relativa entre o custo das melhores soluções χ^b e a solução média χ^a de cada algoritmo de formigas. Para calcular Φ, usamos a equação 6.4.

$$\Phi = \frac{\chi^a}{\chi^b} - 1 \tag{6.4}$$

A Tabela 11 mostra o valor de ν e Φ. Nesta Tabela, concluímos que o *MS-ACS* é o melhor referente as métricas ν e Φ. Isto quer dizer que, dentre os algoritmos de formigas, este algoritmo foi o que mais vezes encontrou a melhor solução conhecida do banco de instâncias e que aquele que apresentou os resultados de melhor qualidade.

	Assimétrico			Simétrico		
Métrica	*AS*	*ACS*	*MS-ACS*	*AS*	*ACS*	*MS-ACS*
ν	4,1%	2.54%	8.95%	1,81%	0,69%	9,22%
Φ	0,178644	0,244575	0,0300307	0,227144	0,2336825	0,0589827

Tabela 11: Variabilidade dos algoritmos de formigas.

As Tabelas 12 e 13 apresentam o tempo médio em segundos gasto por cada heurística. As instâncias foram agrupadas por número de vértices. Nessas Tabelas, o símbolo $*$ indica que o solver concluiu sua execução devido ao consumo excessivo de memória primária. Os resultados demonstram que as heurísticas *naive* que utilizaram procedimentos exatos não se mostraram viáveis para resolver grandes instâncias. Também pode-se observar que o AS foi o algoritmo de formiga mais rápido e o HN4 foi a heurística ingênua mais rápida. Tabelas 36 - 39 (A) e Tabelas 42 - 41 (A) apresentam resultados detalhados sobre o tempo médio exigido por essas heurísticas.

n	AS	ACS	MS-ACS	HN1	HN2	HN3	HN4	HN5
10	0,05	0,06	0,12	5,62	0,37	5,07	0,01	0,01
20	0,10	0,13	0,27	45,95	1,49	42,42	0,02	0,02
30	0,18	0,24	0,49	141,59	3,75	138,94	0,03	0,038
40	0,28	0,38	0,73	306,63	8,49	293,99	0,09	0,07
50	0,40	0,56	5,41	299,49	24,47	294,02	0,12	0,13
100	8,13	13,51	29,17	$*$	415,16	$*$	0,18	0,23
200	22,94	51,92	112,58	$*$	$*$	$*$	0,66	1,01
500	40,53	75,72	127,83	$*$	$*$	$*$	4,42	22,07

Tabela 12: Tempo médio para resolver as instâncias simétricas do problema pelos algoritmos de formigas.

n	AS	ACS	MS-ACS	HN1	HN2	HN3	HN4	HN5
10	0,05	0,06	0,12	6,26	0,29	5,97	0,01	0,01
20	0,10	0,13	0,26	42,64	1,03	41,75	0,02	0,02
30	0,20	0,30	1,9	250,65	3,19	231,31	0,032	0,04
40	0,34	0,48	2,29	240,99	7,60	223,86	0,05	0,092
50	0,41	2,20	5,77	388,36	17,99	372,86	0,11	0,13
100	6,68	28,85	32,69	$*$	388,04	$*$	0,21	0,48
200	31,81	270,94	409,72	$*$	$*$	$*$	1,03	6,11
500	41,72	3477,13	3545,20	$*$	$*$	$*$	12,21	371,62

Tabela 13: Tempo médio para resolver as instâncias assimétricas do problema dos algoritmos de formigas e heurísticas *naive*.

6.5 Análises Comparativas entre os Algoritmos Evolucionários

Nesta seção, relatamos os resultados dos algoritmos evolucionários e comparamos seus resultados com os produzidos pelo solver de otimização e pelas cinco versões das heurísticas *naive*. Também comparamos os algoritmo evolucionários entre si.

A Tabela 14 apresenta a comparação entre os resultados dos algoritmos evolucionários e o solver para instâncias com 10 vértices. A lógica da análise comparativa e a métrica utilizada é mesma adotada na comparação dos algoritmos de formigas e o solver, descritos na seção (6.3). Sendo assim, a distância, υ, entre o melhor resultado encontrado por um dos algoritmos evolucionários e a solução obtida pelo solver é dada pela equação (6.3).

	Assimétrico			Simétrico		
Instância	AG	AM	AM-VC	AG	AM	AM-VC
A-10-3	0,11	0,11	0,11	0,07	0,07	0,00
A-10-4	0,29	0,29	0,12	0,24	0,00	0,00
A-10-5	-0,24	-0,24	-0,24	0,00	0,00	0,00
B-10-3	0,07	0,00	0,00	0,00	0,00	0,00
B-10-4	-0,22	-0,22	-0,22	0,00	0,00	0,00
B-10-5	0,01	0,01	0,01	0,03	0,03	-0,03
C-10-3	0,01	0,01	0,01	0,09	0,09	0,09
C-10-4	-0,15	-0,15	-0,15	-0,01	-0,01	-0,01
C-10-5	-0,01	-0,01	-0,01	-0,23	-0,18	-0,23

Tabela 14: Distância entre os resultados obtidos pelos algoritmos evolucionários e as melhores soluções obtidas no experimento exato.

A Tabela 14 demostra que os resultados dos algoritmos evolucionários estão próximos dos obtidos pelo solver. As adaptações do AG e do AM obtiveram um desempenho semelhante para esse conjunto de instâncias. O AM-VC se sobressaiu sobre as outras duas abordagens evolucionárias neste experimento, pois, este algoritmo foi capaz de encontrar os mesmos resultados encontrados pelo solver para 6 instâncias e melhorar a solução encontrada pelo solver para outras 7 instâncias.

A Tabela 15 apresenta um resumo da comparação entre as heurísticas *naive* e os algoritmos evolucionários. Os dados apresentados na Tabela 15 incluem as 144 instâncias do banco de testes. Comparamos os melhores resultados obtidos por cada método. Os resultados estão no formato $X \times Y$, onde X representa o número de instâncias nas quais a heurística *naive* se saiu melhor e Y denota o número de instâncias para as quais o respectivo algoritmo evolucionário foi mais eficiente. As Tabelas 36 - 39 (A) apresentam

resultados detalhados das heurísticas *naive*. As Tabelas 46 - 49 (C) mostram os resultados dos algoritmos evolucionários.

	Assimétrico			Simétrico		
	AG	AM	AM-VC	AG	AM	AV-VC
$HN1$	17 x 55	18 x 54	9 x 63	10 x 62	12 x 62	1 x 71
$HN2$	12 x 60	13 x 59	5 x 67	6 x 66	6 x 66	0 x 72
$HN3$	0 x 71	1 x 71	1 x 71	1 x 70	2 x 70	2 x 70
$HN4$	3 x 69	0 x 72	0 x 72	16 x 55	1 x 71	1 x 71
$HN5$	8 x 64	2 x 70	1 x 71	23 x 47	5 x 67	4 x 68

Tabela 15: Comparativo entre a heurística *naive* e os algoritmos evolucionários.

A Tabela 15 mostra que os algoritmos evolucionários conseguiram encontrar melhores resultados que as implementações da HN para a maioria dos casos, sendo o AM-VC, o algoritmo evolucionário que mais vezes se sobressaiu sobre as implementações da HN. Seguindo o mesmo formato da Tabela 15, a Tabela 16 sumariza a comparação dos algoritmos evolucionários entre si.

	Assimétrico			Simétrico		
	AG	AM	AM-VC	AG	AM	AM-VC
AG	–	34 x 29	7 x 58	–	24 x 41	3 x 63
AM	29 x 34	–	8 x 64	41 x 24	–	9 x 57
AM-VC	58 x 7	64 x 8	–	63 x 3	57 x 9	–

Tabela 16: Comparativo entre os algoritmos evolucionários

Observando a Tabela 16, nota-se que o AM-VC encontrou melhores resultados que o AG e o AM na grande maioria dos casos. Também pode-se observar que o AM teve um desempenho um pouco melhor que o AG nas instâncias simétricas, enquanto que o AG foi melhor nas instâncias assimétricas em comparação com o AM. Essas conclusões também são suportadas pelos dados das Tabelas 17 e 18.

A Tabela 17 mostra o valor de ν e Φ para os algoritmos evolucionários. A variabilidade dos algoritmos evolucionários foram calculadas utilizando-se as mesmas métricas adotadas na seção 6.4. Assim, a métrica ν denota a média das porcentagens em relação ao número de vezes que o respectivo algoritmo evolucionário encontrou o melhor resultado em 20 execuções independentes. Já Φ indica a distância relativa entre o custo das melhores soluções χ^b e a solução média χ^a de cada algoritmo evolucionário. Φ é calculado com base na equação 6.4. As Tabelas 50 e 51 (C) trazem os resultados específicos destas execuções.

A Tabela 18 apresenta a classificação dos algoritmos evolucionários com base no teste

	Assimétrico			Simétrico		
Métrica	*AG*	*AM*	*AM-VC*	*AG*	*AM*	*AM-VC*
ν	8,3%	8,3%	11,11%	5,5%	5,5%	11,52%
Φ	0,6631271	0,5763565	0,4097157	0,6113851	0,3989601	0,1874568

Tabela 17: Variabilidade dos algoritmos evolucionários.

de Friedman (FRIEDMAN, 1937) e post-hoc de Nemenyi (NEMENYI, 1963). As Tabelas 50 e 51 (C) trazem os resultados específicos destas execuções.

	Assimétrico				Simétrico			
	p-value	AG	AM	AM-VC	p-value	AG	AM	AM-VC
$g10$	0,017982	b	b	a	0,013991	b	b	a
$g20$	0,001502	b	b	a	0,021197	c	b	a
$g30$	0,000925	b	c	a	0,003241	c	b	a
$g40$	0,000693	b	c	a	0,001024	b	b	a
$g50$	0,003601	b	b	a	0,001481	b	b	a
$g100$	0,001786	b	c	a	0,000906	c	b	a
$g200$	0,002412	b	c	a	0,000247	c	b	a
$g500$	0,003434	b	b	a	0,002139	c	b	a

Tabela 18: Valoração dos *p-values* do teste Friedman e rankeamento segundo post-hoc de Nemenyi para os algoritmos evolucionários.

Observando a Tabela 17, concluí-se que, dentre os algoritmos evolucionários, o AM-VC é o melhor referente às métrica ν e Φ. O AM e o AG tiveram o mesmo desempenho quanto a métrica ν. Com relação a métrica Φ, o AM se sobressaiu sobre o AG. Em relação ao nível de significância de 0,05, os *p-values* apresentados na Tabela 18 mostram que o desempenho dos algoritmos evolucionários não foram semelhantes, ou seja, a hipótese nula (NEMENYI, 1963) foi rejeitada em todos os casos.

As Tabelas 19 e 20 apresentam o tempo médio em segundos gasto pelos algoritmos evolucionários. As instâncias foram agrupadas por número de vértices. O AG foi o algoritmo mais ráido dentre as abordagens evolucionárias. Já o AM-VC foi o algoritmo evolucionário que demandou mais tempo durante a execução. As Tabelas 48 - 49 (C) e 46 - 47 (E) apresentam resultados detalhados sobre o tempo médio exigido pelos algoritmos evolucionários.

n	AG	AM	AM-VC
10	0,53	0,30	0,25
20	1,37	1,92	1,49
30	5,96	5,24	4,55
40	12,34	20,29	16,26
50	22,09	51,44	46,33
100	87,07	154,87	134,21
200	231,69	1198,73	1320,74
500	2349,21	5360,39	7476,68

Tabela 19: Tempo médio para resolver as instâncias simétricas do problema pelos algoritmos evolucionários.

n	AG	AM	AM-VC
10	0,18	0,38	0,33
20	3,90	2,57	2,06
30	9,36	12,72	10,59
40	15,45	30,05	25,45
50	22,36	50,76	40,29
100	40,91	133,32	155,57
200	146,98	479,25	453,25
500	1556,34	5284,30	7926,46

Tabela 20: Tempo médio para resolver as instâncias assimétricas do problema pelos algoritmos evolucionários.

6.6 Análises Comparativas das meta-heurísticas ILS e GRASP

Nesta seção, relatamos os resultados das meta-heurísticas ILS e GRASP, e, além de compará-las ente si, comparamos também seus resultados com os produzidos pelas seguintes abordagens:

- *Solvers* de otimização;
- Diferentes implementações da HN;
- Algoritmo de evolucionário mais promissor, o AM-VC.
- Algoritmo de formigas mais promissor, o MS-ACS.

A Tabela 21 apresenta a comparação entre os resultados das meta-heurísticas ILS, GRASP e os melhores resultados obtidos no experimento exato para instâncias com 10 vértices. A lógica da análise comparativa e a métrica utilizada é mesma adotada na comparação dos algoritmos de formigas e o solver, descritos na seção 6.4. Sendo assim, a distância v entre o melhor resultado encontrado por uma destas três meta-heurísticas e a solução obtida pelo solver é dada pela equação (6.3). Nesta métrica, valores positivos mostram que a solução obtida pelo solver foi melhor do que a obtida pelas meta-heurísticas ILS e GRASP. Valores negativos denotam o oposto.

	Assimétrico		Simétrico	
Instância	ILS	GRASP	ILS	GRASP
A-10-3	0,17	0,11	0,00	0,00
A-10-4	0,12	0,12	0,00	0,00
A-10-5	-0,24	-0,24	0,00	0,00
B-10-3	0,00	0,00	0,00	0,00
B-10-4	-0,22	-0,22	0,00	0,00
B-10-5	0,01	0,01	-0,03	-0,03
C-10-3	0,01	0,01	0,11	0,09
C-10-4	-0,15	-0,15	-0,01	-0,01
C-10-5	-0,01	-0,01	-0,23	-0,23

Tabela 21: Distância entre os resultados obtidos pelas abordagens ILS, GRASP e o solver.

A Tabela 21 demonstra a proximidade dos resultados dos algoritmos ILS e GRASP para com aqueles obtidos pelo solver. O algoritmo GRASP teve o melhor desempenho

para o conjunto de instâncias assimétricas. Quanto ao desempenho no conjunto de instâncias simétricas, o desempenho destas duas meta-heurísticas foi semelhante. O algoritmo GRASP obteve 6 soluções de melhor qualidade que as do solver e se equiparou ao mesmo em 3 outros casos de teste. Por fim, o ILS foi melhor que o solver em 6 ocasiões e se equiparou em 3 ocasiões.

A Tabela 22 mostra a análise comparativa entre os resultados das heurísticas *naive* e os algoritmos ILS e GRASP. Os resultados desta Tabela seguem o formato $X \times Y$, onde X e Y representam o número de instâncias nas quais a heurística *naive* encontrou a melhor solução e o número de instâncias nas quais uma destas duas meta-heurísticas encontrou a melhor solução, respectivamente. As Tabelas 54 - 53 (D) mostram os resultados das meta-heurísticas ILS e GRASP.

	Assimétrico		Simétrico	
	ILS	GRASP	ILS	GRASP
$HN1$	2 x 70	7 x 60	0 x 72	5 x 67
$HN2$	2 x 70	11 x 61	0 x 72	1 x 71
$HN3$	1 x 71	2 x 70	2 x 70	2 x 70
$HN4$	0 x 72	1 x 71	1 x 71	4 x 68
$HN5$	1 x 71	4 x 68	2 x 70	12 x 58

Tabela 22: Comparativo entre a heurística *naive* e meta-heurísticas ILS e GRASP.

A Tabela 22 demonstra que os algoritmos ILS e GRASP se saíram melhores que as versões da heurística *naive* na grande maioria das instâncias. A variabilidade destas duas meta-heurísticas foi calculada por meio das mesmas métricas adotadas na seção 6.4. A Tabela 23 mostra o valor de ν e Φ para estas duas meta-heurísticas. As Tabelas 56 e 57 (D) trazem os resultados específicos destas execuções.

	Assimétrico		Simétrico	
Métrica	ILS	$GRASP$	ILS	$GRASP$
ν	7,77%	4,02%	5,21%	4,93%
Φ	0,2334926	0,5562408	0,0857132	0,3316313

Tabela 23: Variabilidade das meta-heurísticas ILS e GRASP.

Na comparação entre ILS e GRASP, ao se observar a Tabela 23, concluí-se que o ILS é o melhor em ambas as métricas do que o GRASP. Isto ocorreu tanto para o grupo de instâncias simétricas quanto para o grupo de assimétricas.

A Tabela 25 apresenta a comparação entre os algoritmos ILS, GRASP, AM-VC e MS-ACS para as instâncias assimétricas. Já a Tabela 24 apresenta a comparação entre estes

mesmos algoritmos para as instâncias simétricas Em ambas as Tabelas, são expostos os resultados do teste de Friedman (FRIEDMAN, 1937) e post-hoc de Nemenyi (NEMENYI, 1963). O nível de significância adotado é de 0,05. Neste experimento, a hipótese nula (NEMENYI, 1963) é rejeitada em todos os casos, pois os *p-values* apresentados demostram que o desempenho dos respectivos algoritmos não são semelhantes. Mais uma vez a meta-heurística MS-ACS se sobressaiu sobre as demais. O algoritmo que obteve o pior desempenho foi o GRASP.

	Simétrico				
	p-value	ILS	GRASP	AM-VC	MS-ACS
$g10$	0,993456	a	a	a	a
$g20$	0,000803	b	c	b	a
$g30$	0,000506	b	c	b	a
$g40$	0,000685	b	c	b	a
$g50$	0,003715	b	c	b	a
$g100$	0,001023	b	c	c	a
$g200$	0,000403	b	c	c	a
$g500$	0,000346	b	c	c	a

Tabela 24: Valoração dos *p-values* do teste Friedman e rankeamento segundo post-hoc de Nemenyi para as meta-heurísticas ILS, GRASP, AM-VC e MS-ACS em grupos de instâncias simétricas.

	Assimétrico				
	p-value	ILS	GRASP	AM-VC	MS-ACS
$g10$	0,985230	a	a	a	a
$g20$	0,000803	b	c	b	a
$g30$	0,000506	b	c	b	a
$g40$	0,000685	b	c	b	a
$g50$	0,003715	b	b	b	a
$g100$	0,001023	b	c	b	a
$g200$	0,000403	b	c	c	a
$g500$	0,000346	b	b	c	a

Tabela 25: Valoração dos *p-values* do teste Friedman e rankeamento segundo post-hoc de Nemenyi para as meta-heurísticas ILS, GRASP, AM-VC e MS-ACS em grupos de instâncias assimétricas.

As Tabelas 26 e 27 apresentam o tempo médio em segundos gasto pelas meta-heurísticas ILS e GRASP. As instâncias foram agrupadas por número de vértices. Observa-se nestas Tabelas que o GRASP consumiu menos tempo que o ILS. As Tabelas 52 - 53 (D) e 54 - 55 (D) apresentam resultados detalhados sobre o tempo médio exigido por estes algoritmos. Também foi constatado que o uso do *Path Relinking* melhorou em média $13,8\%$ o valor das soluções e resultou no aumento de $22,9\%$ do tempo computacional do GRASP.

n	ILS	GRASP
10	0,02	0,02
20	0,03	0,04
30	0,26	0,14
40	0,34	0,29
50	1,09	0,44
100	186,61	163,15
200	1826,69	1198,73
500	6618,66	5667,39

Tabela 26: Tempo médio para resolver as instâncias simétricas do problema pelas metaheurísticas ILS e GRASP.

n	ILS	GRASP
10	0,01	0,02
20	0,04	0,07
30	0,59	0,29
40	0,83	0,54
50	2,36	0,76
100	1971,12	144,21
200	3648,23	2696,25
500	8197,72	5045,42

Tabela 27: Tempo médio para resolver as instâncias assimétricas do problema pelas metaheurísticas ILS e GRASP.

6.7 Análises Comparativas entre as Meta-heurísticas Híbridas

Nesta seção, relatamos os resultados dos algoritmos hibridizados e comparamos seus resultados entre si para concluir sobre a eficiência destas abordagens. Comparamos também os algoritmos hibridizados com a meta-heurística mais promissora, o MS-ACS, para concluir sobre a eficiência das hibridizações. A Tabela 28 apresenta a comparação entre os resultados dos algoritmos hibridizados e os melhores resultados obtidos pelo experimento exato para instâncias assimétricas com 10 vértices. A Tabela 29 apresenta a comparação entre os resultados destes algoritmos e os melhores resultados obtidos pelo experimento exato para instâncias simétricas com 10 vértices. A métrica v utilizada para comparar os resultados dos algoritmos hibridizados e os resultados exatos é dado pela equação (6.3).

	Assimétrico		
Instância	GRASP-ILS	AV-VC-ILS	MS-ACS-ILS
A-10-3	0,17	0,11	0,11
A-10-4	0,12	0,12	0,12
A-10-5	-0,24	-0,24	-0,24
B-10-3	0,00	0,00	0,00
B-10-4	-0,22	-0,22	-0,22
B-10-5	0,01	0,01	0,01
C-10-3	0,01	0,01	0,01
C-10-4	-0,15	-0,15	-0,15
C-10-5	-0,01	-0,01	-0,01

Tabela 28: Distância entre os resultados obtidos pelos algoritmos hibridizados e os melhores resultados obtidos no experimento exato para instâncias assimétricas.

	Simétrico		
Instância	GRASP-ILS	AV-VC-ILS	MS-ACS-ILS
A-10-3	0,00	0,00	0,00
A-10-4	0,00	0,00	0,00
A-10-5	0,00	0,00	0,09
B-10-3	0,00	0,00	0,00
B-10-4	0,00	0,00	0,00
B-10-5	-0,03	-0,03	-0,03
C-10-3	0,11	0,09	0,09
C-10-4	-0,01	-0,01	-0,01
C-10-5	-0,23	-0,23	-0,23

Tabela 29: Distância entre os resultados obtidos pelos algoritmos hibridizados e os melhores resultados obtidos no experimento exato para instâncias simétricas.

As Tabelas 28 e 29 demostram que os resultados dos algoritmos hibridizados estão próximos dos obtidos pelo solver. Nesta Tabela, é possível notar que os algoritmos hibridizados apresentaram o mesmo desempenho para instâncias de tamanho 10.

A Tabela 30 apresenta a comparação dos algoritmos hibridizados e a meta-heurística MS-ACS em instâncias assimétricas. A Tabela 31 traz a mesma comparação para as instâncias simétricas. Em ambas as Tabelas, os algoritmos são classificados com base no teste de Friedman (FRIEDMAN, 1937) e post-hoc de Nemenyi (NEMENYI, 1963). Utilizou-se o nível de significância, denotado por *p-value*, padrão deste teste, que é de 0,05.

	Assimétrico				
	p-value	MS-ACS	GRASP-ILS	AM-VC-ILS	MS-ACS-ILS
$g10$	0,997890	a	a	a	a
$g20$	0,001829	b	b	c	a
$g30$	0,001987	b	b	c	a
$g40$	0,001585	b	b	c	a
$g50$	0,014929	b	b	b	a
$g100$	0,001943	a	b	b	a
$g200$	0,001432	a	c	b	a
$g500$	0,001245	a	c	b	a

Tabela 30: Valoração dos *p-values* do teste Friedman e rankeamento segundo post-hoc de Nemenyi para os algoritmos hibridizados em instâncias assimétricas.

	Simétrico				
	p-value	MS-ACS	GRASP-ILS	AM-VC-ILS	MS-ACS-ILS
$g10$	0,992451	a	a	a	a
$g20$	0,002931	a	b	b	a
$g30$	0,002058	b	b	b	a
$g40$	0,001336	a	b	c	a
$g50$	0,017038	b	c	b	a
$g100$	0,002379	a	b	b	a
$g200$	0,001578	a	c	b	a
$g500$	0,001834	a	c	b	a

Tabela 31: Valoração dos *p-values* do teste Friedman e rankeamento segundo post-hoc de Nemenyi para os algoritmos hibridizados em instâncias simétricas.

Em relação ao nível de significância de 0,05, os *p-values* apresentados nas Tabelas 30 e Tabelas 31 demonstram que o desempenho dos algoritmos hibridizados não foram semelhantes, ou seja, a hipótese nula (NEMENYI, 1963) foi rejeitada em todos os casos.

A variabilidade dos algoritmos hibridizados foram calculadas utilizando-se as mesmas métricas adotadas na seção 6.4. As Tabelas 32 e 33 mostram o valores de ν e Φ destas abordagens para instâncias assimétricas e simétricas, respectivamente. As Tabelas 62 e 63 (E) trazem os resultados específicos destas execuções.

	Assimétrico		
Métrica	GRASP-ILS	AM-VC-ILS	MS-ACS-ILS
ν	6,95%	13,88%	17,5%
Φ	0,1668403	0,2154056	0,0334891

Tabela 32: Variabilidade dos algoritmos hibridizados em instâncias assimétricas.

	Simétrico		
Métrica	GRASP-ILS	AM-VC-ILS	MS-ACS-ILS
ν	7,77%	14,83%	20,83%
Φ	0,1121478	0,1138596	0,0334159

Tabela 33: Variabilidade dos algoritmos hibridizados em instâncias simétricas.

Observando as Tabelas 32 e 33, concluí-se que o MS-ACS-ILS é o melhor referente a ambas as métricas. Neste experimento, o GRASP-ILS obteve um desempenho melhor que o AM-VC-ILS para a metríca Φ em instâncias assimétricas. Na métrica ν, o AM-VC-ILS se saiu melhor que o GRASP-ILS tanto para instâncias quanto para as simétricas.

As Tabelas 34 e 35 apresentam o tempo médio em segundos gasto pelos algoritmos hibridizados. O algoritmo hibridizado que consumiu menos tempo foi o MS-ACS-ILS. As Tabelas 60 - 61 (E) e 58 - 59 (E) apresentam resultados detalhados sobre o tempo médio exigido por estes algoritmos.

n	GRASP-ILS	AM-VC-ILS	MS-ACS-ILS
10	0,09	3,5	0,12
20	0,29	8,14	1,29
30	4,36	55,14	17,15
40	31,34	62,29	29,26
50	83,96	137,44	50,93
100	754,61	1579,15	460,14
200	10082,69	8223,73	3155,74
500	53062,66	45639,39	42427,68

Tabela 34: Tempo médio para resolver as instâncias assimétricas do problema pelos algoritmos hibridizados.

n	GRASP-ILS	AM-VC-ILS	MS-ACS-ILS
10	0,29	3,78	0,50
20	0,37	13,57	1,08
30	2,29	32,29	6,52
40	13,83	108,54	9,25
50	33,69	235,76	23,16
100	121,12	384,21	64,57
200	850,23	628,25	361,25
500	7862,72	11140,42	2880,46

Tabela 35: Tempo médio para resolver as instâncias simétricas do problema pelos algoritmos hibridizados.

Os melhores resultados encontrados pelos algoritmos propostos neste estudo podem ser consultados nas Tabelas 64 - 65 (F).

7 Considerações Finais

Este trabalho apresentou o Problema do Caixeiro Viajante com Cota, Múltiplos Passageiros, Transporte Incompleto e Tempo de Coleta, um problema inédito na literatura. O PCV-CPTIT pode ser um ferramenta de apoio a tomada de decisão em sistemas de mobilidade sob demanda. Estudos realizados sob o escopo da mobilidade sob demanda são orientados para maximizar o número de solicitações de viagem atendidas, consideram penalidades por solicitações de viagens não atendidas e possuem limitações quanto a capacidade de veículos. O modelo proposto não é afetado por estas restrições operacionais e ainda fornece suporte ao desembarque de passageiros em destinos alternativos.

O modelo incentiva a mobilidade compartilhada, em vez dos serviços de transporte tradicionais que seriam subutilizados. Exemplos práticos descritos neste trabalho, elucidam o potencial do modelo proposto para auxiliar o planejamento de rotas no sentido de maximizar índices de ocupação de veículos em centros urbanos e rodovias, minimizando custos com transporte, emissão de poluentes, engarrafamentos e carros estacionados.

A formulação matemática do problema possui função objetivo e restrições não lineares. Como o PCV-CPTIT é um problema original, sua formulação é preliminar e é natural que, no futuro, o problema seja reformulado. Várias instâncias de *benchmarking* foram produzidas para apoiar o processo de otimização por abordagens heurísticas. Um sistema de classificação de instâncias foi desenvolvido para que o conceito de dificuldade a ser enfrentado durante o processo de resolução não fosse associado apenas ao número de vértices. As principais dificuldades enfrentadas para se implementar o algoritmo de geração de instâncias foram:

- Correlacionar e anti-correlacionar as diferentes vertentes do problema de forma a desacoplar a rota ótima do ótimo global;
- Calcular a tarifa máxima que cada passageiro está disposto a pagar;
- Calcular o tempo máximo que cada passageiro está disposto a permanecer no veículo;

- Calcular a penalidade que o caixeiro terá por desembarcar um passageiro em um destino alternativo.

Como resultados satisfatórios foram alcançados pelo solver apenas para instâncias com tamanho 10 devido ao alto número de variáveis e restrições do modelo, heurísticas *naive* foram implementadas para obter referências para outras instâncias de *benchmarking*. Algumas meta-heurísticas com aplicações consagradas a problemas semelhantes ao PCV-CPTIT foram implementadas. Um desses algoritmos, chamado MS-ACS, foi projetado para adaptar o comportamento semi-guloso do ACS original ao PCV-CPTIT. O MS-ACS emprega várias estratégias heurísticas. Este algoritmo, juntamente com a sua versão hibridizada MS-ACS-ILS, forneceram os melhores resultados referentes a um experimento com 144 instâncias de até 500 vértices.

Durante o processo de implementação das abordagens meta-heurísticas, as principais dificuldades observadas foram:

- Soluções com tamanho variável;
- Restrição de transporte incompleto;
- Permear os espaço de soluções em tempo razoável e se desvencilhar de ótimos locais;
- Dimensão dantesca do espaço de soluções viáveis.

A pesquisa apresentada nesta tese pode ser estendida em múltiplas direções. Primeiro, seria interessante .

7.1 Trabalhos Futuros

A pesquisa apresentada nesta tese pode ser estendida em múltiplas direções, dentre as quais destacam-se:

1. Modelar o transporte multi-modal, i. e., expandir a modelagem do problema para suportar a utilização de outras formas de transporte durante o percurso, a exemplo do transporte público, carros elétricos ou até mesmo veículos autônomos;
2. Elaborar variações do PCV-CPTIT com características de demandas dinâmicas, aproximando ainda mais o problema da realidade;

3. Uma vez que a restrição de transporte incompleto de passageiros é inédita na literatura, uma ideia promissora é aplicadá-la em diversas reformulações de problemas de otimização que consideram o gerenciamento de solicitações de viagem.

4. Aplicar outros modelos e estratégias de solução exatos mais eficientes que possam ser adaptadas ao problema;

5. Estudar e adaptar outras meta-heurísticas ao PCV-CPTIT, tal como *Simulated Annealing*, Algoritmos Transgenéticos e Algoritmos Científicos;

6. Implementar para o algoritmo HBL-Mo um mecanismo de seleção de operadores de vizinhança baseado em roleta para induzir a heurística a heurística escolher o operador que se mostrar mais promissor durante as execuções;

7. Pesquisar e analisar estratégias que diminuam o tempo de execução das buscas locais propostas

8. Visto que o MS-ACS foi o algoritmo mais promissor, outra ideia promissora é a implementação de outros algoritmos baseados em formigas com boa aplicabilidade em problemas correlatos ao em estudo. A extensão do projeto de implementação MS-ACS com técnicas de computação paralela (SKINDEROWICZ, 2016) é outra oportunidade interessante para pesquisas futuras;

7.2 Produção Cientifica Associada à Pesquisa

A presente pesquisa, até o momento da edição do atual texto, resultou nos seguintes trabalhos publicados:

- O modelo matemático, o experimento a cerca da sua implementação no *solver* Gurobi, o projeto de implementação das heurísticas auxiliares e dos algoritmos de formigas bem seus resultados foram objeto de publicação internacional na revista *Computers & Operations Research* com o título *Quota Travelling Salesman Problem with Passengers, Incomplete Ride and Collection Time Optimization by Ant-based Algorithms* (SILVA et al., 2020);

- O experimento dos algoritmos de formigas foi melhorado com a adaptação do algoritmo MAX MIN Ant System. Este estudo algorítmico, intitulado como *Multi-Strategy MAX-MIN Ant System for Solving Quota Travelling Salesman Problem*

with Passengers, Incomplete Ride and Collection Time, foi aceito para publicação como capítulo do livro *Operations Management - Emerging Trend in the Digital Era* da editora *IntechOpen*.

Referências

AGATZ, N. et al. Optimization for dynamic ride-sharing: A review. *European Journal of Operational Research*, v. 223, n. 2, p. 295 – 303, 2012.

AL-BEHADILI, H. N. K.; KU-MAHAMUD, K.; SAGBAN, R. Hybrid ant colony optimization and iterated local search for rules-based classification. *J. Theor. Appl. Inf. Technol*, v. 98, n. 04, p. 657–671, 2020.

ALMEIDA, M. W. S. *Utilização de Algoritmos Genéticos para Montagem de Horários Acadêmicos com Foco na Blocagem de Horários*. Dissertação (Mestrado) — Universidade Federal do Rio Grande do Norte, 2015.

ALONSO-MORA, J. et al. On-demand high-capacity ride-sharing via dynamic trip-vehicle assignment. *Proceedings of the National Academy of Sciences*, v. 114, n. 3, p. 462–467, 2017.

APPLEGATE, D.; COOK, W.; ROHE, A. Chained lin-kernighan for large traveling salesman problems. *INFORMS Journal on Computing*, v. 15, p. 82–92, 02 2003.

AWERBUCH, B. et al. New approximation guarantees for minimum-weight k-trees and prize-collecting salesmen. *Annals of Physics*, v. 28, p. 254—-262, 1998.

BALAS, E. *The Prize Collecting Traveling Salesman Problem and Its Applications*. [S.l.: s.n.], 2006. 663-695 p.

BALAS, E.; VAZACOPOULOS, A. Guided local search with shifting bottleneck for job shop scheduling. *Management Science*, v. 44, p. 262–275, 02 1998.

BANIAMERIAN, A.; BASHIRI, M.; TAVAKKOLI-MOGHADDAM, R. Modified variable neighborhood search and genetic algorithm for profitable heterogeneous vehicle routing problem with cross-docking. *Applied Soft Computing*, v. 75, p. 441 – 460, 2019. ISSN 1568-4946. Disponível em: <http://www.sciencedirect.com/science/article/pii/S1568494618306616>.

BAO, L. N. L.; LE, D. H.; NGUYEN, D. A. Application of combinatorial optimization in logistics. p. 329–334, Nov 2018.

BAUM, E. *Iterated descent: A better algorithm for local search in combinatorial optimization problems*. 1986. Technical report, Caltech, Pasadena, CA. manuscript.

BAUM, E. B. Towards practical 'neural' computation for combinatorial optimization problems. *AIP Conference Proceedings*, v. 151, n. 1, p. 53–58, 1986.

BLUM, C. Hybrid metaheuristics in combinatorial optimization: A tutorial. In: DEDIU, A.-H.; MARTÍN-VIDE, C.; TRUTHE, B. (Ed.). *Theory and Practice of Natural Computing*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012. p. 1–10. ISBN 978-3-642-33860-1.

BRANDAO, J. Iterated local search algorithm with ejection chains for the open vehicle routing problem with time windows. *Computers & Industrial Engineering*, v. 120, p. 146 – 159, 2018. ISSN 0360-8352. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0360835218301682>.

C-PLEX. *C-Plex Optimizer*. 2020. `https://www.ibm.com/br-pt/analytics/cplex-optimizer`. Accessed in: 2020-03-03.

CACCHIANI, V. et al. An iterated local search algorithm for the pollution traveling salesman problem. In: _____. [S.l.: s.n.], 2018. p. 83–91. ISBN 978-3-030-00472-9.

CALHEIROS, Z. S. A. *O problema do caixeiro viajante com passageiros*. Dissertação (Mestrado) — Universidade Federal do Rio Grande do Norte, 2017.

CHAMI, Z. A. et al. An advanced grasp-hga combination to solve a multi-period pickup and delivery problem. *Expert Systems with Applications*, v. 105, p. 262 – 272, 2018. ISSN 0957-4174.

CHEN, Z.; LIU, X. C.; WEI, R. Agent-based approach to analyzing the effects of dynamic ridesharing in a multimodal network. *Computers, Environment and Urban Systems*, 2018. ISSN 0198-9715.

CORDEAU, J. F.; LAPORTE, G. A tabu search heuristic for the static multi-vehicle dial-a-ride problem. *Transportation Research Part B: Methodological*, v. 37, p. 579–594, 07 2003.

CORDEAU, J. F.; LAPORTE, G. The dial-a-ride problem: models and algorithms. *Annals of Operations Research*, v. 153, p. 29–46, 2007.

CROES, G. A. A method for solving traveling salesman problems. *Operations Research*, v. 6, p. 791–812, 1958.

DAWKINS, R. *The Selfish Gene*. [S.l.]: Princeton University Press, 2016. 140–142 p.

DONG, Y. et al. An empirical study on travel patterns of internet based ride-sharing. *Transportation Research Part C: Emerging Technologies*, v. 86, p. 1 – 22, 2018.

DORIGO, M.; BONABEAU, E.; THERAULAZ, G. Ant algorithms and stigmergy. *Future Generation Computer Systems*, v. 16, n. 8, p. 851 – 871, 2000. ISSN 0167-739X. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0167739X0000042X>.

DORIGO, M.; GAMBARDELLA, L. M. Ant colony system: a cooperative learning approach to the traveling salesman problem. *IEEE Transactions on Evolutionary Computation*, v. 1, p. 53–66, 1997.

DORIGO, M.; MANIEZZO, V.; COLORNI, A. Ant system: optimization by a colony of cooperating agents. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, v. 26, n. 1, p. 29–41, 1996.

DORIGO, M.; STÜTZLE, T. *Handbook of Metaheuristics*. [S.l.]: Boston: Springer-Verlag, 2003. ISBN 978-1-4020-7263-5.

DORIGO, M.; STÜTZLE, T. Ant colony optimization: overview and recent advances. In: *Handbook of metaheuristics*. [S.l.]: Springer, 2019. p. 311–351.

ERDOGAN, G.; CORDEAU, J.-F.; LAPORTE, G. The attractive traveling salesman problem. *European Journal of Operational Research*, v. 203, n. 1, p. 59 – 69, 2010. ISSN 0377-2217.

FAGNANT, D. J.; KOCKELMAN, K. M. Dynamic ride-sharing and fleet sizing for a system of shared autonomous vehicles in austin, texas. *Transportation*, v. 45, n. 1, p. 143–158, Jan 2018. ISSN 1572-9435.

FARHAN, J.; CHEN, T. D. Impact of ridesharing on operational efficiency of shared autonomous electric vehicle fleet. *Transportation Research Part C: Emerging Technologies*, v. 93, p. 310 – 321, 2018. ISSN 0968-090X.

FEILLET, D.; DEJAX, P.; GENDREAU, M. Traveling salesman problems with profits. *Transportation Science*, INFORMS, v. 39, n. 2, p. 188–205, maio 2005. ISSN 1526-5447.

FEO, T. A.; RESENDE, M. G. A probabilistic heuristic for a computationally difficult set covering problem. *Operations Research Letters*, v. 8, n. 2, p. 67—-71, 1989.

FRIEDMAN, M. The use of ranks to avoid the assumption of normality implicit in the analysis of variance. *Journal of the American Statistical Association*, v. 32, p. 675—701, 1937.

GENDREAU, M.; LAPORTE, G.; SEMET, F. A tabu search heuristic for the undirected selective travelling salesman problem. *European Journal of Operational Research*, v. 106, n. 2, p. 539 – 545, 1998. ISSN 0377-2217.

GOLDBERG, D. Genetic algorithms in search, optimization, and machine learning, addison-wesley, reading, ma, 1989. *NN Schraudolph and J*, v. 3, n. 1, 1989.

GUNAWAN, A.; LAU, H. C.; VANSTEENWEGEN, P. Orienteering problem: A survey of recent variants, solution approaches and applications. *European Journal of Operational Research*, v. 255, n. 2, p. 315 – 332, 2016. ISSN 0377-2217.

GUROBI. *Gurobi Optimizer*. 2018. `http://www.gurobi.com/products/gurobi-optimizer`. Accessed in: 2018-03-12.

HADELI et al. Multi-agent coordination and control using stigmergy. *Computers in Industry*, v. 53, n. 1, p. 75 – 96, 2004. ISSN 0166-3615. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0166361503001234>.

HEIKKILA, S. *Mobility as a Service – A Proposal for Action for the Public Administration, Case Helsinki*. Dissertação (Mestrado) — Aalto University, 2014.

HERBAWI, W.; WEBER, M. *2011 IEEE 23rd International Conference on Tools with Artificial Intelligence*. [S.l.: s.n.], 2011. 282-288 p. ISSN 1082-3409.

HO, S. C. et al. A survey of dial-a-ride problems: Literature review and recent developments. *Transportation Research Part B: Methodological*, v. 111, p. 395 – 421, 2018.

HONG, I.; KAHNG, A. B.; MOON, B.-R. Improved large-step markov chain variants for the symmetric tsp. *Journal of Heuristics*, v. 3, p. 63–81, 1997.

HOU, L.; LI, D.; ZHANG, D. Ride-matching and routing optimisation: Models and a large neighbourhood search heuristic. *Transportation Research Part E: Logistics and Transportation Review*, v. 118, p. 143 – 162, 2018.

HUANG, S. C.; JIAU, M. K.; LIU, Y. P. An ant path-oriented carpooling allocation approach to optimize the carpool service problem with time windows. *IEEE Systems Journal*, p. 1–12, 2018. ISSN 1932-8184.

JACKSON, D. E.; RATNIEKS, F. L. Communication in ants. *Current biology*, Elsevier, v. 16, n. 15, p. R570–R574, 2006.

JOHNSON, D. S. Local optimization and the traveling salesman problem. In: PATERSON, M. S. (Ed.). *Automata, Languages and Programming*. Berlin, Heidelberg: Springer Berlin Heidelberg, 1990. p. 446–461. ISBN 978-3-540-47159-2.

KARP, H. M. On the computational complexity. *Networks*, v. 5, p. 45–68, 1975.

KOCZY, L. T.; FOLDESI, P.; TUU-SZABO, B. Enhanced discrete bacterial memetic evolutionary algorithm - an efficacious metaheuristic for the traveling salesman optimization. *Information Sciences*, v. 460-461, p. 389 – 400, 2018. ISSN 0020-0255.

LEVIN, M. W. et al. A general framework for modeling shared autonomous vehicles with dynamic network-loading and dynamic ride-sharing application. *Computers, Environment and Urban Systems*, v. 64, p. 373 – 383, 2017. ISSN 0198-9715.

LIAW, R.; CHANG, Y.; TING, C. *Advances in Swarm Intelligence*. [S.l.]: Springer International Publishing, 2017. 293–300 p. ISBN 978-3-319-61824-1.

LIRA, V. M. de et al. Boosting ride sharing with alternative destinations. *IEEE Transactions on Intelligent Transportation Systems*, p. 1–11, 2018.

LIU, T.; JIANG, Z.; GENG, N. A memetic algorithm with iterated local search for the capacitated arc routing problem. *International Journal of Production Research*, Taylor & Francis, v. 51, n. 10, p. 3075–3084, 2013.

LOKHANDWALA, M.; CAI, H. Dynamic ride sharing using traditional taxis and shared autonomous taxis: A case study of nyc. *Transportation Research Part C: Emerging Technologies*, v. 97, p. 45 – 60, 2018. ISSN 0968-090X.

LÓPEZ-IBÁÑEZ, M. et al. The irace package: Iterated racing for automatic algorithm configuration. *Operations Research Perspectives*, v. 3, p. 43–58, 2016.

LOURENÇO, H. R.; MARTIN, O. C.; STÜTZLE, T. Iterated local search: Framework and applications. In: _____. *Handbook of Metaheuristics*. Cham: Springer International Publishing, 2019. p. 129–168. ISBN 978-3-319-91086-4.

MARTIN, O.; OTTO, S. Combining simulated annealing with local search heuristics. *Annals of Operations Research*, v. 63, 04 1999.

MASSON, R. et al. Optimization of a city logistics transportation system with mixed passengers and goods. *EURO Journal on Transportation and Logistics*, v. 6, n. 1, p. 81–109, Mar 2017. ISSN 2192-4384.

MCMENEMY, P.; VEERAPEN, N.; OCHOA, G. How perturbation strength shapes the global structure of tsp fitness landscapes. Springer International Publishing, Cham, p. 34–49, 2018.

MILLER, C. E.; TUCKER, A. W.; ZEMLIN, R. A. Integer programming formulation of traveling salesman problems. *Journal of the ACM*, v. 7, n. 4, p. 326–329, 1960.

MORAES, R. F. D. et al. Implementation of a rvnd, vns, ils heuristic for the traveling car renter problem. p. 1–8, July 2018.

MOSCATO, P.; PLATA, L.; NORMAN, M. A memetic approach for the traveling salesman problem implementation of a computational ecology for combinatorial optimization on message-passing systems. v. 1, 07 1999.

NEMENYI, P. B. *Distribution-free multiple comparisons*. Tese (Doutorado) — Princeton University, 1963.

NERI, F.; COTTA, C. Memetic algorithms and memetic computing optimization: A literature review. *Swarm and Evolutionary Computation*, v. 2, p. 1 – 14, 2012. ISSN 2210-6502. Disponível em: <http://www.sciencedirect.com/science/article/pii/S2210650211000691>.

NOURINEJAD, M.; ROORDA, M. A dynamic carsharing decision support system. *Transportation Research*, v. 66, p. 36–50, 2014.

PARRAGH, S.; DOERNER, K.; HARTL, R. A survey on pickup and delivery problems: Part ii: Transportation between pickup and delivery locations. *Journal für Betriebswirtschaft*, v. 58, p. 81–117, 06 2008.

PRAIS, M.; RIBEIRO, C. C. Reactive grasp: An application to a matrix decomposition problem in tdma traffic assignment. *INFORMS Journal on Computing*, v. 12, n. 3, p. 164–176, 2000.

PRINS, C. A grasp× evolutionary local search hybrid for the vehicle routing problem. In: *Bio-inspired algorithms for the vehicle routing problem*. [S.l.]: Springer, 2009. p. 35–53.

REINELT, G. Tsplib - a traveling salesman problem library. *ORSA Journal on Computing*, v. 3, n. 4, p. 376–384, 1991.

RESENDE, M.; RIBEIRO, C. C. *Optimization by GRASP*. New York, USA: Springer-Verlag New York, 2016. ISBN 978-1-4939-6528-1.

RESENDE, M. G.; RIBEIRO, C. C. Grasp with path-relinking: Recent advances and applications. In: _____. *Metaheuristics: Progress as Real Problem Solvers*. Boston, MA: Springer US, 2005. p. 29–63. ISBN 978-0-387-25383-1. Disponível em: <https://doi.org/10.1007/0-387-25383-1_2>.

RESENDE, M. G. C.; RIBEIRO, C. C. A grasp with path-relinking for private virtual circuit routing. *Networks*, v. 41, n. 2, p. 104–114, 2003.

RESENDE, M. G. C.; RIBEIRO, C. C. *Greedy Randomized Adaptive Search Procedures: Advances and Extensions*. Cham: Springer International Publishing, 2019. 169–220 p. ISBN 978-3-319-91086-4.

RIEDLER, M.; RAIDL, G. Solving a selective dial-a-ride problem with logic-based benders decomposition. *Computers & Operations Research*, v. 96, p. 30–54, 2018.

ROSENKRANTZ, D.; STEARNS, R.; LEWIS, I. P. An analysis of several heuristics for the traveling salesman problem. *SIAM Journal on Computing*, v. 6, n. 3, p. 563–581, 1977.

ROSSETI, I. *Heuristicas para o problema de sintese de redes a 2-caminhos*. Tese (Doutorado) — Department of Computer Science, Catholic University of Rio de Janeiro, 2003.

SANTOS, A. G.; CHAGAS, J. B. C. The thief orienteering problem: Formulation and heuristic approaches. p. 1–9, July 2018.

SAVELSBERGH, M.; SOL, M. The general pickup and delivery problem. *Transportation Science*, v. 29, p. 17–29, 1995.

SCHÖNBERGER, J.; KOPFER, H.; MATTFELD, D. C. *A Combined Approach to Solve the Pickup and Delivery Selection Problem*. [S.l.]: Springer Berlin Heidelberg, 2003. 150–155 p. ISBN 978-3-642-55537-4.

SILVA, B. C. et al. Quota travelling salesman problem with passengers, incomplete ride and collection time optimization by ant-based algorithms. *Computers & Operations Research*, v. 120, p. 104950, 2020. ISSN 0305-0548. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0305054820300678>.

SILVA, J. G. S. *Algoritmos de solução para o problema do caixeiro viajante com passageiros e quota*. Dissertação (Mestrado) — Universidade Federal do Rio Grande do Norte, 2017.

SIMONIN, G.; O'SULLIVAN, B. *Optimisation for the Ride-sharing Problem: A Complexity-based Approach*. [S.l.]: IOS Press, 2014. 831-836 p. (ECAI 14). ISBN 978-1-61499-418-3.

SKINDEROWICZ, R. The gpu-based parallel ant colony system. *Journal of Parallel and Distributed Computing*, v. 98, p. 48 – 60, 2016. ISSN 0743-7315. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0743731516300284>.

STÜTZLE, T.; DORIGO, M. et al. Aco algorithms for the traveling salesman problem. *Evolutionary algorithms in engineering and computer science*, v. 4, p. 163–183, 1999.

STÜTZLE, T.; RUIZ, R. Chapter 1 iterated local search : A concise review. In: *IRIDIA - Technical Report Series*. [S.l.]: Universite Libre de Bruxelles, 2018.

TRIPATHY, T. et al. *Advances in Computational Intelligence Systems*. [S.l.]: Springer International Publishing, 2018. 325–336 p. ISBN 978-3-319-66939-7.

UCHOA, E. New benchmark instances for the capacitated vehicle routing problem. *European Journal of Operational Research*, v. 257, n. 3, p. 845 – 858, 2017. ISSN 0377-2217.

WANG, J. et al. Multi-offspring genetic algorithm and its application to the traveling salesman problem. *Applied Soft Computing*, v. 43, p. 415 – 423, 2016. ISSN 1568-4946. Disponível em: <http://www.sciencedirect.com/science/article/pii/S1568494616300722>.

WANG, Y.; ZHENG, B.; LIM, E. Understanding the effects of taxi ride-sharing — a case study of singapore. *Computers, Environment and Urban Systems*, v. 69, p. 124 – 132, 2018. ISSN 0198-9715.

WHITLEY, D. An overview of evolutionary algorithms: practical issues and common pitfalls. *Information and Software Technology*, v. 43, n. 14, p. 817 – 831, 2001. ISSN 0950-5849. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0950584901001884>.

WOOD, D. A. Evolutionary memetic algorithms supported by metaheuristic profiling effectively applied to the optimization of discrete routing problems. *Journal of Natural Gas Science and Engineering*, v. 35, p. 997 – 1014, 2016. ISSN 1875-5100.

YANG, X.-S. Chapter 5 - genetic algorithms. In: YANG, X.-S. (Ed.). *Nature-Inspired Optimization Algorithms*. Oxford: Elsevier, 2014. p. 77 – 87. ISBN 978-0-12-416743-8. Disponível em: <http://www.sciencedirect.com/science/article/pii/B9780124167438000051>.

ZHANG, C. et al. *Algorithm Designs for Dynamic Ridesharing System*. Cham: Springer International Publishing, 2018. 209–220 p. ISBN 978-3-030-04618-7.

ZHAO, M. et al. Ridesharing problem with flexible pickup and delivery locations for app-based transportation service: Mathematical modeling and decomposition methods. *Journal of Advanced Transportation*, v. 2018, p. 1–21, 2018.

ZHOU, Y.; HE, F.; QIU, Y. Optimization of parallel iterated local search algorithms on graphics processing unit. *The Journal of Supercomputing*, v. 72, 05 2016.

ZUFFEREY, N.; FARRES, J.; GLARDON, R. *Computational Logistics*. [S.l.]: Springer International Publishing, 2015. 3–15 p. ISBN 978-3-319-24264-4.

APÊNDICE A – Resultados das Heurísticas *Naive*

	Simétrico									
Instância	$HN1$	$Tempo$	$HN2$	$Tempo$	$HN3$	$Tempo$	$HN4$	$Tempo$	$HN5$	$Tempo$
A-10-3	1172,00	4,04	1172,00	0,38	1172,00	3,00	742,00	0,01	700,75	0,01
A-10-4	971,83	4,35	971,83	0,41	849,00	3,96	618,17	0,01	618,17	0,01
A-10-5	1018,75	9,32	1018,75	0,45	738,17	8,91	738,17	0,01	515,80	0,01
A-20-3	1226,50	19,97	1426,00	1,54	1109,00	18,69	1872,00	0,01	1509,00	0,01
A-20-4	479,33	48,66	479,33	1,42	913,33	47,45	708,40	0,02	708,40	0,02
A-20-5	692,00	77,99	692,00	2,03	812,67	73,07	781,88	0,01	543,08	0,01
A-30-3	830,75	64,37	892,25	3,22	1103,83	61,43	1909,00	0,04	1550,75	0,04
A-30-4	1100,50	86,50	1154,00	4,45	1512,37	80,86	2237,00	0,02	1738,58	0,03
A-30-5	808,50	219,31	819,83	5,11	1021,47	215,69	1108,42	0,03	839,57	0,05
A-40-3	1152,83	231,00	1233,67	13,00	1625,83	219,00	2614,50	0,05	1575,58	0,07
A-40-4	882,83	655,00	1277,83	6,60	1259,00	611,00	1830,60	0,05	1590,00	0,06
A-40-5	626,80	529,00	951,00	9,24	1123,72	533,00	1341,38	0,40	1039,55	0,05
B-10-3	1105,00	4,84	1105,00	0,47	834,67	3,67	834,67	0,02	834,67	0,02
B-10-4	978,25	7,81	1011,25	0,40	649,00	7,77	649,00	0,03	493,58	0,03
B-10-5	1208,50	4,85	1208,50	0,31	1208,50	4,45	1428,00	0,03	851,90	0,03
B-20-3	1313,00	29,16	1313,00	1,17	1449,00	27,85	1371,00	0,05	1351,83	0,05
B-20-4	1452,00	47,58	1485,50	2,22	1184,10	46,05	1452,33	0,02	1396,33	0,02
B-20-5	1246,50	47,57	1246,50	1,62	1095,75	45,77	1192,67	0,02	1003,37	0,02
B-30-3	1104,17	119,83	1291,67	3,83	1704,75	116,12	1740,67	0,04	1313,83	0,04
B-30-4	891,17	137,15	1195,00	3,34	1501,50	134,40	1980,50	0,03	1780,67	0,03
B-30-5	694,50	268,47	1019,17	2,07	1203,00	267,12	1460,42	0,04	1060,22	0,06
B-40-3	1118,33	252,78	1395,83	7,26	1662,00	237,60	1835,75	0,04	1496,50	0,04
B-40-4	1431,08	409,86	1574,08	10,56	1514,00	402,60	2177,00	0,10	1743,50	0,20
B-40-5	1067,58	506,88	1110,08	9,90	1395,58	462,00	1668,00	0,10	1539,62	0,10
C-10-3	636,67	4,76	612,50	0,31	558,08	4,22	612,50	0,01	648,75	0,01
C-10-4	530,83	5,99	571,25	0,31	530,83	5,05	496,25	0,01	496,25	0,01
C-10-5	544,80	4,67	504,33	0,33	682,92	4,62	563,03	0,01	563,03	0,01
C-20-3	1169,83	35,93	1067,92	1,15	906,67	34,20	972,00	0,02	911,75	0,02
C-20-4	786,08	74,78	872,40	1,21	933,08	57,89	1225,62	0,02	1028,80	0,03
C-20-5	973,33	31,96	953,10	1,12	1255,67	30,83	1129,00	0,01	1108,10	0,02
C-30-3	1148,58	116,41	1429,75	4,80	1428,75	110,34	1626,42	0,03	1163,08	0,03
C-30-4	832,38	85,14	1059,65	3,12	1278,92	81,78	1387,67	0,03	1498,45	0,03
C-30-5	719,58	177,16	965,80	3,86	764,68	182,73	1044,07	0,04	702,83	0,04
C-40-3	*	*	1674,25	7,19	*	*	1214,92	0,04	1116,08	0,05
C-40-4	*	*	1181,40	8,87	*	*	1437,80	0,06	1370,60	0,06
C-40-5	*	*	960,50	9,56	*	*	1314,67	0,04	1138,83	0,07

Tabela 36: Resultados das heurísticas HNs para instâncias simétricas de porte pequeno.

	Simétrico									
Instância	$HN1$	$Tempo$	$HN2$	$Tempo$	$HN3$	$Tempo$	$HN4$	$Tempo$	$HN5$	$Tempo$
A-50-3	1258,25	462,00	1812,00	24,42	2054,17	457,00	2747,92	0,10	2203,83	0,10
A-50-4	1213,60	762,96	1222,00	22,44	1883,58	735,00	2868,50	0,08	1702,83	0,09
A-50-5	*	*	1967,33	18,48	*	*	1871,95	0,08	1093,43	0,09
A-100-3	*	*	3360,67	461,03	*	*	3739,83	0,19	2898,50	0,22
A-100-4	*	*	2149,47	398,55	*	*	2247,72	0,19	1848,58	0,23
A-100-5	*	*	2236,08	273,25	*	*	2184,95	0,19	1634,38	0,26
A-200-3	*	*	*	*	*	*	4402,00	0,67	4084,58	0,85
A-200-4	*	*	*	*	*	*	3837,03	0,66	3062,80	1,08
A-200-5	*	*	*	*	*	*	4390,17	0,64	3582,15	0,74
A-500-3	*	*	*	*	*	*	7994,17	4,12	7771,58	15,28
A-500-4	*	*	*	*	*	*	6680,80	4,52	6090,40	45,86
A-500-5	*	*	*	*	*	*	6680,80	4,52	6090,40	45,86
B-50-3	1626,50	523,38	2160,33	21,78	2020,50	498,96	2278,33	0,07	2184,42	0,07
B-50-4	*	*	1644,50	29,70	*	*	1813,82	0,30	1930,07	0,30
B-50-5	982,40	952,08	1320,75	15,18	1857,50	960,30	2902,17	0,30	1758,38	0,30
B-100-3	*	*	2878,67	272,48	*	*	3948,67	0,18	3548,17	0,20
B-100-4	*	*	3522,67	361,55	*	*	4590,67	0,18	3413,75	0,19
B-100-5	*	*	2645,75	775,25	*	*	2425,72	0,19	1778,53	0,29
B-200-3	*	*	*	*	*	*	5224,00	0,65	4098,00	0,86
B-200-4	*	*	*	*	*	*	3715,62	0,68	3465,78	1,28
B-200-5	*	*	*	*	*	*	5799,10	0,69	2935,52	1,12
B-500-3	*	*	*	*	*	*	8827,75	4,21	7915,25	11,84
B-500-4	*	*	*	*	*	*	6921,38	4,25	6414,02	15,43
B-500-5	*	*	*	*	*	*	6802,53	4,63	5546,78	14,51
C-50-3	*	*	1468,75	26,11	*	*	1953,75	0,09	1623,92	0,10
C-50-4	*	*	1440,65	44,02	*	*	2100,25	0,07	1727,30	0,08
C-50-5	*	*	1517,10	18,17	*	*	2046,18	0,06	2074,90	0,12
C-100-3	*	*	2400,83	418,50	*	*	2887,92	0,20	2300,25	0,22
C-100-4	*	*	1591,60	357,69	*	*	2298,80	0,19	1574,00	0,25
C-100-5	*	*	1416,92	418,22	*	*	2063,50	0,18	1651,98	0,21
C-200-3	*	*	*	*	*	*	4237,25	0,64	3673,00	0,96
C-200-4	*	*	*	*	*	*	3819,45	0,64	3145,20	0,72
C-200-5	*	*	*	*	*	*	2372,00	0,67	2330,17	1,44
C-500-3	*	*	*	*	*	*	8071,00	4,15	7524,58	11,31
C-500-4	*	*	*	*	*	*	6377,48	3,91	6167,82	6,98
C-500-5	*	*	*	*	*	*	5793,27	5,48	5219,33	31,64

Tabela 37: Resultados das heurísticas HNs para instâncias simétricas de porte médio e grande.

	Assimétrico									
Instância	$HN1$	$Tempo$	$HN2$	$Tempo$	$HN3$	$Tempo$	$HN4$	$Tempo$	$HN5$	$Tempo$
A-10-3	823,00	2,83	823,00	0,30	478,42	2,58	823,00	0,02	823,00	0,02
A-10-4	959,25	3,71	1044,50	0,32	959,25	3,37	1044,50	0,01	523,57	0,01
A-10-5	934,50	8,55	934,50	0,28	934,50	8,34	934,50	0,01	720,50	0,01
A-20-3	901,17	33,31	930,17	1,02	1164,50	32,24	837,67	0,03	771,25	0,04
A-20-4	955,00	33,35	955,00	0,93	901,67	32,31	1352,00	0,02	878,83	0,02
A-20-5	589,42	50,63	618,92	1,04	810,33	49,88	1113,00	0,03	889,27	0,04
A-30-3	1101,17	89,92	1095,33	2,64	1700,17	87,59	1299,67	0,03	1585,83	0,03
A-30-4	502,98	212,46	792,80	3,14	739,55	210,02	1179,03	0,02	1143,90	0,03
A-30-5	745,63	145,30	1243,67	3,57	1087,20	142,37	1516,00	0,02	1089,55	0,03
A-40-3	781,92	238,26	1258,58	4,62	1573,75	226,90	2126,83	0,06	1539,75	0,08
A-40-4	863,87	442,86	1283,33	5,28	1833,57	474,17	1494,40	0,04	1266,55	0,05
A-40-5	1017,50	544,50	1212,50	7,26	1024,92	501,60	1166,97	0,05	806,78	0,07
B-10-3	947,00	4,54	947,00	0,34	947,00	4,21	947,00	0,01	901,83	0,02
B-10-4	814,00	7,25	814,00	0,28	395,67	6,96	814,00	0,01	547,67	0,01
B-10-5	786,33	9,31	786,33	0,27	786,33	9,06	786,33	0,01	681,67	0,01
B-20-3	1177,50	38,58	1177,50	0,85	1321,25	41,60	1376,33	0,03	1342,33	0,03
B-20-4	1390,00	27,68	1390,00	0,97	1544,50	26,34	1579,00	0,02	1415,67	0,02
B-20-5	1361,00	33,95	1361,00	1,00	1354,83	32,67	1559,50	0,01	1365,50	0,01
B-30-3	1496,00	177,76	1627,17	3,46	1756,17	172,15	2258,50	0,04	1945,67	0,05
B-30-4	1225,00	173,10	1491,00	3,00	1999,83	171,57	2299,50	0,02	1840,17	0,03
B-30-5	977,50	236,71	1274,50	2,62	1333,02	235,63	2322,42	0,04	1403,27	0,05
B-40-3	1575,00	228,36	1654,00	5,28	2411,00	206,58	3061,00	0,03	2031,00	0,05
B-40-4	1618,00	217,00	1634,00	9,90	3304,00	207,90	3505,00	0,10	2772,00	0,20
B-40-5	1506,00	500,94	1918,50	11,22	2638,03	400,62	2799,83	0,10	1718,00	0,10
C-10-3	484,83	6,28	546,33	0,38	494,17	5,86	485,42	0,01	485,42	0,02
C-10-4	402,50	9,27	463,20	0,16	402,50	9,11	463,20	0,01	450,30	0,01
C-10-5	578,00	4,65	595,25	0,34	623,83	4,24	710,75	0,01	710,75	0,01
C-20-3	705,08	50,95	877,33	1,12	941,33	49,32	1228,17	0,02	982,33	0,02
C-20-4	1001,25	56,69	999,40	1,22	1022,33	54,81	1205,10	0,03	1019,98	0,04
C-20-5	757,33	58,62	920,22	1,16	1169,33	56,64	1130,45	0,04	1130,45	0,04
C-30-3	995,17	87,94	1035,00	2,87	1436,75	91,59	1949,25	0,04	1621,92	0,04
C-30-4	1062,17	219,19	1188,23	3,89	1330,78	669,20	1489,62	0,04	1429,30	0,06
C-30-5	869,32	913,52	968,23	3,60	998,33	301,68	1268,72	0,04	1090,68	0,08
C-40-3	*	*	1230,42	8,20	*	*	2164,25	0,04	1781,83	0,05
C-40-4	*	*	1395,55	8,34	*	*	2006,23	0,06	1676,87	0,10
C-40-5	*	*	1127,72	8,38	*	*	2202,97	0,05	1243,75	0,13

Tabela 38: Resultados das heurísticas HNs para instâncias assimétricas de porte pequeno.

	Assimétrico									
Instância	$HN1$	$Tempo$	$HN2$	$Tempo$	$HN3$	$Tempo$	$HN4$	$Tempo$	$HN5$	$Tempo$
A-50-3	1499,50	191,00	2072,17	11,88	2087,00	171,60	3683,00	0,06	4034,75	0,08
A-50-4	887,65	614,46	1725,50	19,14	1948,83	608,52	2763,73	0,07	1800,93	0,09
A-50-5	1120,50	705,54	1693,33	18,57	1631,65	663,30	2101,65	0,05	2030,98	0,08
A-100-3	*	*	2878,67	272,48	*	*	5140,50	0,21	3185,00	0,50
A-100-4	*	*	3522,67	361,55	*	*	4142,23	0,22	3320,90	0,30
A-100-5	*	*	2645,75	775,25	*	*	3222,02	0,21	2502,80	0,46
A-200-3	*	*	*	*	*	*	6279,83	1,07	6121,92	4,40
A-200-4	*	*	*	*	*	*	6710,75	1,01	5252,67	3,35
A-200-5	*	*	*	*	*	*	3988,88	1,02	3775,77	11,71
A-500-3	*	*	*	*	*	*	16613,40	11,39	15037,70	53,12
A-500-4	*	*	*	*	*	*	12934,70	11,43	12480,80	126,93
A-500-5	*	*	*	*	*	*	9392,30	14,94	9011,07	361,66
B-50-3	2044,00	273,24	2381,00	15,18	3449,00	269,94	4973,00	0,05	3274,00	0,07
B-50-4	1695,00	946,44	2081,00	19,80	3114,00	902,22	3655,50	0,30	2798,00	0,30
B-50-5	1998,00	767,58	2565,00	21,78	2753,00	743,16	2900,00	0,30	2234,00	0,30
B-100-3	*	*	3455,67	359,02	*	*	5144,42	0,21	4267,92	0,41
B-100-4	*	*	2831,58	323,68	*	*	3602,68	0,21	3315,77	0,55
B-100-5	*	*	3157,20	251,68	*	*	3761,15	0,21	3761,15	0,45
B-200-3	*	*	*	*	*	*	9448,75	1,03	7999,42	3,51
B-200-4	*	*	*	*	*	*	7759,43	1,02	6004,88	4,67
B-200-5	*	*	*	*	*	*	4927,37	1,03	4413,57	7,65
B-500-3	*	*	*	*	*	*	18131,80	11,49	15587,80	189,71
B-500-4	*	*	*	*	*	*	12636,60	11,43	12055,00	613,32
B-500-5	*	*	*	*	*	*	10181,80	12,35	9840,28	676,18
C-50-3	*	*	1528,67	19,00	*	*	2184,25	0,06	2115,17	0,08
C-50-4	*	*	1764,33	18,28	*	*	3018,60	0,06	2582,60	0,11
C-50-5	*	*	723,90	18,28	*	*	2105,90	0,05	1760,50	0,12
C-100-3	*	*	1933,83	251,07	*	*	4255,17	0,21	3346,75	0,47
C-100-4	*	*	2712,70	274,45	*	*	4434,55	0,21	3883,15	0,43
C-100-5	*	*	2332,27	623,26	*	*	3061,67	0,21	2138,00	0,79
C-200-3	*	*	*	*	*	*	6553,75	1,03	5938,83	5,26
C-200-4	*	*	*	*	*	*	6458,45	1,03	5037,18	6,49
C-200-5	*	*	*	*	*	*	4484,33	1,03	4045,73	7,99
C-500-3	*	*	*	*	*	*	14455,50	11,56	13580,30	176,89
C-500-4	*	*	*	*	*	*	11749,50	11,52	11099,10	781,76
C-500-5	*	*	*	*	*	*	9921,12	13,79	9406,67	365,03

Tabela 39: Resultados das heurísticas HNs para instâncias assimétricas de porte médio e grande.

APÊNDICE B – Resultados dos Algoritmos de Formigas

	AS			ACS			MS-ACS		
Instância	Solução	Média	Tempo	Solução	Média	Tempo	Solução	Média	Tempo
A-10-3	478,42	478,42	0,05	478,42	616,27	0,06	478,42	478,42	0,13
A-10-4	601,08	601,08	0,05	523,57	523,57	0,07	523,57	523,57	0,12
A-10-5	542,00	545,14	0,05	542,00	644,18	0,07	482,60	552,28	0,11
A-20-3	624,58	664,62	0,09	773,42	858,08	0,12	545,50	716,42	0,24
A-20-4	674,33	778,77	0,09	795,63	984,28	0,11	439,67	588,52	0,23
A-20-5	483,17	506,80	0,09	696,25	765,45	0,13	420,78	576,98	0,26
A-30-3	866,25	976,02	0,14	1065,08	1128,60	0,19	760,00	827,45	0,35
A-30-4	451,97	524,57	0,18	606,67	661,52	0,27	489,85	515,22	0,48
A-30-5	576,30	644,73	0,15	683,58	858,84	0,22	539,62	608,37	0,40
A-40-3	898,33	925,63	0,26	912,33	1146,00	0,45	764,75	803,00	0,76
A-40-4	835,42	905,52	0,27	907,38	1071,28	0,39	753,15	845,69	0,79
A-40-5	583,17	622,87	0,29	572,53	714,56	0,53	482,98	582,09	0,83
B-10-3	778,83	842,57	0,04	913,00	913,00	0,05	729,50	865,70	0,09
B-10-4	306,90	377,91	0,04	395,67	395,67	0,05	306,90	395,67	0,10
B-10-5	434,75	545,34	0,05	560,67	633,51	0,07	434,75	479,17	0,12
B-20-3	937,25	976,37	0,09	893,25	1069,87	0,11	893,25	997,62	0,22
B-20-4	1029,83	1079,67	0,08	1305,00	1337,37	0,10	895,25	1114,57	0,22
B-20-5	897,83	946,77	0,08	1128,83	1188,57	0,10	897,00	975,33	0,20
B-30-3	944,17	1004,62	0,15	1021,75	1153,87	0,21	861,08	973,22	0,42
B-30-4	966,42	1039,24	0,16	996,38	1094,91	0,23	771,73	937,93	0,46
B-30-5	841,92	971,15	0,18	1044,67	1189,71	0,24	702,00	899,36	0,46
B-40-3	1164,83	1299,57	0,25	1359,25	1498,23	0,36	951,00	1149,05	0,68
B-40-4	1727,75	1781,68	0,22	1969,50	2025,20	0,31	1403,00	1637,12	0,56
B-40-5	947,03	1073,87	0,27	1215,62	1371,71	0,38	932,48	1097,92	0,76
C-10-3	359,25	359,25	0,05	469,67	481,13	0,06	359,25	438,42	0,12
C-10-4	318,40	344,39	0,07	307,10	383,17	0,08	307,10	393,40	0,17
C-10-5	595,25	601,02	0,05	689,87	698,22	0,06	566,58	585,25	0,12
C-20-3	766,83	769,30	0,12	769,92	811,12	0,16	650,25	745,98	0,32
C-20-4	688,17	741,88	0,13	747,05	966,93	0,20	613,83	764,22	0,33
C-20-5	887,50	905,85	0,14	950,27	1050,01	0,19	693,05	765,28	0,36
C-30-3	925,00	954,65	0,19	1013,08	1101,75	0,27	869,75	936,75	0,47
C-30-4	891,50	937,38	0,26	873,37	967,62	0,36	796,37	937,02	0,67
C-30-5	678,17	704,85	0,47	708,97	879,73	0,71	613,10	751,52	13,39
C-40-3	1195,17	1282,12	0,30	1407,33	1472,42	0,42	1089,83	1137,03	0,85
C-40-4	844,15	907,41	0,50	1092,20	1160,85	0,74	787,30	901,87	14,17
C-40-5	800,67	879,33	0,71	723,90	744,04	0,76	674,90	676,68	1,29

Tabela 40: Resultados dos algoritmos de formigas para instâncias assimétricas de pequeno porte.

	AS			ACS			MS-ACS		
Instância	Solução	Média	Tempo	Solução	Média	Tempo	Solução	Média	Tempo
A-50-3	1541,58	1642,07	0,29	1421,25	1661,65	0,44	1259,42	1410,55	0,80
A-50-4	939,83	1024,66	0,33	1142,40	1324,53	0,55	867,35	926,20	0,93
A-50-5	937,92	998,11	0,35	876,00	1022,22	0,60	728,85	923,74	10,69
A-100-3	1678,33	1764,65	11,38	1750,50	1777,30	25,60	1488,33	1526,65	41,51
A-100-4	1955,88	2045,87	0,97	2111,87	2211,30	23,14	1531,23	1700,35	3,23
A-100-5	1392,50	1491,58	11,06	1456,48	1650,13	27,44	1172,48	1262,39	41,00
A-200-3	3099,08	3218,08	38,57	3160,00	3239,28	406,13	2710,25	2799,92	425,12
A-200-4	2742,58	2809,49	3,63	2678,35	2814,21	329,02	2324,87	2448,07	323,95
A-200-5	2022,92	2117,25	64,16	1936,20	1956,10	111,30	1782,52	1820,53	102,28
A-500-3	8366,25	8432,40	17,31	7547,50	7603,90	656,10	7048,50	7104,10	465,00
A-500-4	6101,20	6239,04	19,49	5412,28	5608,40	1363,30	5238,50	5316,89	1460,10
A-500-5	4969,50	5110,50	34,72	4606,50	4734,33	3230,39	4440,37	4478,59	3501,97
B-50-3	1881,75	1924,23	0,30	1715,08	1970,27	0,44	1536,50	1696,75	0,84
B-50-4	1439,13	1468,57	0,37	1328,25	1553,53	0,68	1084,82	1320,08	10,90
B-50-5	1461,37	1596,70	0,34	1507,12	1576,20	0,59	1178,52	1334,71	0,93
B-100-3	2344,67	2459,05	1,02	2164,92	2255,85	19,64	2015,08	2209,33	37,90
B-100-4	1786,82	1918,52	1,23	1753,35	1889,29	4,05	1504,97	1561,61	5,27
B-100-5	2050,45	2264,07	10,22	2056,47	2390,37	27,34	1504,07	1616,79	42,01
B-200-3	3900,25	4246,38	3,35	3636,08	3913,93	23,26	3302,42	3412,52	322,30
B-200-4	3024,77	3130,56	39,48	2731,17	2836,57	29,86	2532,07	2707,25	495,66
B-200-5	2367,53	2508,99	52,07	2281,67	2372,81	78,99	2097,10	2160,59	893,19
B-500-3	8148,00	8320,15	20,95	7216,43	7395,19	1169,30	6886,08	6731,50	2106,44
B-500-4	5983,57	6062,66	47,45	5815,18	5972,26	7310,88	5517,34	5639,27	7628,10
B-500-5	4951,02	5137,85	45,89	4672,04	4794,44	5928,32	4541,48	4631,19	5614,49
C-50-3	1500,25	1542,87	0,36	1477,58	1625,90	0,48	1288,92	1452,02	0,93
C-50-4	1227,32	1303,70	0,56	1361,88	1539,36	0,88	953,35	1171,58	1,76
C-50-5	664,58	841,59	0,86	904,62	1029,37	15,15	694,10	797,29	24,15
C-100-3	1622,75	1747,95	11,12	1769,58	1853,85	25,64	1452,75	1530,63	40,07
C-100-4	1977,80	2062,60	11,75	1970,30	2216,51	28,06	1496,57	1704,36	4,32
C-100-5	1150,90	1213,31	1,42	1250,50	1281,35	78,80	987,00	1092,96	78,92
C-200-3	2951,50	2986,25	36,04	2693,75	2773,15	272,38	2670,00	2693,32	328,37
C-200-4	2472,20	2510,93	3,91	2319,40	2390,87	597,19	2186,08	2234,64	314,12
C-200-5	2047,00	2137,73	45,13	1962,33	2066,18	590,39	1878,00	1909,56	482,50
C-500-3	6978,56	7028,78	20,37	6557,42	6694,16	1826,19	6470,34	6528,72	1356,29
C-500-4	5640,84	5782,28	46,23	5236,85	5368,92	6894,38	5078,54	5128,40	6901,70
C-500-5	4773,83	4842,15	26,78	4381,17	4425,46	2915,37	4193,38	4257,33	2872,73

Tabela 41: Resultados dos algoritmos de formigas para instâncias assimétricas de médio e grande porte.

	AS			ACS			MS-ACS		
Instância	Solução	Média	Tempo	Solução	Média	Tempo	Solução	Média	Tempo
A-10-3	606,50	670,65	0,05	606,50	668,00	0,07	545,92	587,46	0,12
A-10-4	460,00	553,69	0,05	625,73	647,35	0,06	460,00	492,50	0,13
A-10-5	435,93	474,52	0,06	434,05	453,65	0,09	371,93	398,47	0,16
A-20-3	679,75	848,52	0,09	955,83	1092,48	0,11	786,33	958,83	0,22
A-20-4	346,30	430,65	0,12	500,90	645,50	0,13	348,48	394,34	0,33
A-20-5	413,97	451,31	0,11	473,70	583,70	0,13	432,40	468,85	0,29
A-30-3	652,00	810,03	0,14	915,42	1125,70	0,16	695,17	798,10	0,35
A-30-4	915,25	970,92	0,13	1044,00	1220,60	0,17	912,25	970,23	0,35
A-30-5	572,93	605,48	0,18	579,30	685,76	0,30	541,48	587,16	0,56
A-40-3	922,92	1024,33	0,23	1158,17	1257,32	0,31	906,00	1000,70	0,59
A-40-4	694,68	713,83	0,32	776,87	842,69	0,48	608,75	681,51	0,85
A-40-5	560,92	589,58	0,29	656,62	775,62	0,42	557,92	564,02	0,79
B-10-3	834,67	868,47	0,05	834,67	869,57	0,05	834,67	854,80	0,11
B-10-4	493,58	556,78	0,05	493,58	539,09	0,07	493,58	511,03	0,13
B-10-5	812,97	820,37	0,06	811,67	891,88	0,09	726,35	842,76	0,16
B-20-3	1291,83	1304,67	0,09	1015,42	1361,05	0,11	953,83	1090,60	0,22
B-20-4	1016,67	1065,65	0,09	1016,75	1170,27	0,11	822,82	1016,75	0,23
B-20-5	849,08	915,03	0,10	913,07	981,02	0,11	687,13	738,57	0,24
B-30-3	950,33	1030,30	0,17	875,17	957,02	0,21	792,75	918,03	0,43
B-30-4	948,05	997,14	0,21	855,02	936,53	0,23	715,48	884,37	0,53
B-30-5	602,88	671,92	0,21	655,38	656,60	0,34	510,13	612,13	0,56
B-40-3	1142,25	1226,18	0,27	1137,00	1332,55	0,34	975,50	1010,37	0,66
B-40-4	1021,00	1087,74	0,26	1069,63	1245,72	0,38	714,05	901,46	0,70
B-40-5	994,58	1044,01	0,28	837,85	1065,05	0,39	830,03	902,11	0,74
C-10-3	558,67	589,53	0,04	612,50	633,15	0,05	558,67	613,95	0,10
C-10-4	434,60	434,60	0,05	428,80	428,80	0,06	408,45	424,73	0,11
C-10-5	437,73	507,29	0,05	411,07	454,00	0,07	409,60	453,44	0,14
C-20-3	666,08	734,35	0,13	768,58	902,85	0,14	613,83	740,92	0,30
C-20-4	481,85	659,41	0,14	787,18	872,42	0,21	441,65	596,91	0,39
C-20-5	742,57	831,79	0,10	776,38	968,34	0,14	713,55	815,86	0,27
C-30-3	1021,83	1061,35	0,18	1125,25	1230,07	0,23	923,42	1006,23	0,46
C-30-4	859,35	952,65	0,19	776,50	952,89	0,22	770,15	860,68	0,46
C-30-5	708,17	756,34	0,29	656,67	773,15	0,37	518,33	671,35	0,71
C-40-3	777,75	809,07	0,32	761,75	855,95	0,41	629,00	770,50	0,82
C-40-4	722,60	812,02	0,34	820,55	1036,37	0,36	686,60	733,48	0,71
C-40-5	654,33	681,19	0,50	721,67	808,52	0,87	475,50	638,95	15,89

Tabela 42: Resultados dos algoritmos de formigas para instâncias simétricas de pequeno porte.

	AS			ACS			MS-ACS		
Instância	Solução	Média	Tempo	Solução	Média	Tempo	Solução	Média	Tempo
A-50-3	1186,58	1382,03	0,31	1301,75	1523,38	0,43	1041,25	1190,12	0,85
A-50-4	1062,80	1106,44	0,36	1014,62	1053,05	0,45	862,03	998,79	0,96
A-50-5	694,97	735,78	0,48	680,90	707,84	0,63	553,85	682,10	1,31
A-100-3	1827,33	1947,92	0,99	1860,33	2018,90	1,23	1576,08	1699,58	24,95
A-100-4	1454,30	1492,00	12,89	1403,27	1454,37	21,11	1269,28	1328,74	34,44
A-100-5	1167,05	1193,84	13,63	1063,95	1159,77	19,03	999,42	1046,91	37,00
A-200-3	3276,75	3464,38	35,07	3083,67	3281,62	54,29	2731,17	2932,77	87,59
A-200-4	2489,87	2548,28	4,73	2346,55	2454,79	73,55	2265,92	2311,55	126,35
A-200-5	2657,80	2733,24	32,19	2458,17	2621,34	43,36	2113,38	2240,55	79,78
A-500-3	7158,33	7229,30	41,23	6760,42	6847,89	81,63	6445,75	6532,27	108,38
A-500-4	5605,08	5783,13	91,21	5035,35	5139,81	133,43	5007,38	5191,13	328,15
A-500-5	5618,95	5819,34	20,24	4751,40	4823,16	28,84	4651,00	4793,28	57,18
B-50-3	1245,67	1317,23	0,36	1306,67	1384,48	0,57	1068,92	1195,90	1,04
B-50-4	1051,73	1120,98	0,38	1085,18	1196,22	0,62	931,52	1040,11	10,46
B-50-5	956,65	1019,23	0,39	817,88	940,56	0,53	748,58	854,38	1,07
B-100-3	2576,17	2795,90	0,96	2264,08	2491,07	11,48	1920,75	2042,80	23,29
B-100-4	2632,20	2813,67	0,89	2102,77	2652,72	1,06	2063,63	2221,03	2,19
B-100-5	1312,12	1341,51	17,23	1270,80	1365,09	29,59	1043,22	1142,69	48,26
B-200-3	3625,67	3676,45	34,59	2963,00	3179,53	48,48	2913,67	3025,05	90,87
B-200-4	2748,80	2817,57	4,46	2532,42	2619,81	69,69	2397,13	2455,75	122,99
B-200-5	2218,13	2341,24	51,94	2034,30	2095,53	9,42	1912,87	1972,98	150,29
B-500-3	7541,25	7633,09	30,89	6641,42	6783,36	96,86	4415,60	4537,48	94,18
B-500-4	5818,20	5937,41	45,38	5229,83	5344,13	110,64	5139,48	5224,54	131,72
B-500-5	5089,38	5233,11	31,94	4594,72	4783,45	55,40	4415,60	4537,48	94,18
C-50-3	1122,42	1216,93	0,36	1001,50	1129,42	0,45	896,75	1017,65	0,93
C-50-4	1243,78	1296,77	0,42	1349,03	1482,22	0,53	954,85	1168,31	10,73
C-50-5	880,37	1028,34	0,60	1057,67	1232,51	0,84	840,23	957,49	21,36
C-100-3	1656,50	1694,68	1,09	1537,00	1734,67	1,27	1376,33	1452,47	25,79
C-100-4	1262,00	1304,66	14,14	1193,80	1340,64	21,93	1067,92	1151,15	37,67
C-100-5	1158,95	1259,97	11,42	1241,28	1373,09	14,91	1029,47	1156,45	28,94
C-200-3	2742,50	2915,02	3,51	2675,75	2720,22	4,50	2564,58	2603,33	87,84
C-200-4	2466,38	2549,03	33,51	2361,07	2415,72	45,32	2106,93	2211,53	85,47
C-200-5	1963,67	1995,41	6,46	1888,83	1928,63	118,68	1721,17	1790,24	182,05
C-500-3	6886,36	7017,39	31,84	6161,09	6308,32	62,41	6190,75	6248,55	91,50
C-500-4	5490,10	5549,37	18,29	4985,18	5059,24	21,78	4976,39	5097,15	43,28
C-500-5	4623,17	4732,39	53,75	4138,84	4264,46	90,52	4029,83	4157,45	201,98

Tabela 43: Resultados dos algoritmos de formigas para instâncias simétricas de médio e grande porte.

	Assimétrico			Simétrico		
Instância	*AS*	*ACS*	*MS-ACS*	*AS*	*ACS*	*MS-ACS*
A-10-3	100%	100%	100%	0%	0%	30%
A-10-4	0%	25%	20%	40%	0%	35%
A-10-5	0%	0%	15%	0%	0%	50%
A-20-3	0%	0%	10%	25%	0%	0%
A-20-4	0%	0%	20%	15%	0%	0%
A-20-5	0%	0%	10%	20%	0%	0%
A-30-3	0%	0%	20%	20%	0%	0%
A-30-4	10%	0%	0%	0%	0%	30%
A-30-5	0%	0%	20%	0%	0%	20%
A-40-3	0%	0%	40%	0%	0%	15%
A-40-4	0%	0%	20%	0%	0%	20%
A-40-5	0%	0%	20%	0%	0%	20%
B-10-3	0%	0%	40%	40%	15%	40%
B-10-4	20%	0%	15%	20%	35%	100%
B-10-5	80%	0%	10%	0%	0%	40%
B-20-3	0%	40%	40%	0%	0%	10%
B-20-4	0%	0%	20%	0%	0%	40%
B-20-5	0%	0%	25%	0%	0%	5%
B-30-3	0%	0%	20%	0%	0%	20%
B-30-4	0%	0%	10%	0%	0%	60%
B-30-5	0%	0%	20%	0%	0%	40%
B-40-3	0%	0%	15%	0%	0%	20%
B-40-4	0%	0%	5%	0%	0%	20%
B-40-5	0%	0%	20%	0%	0%	10%
C-10-3	100%	0%	35%	20%	0%	25%
C-10-4	0%	60%	55%	0%	0%	10%
C-10-5	0%	0%	15%	0%	0%	40%
C-20-3	0%	0%	10%	0%	0%	10%
C-20-4	0%	0%	5%	0%	0%	5%
C-20-5	0%	0%	15%	0%	0%	15%
C-30-3	0%	0%	5%	0%	0%	20%
C-30-4	0%	0%	20%	0%	0%	10%
C-30-5	0%	0%	5%	0%	0%	40%
C-40-3	0%	0%	25%	0%	0%	20%
C-40-4	0%	0%	5%	0%	0%	20%
C-40-5	0%	0%	100%	0%	0%	40%

Tabela 44: Percentual da quantidade de vezes em que os algoritmos de formigas encontraram o melhor resultado para instâncias de pequeno porte.

	Assimétrico			Simétrico		
Instância	*AS*	*ACS*	*MS-ACS*	*AS*	*ACS*	*MS-ACS*
A-50-3	0%	0%	15%	0%	0%	20%
A-50-4	0%	0%	20%	0%	0%	35%
A-50-5	0%	0%	20%	0%	0%	20%
A-100-3	0%	0%	20%	0%	0%	5%
A-100-4	0%	0%	5%	0%	0%	15%
A-100-5	0%	0%	15%	0%	0%	5%
A-200-3	0%	0%	10%	0%	0%	25%
A-200-4	0%	0%	20%	0%	0%	20%
A-200-5	0%	0%	10%	0%	0%	5%
A-500-3	0%	0%	5%	0%	0%	15%
A-500-4	0%	0%	10%	0%	0%	15%
A-500-5	0%	0%	100%	0%	0%	5%
B-50-3	0%	0%	5%	0%	0%	20%
B-50-4	0%	0%	10%	0%	0%	10%
B-50-5	0%	0%	10%	0%	0%	10%
B-100-3	0%	0%	10%	0%	0%	20%
B-100-4	0%	0%	15%	0%	0%	20%
B-100-5	0%	0%	5%	0%	0%	25%
B-200-3	0%	0%	10%	0%	0%	5%
B-200-4	0%	0%	10%	0%	0%	10%
B-200-5	0%	0%	5%	0%	0%	5%
B-500-3	0%	0%	5%	0%	0%	20%
B-500-4	0%	0%	5%	0%	0%	5%
B-500-5	0%	0%	10%	0%	0%	20%
C-50-3	0%	0%	100%	0%	0%	20%
C-50-4	0%	0%	40%	0%	0%	20%
C-50-5	0%	0%	20%	0%	0%	15%
C-100-3	0%	0%	5%	0%	0%	20%
C-100-4	0%	0%	20%	0%	0%	10%
C-100-5	0%	0%	15%	0%	0%	5%
C-200-3	0%	0%	25%	0%	0%	15%
C-200-4	0%	0%	20%	0%	0%	20%
C-200-5	0%	0%	5%	0%	0%	10%
C-500-3	0%	0%	10%	0%	25%	0%
C-500-4	0%	0%	10%	0%	0%	5%
C-500-5	0%	0%	5%	0%	0%	15%

Tabela 45: Percentual da quantidade de vezes em que os algoritmos de formigas encontraram o melhor resultado para instâncias de médio e grande porte.

APÊNDICE C – Resultados dos Algoritmos Evolucionários

	AG			AM			AM-VC		
Instância	Solução	Média	Tempo	Solução	Média	Tempo	Solução	Média	Tempo
A-10-3	478,42	682,00	0,05	478,42	478,42	2,31	478,42	478,42	1,65
A-10-4	601,08	601,08	0,01	601,08	601,08	1,79	523,57	523,57	1,74
A-10-5	482,60	571,00	0,09	482,60	482,60	4,02	482,60	482,60	2,69
A-20-3	618,00	717,67	2,39	603,92	603,92	5,59	627,67	627,67	8,51
A-20-4	585,48	717,55	2,37	640,47	640,47	5,53	549,33	549,33	8,56
A-20-5	538,67	567,67	3,16	461,17	461,17	8,14	430,33	430,33	12,70
A-30-3	1128,25	1348,42	3,32	1222,08	1222,08	9,06	1007,67	1007,67	24,49
A-30-4	683,97	824,57	8,35	714,78	714,78	20,51	601,23	601,23	64,26
A-30-5	666,72	862,32	5,41	681,13	681,13	14,96	674,25	674,25	41,92
A-40-3	1077,50	1296,42	9,52	1207,25	1207,25	26,11	899,17	899,17	109,48
A-40-4	1007,25	1122,63	9,42	1140,88	1140,88	26,39	895,27	895,27	117,21
A-40-5	721,55	871,98	12,27	750,82	750,82	31,51	654,65	654,65	142,32
B-10-3	778,83	778,83	0,01	778,83	778,83	1,78	729,50	729,50	1,23
B-10-4	306,90	353,83	0,07	306,90	306,90	2,85	306,90	306,90	1,82
B-10-5	434,75	599,75	1,29	434,75	434,75	3,97	434,75	434,75	3,64
B-20-3	992,17	1293,58	2,09	1063,75	1063,75	5,54	896,33	896,33	9,48
B-20-4	1010,33	1229,50	1,48	1128,17	1128,17	4,19	941,22	941,22	7,30
B-20-5	938,00	1322,83	2,05	1063,17	1063,17	5,23	871,50	871,50	9,72
B-30-3	1254,83	1510,25	4,89	1302,17	1302,17	13,26	996,67	996,67	38,38
B-30-4	1286,47	1703,90	5,44	1302,50	1302,50	13,57	937,17	937,17	37,96
B-30-5	1062,45	1177,77	7,05	1069,05	1069,05	18,94	872,48	872,48	51,20
B-40-3	1788,08	1956,83	6,94	1668,08	1668,08	18,86	1506,08	1506,08	76,03
B-40-4	2171,02	2583,33	3,85	2312,53	2312,53	10,52	2106,05	2106,05	42,32
B-40-5	1302,78	1575,72	9,88	1502,95	1502,95	27,03	1187,70	1187,70	118,84
C-10-3	359,25	429,00	0,13	359,25	359,25	4,63	359,25	359,25	3,55
C-10-4	307,10	307,10	0,04	307,10	307,10	7,16	307,10	307,10	5,29
C-10-5	566,58	566,58	0,02	566,58	566,58	3,22	566,58	566,58	2,13
C-20-3	732,25	852,25	6,21	777,67	777,67	14,09	746,33	746,33	19,61
C-20-4	760,57	920,70	7,00	714,30	714,30	18,24	734,70	734,70	27,05
C-20-5	792,25	1070,83	8,40	837,83	837,83	22,28	766,45	766,45	40,37
C-30-3	1087,50	1434,25	5,66	1057,50	1057,50	14,87	1092,08	1092,08	42,91
C-30-4	1063,00	1169,20	13,15	1180,53	1180,53	36,04	1014,40	1014,40	103,56
C-30-5	873,80	975,53	31,04	886,88	886,88	78,99	795,67	795,67	224,18
C-40-3	1600,42	1901,08	9,00	1664,08	1664,08	24,39	1342,75	1342,75	103,00
C-40-4	1191,30	1558,13	35,05	1230,13	1230,13	68,39	1156,00	1156,00	294,77
C-40-5	1181,30	1601,03	43,13	1327,13	1327,13	136,40	968,83	968,83	615,26

Tabela 46: Resultados dos algoritmos evolucionários para instâncias assimétricas de pequeno porte.

	AG			AM			AM-VC		
Instância	Solução	Média	Tempo	Solução	Média	Tempo	Solução	Média	Tempo
A-50-3	1816,17	1816,17	0,22	1813,33	1813,33	14,56	1505,25	1505,25	69,36
A-50-4	1302,43	1600,10	9,68	1295,50	1295,50	27,04	1050,32	1050,32	164,29
A-50-5	1055,37	1403,30	11,45	1229,07	1229,07	34,06	1035,63	1035,63	214,37
A-100-3	2244,08	2244,08	35,59	2513,67	2513,67	162,38	2153,25	2153,25	2494,66
A-100-4	2745,67	2745,67	22,45	3052,58	3052,58	97,58	2171,22	2171,22	1015,25
A-100-5	2053,12	2053,12	74,03	2275,20	2275,20	156,22	1881,10	1881,10	2433,45
A-200-3	4826,92	4826,92	99,78	5033,92	5033,92	455,94	4939,08	4939,08	382,37
A-200-4	4622,98	4622,98	79,75	3877,90	3877,90	314,66	3939,28	3939,28	273,20
A-200-5	3282,55	3282,55	299,59	3122,17	3122,17	1279,33	3036,73	3036,73	1071,22
A-500-3	15507,92	15507,92	202,31	8420,00	8575,12	3245,79	8138,00	8262,89	3522,88
A-500-4	11213,37	11213,37	577,51	10145,22	10145,22	9862,60	10069,72	10069,72	79924,93
A-500-5	8755,70	8755,70	1696,56	8723,58	8723,58	52628,69	8842,98	8842,98	25123,51
B-50-3	2622,42	2849,08	5,73	2133,83	2133,83	16,20	1970,83	1970,83	87,27
B-50-4	1722,92	2264,52	11,63	1815,73	1815,73	33,42	1559,18	1559,18	216,94
B-50-5	1765,98	2106,82	10,16	1953,25	1953,25	26,98	1643,40	1643,40	172,15
B-100-3	3094,50	3094,50	24,12	4050,25	4050,25	149,63	3263,08	3263,08	1514,92
B-100-4	2595,93	2595,93	44,81	2995,48	2995,48	185,83	2322,25	2322,25	2861,51
B-100-5	2720,32	2720,32	28,27	2258,20	2258,20	141,19	2298,88	2298,88	1925,41
B-200-3	7072,83	7072,83	68,19	5879,42	6401,92	48,08	5678,58	6395,92	2730,40
B-200-4	4732,60	4732,60	125,35	4857,43	4857,43	1096,18	4698,67	4936,62	3290,25
B-200-5	4349,92	4349,92	189,80	3903,68	4180,38	120,07	3676,87	3676,87	480,23
B-500-3	16320,83	16320,83	784,19	7739,42	7911,33	4181,11	7321,15	7529,48	4348,91
B-500-4	12133,73	12133,73	2679,05	5843,40	5908,70	5324,02	5629,68	5728,05	5509,98
B-500-5	10296,57	10296,57	2786,83	4877,92	4986,34	4439,61	4677,92	4722,82	4699,01
C-50-3	1821,33	2361,33	6,73	2107,75	2107,75	26,09	1812,08	1812,08	146,15
C-50-4	1890,00	2134,02	17,51	1868,15	1868,15	61,38	1471,77	1471,77	361,93
C-50-5	1406,83	1595,93	44,72	1206,95	1206,95	136,53	958,83	958,83	845,88
C-100-3	3278,83	3278,83	29,75	2736,67	2736,67	152,63	2824,50	2824,50	2333,20
C-100-4	3203,82	3203,82	23,18	3324,60	3324,60	125,33	2709,43	2709,43	1713,72
C-100-5	2156,57	2156,57	86,04	2117,50	2117,50	383,57	1797,63	1797,63	6923,46
C-200-3	5687,75	5687,75	111,95	4968,42	4968,42	291,75	4624,50	5315,50	3604,95
C-200-4	4610,53	4610,53	139,90	4365,80	4416,70	95,66	3928,60	4217,35	4442,73
C-200-5	3857,33	3857,33	208,51	3745,17	3745,17	610,02	3530,17	3640,73	5804,66
C-500-3	15086,75	15086,75	766,71	13093,50	13093,50	80000,00	13264,42	13264,42	36177,99
C-500-4	12034,60	12034,60	3130,60	5619,40	5655,16	6854,56	5429,89	5557,98	7098,22
C-500-5	9327,33	9327,33	1383,40	4763,25	4776,76	7921,19	4622,58	4719,22	8437,12

Tabela 47: Resultados dos algoritmos evolucionários para instâncias assimétricas de médio e grande porte.

	AG			AM			AM-VC		
Instância	Solução	Média	Tempo	Solução	Média	Tempo	Solução	Média	Tempo
A-10-3	582,50	606,50	0,59	582,50	582,50	1,69	545,92	545,92	1,97
A-10-4	569,75	576,05	0,66	460,00	569,75	3,12	460,00	460,00	1,48
A-10-5	371,93	388,80	1,41	371,93	371,93	3,80	371,93	371,93	4,86
A-20-3	890,08	956,50	1,22	735,92	735,92	4,54	734,33	734,33	7,46
A-20-4	428,90	563,87	2,49	421,55	458,95	7,13	377,15	429,77	7,28
A-20-5	439,50	655,80	3,77	435,92	435,92	13,67	352,42	357,47	11,20
A-30-3	1093,75	1307,75	2,29	1059,67	1059,67	5,75	788,25	788,25	8,79
A-30-4	999,00	1107,02	3,17	927,22	927,22	7,48	715,78	715,78	10,98
A-30-5	662,17	890,92	7,38	667,35	667,35	17,33	544,00	544,00	31,13
A-40-3	1222,92	1372,25	4,55	1264,00	1264,00	13,01	865,00	865,00	25,28
A-40-4	862,45	980,30	9,53	881,07	881,07	31,82	703,53	703,53	72,35
A-40-5	706,93	915,32	9,84	765,05	765,05	29,77	533,10	533,10	67,66
B-10-3	834,67	834,67	0,01	834,67	834,67	4,55	834,67	834,67	0,70
B-10-4	493,58	580,05	0,08	493,58	493,58	4,05	493,58	493,58	2,22
B-10-5	773,45	810,10	0,84	773,45	773,45	2,56	726,35	726,35	2,25
B-20-3	1062,92	1416,00	1,67	1133,83	1133,83	5,13	936,75	936,75	5,18
B-20-4	959,83	1047,37	2,46	1015,85	1015,85	7,57	914,48	914,48	7,63
B-20-5	724,80	836,43	2,41	750,10	750,10	8,03	641,57	641,57	9,50
B-30-3	1072,75	1322,67	3,52	1120,17	1120,17	12,07	957,58	957,58	19,33
B-30-4	905,10	1231,33	4,18	1056,68	1056,68	13,71	696,30	696,30	24,44
B-30-5	663,67	963,07	7,54	732,23	732,23	25,98	603,52	603,52	46,03
B-40-3	1141,67	1640,42	7,71	1322,00	1322,00	22,21	988,83	988,83	52,54
B-40-4	1268,08	1505,63	7,81	1181,47	1181,47	22,17	909,97	909,97	53,63
B-40-5	1081,07	1363,90	9,00	997,03	997,03	27,68	850,32	850,32	64,81
C-10-3	558,67	558,67	0,01	558,67	558,67	2,33	558,67	558,67	1,50
C-10-4	408,45	416,65	0,07	408,45	408,45	5,36	408,45	408,45	2,65
C-10-5	409,60	437,73	1,14	437,73	437,73	4,52	409,60	409,60	2,65
C-20-3	695,25	900,92	3,63	736,75	736,75	11,10	698,67	698,67	11,49
C-20-4	618,00	690,27	5,21	622,45	622,45	19,48	446,05	446,05	23,72
C-20-5	742,75	947,43	3,89	859,37	859,37	11,14	787,20	787,20	11,89
C-30-3	1198,42	1320,00	5,53	1243,08	1243,08	15,55	880,25	880,25	29,89
C-30-4	1018,68	1275,75	5,79	932,73	932,73	17,45	910,70	910,70	29,84
C-30-5	650,00	924,90	13,83	685,00	685,00	42,60	586,07	586,07	89,67
C-40-3	1035,00	1195,75	14,47	986,92	986,92	51,47	804,75	804,75	106,70
C-40-4	1031,20	1147,40	13,14	1064,00	1064,00	41,89	750,00	750,00	107,86
C-40-5	745,17	853,43	33,25	768,50	768,50	90,18	591,83	591,83	248,85

Tabela 48: Resultados dos algoritmos evolucionários para instâncias simétricas de pequeno porte.

	AG			AM			AM-VC		
Instância	Solução	Média	Tempo	Solução	Média	Tempo	Solução	Média	Tempo
A-50-3	1567,42	1808,92	5,61	1232,83	1232,83	17,14	1122,50	1122,50	48,94
A-50-4	1141,42	1517,82	9,44	1062,67	1062,67	28,94	932,72	975,33	70,40
A-50-5	811,17	996,52	20,61	918,70	918,70	67,62	682,50	682,50	208,37
A-100-3	2591,92	2591,92	19,17	2343,67	2343,67	76,10	1714,42	1714,42	331,55
A-100-4	2018,00	2018,00	37,22	1789,83	1789,83	143,45	1405,10	1405,10	784,58
A-100-5	1529,08	1529,08	48,68	1336,98	1336,98	192,94	1208,80	1208,80	864,20
A-200-3	4765,75	4765,75	69,95	3640,50	3640,50	40,40	3451,83	3451,83	35,68
A-200-4	4251,45	4251,45	144,25	2695,40	2695,40	93,30	2898,97	2898,97	85,15
A-200-5	4067,83	4067,83	55,97	2772,32	2772,32	33,34	2674,17	2674,17	30,14
A-500-3	13354,33	13354,33	2230,96	7634,00	7634,00	2424,91	7499,33	7499,33	2149,67
A-500-4	10150,90	10150,90	6171,55	6014,33	6014,33	6444,95	6090,40	6090,40	6055,12
A-500-5	9921,38	9921,38	539,66	5563,57	5563,57	532,67	5437,25	5437,25	488,85
B-50-3	1642,50	1948,83	10,87	1594,50	1689,08	33,04	1160,92	1261,08	90,67
B-50-4	1413,90	1777,38	14,65	1269,70	1399,42	38,08	995,92	995,92	132,05
B-50-5	872,22	1639,02	12,27	1391,80	1409,22	36,91	934,83	934,83	117,81
B-100-3	3329,33	3329,33	15,11	3033,58	3921,25	44,67	2301,75	2431,58	286,94
B-100-4	3170,58	3170,58	15,37	3202,00	3401,87	43,81	2290,73	2290,73	280,29
B-100-5	1750,45	1750,45	78,69	1573,88	1652,22	235,79	1287,05	1287,05	1460,16
B-200-3	5396,25	5396,25	75,89	3896,42	3896,42	40,40	3953,75	3953,75	40,33
B-200-4	4483,07	4483,07	168,57	3035,28	3035,28	86,07	3116,90	3116,90	83,14
B-200-5	3856,73	3856,73	207,00	2486,12	2486,12	325,33	2451,38	2451,38	321,55
B-500-3	13864,33	13864,33	1297,52	5754,00	8017,30	6219,42	5462,00	6512,00	6521,00
B-500-4	12047,92	12047,92	3061,45	6239,47	6239,47	2948,75	6386,87	6386,87	4979,28
B-500-5	10456,97	10456,97	2090,46	5597,78	5597,78	2107,85	5517,72	5517,72	1821,10
C-50-3	1653,75	1653,75	0,50	1421,58	1421,58	32,78	1070,75	1080,08	75,06
C-50-4	1470,00	1710,65	11,18	1575,20	1794,75	29,30	1092,28	1092,28	125,97
C-50-5	1192,20	1535,30	25,79	1176,22	1176,22	85,80	893,87	894,67	248,59
C-100-3	2268,50	2268,50	23,16	2153,92	2153,92	90,48	1685,00	1685,00	467,24
C-100-4	1828,20	1828,20	51,56	1496,53	1496,53	238,85	1239,40	1239,40	1208,47
C-100-5	1802,48	1802,48	44,74	2039,80	2039,80	144,16	1177,70	1177,70	763,87
C-200-3	5215,92	5215,92	86,90	3344,17	3344,17	45,61	3248,58	3248,58	44,33
C-200-4	4367,27	4367,27	78,79	2734,00	2734,00	39,09	2785,03	2785,03	35,64
C-200-5	2809,67	2809,67	295,49	2168,50	2168,50	156,75	2286,58	2286,58	142,15
C-500-3	12484,00	12484,00	1800,76	7464,25	7651,42	5626,34	7134,67	7372,21	5801,98
C-500-4	11432,55	11432,55	562,30	6084,60	6084,60	527,62	6200,20	6200,20	468,10
C-500-5	8306,83	8306,83	3392,14	5064,13	5064,13	3414,83	5193,67	5193,67	2923,51

Tabela 49: Resultados dos algoritmos evolucionários para instâncias simétricas de médio e grande porte.

	Assimétrico			Simétrico		
Instância	*AG*	*AM*	*AM-VC*	*AG*	*AM*	*AM-VC*
A-10-3	100%	100%	100%	100%	100%	100%
A-10-4	0%	0%	100%	0%	0%	100%
A-10-5	100%	100%	100%	100%	100%	100%
A-20-3	0%	0%	0%	0%	0%	0%
A-20-4	0%	0%	0%	0%	0%	0%
A-20-5	0%	0%	0%	0%	0%	0%
A-30-3	0%	0%	0%	0%	0%	0%
A-30-4	0%	0%	0%	0%	0%	0%
A-30-5	0%	0%	0%	0%	0%	0%
A-40-3	0%	0%	0%	0%	0%	0%
A-40-4	0%	0%	0%	0%	0%	0%
A-40-5	0%	0%	0%	0%	0%	0%
B-10-3	0%	0%	100%	0%	0%	100%
B-10-4	100%	100%	100%	100%	100%	100%
B-10-5	100%	100%	100%	100%	100%	100%
B-20-3	0%	0%	0%	0%	0%	0%
B-20-4	0%	0%	0%	0%	0%	0%
B-20-5	0%	0%	0%	0%	0%	0%
B-30-3	0%	0%	0%	0%	0%	0%
B-30-4	0%	0%	0%	0%	0%	0%
B-30-5	0%	0%	0%	0%	0%	0%
B-40-3	0%	0%	0%	0%	0%	0%
B-40-4	0%	0%	0%	0%	0%	0%
B-40-5	0%	0%	0%	0%	0%	0%
C-10-3	100%	100%	100%	100%	100%	100%
C-10-4	100%	100%	100%	100%	100%	100%
C-10-5	10%	10%	10%	10%	10%	10%
C-20-3	0%	0%	0%	0%	0%	0%
C-20-4	0%	0%	0%	0%	0%	0%
C-20-5	0%	0%	0%	0%	0%	0%
C-30-3	0%	0%	0%	0%	0%	0%
C-30-4	0%	0%	0%	0%	0%	0%
C-30-5	0%	0%	0%	0%	0%	0%
C-40-3	0%	0%	0%	0%	0%	0%
C-40-4	0%	0%	0%	0%	0%	0%
C-40-5	0%	0%	0%	0%	0%	0%

Tabela 50: Percentual da quantidade de vezes em que os algoritmos evolucionários encontraram o melhor resultado para instâncias de pequeno porte.

	Assimétrico			Simétrico		
Instância	*AG*	*AM*	*AM-VC*	*AG*	*AM*	*AM-VC*
A-50-3	0%	0%	0%	0%	0%	0%
A-50-4	0%	0%	0%	0%	0%	0%
A-50-5	0%	0%	0%	0%	0%	0%
A-100-3	0%	0%	0%	0%	0%	0%
A-100-4	0%	0%	0%	0%	0%	0%
A-100-5	0%	0%	0%	0%	0%	0%
A-200-3	0%	0%	0%	0%	0%	0%
A-200-4	0%	0%	0%	0%	0%	0%
A-200-5	0%	0%	0%	0%	0%	0%
A-500-3	0%	0%	0%	0%	0%	0%
A-500-4	0%	0%	0%	0%	0%	0%
A-500-5	0%	0%	0%	0%	0%	0%
B-50-3	0%	0%	0%	0%	0%	0%
B-50-4	0%	0%	0%	0%	0%	0%
B-50-5	0%	0%	0%	0%	0%	0%
B-100-3	0%	0%	0%	0%	0%	0%
B-100-4	0%	0%	0%	0%	0%	0%
B-100-5	0%	0%	0%	0%	0%	0%
B-200-3	0%	0%	0%	0%	0%	0%
B-200-4	0%	0%	0%	0%	0%	0%
B-200-5	0%	0%	0%	0%	0%	0%
B-500-3	0%	0%	0%	0%	0%	0%
B-500-4	0%	0%	0%	0%	0%	0%
B-500-5	0%	0%	0%	0%	0%	0%
C-50-3	0%	0%	0%	0%	0%	0%
C-50-4	0%	0%	0%	0%	0%	0%
C-50-5	0%	0%	0%	0%	0%	0%
C-100-3	0%	0%	0%	0%	0%	0%
C-100-4	0%	0%	0%	0%	0%	0%
C-100-5	0%	0%	0%	0%	0%	0%
C-200-3	0%	0%	0%	0%	0%	0%
C-200-4	0%	0%	0%	0%	0%	0%
C-200-5	0%	0%	0%	0%	0%	0%
C-500-3	0%	0%	0%	0%	0%	0%
C-500-4	0%	0%	0%	0%	0%	0%
C-500-5	0%	0%	0%	0%	0%	0%

Tabela 51: Percentual da quantidade de vezes em que os algoritmos evolucionários encontraram o melhor resultado para instâncias de médio e grande porte.

APÊNDICE D – Resultados dos algoritmos ILS e GRASP

	ILS			GRASP		
Instância	Solução	Média	Tempo	Solução	Média	Tempo
A-10-3	504,83	609,62	0,08	478,42	593,07	0,02
A-10-4	523,57	694,47	0,06	523,57	661,30	0,02
A-10-5	482,60	505,90	0,03	482,60	519,26	0,03
A-20-3	550,00	636,16	0,23	591,33	724,99	0,10
A-20-4	439,67	591,78	0,18	455,48	682,88	0,10
A-20-5	399,92	525,64	0,23	504,33	600,43	0,13
A-30-3	716,33	885,25	1,96	888,33	1225,32	0,27
A-30-4	447,50	560,22	4,69	599,65	706,10	0,68
A-30-5	454,98	580,15	2,75	668,97	815,49	0,43
A-40-3	768,75	921,98	8,07	1068,50	1195,64	1,31
A-40-4	685,20	812,46	8,23	914,65	1049,52	1,27
A-40-5	506,82	642,27	11,74	765,28	825,63	1,74
B-10-3	729,50	786,01	0,02	729,50	739,37	0,01
B-10-4	306,90	316,29	0,03	306,90	315,76	0,02
B-10-5	434,75	474,70	0,05	434,75	444,40	0,03
B-20-3	867,92	973,82	0,22	945,25	1094,75	0,10
B-20-4	938,58	1121,33	0,16	1135,75	1211,15	0,07
B-20-5	843,17	1028,19	0,19	983,67	1167,26	0,08
B-30-3	955,25	1036,64	4,08	1128,58	1281,36	0,45
B-30-4	802,65	976,89	3,94	1036,83	1263,50	0,45
B-30-5	678,97	796,06	5,33	772,83	1019,26	0,62
B-40-3	1071,25	1227,29	6,41	1437,42	1667,29	0,92
B-40-4	1448,92	1651,38	3,26	1715,83	2157,52	0,52
B-40-5	810,75	981,20	9,67	1232,83	1337,58	1,31
C-10-3	359,25	390,83	0,05	359,25	414,26	0,03
C-10-4	307,10	310,24	0,10	307,10	317,24	0,04
C-10-5	566,58	627,65	0,04	566,58	597,45	0,02
C-20-3	707,33	816,96	0,61	750,08	868,09	0,24
C-20-4	664,10	778,56	0,69	621,68	843,47	0,27
C-20-5	684,80	812,73	1,04	840,35	927,17	0,42
C-30-3	898,00	1007,93	3,74	1059,92	1212,67	0,49
C-30-4	810,07	929,40	10,26	933,00	1106,96	1,16
C-30-5	580,60	689,05	24,44	786,03	927,13	2,83
C-40-3	1114,33	1328,28	7,57	1525,25	1629,83	1,18
C-40-4	771,00	953,39	22,92	1084,42	1191,95	3,52
C-40-5	690,00	884,08	41,16	901,17	1105,02	6,39

Tabela 52: Resultados dos algoritmos ILS e GRASP para instâncias assimétricas de pequeno porte.

	ILS			GRASP		
Instância	Solução	Média	Tempo	Solução	Média	Tempo
A-100-5	1515,37	1648,72	73,55	2169,72	2353,24	3,45
A-200-3	4158,75	4436,83	1212,31	5476,25	5611,37	48,01
A-200-4	3683,87	3867,20	937,66	4793,12	4900,30	33,40
A-200-5	2816,97	2876,55	9304,11	3202,43	3504,61	134,98
A-50-3	1193,92	1193,92	1,34	2285,17	2285,17	0,06
A-50-4	930,17	1156,37	9,40	1389,47	1794,08	0,95
A-50-5	724,17	724,17	23,61	1281,65	1281,65	1,20
A-500-3	8253,17	8377,37	3379,83	9024,92	9263,18	2321,14
A-500-4	9850,12	9850,12	13993,92	11813,63	11813,63	17478,95
A-500-5	7785,12	7785,12	12377,00	9277,92	9277,92	1723,96
B-100-3	2614,92	2873,38	56,41	3679,42	4132,37	5,95
B-100-4	2045,60	2281,11	97,07	2751,55	3035,65	4,66
B-100-5	2091,97	2192,75	176,21	2770,20	3311,07	8,18
B-200-3	5195,25	5534,72	2342,20	7175,58	7323,85	87,05
B-200-4	4285,98	4559,47	1852,45	5349,48	5692,57	31,99
B-200-5	3226,33	3686,89	3054,02	4328,35	4629,66	36,88
B-50-3	1662,75	1662,75	10,90	2538,25	2716,92	0,58
B-50-4	1200,95	1200,95	25,06	1718,97	1724,02	1,28
B-50-5	1256,72	1498,28	20,18	1825,08	1825,08	1,18
B-500-3	7825,17	7979,82	9215,82	8281,17	8476,00	2117,83
B-500-4	5859,28	5953,86	9814,20	6362,32	6474,82	5434,82
B-500-5	5023,47	5075,54	9913,32	5377,88	5417,44	4049,65
C-100-3	2247,25	2617,68	78,23	3032,75	3292,35	8,63
C-100-4	2404,80	2840,56	53,83	3374,95	3526,37	2,52
C-100-5	1563,33	1738,07	201,68	2043,50	2244,81	10,21
C-200-3	4549,75	4899,65	1969,94	5887,50	6036,17	129,57
C-200-4	3624,50	4133,91	2381,87	4741,80	4866,49	12300,74
C-200-5	3410,60	3596,65	639,80	3819,33	4171,56	39,94
C-50-3	1511,17	1856,50	13,96	2156,08	2156,08	0,89
C-50-4	1587,80	1587,80	46,08	1874,47	1874,47	2,12
C-50-5	924,50	924,50	72,31	1461,20	1461,20	4,89
C-500-3	12456,58	12456,58	42238,86	14858,50	14858,50	868,61
C-500-4	5571,20	5647,65	8221,65	5987,90	6070,23	7046,16
C-500-5	4721,50	4769,04	9913,81	5081,67	5134,36	6219,65

Tabela 53: Resultados dos algoritmos ILS e GRASP para instâncias assimétricas de médio e grande porte.

	ILS			GRASP		
Instância	Solução	Média	Tempo	Solução	Média	Tempo
A-10-3	545,92	560,39	0,06	545,92	563,01	0,01
A-10-4	460,00	537,95	0,07	460,00	588,13	0,02
A-10-5	371,93	378,33	0,12	371,93	381,85	0,04
A-20-3	729,75	886,70	0,15	735,08	944,08	0,06
A-20-4	385,33	510,95	0,22	439,80	556,73	0,10
A-20-5	350,27	434,89	0,37	434,37	509,15	0,14
A-30-3	689,92	751,24	0,80	808,08	1052,43	0,14
A-30-4	634,27	794,84	1,20	882,70	1067,88	0,18
A-30-5	466,43	529,12	5,09	608,68	729,04	0,53
A-40-3	727,75	879,36	2,75	898,00	1165,48	0,43
A-40-4	622,55	713,50	9,43	735,97	875,26	1,74
A-40-5	459,75	537,10	7,30	569,77	728,37	1,16
B-10-3	834,67	851,03	0,04	834,67	843,63	0,01
B-10-4	493,58	508,75	0,10	493,58	525,77	0,03
B-10-5	726,35	747,27	0,09	726,35	802,95	0,03
B-20-3	950,00	1090,15	0,19	1073,00	1216,42	0,06
B-20-4	864,67	995,88	0,19	992,83	1102,15	0,08
B-20-5	676,15	753,28	0,18	767,17	873,73	0,08
B-30-3	738,08	976,10	2,31	1111,67	1262,97	0,23
B-30-4	714,05	824,48	1,94	763,37	1072,57	0,28
B-30-5	514,28	609,72	5,80	687,20	832,30	0,59
B-40-3	875,08	1097,10	4,17	1176,92	1370,76	0,83
B-40-4	724,50	931,66	5,80	1030,22	1241,01	0,80
B-40-5	743,30	897,33	7,50	993,58	1156,97	1,07
C-10-3	568,83	613,45	0,02	558,67	578,56	0,01
C-10-4	408,45	429,17	0,03	408,45	419,40	0,02
C-10-5	409,60	487,08	0,06	409,60	458,54	0,02
C-20-3	634,50	775,44	0,33	664,67	829,60	0,13
C-20-4	457,25	644,24	0,79	625,40	743,39	0,26
C-20-5	711,87	839,28	0,34	802,37	947,57	0,13
C-30-3	847,33	1007,62	2,63	1071,42	1261,38	0,31
C-30-4	751,27	888,38	1,85	952,03	1109,01	0,33
C-30-5	460,83	651,77	6,28	642,50	872,98	0,90
C-40-3	769,25	861,31	10,36	852,50	1037,49	1,77
C-40-4	630,40	788,78	7,85	826,90	1034,78	1,44
C-40-5	547,67	672,93	23,17	734,33	823,94	3,76

Tabela 54: Resultados dos algoritmos ILS e GRASP para instâncias simétricas de pequeno porte.

	ILS			GRASP		
Instância	Solução	Média	Tempo	Solução	Média	Tempo
A-100-3	1616,17	1792,78	123,90	2060,00	2302,18	12,36
A-100-4	1183,12	1392,68	222,32	1598,58	1797,78	19,34
A-100-5	1092,07	1199,36	400,81	1353,30	1508,14	18,98
A-200-3	3265,25	3446,10	56,23	3898,67	4082,82	5,85
A-200-4	2525,20	2589,67	116,23	2871,75	3084,71	12,67
A-200-5	2402,88	2504,20	52,15	3031,40	3210,26	4,42
A-50-3	957,33	1177,30	8,73	1359,50	1522,96	0,84
A-50-4	776,90	915,97	17,51	1026,35	1235,85	1,55
A-50-5	572,00	672,88	48,02	804,37	912,91	3,88
A-500-3	7206,58	7307,63	1140,75	7966,75	8102,98	230,10
A-500-4	5674,00	5674,00	2981,06	6452,10	6452,10	584,88
A-500-5	5456,35	5456,35	370,89	6116,68	6116,68	50,54
B-100-3	2001,92	2585,50	102,88	2556,50	3526,75	4,78
B-100-4	1758,78	2254,23	62,50	2752,60	3359,81	3,72
B-100-5	1193,45	1276,81	752,15	1580,85	1631,78	11,46
B-200-3	3438,08	3606,53	60,86	4408,92	4517,90	6,15
B-200-4	2776,47	2892,96	131,90	3301,28	3458,59	12,40
B-200-5	2201,38	2281,03	368,47	2706,73	2831,20	48,14
B-50-3	1066,00	1201,98	17,95	1383,42	1569,75	1,97
B-50-4	833,05	979,64	26,88	1213,10	1395,33	2,39
B-50-5	754,45	912,09	26,40	1127,45	1249,13	2,38
B-500-3	6808,67	6816,33	5909,00	8245,50	8400,54	5754,00
B-500-4	6122,40	6122,40	1493,68	6876,35	6876,35	277,56
B-500-5	5380,72	5380,72	3261,35	5913,83	5913,83	669,24
C-100-3	1432,25	1539,87	108,87	2009,42	2075,82	8,28
C-100-4	1222,80	1311,60	330,87	1629,00	1677,50	25,48
C-100-5	1036,75	1170,56	172,36	1404,93	1495,58	13,48
C-200-3	2891,50	2973,03	40,37	3524,58	3651,03	7,09
C-200-4	2540,00	2581,31	39,35	3052,95	3154,76	5,37
C-200-5	1961,83	2078,75	192,94	2463,83	2525,72	23,59
C-50-3	883,25	1073,63	14,38	1153,92	1464,03	1,54
C-50-4	921,03	1149,93	16,61	1261,37	1564,03	1,71
C-50-5	774,67	913,31	40,32	1095,35	1237,74	3,90
C-500-3	6578,50	6578,50	4432,00	8080,00	8111,25	4559,70
C-500-4	5938,78	5938,78	270,13	6493,00	6493,00	53,74
C-500-5	4784,50	4784,50	1487,02	5570,00	5570,00	320,74

Tabela 55: Resultados dos algoritmos ILS e GRASP para instâncias simétricas de médio e grande porte.

	Assimétrico		Simétrico	
Instância	*ILS*	*GRASP*	*ILS*	*GRASP*
A-10-3	0%	5%	0%	5%
A-10-4	40%	25%	40%	25%
A-10-5	20%	20%	20%	20%
A-20-3	0%	0%	0%	0%
A-20-4	5%	0%	5%	0%
A-20-5	0%	0%	0%	0%
A-30-3	0%	0%	0%	0%
A-30-4	0%	0%	0%	0%
A-30-5	0%	0%	0%	0%
A-40-3	0%	0%	0%	0%
A-40-4	0%	0%	0%	0%
A-40-5	0%	0%	0%	0%
B-10-3	15%	80%	15%	80%
B-10-4	80%	60%	80%	60%
B-10-5	40%	55%	40%	55%
B-20-3	0%	0%	0%	0%
B-20-4	0%	0%	0%	0%
B-20-5	0%	0%	0%	0%
B-30-3	0%	0%	0%	0%
B-30-4	0%	0%	0%	0%
B-30-5	0%	0%	0%	0%
B-40-3	0%	0%	0%	0%
B-40-4	0%	0%	0%	0%
B-40-5	5%	0%	5%	0%
C-10-3	60%	30%	60%	30%
C-10-4	85%	15%	85%	15%
C-10-5	10%	10%	10%	10%
C-20-3	0%	0%	0%	0%
C-20-4	0%	0%	0%	0%
C-20-5	0%	0%	0%	0%
C-30-3	0%	0%	0%	0%
C-30-4	0%	0%	0%	0%
C-30-5	0%	0%	0%	0%
C-40-3	0%	0%	0%	0%
C-40-4	0%	0%	0%	0%
C-40-5	0%	0%	0%	0%
C-40-5	0%	0%	0%	0%

Tabela 56: Percentual da quantidade de vezes em que os algoritmos ILS e GRASP encontraram o melhor resultado para instâncias de pequeno porte.

	Assimétrico		Simétrico	
Instância	*ILS*	*GRASP*	*ILS*	*GRASP*
A-50-3	100%	0%	100%	0%
A-50-4	0%	0%	0%	0%
A-50-5	100%	0%	100%	0%
A-100-3	0%	0%	0%	0%
A-100-4	0%	0%	0%	0%
A-100-5	0%	0%	0%	0%
A-200-3	0%	0%	0%	0%
A-200-4	0%	0%	0%	0%
A-200-5	0%	0%	0%	0%
A-500-3	0%	0%	0%	0%
A-500-4	0%	0%	0%	0%
A-500-5	0%	0%	0%	0%
B-50-3	0%	0%	0%	0%
B-50-4	0%	0%	0%	0%
B-50-5	0%	0%	0%	0%
B-100-3	0%	0%	0%	0%
B-100-4	0%	0%	0%	0%
B-100-5	0%	0%	0%	0%
B-200-3	0%	0%	0%	0%
B-200-4	0%	0%	0%	0%
B-200-5	0%	0%	0%	0%
B-500-3	0%	0%	0%	0%
B-500-4	0%	0%	0%	0%
B-500-5	0%	0%	0%	0%
C-50-3	0%	0%	0%	0%
C-50-4	0%	0%	0%	0%
C-50-5	0%	0%	0%	0%
C-100-3	0%	0%	0%	0%
C-100-4	0%	0%	0%	0%
C-100-5	0%	0%	0%	0%
C-200-3	0%	0%	0%	0%
C-200-4	0%	0%	0%	0%
C-200-5	0%	0%	0%	0%
C-500-3	0%	0%	0%	0%
C-500-4	0%	0%	0%	0%
C-500-5	0%	0%	0%	0%

Tabela 57: Percentual da quantidade de vezes em que os algoritmos ILS e GRASP encontraram o melhor resultado para instâncias de médio e grande porte.

APÊNDICE E – Resultados dos Algoritmos Hibridizados

	GRASP-ILS			AM-VC-ILS			MS-ACS-ILS		
Instância	Solução	Média	Tempo	Solução	Média	Tempo	Solução	Média	Tempo
A-10-3	504,83	592,83	0,15	478,42	478,42	1,78	478,42	478,42	0,65
A-10-4	523,57	523,57	0,09	523,57	523,57	1,19	523,57	523,57	0,55
A-10-5	482,60	495,25	0,07	482,60	482,60	0,83	482,60	492,09	0,36
A-20-3	585,83	643,73	0,17	519,67	519,67	1,64	539,75	592,50	0,82
A-20-4	443,90	549,19	0,14	492,35	492,35	1,66	443,90	478,55	0,65
A-20-5	424,78	509,44	0,25	415,67	415,67	2,34	418,60	458,28	0,85
A-30-3	868,17	944,42	1,47	780,50	780,50	15,12	711,50	788,05	4,94
A-30-4	543,63	575,56	3,55	483,83	483,83	47,34	471,13	488,99	14,40
A-30-5	610,38	655,53	3,07	454,80	454,80	31,15	491,83	508,54	9,78
A-40-3	833,50	905,62	22,32	880,83	880,83	13,78	707,75	740,35	11,50
A-40-4	811,88	860,02	19,48	861,60	861,60	17,55	772,50	784,79	13,38
A-40-5	609,23	644,68	26,07	634,22	634,22	18,49	500,28	527,99	14,42
B-10-3	729,50	739,37	0,03	729,50	729,50	0,39	729,50	729,70	0,25
B-10-4	306,90	306,90	0,05	306,90	306,90	0,61	306,90	306,90	0,33
B-10-5	434,75	441,31	0,08	434,75	434,75	1,00	434,75	441,31	0,48
B-20-3	921,33	965,60	0,15	905,58	905,58	1,87	869,33	878,87	0,80
B-20-4	895,25	1078,82	0,12	955,58	955,58	1,45	888,83	944,27	0,76
B-20-5	897,83	985,63	0,13	949,83	949,83	1,63	894,50	894,50	0,67
B-30-3	971,17	1030,57	2,37	952,08	952,08	28,66	824,67	884,93	9,12
B-30-4	907,77	973,17	2,39	928,97	928,97	30,75	667,00	815,18	9,34
B-30-5	786,30	831,58	3,67	742,72	742,72	44,49	722,17	745,79	13,07
B-40-3	1224,00	1252,22	16,58	1447,50	1447,50	9,76	945,33	1013,30	13,04
B-40-4	1550,92	1683,31	7,62	1754,67	1754,67	5,53	1384,48	1477,73	5,19
B-40-5	929,43	983,39	23,14	1265,97	1265,97	24,72	851,60	879,80	14,32
C-10-3	359,25	382,58	0,12	359,25	359,25	1,40	359,25	359,25	0,64
C-10-4	307,10	309,35	0,19	307,10	307,10	2,78	307,10	307,10	1,09
C-10-5	566,58	580,57	0,09	566,58	566,58	1,16	566,58	594,55	0,53
C-20-3	712,00	765,43	0,46	711,50	711,50	5,80	650,25	662,45	2,22
C-20-4	715,47	760,35	0,52	659,85	659,85	6,29	706,15	731,57	1,99
C-20-5	771,83	809,05	0,70	743,68	743,68	8,26	719,67	742,60	2,71
C-30-3	969,83	1033,23	2,47	852,92	852,92	30,13	836,75	907,37	9,92
C-30-4	962,27	988,36	6,32	815,27	815,27	80,32	730,00	797,90	24,08
C-30-5	676,07	696,66	13,96	575,00	575,00	187,61	558,77	599,14	62,12
C-40-3	1268,50	1348,10	18,13	1505,58	1505,58	12,06	1012,92	1120,95	11,69
C-40-4	940,80	982,00	51,83	946,25	946,25	38,93	690,13	767,08	34,84
C-40-5	689,70	842,70	95,91	978,73	978,73	72,14	646,17	678,15	53,83

Tabela 58: Resultados dos algoritmos híbridos para instâncias assimétricas de pequeno porte.

	GRASP-ILS			AM-VC-ILS			MS-ACS-ILS		
Instância	Solução	Média	Tempo	Solução	Média	Tempo	Solução	Média	Tempo
A-50-3	1379,08	1419,55	22,18	1229,08	1229,08	8,90	1247,92	1325,75	16,64
A-50-4	911,50	964,38	55,96	1095,70	1095,70	44,31	832,82	854,41	35,97
A-50-5	834,65	862,79	68,74	921,57	921,57	24,88	756,18	777,60	42,63
A-100-3	1989,08	2035,42	1148,82	2095,67	2276,58	367,42	1484,00	1527,33	369,80
A-100-4	2001,92	2101,64	273,97	2413,23	2413,23	252,93	1608,55	1670,05	203,31
A-100-5	1515,45	1567,98	546,62	1605,53	1605,53	312,16	1231,73	1255,46	316,66
A-200-3	2968,70	3130,29	6345,43	2813,07	3118,58	3464,85	2898,67	3085,50	1465,20
A-200-4	2469,76	2617,74	5297,10	2447,58	2660,45	4424,48	2516,65	2516,65	246,29
A-200-5	1961,66	2040,09	25825,40	1874,88	2026,55	25613,32	1847,15	1863,95	4014,53
A-500-3	7547,66	8142,69	12834,45	7359,73	8108,92	7779,48	7257,17	7257,17	24664,36
A-500-4	5650,13	5832,88	79243,49	5436,88	6027,65	64228,06	5502,38	5502,38	80000,00
A-500-5	4890,83	5086,29	80000,00	4533,97	4818,27	80000,00	4516,82	4585,74	80000,00
B-50-3	1565,00	1679,07	36,21	1649,17	1649,17	24,57	1387,75	1511,85	22,39
B-50-4	1153,23	1255,69	82,01	1458,85	1458,85	51,30	1013,30	1124,13	49,67
B-50-5	1290,77	1335,52	61,32	1425,30	1425,30	45,15	1129,80	1190,48	35,29
B-100-3	2487,58	2700,72	450,66	2848,92	2848,92	285,32	1973,17	2050,45	814,93
B-100-4	2127,02	2181,18	736,22	2134,88	2134,88	453,94	1524,52	1578,90	457,57
B-100-5	2108,25	2157,02	485,54	2405,42	2405,42	329,78	1622,30	1706,71	294,77
B-200-3	3568,34	3682,17	11895,07	3452,29	3659,74	6808,31	3503,75	3725,50	1011,69
B-200-4	2608,12	2956,92	8707,33	2693,13	2876,31	8277,81	2717,80	2904,65	1646,46
B-200-5	2188,19	2342,46	13732,81	2159,68	2410,11	10918,29	2094,72	2297,30	2873,25
B-500-3	7491,29	7611,06	41967,26	7401,32	7644,81	29133,67	7065,07	7024,26	26072,35
B-500-4	5636,86	6056,64	57029,06	5545,84	6235,72	34342,95	5522,99	5593,67	26284,72
B-500-5	4701,10	5099,23	41096,61	4683,40	5066,39	29020,00	4675,96	4692,42	29186,31
C-50-3	1550,75	1640,15	48,23	1811,75	1811,75	34,58	1237,75	1296,82	28,61
C-50-4	1243,20	1316,74	118,43	1639,10	1639,10	76,51	938,75	1038,91	68,82
C-50-5	844,70	867,75	262,59	1019,83	1019,83	182,89	621,92	681,86	158,85
C-100-3	2160,50	2338,08	629,58	2404,92	2404,92	377,11	1507,25	1551,25	373,38
C-100-4	2266,00	2519,89	631,29	3048,77	3048,77	279,44	1586,05	1709,31	254,98
C-100-5	1552,00	1607,86	1885,12	1848,50	1848,50	1371,66	1033,20	1067,98	1059,64
C-200-3	2901,73	2976,10	10464,77	2847,45	3033,06	6763,89	2660,08	2757,25	1695,67
C-200-4	2384,62	2477,67	5089,34	2350,91	2519,74	5906,70	2264,55	2300,35	2023,57
C-200-5	1987,28	2152,94	3389,34	2006,49	2122,45	1833,06	1912,93	2007,33	13425,94
C-500-3	6936,72	7337,89	80000,00	7014,33	6941,53	80000,00	6641,65	6674,65	80000,00
C-500-4	5232,35	5930,93	48711,49	5309,54	5793,89	41472,76	5209,91	5266,89	23294,23
C-500-5	4332,36	4688,61	36682,17	4506,96	4804,51	44779,72	4301,58	4284,06	12343,96

Tabela 59: Resultados dos algoritmos híbridos para instâncias assimétricas de médio e grande porte.

	GRASP-ILS			AM-VC-ILS			MS-ACS-ILS		
Instância	Solução	Média	Tempo	Solução	Média	Tempo	Solução	Média	Tempo
A-10-3	545,92	545,92	0,10	545,92	545,92	0,88	545,92	545,92	0,43
A-10-4	460,00	543,82	0,83	460,00	491,10	24,03	460,00	478,55	0,53
A-10-5	371,93	373,89	1,01	371,93	371,93	2,72	371,93	371,93	0,58
A-20-3	709,92	839,02	0,36	679,75	737,58	18,27	747,75	762,83	0,49
A-20-4	380,40	477,79	0,88	348,48	395,33	43,32	346,30	349,38	1,44
A-20-5	343,32	439,68	0,93	315,67	362,92	45,23	315,67	355,26	1,07
A-30-3	599,67	728,83	2,33	553,42	626,58	113,12	574,33	616,97	1,85
A-30-4	650,08	747,32	3,19	597,75	681,25	145,27	650,10	682,02	2,31
A-30-5	485,15	523,21	2,45	458,70	496,02	2001,25	469,15	477,70	10,35
A-40-3	775,67	885,17	4,61	728,08	798,58	185,96	744,42	784,92	4,33
A-40-4	641,78	717,30	13,98	592,90	617,13	702,05	561,72	609,98	10,02
A-40-5	472,45	564,18	9,72	457,10	477,80	662,74	456,40	499,02	6,99
B-10-3	834,67	834,67	0,08	834,67	834,67	0,88	834,67	834,67	0,40
B-10-4	493,58	493,58	0,23	493,58	493,58	1,72	493,58	493,58	0,51
B-10-5	726,35	743,41	0,19	726,35	726,35	1,92	726,35	726,35	0,81
B-20-3	1002,50	1047,05	0,11	909,58	909,58	1,39	952,25	1028,03	0,61
B-20-4	1020,20	1046,93	0,13	910,17	910,17	1,37	937,68	955,49	0,68
B-20-5	690,13	789,82	0,13	697,00	697,00	1,23	697,00	721,39	0,67
B-30-3	864,50	937,03	1,01	832,58	832,58	14,44	723,75	818,97	4,48
B-30-4	752,85	801,35	1,35	716,80	716,80	17,19	710,12	736,26	6,55
B-30-5	565,82	606,66	3,42	559,15	559,15	39,83	522,77	540,67	9,76
B-40-3	965,33	1051,33	1,36	934,25	934,25	17,24	958,33	987,08	5,84
B-40-4	834,40	947,30	8,20	807,15	905,82	5,83	817,53	869,71	7,31
B-40-5	780,25	907,28	10,97	781,58	781,58	27,68	723,35	798,63	8,23
C-10-3	568,83	590,08	0,06	558,67	558,67	0,27	558,67	558,67	0,27
C-10-4	408,45	408,45	0,06	408,45	408,45	0,44	408,45	424,73	0,41
C-10-5	409,60	410,48	0,09	409,60	409,60	1,20	409,60	409,60	0,62
C-20-3	701,33	753,63	0,20	668,50	668,50	2,46	622,83	655,63	1,15
C-20-4	670,40	711,31	0,48	446,05	446,05	5,94	501,05	539,64	2,30
C-20-5	726,18	825,63	0,18	729,20	729,20	2,43	706,88	729,42	1,00
C-30-3	822,25	993,32	1,27	853,67	853,67	13,48	841,50	914,70	6,37
C-30-4	856,60	898,89	1,02	737,68	737,68	12,17	769,30	795,94	3,85
C-30-5	586,58	597,60	4,55	489,17	489,17	54,05	426,92	477,91	13,24
C-40-3	794,50	816,70	18,00	799,00	799,00	14,85	624,50	644,75	10,48
C-40-4	745,60	797,32	14,70	871,20	871,20	6,38	697,00	734,06	8,69
C-40-5	574,50	623,64	35,95	813,33	813,33	15,15	475,67	505,23	24,03

Tabela 60: Resultados dos algoritmos híbridos para instâncias simétricas de pequeno porte.

	GRASP-ILS			AM-VC-ILS			MS-ACS-ILS		
Instância	Solução	Média	Tempo	Solução	Média	Tempo	Solução	Média	Tempo
A-50-3	1040,33	1166,43	13,85	956,33	1008,08	652,31	1010,42	1044,73	11,36
A-50-4	809,32	930,28	27,14	760,12	811,22	1188,63	775,40	849,49	15,69
A-50-5	607,77	678,98	60,78	601,92	601,92	144,87	558,88	604,64	51,88
A-100-3	1771,75	1903,55	60,34	1692,25	1692,25	168,65	1517,67	1610,05	47,56
A-100-4	1310,97	1494,30	121,70	1329,50	1329,50	275,99	1201,63	1256,96	57,49
A-100-5	1072,00	1175,24	176,94	1075,50	1075,50	2602,04	961,53	1048,88	89,68
A-200-3	3196,58	3196,58	443,69	3324,92	3324,92	327,93	2957,67	2957,67	215,64
A-200-4	2490,53	2490,53	1046,03	2648,35	2648,35	745,36	2138,43	2138,43	434,07
A-200-5	2383,15	2383,15	398,28	2239,22	2239,22	181,75	2169,43	2169,43	182,53
A-500-3	6606,41	7478,30	5560,09	6550,20	7130,73	3147,88	6340,17	6340,17	3103,03
A-500-4	5123,07	5735,94	14674,37	5298,85	5743,41	7553,87	4972,30	4972,30	8538,29
A-500-5	4901,30	5305,29	1064,82	4854,91	5215,79	1268,21	4555,58	4555,58	959,56
B-50-3	1101,92	1135,15	25,60	1122,25	1122,25	10,97	1037,08	1077,28	20,17
B-50-4	929,68	949,57	35,20	1057,47	1057,47	25,28	821,05	872,33	15,69
B-50-5	832,18	865,62	36,65	877,62	877,62	30,53	711,40	756,93	21,46
B-100-3	2077,92	2249,05	42,73	2331,08	2331,08	19,62	1785,42	1925,72	26,34
B-100-4	2044,42	2158,14	34,00	2348,35	2348,35	18,29	1849,37	1965,52	20,91
B-100-5	1120,17	1173,32	302,97	1362,53	1362,53	185,87	1065,38	1116,47	168,41
B-200-3	3421,00	3421,00	464,61	3421,25	3421,25	309,79	3011,42	3011,42	195,45
B-200-4	2815,53	2815,53	1106,91	2844,50	2844,50	737,45	2360,13	2360,13	444,32
B-200-5	2251,03	2251,03	1517,67	2405,58	2405,58	974,90	1991,45	1991,45	631,27
B-500-3	4878,69	4892,57	17018,10	4615,69	5097,69	32519,55	6322,50	6322,50	1703,26
B-500-4	5283,22	5640,28	3191,22	5248,88	5802,69	4772,32	5084,73	5084,73	3364,79
B-500-5	4851,25	4834,29	11262,51	4756,53	4741,29	19218,61	4485,22	4485,22	2204,83
C-50-3	916,50	985,83	18,16	1083,83	1083,83	7,91	887,75	931,90	12,69
C-50-4	995,72	1103,30	26,03	1190,47	1190,47	22,55	942,52	1011,84	16,67
C-50-5	804,07	854,24	59,89	865,37	865,37	38,93	751,30	817,01	42,90
C-100-3	1454,75	1575,33	62,10	1674,50	1674,50	32,36	1376,25	1407,72	30,18
C-100-4	1216,80	1289,80	194,05	1267,60	1267,60	90,25	1070,05	1132,90	87,26
C-100-5	1195,67	1212,85	100,57	1233,27	1233,27	38,10	973,97	1069,95	49,90
C-200-3	2870,25	2870,25	412,30	2896,00	2896,00	237,50	2626,58	2626,58	169,91
C-200-4	2423,80	2423,80	406,62	2410,10	2410,10	238,17	2225,60	2225,60	181,86
C-200-5	1987,50	1987,50	1854,00	2008,93	2008,93	1004,67	1772,17	1772,17	797,83
C-500-3	6313,93	6672,06	10896,58	6402,82	6794,65	23330,39	6130,50	6130,50	1865,37
C-500-4	5278,21	5679,75	1391,24	5150,55	5493,92	562,64	4975,80	4975,80	614,25
C-500-5	4262,96	4526,49	5703,26	4213,29	4531,16	7895,52	4064,33	4064,33	3572,53

Tabela 61: Resultados dos algoritmos híbridos para instâncias simétricas de médio e grande porte.

	Assimétrico			Simétrico		
Instância	GRASP-ILS	AM-VC-ILS	MS-AM-VC-ILS	GRASP-ILS	AM-VC-ILS	MS-AM-VC-ILS
A-10-3	0%	100%	100%	0%	100%	100%
A-10-4	100%	100%	100%	100%	100%	100%
A-10-5	20%	100%	40%	20%	100%	40%
A-20-3	0%	100%	0%	0%	100%	0%
A-20-4	0%	0%	0%	0%	0%	0%
A-20-5	0%	0%	0%	0%	0%	0%
A-30-3	0%	0%	20%	0%	0%	20%
A-30-4	0%	0%	0%	0%	0%	0%
A-30-5	0%	100%	0%	0%	100%	0%
A-40-3	0%	0%	20%	0%	0%	20%
A-40-4	0%	0%	0%	0%	0%	0%
A-40-5	0%	0%	0%	0%	0%	0%
B-10-3	80%	100%	40%	80%	100%	40%
B-10-4	100%	100%	100%	100%	100%	100%
B-10-5	80%	100%	80%	80%	100%	80%
B-20-3	0%	0%	0%	0%	0%	0%
B-20-4	0%	0%	20%	0%	0%	20%
B-20-5	0%	0%	0%	0%	0%	0%
B-30-3	0%	0%	20%	0%	0%	20%
B-30-4	0%	0%	20%	0%	0%	20%
B-30-5	0%	0%	0%	0%	0%	0%
B-40-3	0%	0%	20%	0%	0%	20%
B-40-4	0%	0%	20%	0%	0%	20%
B-40-5	0%	0%	0%	0%	0%	0%
C-10-3	60%	100%	100%	60%	100%	100%
C-10-4	60%	100%	100%	60%	100%	100%
C-10-5	10%	10%	10%	10%	10%	10%
C-20-3	0%	0%	80%	0%	0%	80%
C-20-4	0%	0%	0%	0%	0%	0%
C-20-5	0%	0%	0%	0%	0%	0%
C-30-3	0%	0%	0%	0%	0%	0%
C-30-4	0%	0%	0%	0%	0%	0%
C-30-5	0%	0%	0%	0%	0%	0%
C-40-3	0%	0%	20%	0%	0%	20%
C-40-4	0%	0%	20%	0%	0%	20%
C-40-5	0%	0%	20%	0%	0%	20%

Tabela 62: Percentual da quantidade de vezes em que os algoritmos híbridos encontraram o melhor resultado para instâncias de pequeno porte.

	Assimétrico			Simétrico		
Instância	GRASP-ILS	AM-VC-ILS	MS-AM-VC-ILS	GRASP-ILS	AM-VC-ILS	MS-AM-VC-ILS
A-50-3	0%	0%	0%	0%	0%	0%
A-50-4	0%	0%	0%	0%	0%	0%
A-50-5	0%	0%	0%	0%	0%	0%
A-100-3	0%	0%	20%	0%	0%	20%
A-100-4	0%	0%	0%	0%	0%	0%
A-100-5	0%	0%	0%	0%	0%	0%
A-200-3	0%	0%	0%	0%	0%	0%
A-200-4	0%	0%	0%	0%	0%	0%
A-200-5	0%	0%	0%	0%	0%	0%
A-500-3	0%	0%	0%	0%	0%	0%
A-500-4	0%	0%	0%	0%	0%	0%
A-500-5	0%	0%	0%	0%	0%	0%
B-50-3	0%	0%	20%	0%	0%	20%
B-50-4	0%	0%	20%	0%	0%	20%
B-50-5	0%	0%	20%	0%	0%	20%
B-100-3	0%	0%	0%	0%	0%	0%
B-100-4	0%	0%	0%	0%	0%	0%
B-100-5	0%	0%	0%	0%	0%	0%
B-200-3	0%	0%	0%	0%	0%	0%
B-200-4	0%	0%	0%	0%	0%	0%
B-200-5	0%	0%	100%	0%	0%	100%
B-500-3	0%	0%	0%	0%	0%	0%
B-500-4	0%	0%	0%	0%	0%	0%
B-500-5	0%	0%	0%	0%	0%	0%
C-50-3	0%	0%	0%	0%	0%	0%
C-50-4	0%	0%	20%	0%	0%	20%
C-50-5	0%	0%	20%	0%	0%	20%
C-100-3	0%	0%	0%	0%	0%	0%
C-100-4	0%	0%	0%	0%	0%	0%
C-100-5	0%	0%	0%	0%	0%	0%
C-200-3	0%	0%	100%	0%	0%	100%
C-200-4	0%	0%	0%	0%	0%	0%
C-200-5	0%	0%	0%	0%	0%	0%
C-500-3	0%	0%	0%	0%	0%	0%
C-500-4	0%	0%	0%	0%	0%	0%
C-500-5	0%	0%	0%	0%	0%	0%

Tabela 63: Percentual da quantidade de vezes em que os algoritmos híbridos encontraram o melhor resultado para instâncias de médio e grande porte.

APÊNDICE F – Melhores Resultados Encontrados pelos Algoritmos Propostos

	Assimétrico			Simétrico		
Instância	Algoritmo	Solução	Tempo	Algoritmo	Solução	Tempo
A-10-3	GRASP	478,42	0,02	GRASP	545,92	0,01
A-10-4	GRASP	523,57	0,02	GRASP	460,00	0,02
A-10-5	GRASP	482,60	0,03	GRASP	371,93	0,04
A-20-3	AM-VC-ILS	519,67	1,64	AS	679,75	0,09
A-20-4	ILS	439,67	0,18	AS	346,30	0,12
A-20-5	ILS	399,92	0,23	MS-ACS-ILS	315,67	1,07
A-30-3	MS-ACS-ILS	711,50	4,94	AM-VC-ILS	553,42	113,12
A-30-4	ILS	447,50	4,69	AM-VC-ILS	597,75	145,27
A-30-5	AM-VC-ILS	454,80	31,15	AM-VC-ILS	458,70	2001,25
A-40-3	MS-ACS-ILS	707,75	11,50	ILS	727,75	2,75
A-40-4	ILS	685,20	8,23	MS-ACS-ILS	561,72	10,02
A-40-5	MS-ACS	482,98	0,83	MS-ACS-ILS	456,40	6,99
B-10-3	GRASP	729,50	0,01	AG	834,67	0,01
B-10-4	GRASP	306,90	0,02	GRASP	493,58	0,03
B-10-5	GRASP	434,75	0,03	GRASP	726,35	0,03
B-20-3	ILS	867,92	0,22	AM-VC-ILS	909,58	1,39
B-20-4	MS-ACS-ILS	888,83	0,76	MS-ACS	822,82	0,23
B-20-5	ILS	843,17	0,19	AM-VC	641,57	9,50
B-30-3	MS-ACS-ILS	824,67	9,12	MS-ACS-ILS	723,75	4,48
B-30-4	MS-ACS-ILS	667,00	9,34	AM-VC	696,30	24,44
B-30-5	ILS	678,97	5,33	MS-ACS	510,13	0,56
B-40-3	MS-ACS-ILS	945,33	13,04	ILS	875,08	4,17
B-40-4	MS-ACS-ILS	1384,48	5,19	MS-ACS	714,05	0,70
B-40-5	ILS	810,75	9,67	MS-ACS-ILS	723,35	8,23
C-10-3	GRASP	359,25	0,03	AG	558,67	0,01
C-10-4	AG	307,10	0,04	GRASP	408,45	0,02
C-10-5	AG	566,58	0,02	GRASP	409,60	0,02
C-20-3	MS-ACS	650,25	0,32	MS-ACS	613,83	0,30
C-20-4	MS-ACS	613,83	0,33	MS-ACS	441,65	0,39
C-20-5	ILS	684,80	1,04	MS-ACS-ILS	706,88	1,00
C-30-3	MS-ACS-ILS	836,75	9,92	GRASP-ILS	822,25	1,27
C-30-4	MS-ACS-ILS	730,00	24,08	AM-VC-ILS	737,68	12,17
C-30-5	MS-ACS-ILS	558,77	62,12	MS-ACS-ILS	426,92	13,24
C-40-3	MS-ACS-ILS	1012,92	11,69	MS-ACS-ILS	624,50	10,48
C-40-4	MS-ACS-ILS	690,13	34,84	ILS	630,40	7,85
C-40-5	MS-ACS-ILS	646,17	53,83	MS-ACS	475,50	15,89

Tabela 64: Melhores resultados encontrados pelos algoritmos propostos para instâncias de pequeno porte.

	Assimétrico			Simétrico		
Instância	Algoritmo	Solução	Tempo	Algoritmo	Solução	Tempo
A-50-3	ILS	1193.92	1.34	AM-VC-ILS	956.33	652.31
A-50-4	MS-ACS-ILS	832.82	35.97	AM-VC-ILS	760.12	1188.63
A-50-5	ILS	724.17	23.61	MS-ACS	553.85	1.31
A-100-3	MS-ACS-ILS	1484.00	369.80	MS-ACS-ILS	1517.67	47.56
A-100-4	MS-ACS	1531.23	3.23	ILS	1183.12	222.32
A-100-5	MS-ACS	1172.48	41.00	MS-ACS-ILS	961.53	89.68
A-200-3	MS-ACS	2710.25	425.12	MS-ACS	2731.17	87.59
A-200-4	MS-ACS	2324.87	323.95	MS-ACS-ILS	2138.43	434.07
A-200-5	MS-ACS	1782.52	102.28	MS-ACS	2113.38	79.78
A-500-3	MS-ACS	7048.50	465.00	MS-ACS-ILS	6340.17	3103.03
A-500-4	MS-ACS	5238.50	1460.10	MS-ACS-ILS	4972.30	8538.29
A-500-5	MS-ACS	4440.37	3501.97	MS-ACS-ILS	4555.58	959.56
B-50-3	MS-ACS-ILS	1387.75	22.39	MS-ACS-ILS	1037.08	20.17
B-50-4	MS-ACS-ILS	1013.30	49.67	MS-ACS-ILS	821.05	15.69
B-50-5	MS-ACS-ILS	1129.80	35.29	MS-ACS-ILS	711.40	21.46
B-100-3	MS-ACS-ILS	1973.17	814.93	MS-ACS-ILS	1785.42	26.34
B-100-4	MS-ACS	1504.97	5.27	ILS	1758.78	62.50
B-100-5	MS-ACS	1504.07	42.01	MS-ACS	1043.22	48.26
B-200-3	MS-ACS	3302.42	322.30	MS-ACS	2913.67	90.87
B-200-4	MS-ACS	2532.07	495.66	MS-ACS-ILS	2360.13	444.32
B-200-5	MS-ACS-ILS	2094.72	2873.25	MS-ACS	1912.87	150.29
B-500-3	MS-ACS	6886.08	2106.44	MS-ACS	4415.60	94.18
B-500-4	MS-ACS	5517.34	7628.10	MS-ACS-ILS	5084.73	3364.79
B-500-5	MS-ACS	4541.48	5614.49	MS-ACS	4415.60	94.18
C-50-3	MS-ACS-ILS	1237.75	28.61	ILS	883.25	14.38
C-50-4	MS-ACS-ILS	938.75	68.82	ILS	921.03	16.61
C-50-5	MS-ACS-ILS	621.92	158.85	MS-ACS-ILS	751.30	42.90
C-100-3	MS-ACS	1452.75	40.07	MS-ACS-ILS	1376.25	30.18
C-100-4	MS-ACS	1496.57	4.32	MS-ACS	1067.92	37.67
C-100-5	MS-ACS	987.00	78.92	MS-ACS-ILS	973.97	49.90
C-200-3	MS-ACS-ILS	2660.08	1695.67	MS-ACS	2564.58	87.84
C-200-4	MS-ACS	2186.08	314.12	MS-ACS	2106.93	85.47
C-200-5	MS-ACS	1878.00	482.50	MS-ACS	1721.17	182.05
C-500-3	MS-ACS	6470.34	1356.29	MS-ACS-ILS	6130.50	1865.37
C-500-4	MS-ACS	5078.54	6901.70	MS-ACS-ILS	4975.80	614.25
C-500-5	MS-ACS	4193.38	2872.73	MS-ACS	4029.83	201.98

Tabela 65: Melhores resultados encontrados pelos algoritmos propostos para instâncias de médio e grande porte.