PONTIFÍCIA UNIVERSIDADE CATÓLICA DE GOIÁS
ESCOLA POLITÉCNICA E DE ARTES
GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

HIBRIDIZAÇÃO DO ALGORITMO GENÉTICO E COLÔNIA DE FORMIGAS PARA O PROBLEMA DE ROTEAMENTO E LOCALIZAÇÃO DO SISTEMA DE VEÍCULOS ELÉTRICOS AUTÔNOMOS COMPARTILHADOS

VANESSA OLIVEIRA ALVES

GOIÂNIA
2023

Vanessa Oliveira Alves

Hibridização Do Algoritmo Genético E Colônia De Formigas Para O Problema De Roteamento E Localização Do Sistema De Veículos Elétricos Autônomos Compartilhados

Trabalho de Conclusão de Curso apresentado à Escola Politécnica e de Artes da Pontifícia Universidade Católica de Goiás, como parte dos requisitos para obtenção do título de Bacharel em Ciência da Computação.

Orientador:

Prof. Me. Alexandre Ribeiro

GOIÂNIA

2023

VANESSA OLIVEIRA ALVES

HIBRIDIZAÇÃO DO ALGORITMO GENÉTICO E COLÔNIA DE FORMIGAS PARA O PROBLEMA DE ROTEAMENTO E LOCALIZAÇÃO DO SISTEMA DE VEÍCULOS ELÉTRICOS AUTÔNOMOS COMPARTILHADOS

Este Trabalho de Conclusão de Curso foi julgado adequado para a obtenção do grau de Bacharel em Ciência da Computação e aprovado em sua forma final pela Escola Politécnica e de Artes, da Pontifícia Universidade Católica de Goiás, em _____/_____/_____.

_____

Profª. Ma. Ludmilla Reis Pinheiro dos Santos
Coordenadora de Trabalho de Conclusão de Curso

Banca examinadora:

_____

Orientador: Prof. Me. Alexandre Ribeiro

_____

Profª. Dra. Maria José Pereira Dantas

_____

Prof. Me. Max Gontijo de Oliveira

GOIÂNIA
2023

RESUMO

O presente trabalho teve como foco principal o problema de roteamento e localização do sistema de veículos elétricos autônomos compartilhados (LRP-SAEV), que apresenta quatro variações. No contexto de problemas computacionais da classe NP-difícil, dada à crescente necessidade de otimização do processamento de massas de dados cada vez maiores e na importância de uma solução para tais problemas o mais próximo do ótimo possível. Soma-se a isso o grande potencial na melhoria do trânsito urbano com sistema de transporte inteligente, sustentável e autônomo. Este projeto propõe um algoritmo que envolve a hibridização de duas metaheurísticas distintas, algoritmo genético e colônia de formigas, com o intuito de otimizar os resultados do LRP-SAEV e suas variantes. Para isso, as instâncias de cada uma das quatro variantes do LRP-SAEV serão resolvidas com a hibridização de duas metaheurísticas, gerando uma classe de solução para cada variante. Dessa forma, esse estudo apresenta uma comparação entre os resultados alcançados em cada cenário para o algoritmo hibrido e as soluções apresentadas com base nas metaheurísicas puras, ou seja, sem a hibridização. As metaheurísticas consideradas para comparação são o algoritmo genético e de colônia de formigas.

Palavras-chave: Veículos autônomos compartilhados, veículos elétricos, problema de localização de veículos, metaheurística hibrida.

ABSTRACT

The following study has as main focus the location routing problem of the car-sharing system with Autonomous Electric Vehicles (LRP-SAEV), which presents four different variations. This LRP-SAEV is considered a combinatorial optimization problem which is classified as NP – hard. In the context of given the virtue of mass data optimization processing being gradually more relevant and in the importance of a solution for those problems near the best solution. In addition, the great potential in the optimization of the urban traffic with smart, sustainable and automatous transport system. This project proposes an algorithm which involves a hybridization of two different metaheuristics, genetic algorithm and ant colony optimization, with the intention of optimizing the results of the LRP-SAEV and its variants. In order to do so the instance of each one of the four variants of the LRP-SAEV will be solved with the hybridization of two metaheuristics, generating a group of solution for each variant. Therefore, this study presents a comparison among the results reached in each scenario for the hybrid algorithm and the solutions presented based on the metaheuristics. The metaheuristics used for the comparison are the genetic algorithm and ant colony.

Keywords*: Shared autonomous vehicles, Electric vehicles, Location routing problem, hybrid meta heuristics*

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ALGORITMOS

SUMÁRIO

1 INTRODUÇÃO

Alternativas aos modelos de transporte atuais têm conquistado espaço dentre os habitantes de áreas de grandes centros urbanos. Cada vez mais as pessoas aderem ao modelo de veículos compartilhados (SHAHEEN; COHEN, 2012). Somada à aversão ao movimento caótico das cidades, também existe a questão ambiental. O uso de veículos elétricos é eficaz no combate à emissão de gases poluentes decorrentes de veículos que operam usando combustíveis fósseis (WEGSCHEIDER et al., 2022).

Dadas essas características, é notável a oportunidade de soluções que atendam às novas demandas. Segundo Wegscheider et al. (2022) o modelo de veículos elétricos autônomos compartilhados é uma boa solução para os problemas de mobilidade e poluição urbana, apresentando, inclusive, dados sobre o crescimento da representatividade de vendas de veículos elétricos e do uso de serviços de compartilhamento de veículos.

O presente trabalho tem como objetivo abordar o Problema de Roteamento e Localização do Sistema de Veículos Elétricos Autônomos Compartilhados (*The Location Routing Problem of the Car-Sharing System with Autonomous Electric Vehicles* - LRP-SAEVs).

A principal motivação é encontrar soluções que incentivem a adoção de novos modelos de mobilidade para atender às necessidades da sociedade contemporânea e, ao mesmo tempo, exerçam um efeito ambiental positivo, sem elevar muito o custo de construção, utilização e de serviço.

O LRP-SAEVs é um problema de otimização combinatória que possui como objetivo principal dimensionar frotas e determinar a localização de estações, de forma a minimizar o custo de implantação e operação. Nesse problema, são apresentadas uma variedade de estações candidatas e uma variedade de demandas de clientes. As demandas de cliente são compostas pelo local de origem, destino e o tempo que o veículo precisa estar na origem. Os veículos são encaminhados para atender uma solicitação de usuário de acordo com o tempo de chegada na origem que o usuário estipulou. As estações podem ser depósito ou pontos de carregamento, os quais podem ser localizadas na mesma área. Os depósitos são o local onde o veículo parte no início do dia e onde eles ficam ao final do dia; pontos de carregamento são os locais onde os veículos ficam estacionados e fazem a recarga.

Cada veículo inicia o dia de atendimento em um depósito e tenta atender ao maior número possível de solicitações de usuários. Após finalizar um atendimento, ele pode continuar atendendo mais requisições com a energia disponível ou pode ir para uma estação e fazer a recarga da bateria, em situações em que o nível de energia no veículo esteja abaixo do mínimo requerido, ou seja, insuficiente para atendimentos futuros.

Apresenta-se, para esse problema, duas formas de recarga de bateria e duas formas de atendimento aos usuários.

Em relação a forma de recarga da bateria, o veículo pode ser recarregado de forma total ou de forma parcial. Na recarga total, o veículo segue para uma estação e só sai dela quando a bateria estiver completamente carregada. No modo de recarga parcial, o veículo pode sair antes de ser recarregado por completo, sendo o nível de recarga determinado pelo tempo em que o veículo esteve na estação.

Com relação à forma de servir os usuários, o atendimento pode ser feito de forma total ou de forma parcial. Na forma total, todas as requisições dos usuários devem ser atendidas. Na forma parcial, uma quantidade limitada de passageiros pode ter suas solicitações rejeitadas. O não atendimento desses clientes gera um custo adicional de recusa de atendimento.

De acordo com Ma, Hu e Wu (2021) a combinação das duas formas de recarga de bateria e das duas formas de atendimento(serviço) geram os seguintes cenários:

I. Recarga Total, Serviço Total (RTST): o problema é modelado para que toda parada de recarga seja feita até que a bateria do veículo atinja 100% de carga e todas as requisições sejam atendidas;

II. Recarga Total, Serviço Parcial (RTSP): mesma modelagem para recarga em (I) porém o serviço é parcial, podendo recusar determinadas requisições durante a minimização da função objetivo;

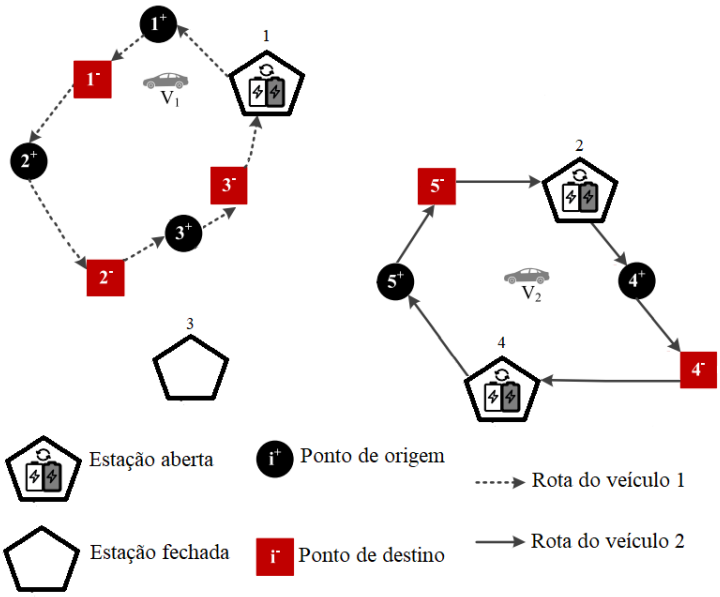
III. Recarga Parcial, Serviço Total (RPST): os veículos podem realizar recargas parciais, utilizando o ganho de tempo para atender requisições e, o serviço será completo, seguindo a descrição em (I);

IV. Recarga Parcial, Serviço Parcial (RPSP): a modelagem leva em conta a definição de recarga parcial dada em (II) e a definição de serviço parcial dada em (III).

Na Figura 1 é representado o problema do LRP-SAEVs tendo quatro estações possíveis (pentágonos) como opção e apenas três estão sendo utilizadas (pentágonos com bateria), sendo eles 1, 2 e 4, também são apresentadas cinco requisições, em

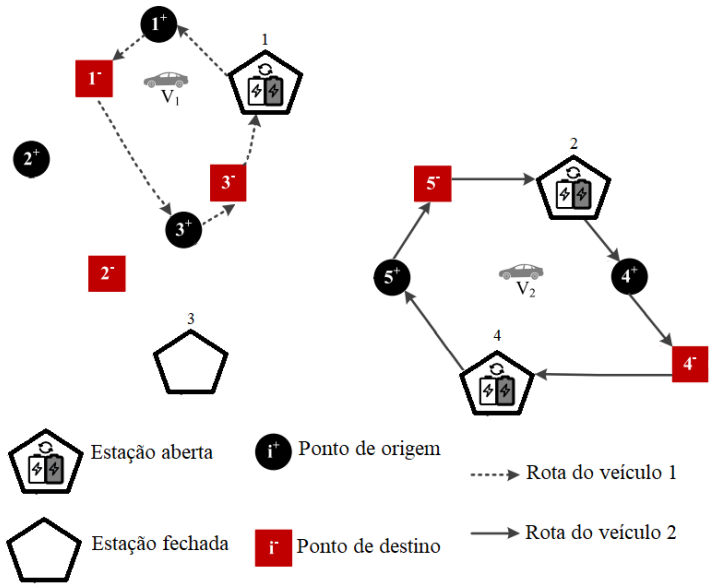
que o ponto de origem de um usuário é a bolinha e o ponto de destino de um usuário é o quadrado.

Figura 1- LRP-SAEVs para serviço total



Fonte: Adaptada de MA et al. (2022)

Figura 2 – LRP-SAEVs para serviço parcial



Fonte: Adaptada de MA et al. (2022)

A rota do veículo 1 (v1) está representada com tracinhos. Note que ele consegue atender as três requisições sem precisar realizar recargas. A rota do veículo 2 (v2) está representada por linhas contínuas, o veículo consegue atender 2 requisições, mas precisa passar na estação para reabastecer.

Na Figura 2 tem-se um esquema muito parecido ao da Figura 1, porém a Figura 1 é serviço total e a Figura 2 é serviço parcial. Como a Figura 2 é serviço parcial, pode ocorrer de requisições serem rejeitadas; no caso, o usuário 2 teve sua requisição recusada.

O recente desenvolvimento de soluções para resolver problemas de otimização combinatória introduz metaheurísticas, como busca tabu, algoritmo genético, *simutated annealing* e colônia de formigas. Todas elas alcançam busca mais efetiva no espaço de solução do que métodos convencionais por diferentes estratégias. As metaheurísticas conseguem encontrar boas soluções para difíceis problemas de otimização combinatória como o LRP (CHEN et al., 2009).

A combinação de várias ideias de múltiplos algoritmos obtém um melhor desempenho do sistema explorando vantagens e unindo as melhores estratégias (RAIDL; PUCHINGER, 2019). Para esse artigo, foram aplicadas duas metaheurísticas, algoritmo genético (GA) e colônia de formigas (ACO), e um algoritmo híbrido para a resolução do problema LRP-SAEV, aplicando em cada uma das quatro variações do problema. O algoritmo híbrido proposto é uma junção do algoritmo genético e do algoritmo de colônia de formigas. São apresentadas as comparações entre as metaheurísticas com o algoritmo híbrido proposto, todos aplicados ao problema LRP-SAEVs, e suas variações.

O presente trabalho está organizado da seguinte forma: o Capítulo 2 apresenta o referencial teórico, explicando um pouco sobre o que é problema de otimização combinatória, as duas metaheurísticas (algoritmo genético e colônia de formigas), e um explicação sobre hibridização; no Capítulo 3 são apresentados os métodos para desenvolvimento desse trabalho; o Capítulo 4 introduz os principais trabalhos relacionados ao tema; o Capítulo 5 descreve o problema LRP-SAEVs, com suas variações e as suas formulações matemáticas; o Capítulo 6 mostra as abordagens utilizadas para a resolução do problema; o Capítulo 7 apresenta os resultados dos experimentos computacionais realizados; e, por fim, o Capítulo 8 apresenta algumas considerações finais.

2 REFERENCIAL TEÓRICO

Problemas reais podem ser transformados em problemas de otimização combinatória. Um problema consiste em uma questão geral que precisa ser resolvida e é dada pelos parâmetros de entrada. Uma instância de problema é dada pelo problema junto com a especificação dos parâmetros. Então, em um problema de otimização combinatória, cada configuração de parâmetros será uma instância. Ele pode ser definido por 3 fatores primordiais (S, f, Ω), em que S é um espaço de busca, f é função objetivo do problema, que pode ser para maximizar o valor da resposta ou para minimizá-la, e o Ω representa as restrições que definem os critérios para que uma solução esteja no espaço viável (NEUMANN; WITT, 2010).

Problemas de otimização combinatória têm um espaço de busca finito com variáveis de decisão discretas. Mesmo o espaço sendo finito, é difícil para uma busca exaustiva encontrar uma solução ótima em tempo hábil. Sendo assim, o principal objetivo é encontrar a solução ótima global, sendo global a consideração de todo o espaço de busca. No caso de maximização, busca-se encontrar o maior valor naquele espaço de busca atendendo todas as restrições (RAHMAN et al., 2021).

Só são consideradas soluções viáveis, quando a solução apresentada respeita todas as restrições do problema. Soluções inviáveis são desconsideradas, mesmo tendo resultados menores. O interessado está em encontrar a solução do problema em um tempo hábil, com o menor ou maior resultado da função objetivo e respeitando as restrições definidas.

Grande parte dos problemas de otimização combinatória são NP-difíceis e, até o momento, não possuem um método eficiente para resolvê-los de forma ótima. Dada a importância desses problemas, pesquisadores têm desenvolvido estudos sobre metaheurísticas. Elas têm recebido muita atenção e têm sido muito aplicadas. Os avanços em metaheurísticas dependem de melhorias e inovações de técnicas numéricas para calcular problemas de otimização (ZHAO et al., 2018).

O termo Metaheuristica descreve um alto nível de heurísticas, que podem explorar todo espaço do problema com diferentes aplicações por um processo iterativo e inteligente. Normalmente, metaheurísticas simulam estratégia de evolução natural para produzir convergência (RAHMAN et al., 2021).

Metaheurísticas são usadas para resolver problema de otimização pelo processo de busca da solução ótima para o problema de interesse. O processo de

busca pode ser realizado usando múltiplos agentes no qual o sistema essencialmente usa um conjunto de regras ou equações matemáticas durante as várias iterações. Essas iterações acontecem até que a solução atenda a um critério de parada. Essa solução final, que é uma solução aceitável, é dita assim e o sistema é considerado como se ele tivesse alcançado a convergência (HUSSAIN et al., 2019)

Como o termo meta em metaheurística sugere, metaheurística está a um nível mais alto do que abordagens de heurísticas. Técnicas de metaheurística têm tido muito sucesso, pois encontram soluções aceitáveis e com bom custo computacional (HUSSAIN et al., 2019).

Ancioto (2019) utilizou a hibridização do algoritmo genético e de colônia de formigas aplicado aos problemas do caixeiro viajante e roteamento de veículo. Conseguindo bons resultados, assim decidiu se utilizar esses algoritmos.

Neste capítulo, é apresentado o referencial teórico para um melhor entendimento do trabalho. Nas seções 2.1 e 2.2 são introduzidos os conceitos de algoritmo genético e algoritmo de colônia de formigas, que foram utilizados nesse trabalho. O método principal que foi utilizado nesse trabalho é a hibridização do algoritmo genético e de colônia de formigas que é apresentado na Seção 2.3.

2.1 Algoritmo Genético

Algoritmo genético do inglês *Genetic Algorithm* (GA) foi introduzido por Holland (1975), e é uma abstração de biologia evolutiva, baseada na teoria da seleção natural proposta por Charles Darwin. John Holland foi o primeiro a utilizar técnicas que são as peças chaves para as operações de resolver problemas utilizando o GA. Essas operações genéticas são cruzamento, mutação e seleção (YANG, 2021).

O GA utiliza a ideia do processo evolucionário que pode ser aplicado para solucionar uma variedade grande de problemas. De acordo com Whitley (2019), todos os indivíduos na população são avaliados de acordo com uma função objetivo. A seguir, uma população intermediária é criada utilizando a seleção. O processo de seleção mantém os melhores indivíduos na população intermediária. Finalmente, com a população intermediária definida, o cruzamento e a mutação são aplicados para criar a próxima população na próxima geração. Na proposta básica apresentada por Holland, dois indivíduos poderiam ser recombinados para a criação de um novo, e

esse novo indivíduo poderia substituir os indivíduos que o criaram. Isso significa que a melhor solução de uma geração poderia não sobreviver para a próxima.

O algoritmo em geral utiliza a ideia de que evolução adaptativa é um sistema modificador poderoso, que é responsável pela diversidade e adaptação da vida na Terra. Em um sistema natural, o ambiente está em constante mudança e evolução, o que tem como resultado uma alta adaptação da vida formada de comportamentos complexos (WHITLEY, 2019).

Segundo Xidias, Panagiotopoulos, Zacharia (2023), o GA é uma metaheurística muito conhecida em problemas de roteamento de veículos, a qual tem tido muito sucesso. O algoritmo é aplicado para uma larga e complexa combinação de problemas de otimização, desde coloração de grafo até reconhecimento de padrões.

Para entender o algoritmo, algumas definições iniciais devem ser apresentadas, como geração, população e indivíduo.

Entenda como geração, a simulação do processo evolutivo. Então, uma geração é uma execução de todo o processo evolutivo, sendo ela a geração de uma população, utilizando os processos de cruzamento, mutação e seleção. Assim, o número de gerações é a quantidade de iterações, ou a quantidade de vezes que ocorre alteração na população (BRABAZON; O'NEILL; MCGARRAGHY, 2015).

A população é um conjunto de indivíduos, que armazena na memória várias soluções possíveis do problema. Para cada nova geração são aplicadas, nos indivíduos, operações genéticas que podem melhorar os próximos indivíduos; e a população tem como função propagar para as próximas gerações melhores resultados (WHITLEY, 2019).

O indivíduo é o conjunto de valores para as variáveis do problema que contém uma resposta para a instância do problema (BRABAZON; O'NEILL; MCGARRAGHY, 2015).

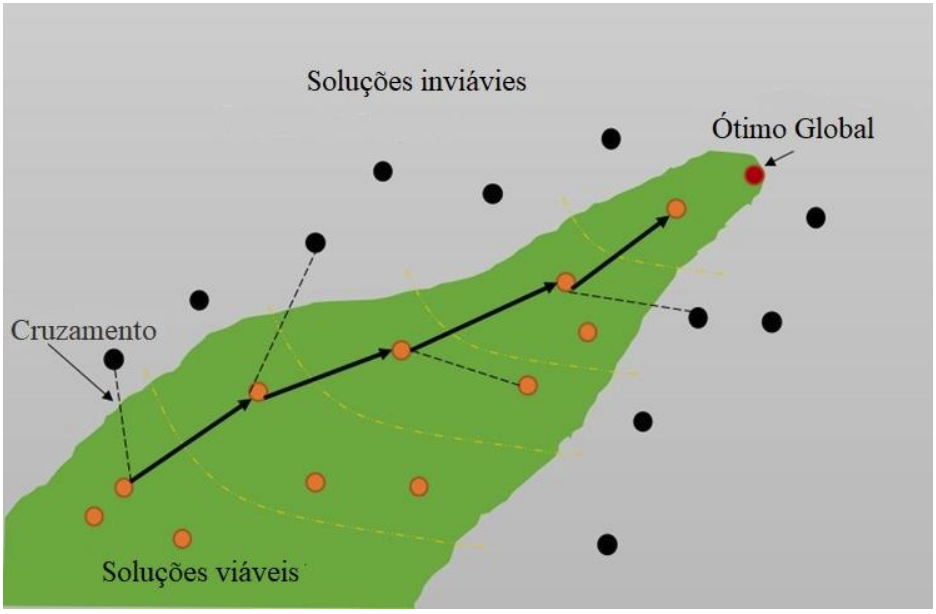
Com relação aos processos que ocorrem no GA, tem-se o cruzamento, a mutação e a seleção. Existem várias formas de fazer a implementação desses processos.

No cruzamento foi utilizada uma abordagem de escolha de dois indivíduos aleatórios, em que partes de duas soluções são trocadas uma com a outra. O objetivo aqui é promover um mix de soluções e convergência no subespaço. (YANG, 2021). A Figura 3 mostra que o processo de cruzamento pode levar para soluções piores que não atendem todas as restrições do problema, inviáveis, assim como podem levar por

melhores caminhos, que tenham soluções viáveis, chegando à solução considerada ótima global.

Em relação à mutação, o objetivo principal é introduzir uma nova variação e uma exploração local. Existem várias abordagens, como gerar uma nova variante aleatória e completamente diferente da inicial, ou apenas partes de um indivíduo são alteradas e um novo é criado. A mutação gera diversidade na população e tenta escapar do ótimo local (WHITLEY, 2019).

Figura 3 – Processo evolucionário do GA



Fonte: Adaptada de YU; ZHOU; LIU, (2019)

A seleção não é diferente, existe uma grande variedade de formas de se fazer a seleção dos melhores indivíduos, uma delas é a *ordinal selection,* que foi utilizada nesse trabalho*.* Essa abordagem utiliza a seleção com base no *rank*. Os indivíduos serão ordenados da melhor para a pior solução, que, baseado na função objetivo, tem-se o quão boa uma solução é. Assim, para cada indivíduo é calculada a função objetivo e cria-se o *ranking* dos valores de acordo com o objetivo do problema, de maximizar ou minimizar (BRABAZON; O'NEILL; MCGARRAGHY, 2015). Na escolha de quais irão para a próxima geração, é usado o elitismo, um mecanismo em que as melhores soluções da geração são adicionadas à próxima geração. O elitismo é parametrizado em uma constante que define a quantidade que irá direto para a próxima geração (WHITLEY, 2019)

O GA não tem um ponto de partida bem definido, o que torna necessária a criação de uma população inicial. Encontrar a melhor população inicial não é uma tarefa fácil. Se o ponto de início fosse conhecido, a eficiência do GA poderia ser aperfeiçoada usando essa informação como população inicial. Como não é possível isso, comumente um bom ponto de partida não é conhecido e a população inicial é gerada aleatoriamente (BRABAZON; O'NEILL; MCGARRAGHY, 2015).

Algoritmo 1 – Algoritmo básico do GA

```
1  entrada: uma instância do problema a ser resolvido
2  saída: a solução ótima ou a melhor
3  Inicializar a população de soluções candidatas
4  enquanto t < máximo número de gerações faça
5      Gera novas soluções pelo cruzamento e mutação
6      Aceita novas soluções se a função objetivo reduzir
7      Calcular a função objetivo para cada indivíduo da população
8      Seleciona alguns ou todos da população atual, apenas os melhores vão
       para a próxima geração(elitismo)
9      Atualiza t = t + 1
10 fim enquanto
```

O Algoritmo 1 foi adaptado de Yang (2021), a entrada do algoritmo, como mostrado na linha 1, é uma instância do problema que será resolvido. Já a saída, na linha 2 é a melhor solução ou a solução ótima.

Como o GA não apresenta nenhum ponto de partida, uma população inicial deve ser criada de forma aleatória, na linha 3 é feita essa inicialização.

A repetição definida do algoritmo ocorre entre as linhas 5 e 10, em que uma nova geração é criada a cada iteração. Para cada geração novos indivíduos são criados utilizando o cruzamento e a mutação, são gerados uma nova população com esses indivíduos, como mostrado na linha 5. Aceita apenas as melhores soluções geradas, linha 6. Para essa população é necessário fazer o cálculo da função objetivo para cada indivíduo, linha 7. Seleciona alguns ou todos indivíduos para a população da próxima geração, utilizando o elitismo, linha 8. Finaliza retornando a melhor solução encontrada.

2.2 Algoritmo Colônia de Formigas

O Algoritmo de Colônia de Formigas do inglês *Ant Colony Optimization Algorithms* (ACO) foi inspirado no comportamento de algumas espécies de formigas. Essas formigas marcam o caminho percorrido com feromônio, assim os membros de sua colônia podem seguir pelo mesmo caminho. O algoritmo explora o mesmo mecanismo para resolver problemas de otimização (DORIGO; BIRATTARI; STUTZLE, 2006).

O ACO originalmente foi proposto para resolver o problema do caixeiro viajante. Proposto por Colorni, Dorigo, Maniezzo (1992), o ACO é inspirado pela observação das colônias de formigas reais, na qual apresenta o recurso de encontrar o menor caminho entre a colônia e o alimento (VALDEZ; MELIN; CASTILLO, 2014).

Esse algoritmo vem sendo muito aplicado para resolução de problemas de otimização combinatória tidos como difíceis de resolver. Esses problemas se encaixam na classe NP-difícil, em que o tempo para resolver esse problema, no pior caso, aumenta exponencialmente em relação ao tamanho da instância (DORIGO; BIRATTARI; STUTZLE, 2006).

Algoritmos de aproximação são utilizados para gerar uma solução possível, que seja próxima da solução ótima e com custo computacional relativamente baixo. O ACO é um método de busca estocástico, que está centrado no feromônio, o qual é usado probabilisticamente em conjunto a alguma busca espacial (DORIGO; BLUM, 2005).

Muitas variações do algoritmo foram propostas durante os anos, como *ant system, max-min* e o *ant colony system* (DORIGO; BIRATTARI; STUTZLE, 2006). Existem várias outras variações do algoritmo, porém o foco nesse trabalho será o algoritmo básico que está apresentado no Algoritmo 2. Algumas definições são apresentadas a seguir.

Na inicialização do feromônio, todas as variáveis de feromônio são inicializadas com o valor τ_0, que é um parâmetro que não pode ser menor do que ou igual a zero e indica o valor que começa o feromônio (DORIGO; BLUM, 2005)

Cada formiga, será uma solução do problema. A formação de uma formiga é realizada de forma que, a formiga definida como s_p, inicializa vazio. A cada passo da construção da resposta, s_p será uma solução parcial do problema, escolhendo dentro

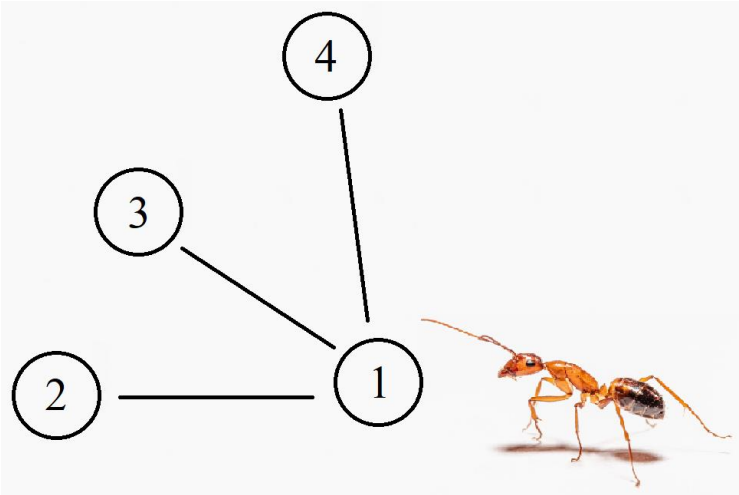
das componentes um resultado viável $c_i^j \in \mathcal{N}(s_p) \subseteq C$ e adicionando o elemento à solução parcial. $\mathcal{N}(s_p)$ são as componentes da solução que podem ser adicionadas e que mantem a viabilidade da solução, sendo definido implicitamente pelo processo de solução que é implementado pela formiga. Caso a componente não seja viável, ela é desconsiderada.

A escolha de uma componente de solução é adicionada probabilisticamente para cada passo da construção de uma formiga. Várias maneiras de definir a distribuição probabilística tem sido considerada, porém a mais usada é a *Ant System*(AS) (DORIGO; STUTZLE, 2019).

$$p(c_i^j|S_p) = \frac{\tau_{ij}^{\alpha}.[\eta(c_j^i)]^{\beta}}{\sum_{c_i^l \epsilon \mathcal{N}(s_p)} \tau_{il}^{\alpha}.[\eta(c_i^l)]^{\beta}}, \forall c_i^j \epsilon \mathcal{N}(s_p) \quad (1)$$

A Equação 1 apresenta $\eta(.)$, que é uma função que atribui o valor da heurística η_{ij} para cada componente viável $c_i^j \in \mathcal{N}(s_p)$, é geralmente conhecido como informações da heurística. Para a influência do feromônio sobre a solução é utilizado os parâmetros α e β. Se α = 0, a seleção da probabilidade fica proporcional a $[\eta_{ij}]^{\beta}$ e a componente de solução com o maior valor será selecionado. Se β = 0, apenas o feromônio é considerado (DORIGO; STUTZLE, 2019).

Figura 4 – Formiga no ponto 1 definindo qual dos três pontos será escolhido



Fonte: Adaptada de DORIGO; BIRATTARI; STUTZLE, (2006).

Na Figura 4 é apresentada uma formiga estando na componente 1, que já foi adicionada à solução parcial dessa formiga em questão, e que vai decidir entre qual das 3 componentes será adicionada à solução parcial, 2 ou 3 ou 4.

A busca local é um processo que pode ser aplicado após a construção de uma formiga, porém é opcional. O processo é aplicado para melhorar a construção da solução (DORIGO; BLUM, 2005).

A atualização do feromônio é aplicada para que boas soluções sejam desejadas para as próximas iterações. Dois mecanismos são usados na atualização, a adição e a evaporação de feromônio (DORIGO; STUTZLE, 2019).

A adição de feromônio consiste em depositar o feromônio das boas soluções, o qual eleva o valor do feromônio das componentes associadas com a escolha do S_{upd} da boa solução. Assim, possibilita que as próximas formigas tenham essas componentes mais atrativas (DORIGO; STUTZLE, 2019).

O outro mecanismo é a evaporação do feromônio, que é a redução de todas as variáveis que armazena o feromônio. Esse processo ocorre para evitar que aconteça uma convergência muito rápida do algoritmo para uma região sub-ótima (DORIGO; STUTZLE, 2019).

$$\tau_{ij} = (1-\rho)\tau_{ij} + \sum_{s \in S_{upd} | c_j^i \in s} g(s) \qquad (2)$$

Finalmente, a atualização do feromônio é implementada como na Equação 2, onde S_{upd} é tido como o conjunto da solução que será adicionado o feromônio, $\rho \in (0,1]$ é um parâmetro chamado taxa de variação. E para determinar a qualidade da solução tem-se $g(.): S \rightarrow R+$ é a função tal que $f(s) < f(s') \Rightarrow g(s) \geq g(s')$ (DORIGO; STUTZLE, 2019).

O Algoritmo 2 retirado de Dorigo e Blum (2005) demonstra o algoritmo básico do ACO. Na linha 1 é apresentada a entrada sendo uma instância do problema que será resolvido. Já na linha 2 é apresentado o retorno da função que é a melhor solução encontrada ou a solução ótima.

Na linha 3, ao inicializar o algoritmo, todos os parâmetros recebem os valores definidos, junto com todas as variáveis de feromônio.

Na linha 4, em que o sbs armazena a melhor solução encontrada, é definido como nulo. A variável passa a receber valor apenas após a execução das repetições do algoritmo. As repetições se iniciam na linha 5, que ocorre até que a condição de parada seja satisfeita.

A cada iteração uma quantidade de formigas m é construída. Cada formiga construída é armazenada em s, linha 7. Na linha 8, verifica se a solução/formiga é uma solução válida. Na linha 9, se ela for válida, uma busca local pode ser aplicada para melhorar a solução, sendo essa busca opcional. Então, com o resultado de s, verifica se a nova formiga encontrada é uma solução melhor que a armazenada em sbs, para isso deve-se utilizar a função objetivo. Na linha 10, o primeiro resultado não tem o que comparar, então ele é o melhor resultado. A aplicação de feromônio deve ocorrer ao final de cada iteração, linha 13.

Algoritmo 2 – Algoritmo básico do ACO

```
1  entrada: Uma instância do problema a ser resolvido
2  saída: a solução ótima ou a melhor
3  Inicializar os valores do feromônio
4  sbs ← Nulo
5  enquanto a condição de fim não for atendida faça
6      para j = 1, ..., m faça
7          s ← construção da formiga
8          se s é uma solução válida então
9              s ← busca local (opcional)
10             se (f (s) < f(sbs)) ou (sbs = Nulo) então sbs ← s fim se
11         fim se
12     fim para
13     Aplica a atualização do feromônio
14 fim enquanto
```

2.3 Hibridização GA-ACO

As metaheurísticas são adequadas para a resolução de problemas complexos. A motivação na utilização da hibridização de diferentes algoritmos é, geralmente, obter um melhor desempenho do sistema, que explora e une as vantagens individuais de cada algoritmo por uma estratégia pura, no qual o híbrido se beneficia da sinergia. A chave para atingir o melhor desempenho na resolução dos maiores problemas de

otimização é escolher de uma combinação adequada dos conceitos de múltiplos algoritmos (RAIDL; PUCHINGER, 2019).

Segundo Raidl, Puchinger (2019), a criação de uma solução híbrida eficiente é, sem dúvida, uma tarefa complexa e que pode ser considerada uma arte. Existem diversas estratégias que se mostraram eficazes em diversas ocasiões e que fornecem uma base para as futuras aplicações.

Em trabalhos como Ancioto (2019), a aplicação da hibridização do GA e do ACO, para os problemas do caixeiro viajante e do roteamento de veículo, demonstrou ser bastante eficaz se comparada à utilização das abordagens metaheurísticas puras.

A hibridização tem como objetivo melhorar a capacidade de busca local dos algoritmos, a qualidade da solução e a convergência. Na abordagem de hibridizar o GA e o ACO há um processo de integração. Acontece a migração das soluções encontradas pelos algoritmos, sempre que que um algoritmo não consegue alcançar um ponto mínimo local, o outro oferece essa capacidade para ele (CARVALHO; YAMAKAMI, 2008).

O GA e ACO são algoritmos que têm como base uma população. O GA é muito eficiente em busca global, enquanto o ACO é uma combinação de mecanismos de *feedback* positivo. Em contrapartida, o GA não usa o *feedback* no sistema de informação e a eficiência da acurácia é baixa. Para o ACO, a velocidade de solução é lenta e falta informação de forma precoce. A combinação desses algoritmos resulta num auxílio mútuo, em que o GA gera a informação distribuída e o ACO tenta obter a acurácia da solução (GUANDONG, PING, QUN; 2007).

3 MÉTODO

Antes de especificar qualquer método ou atividade que esteja envolvida na execução da pesquisa, é importante delimitar suas características com base nas conceituações fornecidas por Gil (2002) e Wazlawick (2014). Em relação à sua natureza, pode ser classificada como um resumo de assunto, uma vez que serão descritos os conceitos usados, além de acrescentar o que tem sido pesquisado e publicado na comunidade científica. Em termos de procedimentos técnicos, esse trabalho de conclusão de curso pode ser classificado como pesquisa bibliográfica ou como uma pesquisa experimental. Na pesquisa bibliográfica, dada a necessidade de construir um referencial teórico sólido sobre os temas abordados e o estado da arte. Já a pesquisa experimental é caracterizada pelo fato de os pesquisadores terem a atuação direta sobre os objetos de estudo, alterando parâmetros, instâncias e técnicas, bem como avaliando os resultados alcançados.

A proposta desta pesquisa é utilizar a hibridização das metaheurísticas do GA e do ACO para resolver o LRP-SAEVs e avaliar os resultados alcançados. O estudo de Ma, Hu e Wu (2021) apresentou uma solução utilizando um algoritmo genético e é realizada uma comparação com o *General Algebraic Modeling System* (GAMS) que é *software* projetado para modelar e resolver problemas de otimização linear, não linear e de inteiro misto.

As instâncias utilizadas nesse trabalho foram criadas pelo autor. O objetivo era obter resultados comparáveis à literatura, porém isso não será possível. As instâncias utilizadas no artigo de Ma, Hu e Wu (2021) foram solicitadas para serem como *benchmark*. Porém, não houve retorno dos autores.

Em todas as instâncias serão analisadas as soluções apresentadas nos quatro possíveis cenários para o LRP-SAEVs, apresentados por Ma, Hu e Wu (2021), que estão na introdução desse trabalho. Cada problema e instância foram resolvidos com o GA, ACO e a hibridização.

4 TRABALHOS RELACIONADOS

O problema do LRP-SAEVs proposto por Ma, Hu, Wu (2021) é muito recente na literatura, logo, não foi possível encontrar trabalhos que se refiram ao problema. Dessa forma, são apresentados trabalhos que tratam de três tópicos que se baseiam o problema e para Ma, Hu, Wu (2021) são considerados interessantes, sendo Sistema de Veículos Elétricos Autônomos Compartilhados (SAEVs), Problema de Roteamente e Localização de Elétricos (ELRP) e *dial-a-ride problem* (DARP).

A estimativa dos principais parâmetros do sistema de SAEV pode ser feito por dois processos: otimização e simulação. Para o processo de otimização, é necessário elaborar uma formulação desenvolvida para o SAEV. A opção de simulação visa replicar o sistema da forma mais realista possível (SANTOS; BIROLINI; CORREIA, 2023). Pode-se acrescentar diversas componentes estocásticos, como variações de demanda e tempo de viagem, e praticamente não há limitações nos componentes do modelo, mas à medida que o modelo cresce, dificulta a interpretação (MARTINEZ; CORREIA; VIEGAS, 2014). Vários aspectos do modelo SAEV têm sido discutidos e apresentados em trabalhos como Chen, Kockelman, Hanna (2016); Lee, Kang, Lee (2020); Martinez, Correia, Viegas (2014) e Farhan, Chen (2018).

Em Lee, Kang, Lee (2020) consideram a importância dos SAEVs, que têm sido de bastante interesse. Os SAEVs simplificam o acesso a veículos, evitam custos de estacionamento, reduzem tempo e dinheiro. Em seu artigo, Lee, Kang, Lee (2020) abordam um problema real na região do Texas, usando a simulação. Santos, Birolini, Correia (2023) propuseram um programa completo que se baseia em fluxo, espaço de tempo e energia para projetar um sistema SAEV. O processo de otimização é considerado, assim como, apresenta um modelo para um estudo de caso real.

De acordo com Yu, Zhou, Liu (2019), o problema de roteamento de localização (LRP) é um problema popular e um tópico desafiador para a área de logística, pois envolve a localização dos depósitos que fizeram o atendimento e a rota dos veículos para que esse atendimento seja possível. No trabalho de Chen et al. (2009) é apresentada uma solução do LRP que usa o algoritmo genético. Yu, Zhou, Liu (2019) apresentam uma solução para o LRP considerando a capacidade do veículo, e propõem uma hibridização do algoritmo genético. Já no problema proposto neste trabalho, a capacidade do veículo não será considerada, mas sim uma proposta de hibridização.

Schiffer, Walther (2017) apresentam uma variação do problema do LRP, que considera que os veículos usados são elétricos (ELRP). A recarga parcial também é apresentada nesse estudo. Outro ponto relevante é a janela de tempo, ou seja, a necessidade de atender o usuário dentro do prazo estabelecido. Este é um problema muito próximo ao LRP-SAEV

Cordeau, Laporte (2007) definem o DARP como um problema que envolve a criação de rotas de veículos e horários para diversos usuários, que requerem especificação de local e horário de origem e destino da solicitação. O objetivo é elaborar um conjunto de rotas de veículos de custo mínimo que permitam acomodar o maior número possível de usuários, sem limitações. No trabalho proposto por Cordeau, Laporte (2007) é apresentado um modelo e vários algoritmos para solução do DARP. Parragh, Sousa, Almada-lobo (2014) apresentam uma solução para o problema, tendo em vista que, pode haver passageiros de origens e destinos diferentes, no mesmo carro ao mesmo tempo.

5 O PROBLEMA LRP-SAEVS

O LRP-SAEVs é um problema de otimização combinatória que tem como objetivo principal otimizar a frota e a localização das estações. Nas definições do modelo, são assumidas e simplificadas as definições a seguir.

Ao sair do depósito para a primeira demanda daquele veículo, é garantido que sua bateria esteja totalmente carregada. Os SAEVs operam na cidade com estradas fáceis e não haverá diferença na capacidade de carga. Fatores como altitude, carga e qualidade da estrada que influencia na variação do consumo de energia serão desconsiderados. Dessa forma, é possível supor que a velocidade é constante, em que o consumo de energia pode ser simplificado como uma função linear da distância percorrida e a recarga é dependente de um tempo de recarga linear.

Apenas a janela de tempo dos usuários no ponto de origem é considerada. Não é relevante, nesse caso, o horário em que o usuário chega ao seu destino. Apenas é garantido que ele chegará, se sua solicitação for considerada. O tempo de serviço dos veículos ao fazer o atendimento pode ser desconsiderado. A frota é operada de forma centralizada, para que os usuários sejam notificados da proximidade do veículo e estejam preparados para o embarque. Nesse sentido, sendo a primeira demanda do veículo no dia, o tempo de sair do depósito e atender esse indivíduo, para tal, é garantido que isso seja possível para qualquer usuário.

Todas as nomenclaturas, parâmetros e variáveis estarão descritas na Seção 5.2. Não existe instalação de carregamento adicional para veículos, exceto para as pilhas de carregamento do depósito, e múltiplas visitas para o carregamento são permitidas. Alguns pontos fictícios são definidos para cada depósito e pontos de recarga são definidos para todos as estações. Outro ponto relevante, é o de atendimento de clientes, que é definido como demanda ou serviço. Todo ponto de demanda de pedidos de viagem é dividido em 2 conjuntos, o conjunto de pontos de origem dos passageiros e o conjunto de pontos de destino. A seguir, é apresentada a formulação matemática, que se encontra dividida para os 4 tipos de problema. Primeiro, é apresentada a formulação para serviço total e a combinação de recarga parcial e total; depois, é apresentada a formulação para serviço parcial e as combinações de recarga total e parcial.

5.1 Formulação matemática

A formulação apresentada nessa seção foi retirada de Ma, Hu, Wu (2021), que define o LRP-SAEV como um grafo direcionado $G = (N, A)$, sendo N o conjunto de vértices e as arestas $(i, j) \in A$. O grafo não pode ser considerado completo, pois algumas restrições não permitem que algumas arestas sejam utilizadas. Sendo $N = H \cup P \cup O \cup D$, que são todos os pontos existentes no problema, como conjunto de localizações candidatas, conjunto de pontos de recarga, conjunto de origens e de destinos das viagens solicitadas. As arestas podem existir, todas ligando um vértice a outro, porém algumas restrições tornam essas arestas proibidas.

5.1.1 Serviço Total

A função objetivo para o atendimento total é formulada na Equação 3, tendo como propósito minimizar o custo total diário. Sendo o primeiro, o custo da estação ($C_{estação}$), em que está incluso nesse custo a construção do espaço de estacionamento e os carregadores. O valor do custo da estação é considerado se ela está aberta, por isso y_i são uma variável binária que é 1 caso a estação i esteja aberta e 0 caso esteja fechada. Quanto ao custo do veículo ($C_{veículo}$), já está incluso a compra e o custo de carregamento. Esse valor é multiplicado pela variável x_{ijk} que também é uma variável binária, que é 1 se o veículo k viajou do ponto i para o ponto j. E por último, o custo da distância (C_{dist}) de ir de um ponto a outro, multiplicado pela variável x_{ijk}.

$$\min \sum_{i \in H} y_i * C_{estação} + \sum_{i \in H} \sum_{j \in N \setminus H} \sum_{k \in K} x_{ijk} * C_{veículo} + \sum_{i \in N} \sum_{j \in N, i \neq j} \sum_{k \in K} x_{ijk} * C_{dist} \qquad (3)$$

Neste problema, as restrições serão divididas em três tipos: de rota dos veículos e das estações; do nível de energia para recarga e; de tempo. Inicialmente, serão apresentadas as restrições de rota dos veículos e estações. A restrição 4 garante que um veículo $k \in K$ não pode ir de um depósito $i \in H$ para um ponto de carregamento $j \in P$. A restrição 5 garante que um veículo $k \in K$ que se inicie do depósito $i \in H$, não podendo iniciar os atendimentos daquele carro de outro ponto.

As Restrição 6 e 7 indicam que apenas estações que são consideradas como abertas recebem veículos. E, cada estação candidata, tem um ponto virtual correspondente sobre ela, podendo estar aberta ou não, a Restrição 8. A Restrição 9 mostra que cada ponto de demanda só pode ser acessado por um veículo e a conservação do fluxo é controlada pela Restrição 10. A Restrição 11 garante que um veículo não vá para a estação de carregamento com passageiros no veículo. No geral, o veículo vai para a origem do passageiro faz a coleta e vai direto para o local de destino dele.

$$x_{ijk} = 0 \ \forall i,j \in H \cup P, \forall k \ \in K \tag{4}$$

$$\sum_{i \in H} \sum_{j \in N \setminus H} x_{ijk} \ \leq 1, \forall k \ \in K \tag{5}$$

$$x_{ijk} \leq y_i \ \forall i \ \in H \cup P, j \in N, i \neq j, k \in K \tag{6}$$

$$x_{jik} \leq y_i \ \forall i \ \in H \cup P, j \in N, i \neq j, k \in K \tag{7}$$

$$y_i = y_j \ \forall i \ \in P_j, j \ \in H \tag{8}$$

$$\sum_{k \in K} \sum_{i \in N, i \neq j} x_{ijk} = \ 1, \forall j \ \in O \cup D \tag{9}$$

$$\sum_{j \in N, j \neq i} x_{ijk} - \sum_{j \in N, j \neq i} x_{jik} = \ 0 \ \forall i \ \in N \setminus H, k \ \in K \tag{10}$$

$$\sum_{k \in K} x_{ijk} = \ 1 \ \forall i \ \in \ O_m, j \ \in \ D_m \tag{11}$$

Para as restrições de controle de níveis de energia tem-se as restrições, sendo que a Restrição 12 garante que o veículo esteja completamente carregado quando ele sair do depósito. A Restrição 13 e 14 definem que, quando o veículo passar por uma estação de carregamento, ele deve fazer o carregamento; para a Restrição 13 o veículo tem de estar totalmente carregado e para a Restrição 14 acontece a relaxação, para aceitar recarga parcial. Após o carro visitar os pontos de origem e destino do usuário, ele tem que ter energia o suficiente para fazer todo o atendimento e voltar para um ponto de recarga, o que é garantido pela Restrição 15. A Restrição 16 rastreia o nível de energia do veículo de acordo com a sequência de pontos visitados, a redução do nível de energia ocorre de acordo com o consumo no deslocamento na aresta (i, j). A Restrição 17 impõe que o nível de energia, ao chegar em qualquer ponto, não pode ser menor que o nível limite da bateria, sendo o limite o valor que o

veículo necessita ir para uma estação de carregamento e fazer a recarga para que não haja danos à bateria.

$$de_{ik} = E * \sum_{j \in N \setminus H} x_{ijk} \ \forall i \in P, k \in K \tag{12}$$

$$ae_{ik} * \sum_{j \in N, i \neq j} x_{ijk} \ \leq \ de_{ik} \ \leq E * \sum_{j \in N, i \neq j} x_{ijk} \ \ \forall i \ \in P, k \ \in K \tag{13}$$

$$de_{ik} = E * \sum_{j \in N, i \neq j} x_{ijk} \, \forall i \ \in P, k \ \in K \tag{14}$$

$$de_{ik} = ae_{ik} \, \forall i \ \in O \cup D, k \ \in K \tag{15}$$

$$ae_{jk} \leq \left(de_{ik} - r * d_{ij}\right) * x_{ijk} + E * \left(1 - x_{ijk}\right) \forall i, j \in N, k \ \in K \tag{16}$$

$$\alpha * E * \sum_{j \in N, j \neq i} x_{ijk} \ \leq \ ae_{ik} \ \leq E * \sum_{j \in N, j \neq i} x_{jik} \ \ \forall i \ \in N, k \ \in K \tag{17}$$

As Restrições de tempo são expressas a seguir: A Restrição 18 especifica que no momento que ele sai do depósito para sua primeira demanda, o tempo é zero, determinando que é possível atender qualquer usuário. A Restrição 19 considera a mudança de tempo do veículo de acordo com a sequência de pontos visitados e a distância entre eles. A Restrição 20 mostra a relação de permanência do veículo no ponto de carregamento e a energia suplementar, garantindo que o tempo de saída não ultrapasse o tempo máximo de operação. As Restrição 21 e 22 impõem a hora que o veículo chega ao ponto de embarque e sai dele respectivamente. O tempo de serviço do ponto de origem para o ponto de destino são desprezados, então, a Restrição 23 sincroniza quando o veículo chega no ponto de destino e sai desse ponto.

$$dt_{ik} = 0 \ \forall i \ \in H, k \ \in K \tag{18}$$

$$at_{jk} \geq \left(dt_{ik} + t_{ij}\right) * x_{ijk} - T_{max} * \left(1 - x_{ijk}\right) \forall i, j \ \in N, i \neq j, k \ \in K \tag{19}$$

$$\left[at_{ik} + \frac{de_{ik} - ae_{ik}}{g}\right] * \sum_{i \in N, i \neq j} x_{ijk} \leq dt_{ik} \leq T_{max} * \sum_{i \in N, i \neq j} x_{ijk} \ \forall i \ \in H, k \ \in K \tag{20}$$

$$at_{ik} \leq lt_i * \sum_{j \in N, i \neq j} x_{jik} \ \ \forall i \ \in O, k \ \in K \tag{21}$$

$$dt_{ik} = \max\left\{at_{ik}, et_i\right\} \forall i \ \in O, k \ \in K \tag{22}$$

$$dt_{ik} = at_{ik} * \sum_{j \in N, j \neq i} x_{ijk} \ \ \forall i \ \in D, k \ \in K \tag{23}$$

5.1.1.1 Formulação para a Recarga Total e Serviço Total (I - RTST)

O problema RTST considera a recarga total, sendo assim, sempre que o veículo parar para fazer o abastecimento de energia, ele sai do ponto de recarga com o nível de energia totalmente carregado. O serviço é definido como total, todos os usuários serão atendidos. A função objetivo apresentada na Equação , será considerada para esse problema; as Restrições serão consideradas da 4 a 23, exceto a 14, que é considera para recarga parcial.

5.1.1.2 Formulação para o Recarga Parcial e Serviço Total (III - RPST)

O problema RPST, a recarga é parcial, quando o veículo para em um ponto de recarga ele não precisa sair com a bateria completa, por isso tem se a Restrição 14 que evidência isso. Todos os usuários são atendidos, pois o serviço é definido como total. A função objetivo também é a Equação 3 e as Restrições são da 4 a 23, exceto a 13 que considera a recarga total.

5.1.2 Serviço Parcial

Nesse caso, alguns atendimentos podem ser recusados, sendo assim, um custo adicional é adicionado à função objetivo para cada viagem não atendida. Na primeira parte da função objetivo, vale o mesmo apresentado na função objetivo 3, porém, aqui é adicionado um C_{reject} que é o custo unitário para cada viagem que foi recusada, que é apresentado na Equação 24.

$$\min \sum_{i \in H} y_i * C_{estação} + \sum_{i \in H} \sum_{j \in N \setminus H} \sum_{k \in K} x_{ijk} * C_{veículo} + \sum_{i \in N} \sum_{j \in N, i \neq j} \sum_{k \in K} x_{ijk} * C_{dist} + \left(m - \sum_{i \in N} \sum_{j \in O, i \neq j} \sum_{k \in K} x_{ijk} \right) * C_{rejeita} \quad (24)$$

As restrições do serviço total (5.1.1) também valem para o serviço parcial, porém, é necessária uma relaxação das restrições. A Restrição 9 será relaxada para a Restrição 25, indicando que nem todo ponto de atendimento será visitado pelos

veículos. Também é modificada a Restrição 11 pela 26, para ter certeza que um ponto de origem e um ponto de entrega ou serão os dois visitados, ou nenhum deles será visitado.

$$\sum_{k \in K} \sum_{i \in N, i \neq j} x_{ijk} \leq 1 \ \forall j \in O \cup D \tag{25}$$

$$\sum_{k \in K} x_{ijk} = \sum_{k \in K} \sum_{h \in N, h \neq i} x_{hik} = 1, \forall i \in O_m, j \in D_m \tag{26}$$

5.1.2.1 Formulação para o Recarga Total e Serviço Parcial (II - RTSP)

O RTSP considera o serviço parcial, em que pode haver a recusa de algumas solicitações de usuários, nem todos precisam ser atendidos. Nesse caso, é necessário garantir que o veículo, ao recusar uma viagem, não vá nem ao seu ponto de origem, nem ao de destino. A recarga para esse problema é considerada total, ao deixar o ponto de recarga o nível de energia está com o máximo permitido para a bateria. A função objetivo para o serviço parcial muda, pois tem que considerar um custo a mais para casos de recusa de viagem, então, a equação da função objetivo é a Equação 24. As Restrições consideras são da 4 a 23, não considerando a restrição 14, e tem a troca das Restrições 9 pela 25 e da 11 pela 26, para considerar o serviço parcial.

5.1.2.2 Formulação para o Recarga Parcial e Serviço Parcial (IV - RPSP)

O RPSP também considera o serviço parcial, relaxando algumas restrições para aceitar a recusa de viagem e na função objetivo é considerado um custo a mais, pois toda viagem que for recusada tem um custo. A recarga parcial é considerada, podendo o veículo ir para um ponto de recarga e fazer apenas a quantidade de recarga necessária. A função objetivo considerada é a Equação 23 e as Restrições são da 4 a 23, não considerando a 13, e tem a troca das Restrições 9 pela 25 e da 11 pela 26.

5.2 Nomenclatura

ae_{ik}= A energia remanescente de um veículo k quando ele chega no ponto i
at_{ik}= O tempo de chegada de um veículo k no ponto i
C_{dist}= O custo de uma unidade de distância percorrida pelo veículo, tendo essa unidade, como o custo do veículo k de ir do ponto i para j
$C_{rejeita}$ = O custo da penalidade por rejeitar uma viagem

$C_{estação}$ = O custo da construção de uma estação, incluindo custo de instalação de pontos de recarga
$C_{veículo}$ = Custo da aquisição de um veículo
m = número de requisições de serviço
D = Conjunto das demandas de serviço, apenas para os pontos de destino
de_{ik} = A energia remanescente de um veículo k quando ele sai no ponto i
d_{ij} = Distância do ponto i para o ponto j
dt_{ik} = O tempo de partida do veículo k para o ponto i
E = Capacidade da bateria do veículo
et_m= Tempo mais cedo permitido para a demanda m
g = Taxa de recarga da infraestrutura
H = Conjunto da localização dos depósitos
h = Número de depósitos
K = Conjunto de veículos
k = Número de veículos
lt_m = Último período de tempo de viagem permitido, $lt_m = et_m + w$
w = é a tamanho da janela de tempo, quanto um veículo pode atrasar para chegar até o ponto
M = Conjunto de todas as demandas de serviço
N = Conjunto de todos os pontos $(N = HUPUOUD)$
O = Conjunto das demandas de serviço, apenas para ponto de origem
P = Conjunto de todos os pontos de recarga
r = Taxa de consumo do veículo elétrico
t_{ij} = Tempo de viagem do ponto i ao ponto j
T_{max} = Tempo máximo de operação
x_{ijk} = Valor Binário (0 ou 1), se o veículo k viaja de i para j
y_i = Valor Binário (0 ou 1), se a localização candidata $i \epsilon H$ está aberta, estação do SAEVs

6 ABORDAGENS UTILIZADAS

Nesse capítulo, é mostrado como foi realizada a implementação dos algoritmos utilizados para a resolução das variantes do LRP-SAEVs. Antes, é explicado como foram elaboradas as representações das soluções. Então, as duas metaheurísticas são apresentadas, sendo elas o algoritmo genético e colônia de formigas. Por fim, tem-se a hibridização do GA com o ACO.

6.1 Representação de uma solução

A representação da solução do problema define se um algoritmo poderá ter sucesso, por isso é importante a escolha de uma boa representação da solução. O aspecto que tem que ser levado em consideração para isso é a decisão de como o indivíduo será representado, de forma a potencializar a solução de acordo com o problema (BRABAZON; O'NEILL; MCGARRAGHY, 2015).

A solução foi representada em um alto nível, com as informações de quais estações estão abertas, lista de carros e lista de recusa, apenas em casos de serviço parcial. Esse tipo de representação é mais intuitivo e permite que o a execução da função objetivo seja mais fácil. Também traz um entendimento melhor da representação das soluções do problema (LOPES; FERREIRA; SANTOS, 2016).

Para o problema em questão, a solução foi representada de forma a armazenar as informações relevantes do problema, sendo que cada solução corresponde a um indivíduo ou formiga para o GA e o ACO, respectivamente.

Em termos práticos, cada solução tem um conjunto com todas as estações, indicando quais estão ativas e quais não estão. Para a variante que considera o serviço parcial, é armazenada uma lista com os atendimentos recusados. Uma lista com todos os veículos também é armazenada. Cada veículo contém as informações das arestas que ele passou, o tempo de percurso e o nível de energia atual.

6.2 Implementação do GA

No GA básico, a representação da solução envolve principalmente a codificação e a decodificação da solução, que é definida por um vetor de bits ou caracteres que representam um indivíduo, sendo essa uma solução possível. Para

calcular a função objetivo, o indivíduo tem que ser decodificado, assim como para ser utilizado nos outros problemas (YANG, 2021).

O GA apresentado foi utilizado para a composição do algoritmo híbrido; para uma melhor compatibilidade entre os dois algoritmos da representação proposta em 6.1 foi utilizada.

O Algoritmo 3 demonstra como foi implementada a solução do GA, que se baseia no Algoritmo 1. Não é considerada a solução de codificação e decodificação, além de não considerar a probabilidade do cruzamento e da mutação. Em todas gerações de novos indivíduos foram consideradas mutação e crossover.

Algoritmo 3 – Algoritmo do GA utilizado

```
1  entrada: uma instância do problema a ser resolvido
2  saída: uma solução ótima ou a melhor encontrada
3  Inicializar a população de soluções candidatas
4  Calcular a função objetivo para cada indivíduo da população
5  enquanto t < máximo número de gerações faça
6      Gera novas soluções pelo cruzamento e mutação
7      Aceita novas soluções se a função objetivo reduzir
8      Calcular a função objetivo para cada indivíduo da população
9      Seleciona alguns ou todos da população atual pelos melhores para a
       próxima geração, apenas os melhores(elitismo)
10     Atualiza t = t + 1
11 fim enquanto
```

O GA inicializa com uma população aleatória. Para cada demanda, um valor aleatório é gerado, esse valor indica em qual carro será feita a tentativa de inserção daquela solicitação de usuário. Se for possível aquele carro atender essa demanda, ela é inserida. Caso não seja possível, um novo carro é inserido na lista para atender a demanda. Quando todas as demandas forem atendidas, então tem-se um novo indivíduo, adicionado a população inicial.

Para todos os indivíduos da população inicial utiliza a função objetivo. Para selecionar os melhores indivíduos, até o valor do elitismo se mantém na população, assim, os indivíduos que estiverem abaixo são descartados.

Cada repetição é uma nova geração. O valor que falta para completar a quantidade da população é obtido a partir de quantos novo indivíduos foram gerados pelo cruzamento e mutação. Os indivíduos que foram para o cruzamento ou mutação são escolhidos gerando um valor aleatório que indica um indivíduo na população, sendo ele o mandado para a mutação e ou cruzamento. Para cada um, um valor aleatório é gerado.

Com a nova população é feita a seleção, ordena os indivíduos de acordo com o resultado da função objetivo. E os melhores se mantém para a próxima geração, dada uma quantidade de melhores. O elitismo define quantos vão sobreviver para a próxima geração.

O cruzamento será executado como definido por Lopes, Ferreira, Santos (2016), com algumas alterações pois no artigo citado ocorria a troca entre estações. Neste trabalho, foi realizada a troca entre a lista de carros dos indivíduos. Uma quantidade de carros do primeiro é levada para compor a solução do segundo, e uma quantidade de carros no segundo é levada para compor a solução do primeiro. No final, os indivíduos selecionados foram alterados e retornados.

Na mutação, é selecionado um novo indivíduo aleatoriamente, sendo que dois carros da lista de carros desse indivíduo são selecionados aleatoriamente. Um novo carro é criado, esse carro tenta atender todas as demandas que os dois anteriores possuíam. Se for possível atender, elas são alocadas ao carro criado. caso não seja possível, repete-se o processo com os que não foram possível. Pode ser criado uma lista com apenas um carro ou com vários carros. No final, o indivíduo selecionado foi alterado e retornado.

A quantidade de gerações e de população foi definida utilizando pré-teste de soluções, que foi definido nos resultados.

6.3 Implementação do ACO

Neste algoritmo foi necessário utilizar uma quantidade maior de memória para armazenar os dados da solução. Ao guardar os valores do feromônio para cada aresta, é aumentada a complexidade do método (JUNIOR, 2019).

No Algoritmo 4, é apresentada a solução utilizada para execução do ACO, o qual é semelhante ao Algoritmo 2, que é o básico do ACO. Apenas alguns pontos sofreram alteração.

O feromônio foi representado como uma lista de adjacência. Por não ser um grafo completo, uma lista de adjacência é computacionalmente mais barata do que uma matriz de adjacência. Ela informa o valor de feromônio para cada aresta, considerando as restrições do problema.

Ao inicializar uma geração de uma formiga, cada demanda é analisada de acordo com as opções disponíveis para cumpri-la. As opções para a primeira demanda será escolher uma estação para início do carro, ou na variante serviço parcial, essa demanda pode ser recusada. As próximas demandas também têm as mesmas opções do primeiro, mas agora já existem carros que essas demandas podem ser adicionadas, então é adicionado a opção.

A probabilidade de seleção de cada opção é calculada a partir do inverso do valor excedente que será adicionado à função objetivo já existente na próxima aresta da formiga. O valor encontrado é a probabilidade de escolha para aquela aresta, caso ela seja escolhida, ela deve ser adicionada à parcial da formiga. Ao adicionar todas as demandas, sendo atendidas ou não, tem-se uma nova formiga.

A nova formiga é adicionada a uma lista que será utilizada para fazer a atualização do feromônio. Todos os valores da lista são aplicados ao feromônio.

Algoritmo 4 – Algoritmo do ACO utilizado

```
1  entrada: uma instância do problema a ser resolvido
2  saída: uma solução ótima ou a melhor
3  Inicializar os parâmetros de entrada;
4  Inicializa a variável de feromônio
5  sbs -> nulo
6  enquanto a condição de fim não for atendida faça
7      para j = 1, ..., m faça
8          s ← gerar formiga
9          se (f (s) < f(sbs)) ou (sbs = Nulo) então sbs ← s fim se
10     fim para
11     Aplica a atualização do feromônio
12 fim enquanto
```

No Algoritmo 4, a linha 3, os parâmetros do problema são adicionados a suas respectivas variáveis. A inicialização da lista de adjacência que armazena o feromônio

é executada. O sbs, variável que armazena o melhor valor encontrado, é tido como nulo no início. Na linha 6 é repetido na quantidade de vezes definida como quantidade de geração. Na linha 7 foi definida a quantidade de formigas que serão geradas para aquela iteração, sendo m a quantidade de formigas. A linha 9 verifica se o valor encontrado é melhor do que o já armazenado. Por fim, é aplicada a atualização do feromônio na linha 11. A iteração ocorre até que a condição de parada seja atendida.

6.4 Implementação do algoritmo híbrido GA-ACO

A maneira mais natural de hibridizar dois algoritmos é embutir um algoritmo em outro para obter tanto uma solução inicial promissora, quanto soluções intermediarias melhoradas (RAIDL; PUCHINGER, 2019).

Dessa forma, a solução utilizada será uma junção do GA com o ACO, de forma que, a base será o algoritmo genético com a adição do algoritmo de colônias de formigas, adicionando ao novo algoritmo as vantagens de cada uma.

Um desafio importante para várias metaheurísticas, como o GA, é encontrar soluções iniciais que melhorem o avanço da busca local ou da metaheurística (RAIDL; PUCHINGER, 2019). Na implementação do GA, apresentada no item 6.2, a população é gerada de forma aleatória. No algoritmo híbrido, a população inicial é gerada pelo ACO.

Na geração de novas populações, a ideia é similar à apresentada por ZHAO et al., (2018), na qual, a cada geração, são criados novos indivíduos, utilizando tanto o cruzamento e a mutação presentes no GA, quanto a geração de formigas presentes no ACO.

No Algoritmo 5, é apresentado a implementação do GA-ACO. As linhas 1 e 2 apresentam a entrada de uma instância do problema e a saída da melhor solução encontrada, respectivamente. Na linha 3, é executada a inicialização dos parâmetros, como quantidade de geração, quantidade da população, dentre outros. Na linha 4, os valores do feromônio são inicializados. Na linha 5, uma população inicial é gerada utilizando a geração das formigas. A linha 6 itera sobre cada nova geração. A atualização do feromônio para cada nova população é executada na linha 7. A população da geração anterior é armazenada na linha 8. A função objetivo é utilizada para definir os melhores indivíduos daquela população na linha 9. A seleção da população acontece na linha 10, utilizando o valor de elitismo para que os indivíduos

piores daquela população sejam abandonados na nova geração. As linhas 11, 12 e 13 são utilizadas para gerar novos indivíduos, utilizando o cruzamento e a mutação, como definidos pelo GA, e a geração das formigas, como definido pelo ACO. A atualização do contador de gerações é executada na linha 15.

Algoritmo 5 – Algoritmo do GA-ACO utilizado

```
1  entrada: uma instância do problema a ser resolvido
2  saída: uma solução ótima ou a melhor
3  Inicializar os parâmetros
4  Inicializar os valores do feromônio
5  Inicializar a população com geração de formigas (tamanho população)
6  enquanto t < máximo número de gerações faça
7      Atualizar o feromônio com a população
8      População da geração anterior armazenada
9      Definir a função objetivo para minimizar
10     Selecionar os melhores(elitismo) para essa geração
11     enquanto o total da população não for alcançado faça
12         Gerar novos indivíduos usando cruzamento e mutação (selecionar
           aleatoriamente)
13         Gerar nova formiga
14     fim enquanto
15     Atualiza t = t + 1
15 fim enquanto
```

7 EXPERIMENTOS COMPUTACIONAIS

Neste capítulo são descritos os experimentos computacionais realizados para os 4 problemas. Cada um dos 3 algoritmos foi aplicado a todas as variantes.

Os experimentos foram executados em um computador com processador *AMD Ryzen* 5 2500U sobre o sistema operacional *Microsoft Windows* 10 *Pro* (64 bits).

Os algoritmos propostos foram implementados utilizando a linguagem *Python* com o ambiente de desenvolvimento *Spyder* presente no *Anaconda Navigator* (*anaconda3*). A versão do *Python* era 3.9.12 64-bit.

7.1 Elaboração das instâncias

No artigo de Ma, Hu, Wu (2021), foram utilizadas as instâncias adaptadas do trabalho de Parragh, Sousa, Almada-lobo (2014). Para uma melhor comparação dos resultados apresentados neste trabalho com o artigo principal, seria interessante ter as mesmas instâncias. Como não foi possível obter essas instâncias, as instâncias apresentadas nesse trabalho foram criadas pelo autor.

Cada nova instância foi criada utilizando algumas definições do artigo principal. Para a coordenada das estações candidatas, foi utilizado o proposto pelo autor, quando se tem apenas duas estações, sendo os pontos (-5, -5) e (5, 5) para as estações.

Com relação ao tempo, todos os atendimentos foram definidos dentro do tempo de 24 horas, pois o problema considera o atendimento para um dia de serviço. Apenas para ajudar na execução do algoritmo, o tempo ficou definido em minutos, de forma que o tempo dos atendimentos seja menor que 1440 minutos.

Para os pontos de atendimento, foram considerados aqueles em que existe a possibilidade de que um novo veículo saia do depósito, execute o atendimento e retorne para a estação, para tanto, pontos que não atendam esse requisito não foram considerados.

Foram criadas 10 instâncias, utilizando a quantidade de atendimento definidos pelo autor e adicionando mais opções para teste. Na Tabela 1, são apresentadas todas as instâncias que foram utilizadas para executar os experimentos.

Tabela 1 – Instâncias do problema

Instância	Quantidade de estações	Quantidade de atendimentos
S2_A4_T1	2	4
S2_A8_T1	2	8
S2_A12_T1	2	12
S2_A16_T1	2	16
S2_A20_T1	2	20
S2_A25_T1	2	25
S2_A30_T1	2	30
S2_A50_T1	2	50
S2_A75_T1	2	75
S2_A100_T1	2	100

Fonte: Autoria Própria

7.2 Definição dos parâmetros

Baseando-se no trabalho de Ma, Hu, Wu (2021), nesta seção são apresentados alguns parâmetros para as instâncias e para execução dos algoritmos.

O nível de bateria máximo foi estabelecido em 15 kWh. Para a taxa de consumo de energia do veículo tem-se 0,15kWh/km, e para a taxa de recarga, utilizou-se como 0,4 kWh/minuto. A velocidade média do veículo foi fixada em 35 km/h. O limite do veículo, isto é, o menor nível de bateria que o veículo pode atingir, foi definido como 3 kWh. O tempo limite de atraso para estar na origem do usuário é de 5 minutos. Em relação à recusa de viagens, apenas 30% das demandas podem ser recusadas.

Os custos definidos no artigo são apresentados na moeda RMB. Sabendo que essa moeda tem um valor diferente do real, apenas para fins acadêmicos, e em uma tentativa de aproximar dos resultados encontrados no artigo, será considerado que 1 real equivale a 1 RMB. O custo do veículo para um dia ficou definido como R$ 137,00. O custo das estações, tendo incluso a construção e a instalação de pontos de carregamento para um dia de utilização, ficou estipulado em R$ 2328,76. Para cada rota que o veículo passa, o custo unitário é de R$ 19,37. O custo de recusar uma viagem, isto é, o custo de não fazer um atendimento, é R$ 19,73.

Para a execução dos algoritmos foi considerado o número de gerações igual a 300 e a quantidade de indivíduos/formigas igual a 300. Esse valor foi definido após um teste inicial com 4 instâncias para o problema com recarga total e serviço total. As instâncias utilizadas foram as S2_A4_T1, S2_A25_T1, S2_A50_T1 e S2_A75_T1. Variando a quantidade de geração e a quantidade da população de 50 em 50, começando como as duas em 50, até chegar nas em 400, executando cada algoritmo para cada combinação de quantidade e para cada instância 10 vezes. O elitismo foi considerado com 0,6. A evaporação de feromônio é igual a 0,3.

7.3 Resultados

Como as instâncias foram criadas pelo autor, não se conhece ainda o valor da solução ótima. Então, o melhor resultado encontrado após a execução de todos os algoritmos foi utilizado como uma referência do valor ótimo.
Com relação à métrica para comparação das soluções obtidas foi utilizada a abordagem de cálculo do gap.

$$gap = \left(\frac{val(encontrado) - val(\acute{o}timo)}{val(\acute{o}timo)}\right) * 100 \qquad (27)$$

A Equação 27 apresenta o cálculo do gap, sendo val(encontrado) o valor da melhor solução obtido pela abordagem, e val(ótimo) é o menor valor encontrado dentro de todas as execuções realizadas.

Os resultados foram apresentados para cada um dos quatro problemas, aplicados às instâncias definidas anteriormente, para os três algoritmos.

Para cada uma das variantes do problema, é exibida uma tabela com os resultados das 10 instâncias apresentadas na Seção 7.1. Para cada instância, os três algoritmos foram executados 20 vezes.

Para cada instância e algoritmo são apresentadas quatro informações, sendo elas o $f_{médio}$, $f_{mín}$, o gap e o tempo. O $f_{médio}$ é a média dos valores encontrados após as 20 execuções. O $f_{mín}$ é o menor valor encontrado para a execução daquele algoritmo. O gap é a distância percentual entre o $f_{mín}$ do algoritmo atual e o melhor $f_{mín}$ dentre os três algoritmos. O tempo indica o tempo médio de execução de cada algoritmo em

segundos. Os valores em negritos indicam a melhor solução para aquela instância dentre os três algoritmos.

Ao longo deste capítulo, considera-se o termo menor valor como o menor dos valores obtidos pelos algoritmos apresentados para uma instância em específico.

I. Recarga Total, Serviço Total (RTST)

A Tabela 2 apresenta os resultados dos experimentos computacionais para a variante RTST. Analisando os resultados, o GA encontrou soluções piores para quase todas as instâncias. Para os casos em que as instâncias são menores, como aquelas que apresentam 4 e 8 serviços a serem atendidos, o gap foi menor, porém as outras soluções conseguiram encontrar o melhor resultado em todas as execuções. Um ponto a se observar é que na instância com 100 demandas o gap do ACO foi 24%, tendo a pior solução e o GA conseguiu encontrar um valor melhor que o apresentado pelo ACO. O gap do GA foi 6,11%, menor até mesmo que os anteriores.

O GA-ACO para a maioria dos casos foi a melhor solução, porém se avaliarmos o caso em que ele não encontrou o melhor, o gap dessa instância é 0,57%, o que é muito próximo do melhor valor.

Nas instâncias menores, o ACO e o GA-ACO conseguiram encontrar o melhor valor; o ACO conseguiu no seu mínimo encontrar o menor resultado, porém, quando é avaliada a média dos valores encontrados, os valores do ACO se encontram bem distante do valor ótimo; já o GA mostrou que sua média e seu valor ótimo permaneceram próximos. Com isso, a hibridização foi uma ótima opção, conseguindo utilizar o melhor do ACO e o melhor do GA.

O tempo de execução dos algoritmos é um valor muito importante para avaliação. O GA apresentou o menor tempo de execução dentre todos os algoritmos e dentre quase todas as instâncias, porém, se for observado os tempos do GA-ACO, foram muito próximos dos tempos do GA. Assim sendo, pode-se considerar até mesmo irrelevante a diferença de tempo dos dois.

Para o ACO, os tempos de execução foram mais elevados. Em instâncias menores, o tempo foi 2 vezes maior em comparação com os outros algoritmos, já para instâncias maiores esse valor aumenta muito. Para a instância com demanda igual a 100, o tempo do ACO chegou a ser 6 vezes maior que o GA-ACO e 8 vezes maior

que o GA. Tais valores são relevantes, considerando que esse problema tende a aumentar cada vez mais a quantidade de demanda dentro do sistema.

Tabela 2 – Valores das soluções para RTST

Instâncias		GA	ACO	GA-ACO
S2_A4_T1	$f_{médio}$	2678,83	2620,72	2620,72
	$f_{mín}$	2678,83	**2620,72**	**2620,72**
	gap (%)	2,22	0,00	0,00
	tempo (s)	6,80	13,09	6,31
S2_A8_T1	$f_{médio}$	2911,27	2775,68	2775,68
	$f_{mín}$	2911,27	**2775,68**	**2775,68**
	gap (%)	4,88	0,00	0,00
	tempo (s)	10,69	27,70	10,21
S2_A12_T1	$f_{médio}$	3143,71	2950,01	2950,01
	$f_{mín}$	3143,71	**2950,01**	**2950,01**
	gap (%)	6,57	0,00	0,00
	tempo (s)	14,77	52,61	14,29
S2_A16_T1	$f_{médio}$	3376,15	3085,60	3085,60
	$f_{mín}$	3376,15	**3085,60**	**3085,60**
	gap (%)	9,42	0,00	0,00
	tempo (s)	18,61	68,28	18,95
S2_A20_T1	$f_{médio}$	3706,85	3416,30	3396,93
	$f_{mín}$	3706,85	**3377,56**	3396,93
	gap (%)	9,75	0,00	0,57
	tempo (s)	26,69	97,65	27,53
S2_A25_T1	$f_{médio}$	3958,66	3845,26	3668,11
	$f_{mín}$	3958,66	**3610,00**	**3610,00**
	gap (%)	9,66	0,00	0,00
	tempo (s)	33,41	150,65	35,67
S2_A30_T1	$f_{médio}$	4229,84	3900,55	3861,81
	$f_{mín}$	4229,84	3823,07	**3803,70**
	gap (%)	11,20	0,51	0,00
	tempo (s)	38,23	179,16	41,27
S2_A50_T1	$f_{médio}$	5432,19	5044,79	4957,63
	$f_{mín}$	5432,19	4949,35	**4891,24**
	gap (%)	11,06	1,19	0,00
	tempo (s)	75,61	294,68	87,76
S2_A75_T1	$f_{médio}$	6613,76	6555,65	6187,62
	$f_{mín}$	6613,76	5977,37	**5937,22**
	gap (%)	11,39	0,68	0,00
	tempo (s)	109,27	76,66	125,14
S2_A100_T1	$f_{médio}$	8080,43	9783,89	8070,74
	$f_{mín}$	8070,74	9431,00	**7605,86**
	gap (%)	6,11	24,00	0,00
	tempo (s)	152,71	1259,55	203,10

Fonte: Autoria Própria

De forma geral, para o problema de recarga total e serviço total, o GA-ACO apresentou os melhores resultados em tempo hábil, considerando que o algoritmo encontrou as melhores soluções e as médias de soluções ficaram muito próximas dos melhores resultados.

II. Recarga Total, Serviço Parcial (RTSP)

Na Tabela 3, são apresentados os resultados para o problema com recarga total e serviço parcial. Nesse problema, o ACO conseguiu ser melhor do que no problema RTST, sendo o seu maior gap 4,38%. O ACO conseguiu encontrar as melhores soluções em instâncias menores; instâncias como a que apresenta 25 a 30 demandas, ele apresentou o melhor valor, porém, ao considerar os valores do GA-ACO para as mesmas instâncias, o valor encontrado foi próximo, apresentando um gap de 1,75% para a instância com 25 demandas e 3,92% para a instância com 30 demandas. Entretanto, se considerar a média dos resultados, o GA-ACO mostrou uma média menor que a do ACO para as duas instâncias.

Em instâncias maiores, o GA-ACO encontrou o melhor resultado, mesmo os resultados do ACO ficando próximos aos melhores resultados. Vale ressaltar que, a média do ACO foi maior do que o seu melhor resultado, já para o GA-ACO sua média se aproximou, consideravelmente, do seu melhor resultado. O GA conseguiu se aproximar dos melhores resultados apenas em instâncias pequenas, já em instâncias maiores o gap foi mais expressivo. Contudo, o $f_{médio}$ e o $f_{mín}$ ficaram próximos.

O tempo de execução dos algoritmos também foi levado em consideração. O GA teve o menor tempo de execução na maioria das instâncias, não sendo o menor apenas para instâncias menores, em que o GA-ACO foi o menor. O ACO teve um tempo muito maior que os outros algoritmos, sendo que na menor instância seu tempo chegou a ser 2 vezes maior. Para a maior instância, o ACO teve um tempo 10 vezes maior que o menor tempo. Assim, pode-se considerar que, diante de todos os fatores apresentados, a hibridização melhorou os resultados, conseguindo considerar o melhor de cada algoritmo.

Tabela 3 – Valores das soluções para RTSP

Instâncias		GA	ACO	GA-ACO
S2_A4_T1	$f_{médio}$	2640,45	2601,71	2601,71
	$f_{mín}$	2640,45	**2601,71**	**2601,71**
	gap (%)	1,49	0,00	0,00
	tempo (s)	7,05	14,22	6,37
S2_A8_T1	$f_{médio}$	2834,51	2737,66	2737,66
	$f_{mín}$	2834,51	**2737,66**	**2737,66**
	gap (%)	3,54	0,00	0,00
	tempo (s)	10,91	31,66	10,58
S2_A12_T1	$f_{médio}$	3028,57	2873,61	2873,61
	$f_{mín}$	3028,57	**2873,61**	**2873,61**
	gap (%)	5,39	0,00	0,00
	tempo (s)	14,72	53,63	17,75
S2_A16_T1	$f_{médio}$	3222,63	3048,30	3028,93
	$f_{mín}$	3222,63	**3009,56**	**3009,56**
	gap	7,08	0,00	0,00
	tempo	18,58	77,68	22,47
S2_A20_T1	$f_{médio}$	3358,94	3358,94	3224,06
	$f_{mín}$	3358,94	**3145,87**	**3145,87**
	gap (%)	6,77	0,00	0,00
	tempo (s)	21,61	108,51	26,31
S2_A25_T1	$f_{médio}$	3572,37	3573,11	3525,36
	$f_{mín}$	3553,00	**3320,56**	3378,67
	gap (%)	7,00	0,00	1,75
	tempo (s)	26,69	143,30	33,78
S2_A30_T1	$f_{médio}$	3903,79	3884,42	3768,20
	$f_{mín}$	3884,42	**3495,61**	3632,61
	gap (%)	11,12	0,00	3,92
	tempo (s)	33,74	179,11	41,63
S2_A50_T1	$f_{médio}$	4748,37	4758,47	4546,57
	$f_{mín}$	4738,86	4507,83	**4469,09**
	gap (%)	6,04	0,87	0,00
	tempo (s)	60,46	430,92	72,12
S2_A75_T1	$f_{médio}$	5767,99	5816,69	5459,48
	$f_{mín}$	5690,51	5420,74	**5323,53**
	gap (%)	6,89	1,83	0,00
	tempo (s)	80,74	655,96	107,21
S2_A100_T1	$f_{médio}$	6847,14	7155,59	6567,86
	$f_{mín}$	6798,89	6509,75	**6236,80**
	gap (%)	9,01	4,38	0,00
	tempo (s)	122,87	1259,54	166,26

Fonte: Autoria Própria

A diferença desse problema para o problema anterior é o serviço ser parcial, enquanto no outro o serviço era total. Os resultados apresentados aqui são menores

que os da Tabela 2, isso acontece porque o custo de se recusar uma rota é muito próximo do valor do custo de passar por uma aresta. Para trabalhos futuros, poderia ser avaliado o aumento do preço de rejeitar uma viagem em relação ao preço de se aceitar.

III. Recarga Parcial, Serviço Total (RPST)

Analisando a Tabela 4, para o problema de recarga parcial e serviço total, o GA-ACO foi o melhor em quase todas as instâncias; na única instância que ele não foi o melhor, o seu gap foi de 0,54%, o que é bastante próximo à melhor solução.

Para instâncias menores, o ACO conseguiu encontrar o menor valor, enquanto para a maior instância o seu gap foi de 20,97%, sendo, consideravelmente, maior que todos os outros algoritmos, nesse problema os valores das médias dos resultados e os seus respectivos melhores valores foram próximos.

O GA encontrou valores piores que os melhores, tendo um gap entre 9% e 11%. Porém, considerando a maior instância, ele conseguiu ser menor que o ACO, sendo o ACO o pior resultado para a instância com 100 demandas.

O tempo de execução do GA nesse problema também foi o menor entre todos os algoritmos, mas comparado ao tempo do GA-ACO, o tempo dos dois algoritmos foram muito próximos. Já, o ACO apresentou um tempo quase 9 vezes maior que o menor tempo.

Comparando o RPST com RTSP, com relação à recarga, os resultados do primeiro foram maiores que o segundo, isso acontece pela questão do custo que foi dito no problema do RTSP. Dado que os custos são muito diferentes, não é possível fazer uma comparação com relação ao serviço.

O RPST e o RTST são problemas em que o serviço é o mesmo, ou seja, todos os usuários tem que ser atendidos. O diferencial entres os dois é a recarga; para algumas instâncias, o menor custo foi o RPST, porém, em outras foi o RTST.

Tabela 4 – Valores das soluções para RPST

Instâncias		GA	ACO	GA-ACO
S2_A4_T1	$f_{médio}$	2678,83	2620,72	2620,72
	$f_{mín}$	2678,83	**2620,72**	**2620,72**
	gap (%)	2,22	0,00	0,00
	tempo (s)	7,06	13,17	5,91
S2_A8_T1	$f_{médio}$	2911,27	2775,68	2775,68
	$f_{mín}$	2911,27	**2775,68**	**2775,68**
	gap (%)	4,88	0,00	0,00
	tempo (s)	11,49	32,60	10,24
S2_A12_T1	$f_{médio}$	3143,71	2950,01	2950,01
	$f_{mín}$	3143,71	**2950,01**	**2950,01**
	gap (%)	6,57	0,00	0,00
	tempo (s)	15,43	51,11	14,61
S2_A16_T1	$f_{médio}$	3376,15	3085,60	3085,60
	$f_{mín}$	3376,15	**3085,60**	**3085,60**
	gap (%)	9,42	0,00	0,00
	tempo (s)	19,67	73,21	19,00
S2_A20_T1	$f_{médio}$	3706,85	3455,04	3396,93
	$f_{mín}$	3706,85	**3396,93**	**3396,93**
	gap (%)	9,12	0,00	0,00
	tempo (s)	27,29	104,30	27,58
S2_A25_T1	$f_{médio}$	3958,66	3668,82	3648,74
	$f_{mín}$	3958,66	**3590,63**	3610,00
	gap (%)	10,25	0,00	0,54
	tempo (s)	34,17	162,13	35,83
S2_A30_T1	$f_{médio}$	4229,84	3940,70	3861,81
	$f_{mín}$	4229,84	**3803,70**	**3803,70**
	gap (%)	11,20	0,00	0,00
	tempo (s)	39,62	187,53	41,79
S2_A50_T1	$f_{médio}$	5412,82	5018,56	4986,68
	$f_{mín}$	5412,82	4967,31	**4929,98**
	gap (%)	9,79	0,76	0,00
	tempo (s)	77,86	447,20	88,53
S2_A75_T1	$f_{médio}$	6613,76	6169,66	6052,03
	$f_{mín}$	6613,76	5935,81	**5858,33**
	gap (%)	12,89	1,32	0,00
	tempo (s)	112,33	643,90	128,82
S2_A100_T1	$f_{médio}$	8128,85	9411,63	8109,48
	$f_{mín}$	8128,85	9411,63	**7780,19**
	gap (%)	4,48	20,97	0,00
	tempo (s)	152,01	1364,04	204,22

Fonte: Autoria Própria

IV. Recarga Parcial, Serviço Parcial (RPSP)

Na Tabela 5, são apresentados os resultados para recarga parcial e serviço parcial. O GA-ACO conseguiu encontrar os melhores resultados para a maioria das instâncias. Apenas para as instâncias com 25 e 30 solicitações de viagem, o ACO foi o melhor, porém, para essas instâncias o gap do GA-ACO foi 2,33% e 3,90%, respectivamente, os quais, são muito próximo do melhor resultado.

Para esse problema, o ACO conseguiu encontrar boas soluções, mesmo em instâncias maiores, em que o seu gap, para instâncias com demanda entre 75 e 100 de demanda, foi 2,55% e 4,98%, respectivamente. Sendo para a instância com 100, um valor de gap maior que os gaps apresentados pelo GA-ACO. Em comparação com os outros problemas, esse foi um dos menores gap para essa instância junto com o gap do problema II. Assim, para serviços parciais, o ACO conseguiu encontrar boas soluções, enquanto, para serviços total, o gap foi acima das melhores soluções, comparado, principalmente, à instância com 100 demandas.

O GA, para todos os problemas e para todas as instâncias, apresentou um pior resultado do que os outros algoritmos. Se comparar a média dos resultados do GA com os melhores resultados apresentados por ele, os valores são muito próximos ou idênticos, não tendo uma oscilação de valor.

A respeito do tempo de execução, para todos os problemas, o GA apresentou o menor tempo e o GA-ACO conseguiu apresentar um tempo muito próximo do melhor.

Tabela 5 – Valores das soluções para RPSP

Instâncias		GA	ACO	GA-ACO
S2_A4_T1	$f_{médio}$	2640,45	2601,71	2601,71
	$f_{mín}$	2640,45	**2601,71**	**2601,71**
	gap (%)	1,49	0,00	0,00
	tempo (s)	7,42	15,12	6,62
S2_A8_T1	$f_{médio}$	2834,51	2737,66	2737,66
	$f_{mín}$	2834,51	**2737,66**	**2737,66**
	gap (%)	3,54	0,00	0,00
	tempo (s)	11,12	34,98	10,76
S2_A12_T1	$f_{médio}$	3028,57	2873,61	2892,98
	$f_{mín}$	3028,57	**2873,61**	**2873,61**
	gap (%)	5,39	0,00	0,00
	tempo (s)	14,85	52,41	18,42
S2_A16_T1	$f_{médio}$	3222,63	3009,56	3126,49
	$f_{mín}$	3222,63	**3009,56**	**3009,56**
	gap (%)	7,08	0,00	0,00
	tempo (s)	18,76	79,19	22,62
S2_A20_T1	$f_{médio}$	3358,94	3145,87	3331,30
	$f_{mín}$	3358,94	**3145,87**	**3145,87**
	gap (%)	6,77	0,00	0,00
	tempo (s)	21,92	103,72	25,69
S2_A25_T1	$f_{médio}$	3582,06	3330,07	3572,37
	$f_{mín}$	3553,00	**3320,56**	3398,04
	gap (%)	7,00	0,00	2,33
	tempo (s)	25,97	148,08	32,83
S2_A30_T1	$f_{médio}$	3903,79	3583,48	3671,35
	$f_{mín}$	3766,79	**3514,98**	3651,98
	gap (%)	7,16	0,00	3,90
	tempo (s)	34,16	185,28	41,33
S2_A50_T1	$f_{médio}$	4748,55	4565,94	4556,26
	$f_{mín}$	4719,49	**4390,20**	**4390,20**
	gap (%)	7,50	0,00	0,00
	tempo (s)	62,24	443,88	74,33
S2_A75_T1	$f_{médio}$	5787,36	5635,04	5499,63
	$f_{mín}$	5709,88	5499,63	**5362,63**
	gap (%)	6,48	2,55	0,00
	tempo (s)	86,77	688,75	115,20
S2_A100_T1	$f_{médio}$	6847,32	8547,96	6606,60
	$f_{mín}$	6740,78	6857,00	**6531,94**
	gap (%)	3,20	4,98	0,00
	tempo (s)	128,42	1290,06	174,59

Fonte: Autoria Própria

8 CONCLUSÃO

Neste trabalho de conclusão de curso, foi abordado um problema de otimização combinatória, chamado LRP-SAEVs. Para esse problema, foram apresentadas quatro variantes. Para todas as variantes foi utilizado o mesmo conjunto de instâncias, definindo as estações candidatas com sua localização e os pontos de demanda. Para cada demanda, foi apresentado o tempo que o carro tem para chegar na origem, a localização da origem e do destino. O objetivo do problema é dimensionar frotas e localização de estações, de forma a minimizar o custo de implantação. Para cada problema, as restrições para se atingir esse objetivo são diferentes.

Para tratar o problema, foram utilizadas três abordagens diferentes: duas metaheurísticas e uma hibridização. A primeira metaheurística apresentada foi o GA, que conseguiu encontrar soluções em um tempo hábil; a segunda metaheurística foi o ACO, a qual apresentou boas soluções, porém, o tempo não foi hábil. E a hibridização foi realizada utilizando o GA e o ACO, que encontrou boas soluções em um tempo mais razoável.

Os quatro problemas apresentados se dividiram da seguinte forma:

I. Recarga Total, Serviço Total (RTST):

Onde o nível de bateria do veículo ao passar por um ponto de recarga deve ser totalmente carregado, e todas as solicitações têm que ser atendidas.

Com relação às instâncias menores, o ACO e o GA-ACO encontraram a melhor solução. Para as instâncias com demanda maior ou igual a 50, apenas o GA-ACO apresentou as melhores soluções. O ACO na instância com 100 demandas mostrou um valor muito maior do que o esperado.

II. Recarga Total, Serviço Parcial (RTSP):

Onde, ao passar por um ponto de recarga, a bateria tem que ser completamente carregada e algumas soluções podem ser rejeitadas. Para as instâncias menores, novamente, o ACO e o GA-ACO encontraram as mesmas soluções, sendo elas as melhores soluções; enquanto, para instâncias com demanda maior ou igual a 50, o GA-ACO encontrou os melhores resultados entre os algoritmos apresentados.

III. Recarga Parcial, Serviço Total (RPST):

Onde considera-se que, ao chegar no ponto de recarga, o veículo pode carregar apenas parte ou toda a bateria, e que todos os clientes têm que ser atendidos. O GA-

ACO foi o melhor em todos as instâncias, mesmo que em instâncias menores o ACO ficou igual aos melhores resultados.

IV. Recarga Parcial, Serviço Parcial (RPSP):

Onde considera-se que, ao chegar para recarregar, o veículo pode decidir quanto será recarregado e que pode haver demandas que serão recusadas. O ACO foi o melhor em alguns casos, em instâncias intermediárias, e igual ao GA-ACO em instâncias pequenas. Porém, o GA-ACO foi o melhor, pois, mesmo nas instâncias em que ele não foi o melhor, ficou muito próximo do melhor. E nas instâncias maiores os melhores resultados foram do GA-ACO.

Com relação às diferenças quase imperceptíveis, em relação a recarga total e a recarga parcial, talvez considerar novas instâncias, distâncias entre pontos maiores ou menores, são pontos interessantes.

Com relação ao tempo, o GA apresentou o menor tempo entre todos os algoritmos, porém, o GA-ACO, mesmo um pouco maior, apresentou um tempo muito próximo do GA. Para o ACO, o tempo chegou a ser 4 vezes maior que o tempo dos outros dois algoritmos.

De forma geral, nos quatro problemas, o GA-ACO foi o mais eficiente e encontrou os melhores resultados em um tempo hábil, concluíndo assim, que a hibridização se provou melhor que as metaheurísticas puras.

REFERÊNCIAS

ANCIOTO, E. M. **PLATAFORMA WEB HYBROO: AMBIENTE EXPERIMENTAL VOLTADO À HIBRIDIZAÇÃO DE ALGORITMOS DE OTIMIZAÇÃO**. Programa de Mestrado em Engenharia de Produção e Sistemas. Pontifícia Universidade Católica de Goiás. Goiânia, GO, 2019.

BRABAZON, A.; O'NEILL, M.; MCGARRAGHY, S. **Natural Computing Algorithms**. Berlin, Heidelberg: Springer Berlin Heidelberg, 2015.

CARVALHO, M. B.; YAMAKAMI, A. **Meta-heurística Híbrida de Sistema de Colônia de Formigas e Algoritmo Genético para o Problema do Caixeiro Viajante.** Departamento de Telemática, Faculdade de Engenharia Elétrica e de Computação, UNICAMP, 13083-852 Campinas, SP, Brasil

CHEN, T. D.; KOCKELMAN, K. M.; HANNA, J. P. Operations of a shared, autonomous, electric vehicle fleet: Implications of vehicle & charging infrastructure decisions. **Transportation Research Part A: Policy and Practice**, v. 94, p. 243–254, dez. 2016.

CHEN, X. et al. **Evolutionary Algorithm of Port Based Location Routing Problem**. 2009 WRI Global Congress on Intelligent Systems. **Anais**... Em: 2009 WRI GLOBAL CONGRESS ON INTELLIGENT SYSTEMS. Xiamen, China: IEEE, 2009. Disponível em: <http://ieeexplore.ieee.org/document/5209131/>. Acesso em: 4 jun. 2023

COLORNI, A.; DORIGO, M.; MANIEZZO, V. **Distributed Optimization by Ant Colonies**. Dipartimento di Elettronica, Politecnico di Milano. Piazza Leonardo da Vinci 32, 20133. Milano, Italy.

CORDEAU, J.-F.; LAPORTE, G. The dial-a-ride problem: models and algorithms. **Annals of Operations Research**, v. 153, n. 1, p. 29–46, 6 jun. 2007.

DORIGO, M.; BIRATTARI, M.; STUTZLE, T. Ant colony optimization. **IEEE Computational Intelligence Magazine**, v. 1, n. 4, p. 28–39, nov. 2006.

DORIGO, M.; BLUM, C. Ant colony optimization theory: A survey. **Theoretical Computer Science**, v. 344, n. 2–3, p. 243–278, nov. 2005.

DORIGO, M.; STUTZLE, T. Ant Colony Optimization: Overview and Recent Advances. Em: **Handbook of Metaheuristics**. International Series in Operations Research & Management Science. [s.l.] Springer International Publishing, 2019. v. 272, c. 10,p. 311–351.

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, ago. 2018.

GUANGDONG, H.; PING, L.; QUN, W. **A Hybrid Metaheuristic ACO-GA with an Application in Sports Competition Scheduling**. Eighth ACIS International Conference on Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing (SNPD 2007). Anais... Em: EIGHTH ACIS INTERNATIONAL CONFERENCE ON SOFTWARE ENGINEERING, ARTIFICIAL

INTELLIGENCE, NETWORKING, AND PARALLEL/DISTRIBUTED COMPUTING (SNPD 2007). Qingdao, China: IEEE, jul. 2007. Disponível em: <http://ieeexplore.ieee.org/document/4287925/>. Acesso em: 12 jun. 2023

GIL, A. C. **Como elaborar projetos de pesquisa.** 4. ed. São Paulo: Atlas, 2002.

HOLLAND, J. H. **Adaptation in natural and artificial systems: An introductory analysis with applications to biology, control and artificial intelligence**. [s.l.] MI: University of Michigan Press, 1975.

HUSSAIN, K. et al. Metaheuristic research: a comprehensive survey. **Artificial Intelligence Review**, v. 52, n. 4, p. 2191–2233, dez. 2019.

LEE, U.; KANG, N.; LEE, I. Shared autonomous electric vehicle design and operations under uncertainties: a reliability-based design optimization approach. **Structural and Multidisciplinary Optimization**, v. 61, n. 4, p. 1529–1545, abr. 2020.

LOPES, R. B.; FERREIRA, C.; SANTOS, B. S. A simple and effective evolutionary algorithm for the capacitated location–routing problem. **Computers & Operations Research**, v. 70, p. 155–162, jun. 2016.

MA, B. et al. Time-dependent Vehicle Routing Problem with Departure Time and Speed Optimization for Shared Autonomous Electric Vehicle Service. **Applied Mathematical Modelling**, v. 113, p. 333–357, set. 2022.

MA, B.; HU, D.; WU, X. The Location Routing Problem of the Car-Sharing System with Autonomous Electric Vehicles. **KSCE Journal of Civil Engineering**, v. 25, n. 8, p. 3107–3120, ago. 2021.

MARTINEZ, L. M.; CORREIA, G. H. A.; VIEGAS, J. M. **An agent-based simulation model to assess the impacts of introducing a shared-taxi system**: an application to Lisbon (Portugal). 28 jul. 2014.

NEUMANN, F.; WITT, C. (EDS.). Combinatorial Optimization and Computational Complexity. Em: **Bioinspired Computation in Combinatorial Optimization**. Natural Computing Series. [s.l.] Springer International Publishing, 2010.

PARRAGH, S. N.; PINHO DE SOUSA, J.; ALMADA-LOBO, B. The Dial-a-Ride Problem with Split Requests and Profits. **Transportation Science**, v. 49, n. 2, p. 311–334, jun. 2014.

RAHMAN, M. A. et al. Nature-Inspired Metaheuristic Techniques for Combinatorial Optimization Problems: Overview and Recent Advances. **Mathematics**, v. 9, n. 20, p. 2633, 19 out. 2021.

RAIDL, G. R.; PUCHINGER, J. Metaheuristic Hybrids. Em: **Handbook of Metaheuristics**. International Series in Operations Research & Management Science. Cham: Springer International Publishing, 2019. v. 272.

SANTOS, G. G. D.; BIROLINI, S.; CORREIA, G. H. D. A. A space–time-energy flow-based integer programming model to design and operate a regional shared automated

electric vehicle (SAEV) system and corresponding charging network. **Transportation Research Part C: Emerging Technologies**, v. 147, p. 103997, fev. 2023.

SHAHEEN, Susan A.; COHEN, Adam P. Carsharing and Personal Vehicle Services: Worldwide Market Developments and Emerging Trends, **International Journal of Sustainable Transportation**, 7:1, 5-3; (2012).

SCHIFFER, M.; WALTHER, G. The electric location routing problem with time windows and partial recharging. **European Journal of Operational Research**, v. 260, n. 3, p. 995–1013, ago. 2017.

VALDEZ, F.; MELIN, P.; CASTILLO, O. A survey on nature-inspired optimization algorithms with fuzzy logic for dynamic parameter adaptation. **Expert Systems with Applications**, v. 41, n. 14, p. 6459–6466, out. 2014.

WAZLAWICK, Raul Sidnei. **Metodologia de Pesquisa para Ciência da Computação**. 2ª ed. - Rio de Janeiro: Elsevier, 2014. cap.4, p. 21 – 26.

WEGSCHEIDER, Augustin K.; HAGENMAIER, Markus; BERT, Julien; COLLIE, Brian; PALME, Thomas; ROSE, Justin. **Shared, Autonomous, and Electric: An Update on the Reimagined Car**. BCG, 26 de julho de 2022. Disponível em: <https://www.bcg.com/publications/2022/update-on-shared-autonomous-electric-vehicles-market>. Acesso em: 7 de setembro de 2022.

WHITLEY, D. Next Generation Genetic Algorithms: A User's Guide and Tutorial. Em: **Handbook of Metaheuristics**. International Series in Operations Research & Management Science. [s.l.] Springer International Publishing, 2019. v. caption 8p. 245–274.

XIDIAS, E. K.; PANAGIOTOPOULOS, I. E.; ZACHARIA, P. TH. An intelligent management system for relocating semi-autonomous shared vehicles. **Transportation Planning and Technology**, v. 46, n. 1, p. 93–118, 2 jan. 2023.

YANG, X.-S. Genetic Algorithms. Em: **Nature-Inspired Optimiztion Algorithms**. [s.l: s.n.]. p. 91–100.

YU, X.; ZHOU, Y.; LIU, X.-F. A novel hybrid genetic algorithm for the location routing problem with tight capacity constraints. **Applied Soft Computing**, v. 85, p. 105760, dez. 2019.

ZHAO, H. et al. Study on an Adaptive Co-Evolutionary ACO Algorithm for Complex Optimization Problems. **Symmetry**, v. 10, n. 4, p. 104, 11 abr. 2018.