ISSN: 1983 7402 ITA, 28 a 29 SET 2021

Abordagens Exata e Heurística na Otimização do Problema do Caixeiro Viajante com Drone

André Rossi Kuroswiski[1], Humberto Baldessarini Pires[1], Angelo Passaro[2], Lamartine Nogueira Frutuoso Guimarães[2], Edson Luiz França Senne[3]
[1]Instituto Tecnológico de Aeronáutica (ITA), São José dos Campos/SP – Brasil
[2]Instituto de Estudos Avançados (IEAv), São José dos Campos/SP – Brasil
[3]Universidade Federal de São Paulo (UNIFESP), São José dos Campos/SP – Brasil

***Resumo* – O aumento na utilização de drones para cumprir as mais variadas tarefas tem motivado um crescimento exponencial de pesquisas que buscam otimizar o emprego desses meios, beneficiando tanto aplicações militares quanto civis, dentre as quais a logística de entregas. Nesse sentido, o uso conjugado de caminhões e drones vêm sendo explorado com bastante interesse pela Pesquisa Operacional. Este trabalho apresenta formulações matemáticas em Programação Linear Inteira Mista e propõe um Algoritmo Genético híbrido (HGenFS) para a otimização de uma variação do Problema do Caixeiro Viajante (*Traveling Salesman Problem*, TSP) chamada de *Flying Sidekick* TSP (FSTSP), em que caminhão e drone atuam cooperativamente. Os resultados obtidos confirmaram que a formulação adotada para a solução exata é adequada para resolver problemas de até dez clientes e o HGenFS demonstrou ser capaz de encontrar soluções ótimas para o FSTSP, em poucos segundos, ao incorporar heurísticas específicas e uma fase de busca local.**

Palavras-Chave* – *Drone, Modelagem matemática, Meta-heurísticas.

I. INTRODUÇÃO

Historicamente, o emprego de drones para propósitos militares data da primeira metade do século passado [1]. Contudo, com o avanço tecnológico desses dispositivos nas duas últimas décadas, a aplicação militar de drones tornou-se mais ampla e intensa, e a sua utilização tem sido estudada a partir de várias perspectivas, inclusive aquelas relacionadas à área civil, como as operações logísticas [2].

Nesse ínterim, o serviço de entrega de bens expandiu-se consideravelmente e, devido à alta competição nesse nicho de mercado, os consumidores optam pelas empresas em função da rapidez, flexibilidade e custos de frete oferecidos. Isso motivou as companhias do ramo a implementarem inovações tecnológicas nos serviços de entrega, em especial na etapa chamada *last mile delivery* [3], termo que, em uma tradução literal, significa a "entrega de última milha". Tal expressão é utilizada para definir o transporte de produtos dos centros de distribuição até o destino, como uma casa ou uma empresa [4].

A opção pela utilização de drones justifica-se pelo fato de esses dispositivos não serem restritos à malha rodoviária nem limitados pelo relevo, ao contrário dos caminhões, veículos tradicionais utilizados para o fim [5], [6]. Contudo, as desvantagens dos drones referem-se à restrita capacidade de transporte de carga e ao tempo de autonomia da aeronave, limitado pela bateria [7].

André Rossi Kuroswiski, andrekuros@gmail.com; Humberto Baldessarini Pires, humbertohbp@ita.br; Angelo Passaro, angelopassaro@gmail.com; Lamartine Nogueira Frutuoso Guimarães, lamar.guima@gmail.com, Edson Luiz França Senne, edson.senne@unifesp.br.

Em função das vantagens inerentes a esses dois tipos de veículos, o emprego de um sistema caminhão-drone compreende uma combinação que potencializa as melhores características de caminhões e drones. A otimização desse processo pode tornar o serviço de entrega mais eficiente e, possivelmente, mais barato. Consequentemente, isso originou diferentes novos problemas de logística, despertando um interesse crescente da área de Pesquisa Operacional [8].

Esse tipo de problema possui ampla aplicação na área militar em que, por exemplo, os clientes atendidos podem ser tropas ou frações de tropa em um ambiente operacional, e veículos de apoio ou bases móveis executam a função do caminhão. Prova disso é que, na segunda metade da última década, houve o surgimento de programas militares nos EUA [9] e no Reino Unido [10], por exemplo, visando ao desenvolvimento de soluções de otimização da logística de distribuição (com o emprego de drones e veículos de apoio) de equipamentos médicos, alimentos, combustível e peças de reposição, dentre outros itens, para contingentes desdobrados [10]. O objetivo dessas iniciativas é reduzir os riscos associados às operações de (re)abastecimento complexas, em que os operadores podem ser expostos a perigos significativos em ambientes hostis [10].

Nesse sentido, este trabalho possui como proposta a abordagem de uma variação do Problema do Caixeiro Viajante (PCV), introduzida em [11], em que um caminhão e um drone operam de forma colaborativa a fim de realizar as entregas. Os serviços são realizados por meio da exploração das melhores características de cada veículo: a alta velocidade média do drone e a grande capacidade de carga do caminhão. Para a resolução do problema, é apresentada uma formulação matemática modelada por meio de uma Programação Linear Inteira Mista (PIM), além de ser proposto um método heurístico em que Algoritmos Genéticos híbridos foram utilizados para o alcance das soluções viáveis.

II. REVISÃO DA LITERATURA

Nos últimos anos, mais de trezentos artigos foram publicados sobre problemas de otimização com o emprego de caminhões e drones, sendo possível classificá-los em diferentes categorias de sistemas caminhão-drone [12]: caminhões como meio de apoio às operações dos drones; drones apoiando os serviços de entrega efetuados por caminhões; drones e caminhões em tarefas independentes; e drones e caminhões operando de forma sincronizada.

No trabalho, o escopo da revisão da literatura estará restrito às pesquisas julgadas mais promissoras relacionadas à categoria de sistema caminhão-drone sincronizado. A utilização de drones na otimização de um processo de entrega

foi primeiramente proposta em [11], trabalho que explorou uma variação do PCV (em inglês, *Traveling Salesman Problem*, ou TSP) definida como FSTSP (*Flying Sidekick* TSP), em que um drone e um caminhão compõem um sistema que atua de maneira cooperativa.

No FSTSP, cada cliente é atendido somente uma vez, seja pelo drone ou pelo caminhão, que é utilizado para realizar as entregas que ultrapassam a capacidade de carga ou a autonomia do drone. Além disso, cada nó cliente somente pode ser visitado uma vez. O drone pode ser lançado do depósito ou do caminhão. Cada voo realizado é chamado de surtida e compreende a decolagem de um nó cliente do caminhão para a entrega da mercadoria a apenas um cliente, bem como o reencontro em um diferente nó cliente do caminhão. Lançamentos e pousos devem ser realizados com o caminhão parado, não sendo permitidos pousos intermediários no intuito de economizar a bateria. Caminhão e drone devem aguardar um ao outro em um nó cliente do caminhão. A função-objetivo é minimizar o tempo de conclusão das rotas [11].

Abordagens como o FSTSP constituem um problema NP-difícil do ponto de vista da complexidade computacional [13]. Em função disso, alguns métodos heurísticos têm sido explorados para a resolução desse tipo de problema [14]. A formulação de PIM apresentada em [11] não foi capaz de resolver, de forma otimizada e em trinta minutos, as instâncias relativas a dez clientes, sendo propostos três métodos heurísticos baseados em algoritmos KNN (*K-Nearest Neighbor*) e de varredura.

Outro problema que compartilha as principais características do FSTSP é o chamado TSP com Drone (TSP-D). Em [15] os autores modelaram o problema com Programação Linear Inteira (PI) e desenvolveram heurísticas baseados em Busca Local e Programação Dinâmica, encontrando soluções ótimas em até sessenta minutos para instâncias de até quinze clientes.

Na Fig. 1, é possível visualizar possíveis soluções para o clássico TSP (a), o FSTSP (b) e o TSP-D (c).

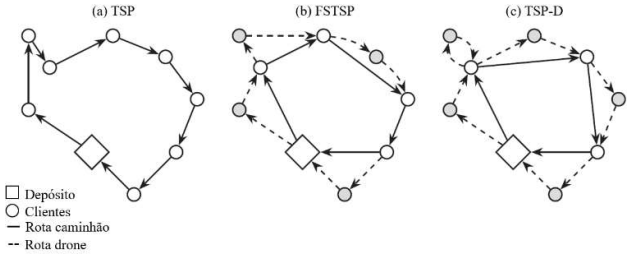

Fig. 1. Comparação entre possíveis soluções do TSP (a), FSTSP (b) e TSP-D (c). Fonte: [8].

As principais diferenças do TSP-D em relação ao FSTSP são: os clientes podem ser visitados mais de uma vez pelo caminhão, caso isso seja conveniente para o lançamento e a recuperação do drone; os locais de lançamento e reencontro de drone e caminhão podem coincidir; a autonomia do drone é considerada ilimitada; e os tempos de lançamento e reencontro podem ser considerados insignificantes.

Pela definição apresentada em [16], apesar do autor chamar a abordagem de TSP-D, são utilizas premissas do FSTSP (não permite a revisita de nós clientes nem o lançamento e o reencontro em um mesmo nó) e emprega um algoritmo de Busca Local e uma meta-heurística GRASP (*Greedy Randomized Adaptive Search Procedure*). Em um trabalho posterior, os mesmos autores utilizaram um Algoritmo Genético (AG) híbrido aperfeiçoado por procedimento de Busca Local para minimizar o custo operacional total e o tempo das entregas. O algoritmo proveu soluções ótimas para instâncias de até cem clientes [17].

Neste trabalho, o problema da otimização do serviço de entrega por um sistema caminhão-drone será abordado conforme as definições estabelecidas para o FSTSP em [11], mas com a formulação matemática do problema em PIM baseada em [18], que permitiu a implementação de um modelo exato com apenas treze restrições (diferentemente do modelo proposto em [11], que continha vinte e oito restrições).

III. DEFINIÇÃO DO PROBLEMA E FORMULAÇÃO MATEMÁTICA

No problema, o nó inicial, definido por 0, e o nó final, c+1, serão compostos por um mesmo depósito, e pretende-se que o conjunto de dez clientes C = {1,..., c}, com c = 10 sejam atendidos pelo caminhão ou pelo drone. As posições dos clientes são previamente conhecidas. Todos os dez clientes do conjunto C podem ser atendidos pelo caminhão, porém somente os clientes do subconjunto $C' \subseteq C$ podem ser atendidos pelo drone.

O problema foi construído no dígrafo $G = (N, A)$, onde: $N = \{0, 1,..., c+1\}$ contém todos os nós possíveis; $N_0 = \{0, 1,..., c\}$ representa os possíveis nós iniciais de um arco; e $N_+ = \{1,..., c+1\}$, os possíveis nós de término de um arco. Assim, seja A o conjunto de todos os arcos (i, j), com $i \in N_0$ e $j \in N_+$, com $i \neq j$. Cada arco (i, j) está associado a dois tempos de viagem (em minutos) não negativos, τ_{ij}^C e τ_{ij}^D, que representam o tempo de viagem daquele arco percorrido pelo caminhão e pelo drone, respectivamente.

As matrizes de tempo de viagem do drone e do caminhão são normalmente diferentes, e o tempo de viagem entre os nós 0 e c+1 é definido como 0, porque representam o mesmo ponto físico (depósito). Os tempos gastos nos atendimentos dos clientes, tanto para o drone quanto para o caminhão, estão incluídos nos respectivos tempos de viagem. Foi considerado um tempo de preparação do drone para o seu lançamento, dado por σ^L, e de recuperação (pouso no caminhão), dado por σ^R. Ambas as ações são realizadas pelo motorista do caminhão, que permaneceu estacionado em um nó cliente durante esses procedimentos. Contudo, o drone também poderia ser lançado do depósito. Nessa situação, σ^L é igual a zero. O parâmetro E representa a autonomia, em minutos, do drone, restringindo a sua utilização durante as surtidas. O tempo de recuperação σ^R contribui para o cálculo total da autonomia, enquanto σ^L não a afeta, uma vez que o drone permanece no caminhão até a saída. Assim, somente σ^L contribuirá para o tempo total de entrega.

Uma surtida é formalmente definida por $\langle i, j, k \rangle$, com $i \neq j \neq k$, onde: $i \in N_0$ é o nó de lançamento; $j \in C'$ representa o cliente a ser atendido; e $k \in N_+$, o nó de reencontro entre drone e caminhão. Seja F o conjunto de todas as surtidas que podem ser realizadas em conformidade com a autonomia E (para as quais é válida a condição $\tau_{ij}^D + \tau_{kj}^D + \sigma^R \leq E$). O drone não poderá ser lançado do depósito antes que o caminhão inicie a sua rota e, após cada surtida, a bateria é substituída ou recarregada para uma nova surtida. Uma surtida $\langle i, j, k \rangle$ ainda poderá se tornar inviável caso o caminhão exceda a autonomia E do drone ao percorrer o arco (i, k).

Finalmente, o caminhão pode atender clientes durante uma surtida do drone. Para o reencontro, faz-se necessária uma

sincronização, em que o veículo que chegar primeiro a um ponto de reencontro deve esperar pelo outro.

Detalhados os conjuntos e parâmetros do problema, é necessário definir as variáveis de decisão. Seja $x_{ij} \in \{0, 1\}$ igual a 1 se o nó $j \in N_+$ é visitado após o nó $i \in N_0$, sendo $i \neq j$; 0, caso contrário. As surtidas do drone são representadas pela variável tri indexada $y_{ijk} \in \{0, 1\}$, com $\langle i, j, k\rangle \in F$, sendo igual a 1 se a surtida é realizada; 0, caso contrário. A variável $z_i \in \{0,1\}$ é igual a 1 se o drone encontra-se no caminhão; 0, caso contrário. A variável não-negativa w_i representa o tempo em que o caminhão espera o drone no nó $i \in N$, enquanto a variável não-negativa t_i, com $i \in N$, é utilizada para representar o tempo de sincronização entre caminhão e drone, sendo computada nela eventuais esperas do drone pelo caminhão.

A função-objetivo (1) visa minimizar o tempo de chegada de caminhão e drone ao depósito final, sendo possível que o drone retorne ao depósito antes do caminhão. Quando isso ocorrer, a diferença de tempo será considerada pela variável de tempo de espera w_{c+1}. O tempo de conclusão do serviço de entrega pode ser decomposto nos seguintes componentes: o tempo total de rota gasto pelo caminhão; o tempo total necessário para os lançamentos e recuperações do drone durante as surtidas realizadas; e o tempo total de espera do caminhão pelo drone.

$$\min \sum_{(i,j)\in A} \tau_{ij}^{C} x_{ij} + \sigma^{R} \sum_{\langle 0,j,k\rangle \in F} y_{0jk} + (\sigma^{L} + \sigma^{R}) \sum_{\langle i,j,k\rangle \in F, i\neq 0} y_{ijk} + \sum_{i\in N_+} w_i \quad (1)$$

As restrições (2) e (3) obrigam um dos dois veículos a atenderem cada cliente somente uma vez. Em (4), é imposta a necessidade de o caminhão iniciar e finalizar o seu percurso no depósito e, em (5), é definida a necessidade de conservação do fluxo do caminhão, pela combinação de (2) e (3).

$$\sum_{(i,j)\in A} x_{ij} + \sum_{\langle i,j,k\rangle \in F} y_{ijk} = 1, \text{para } j \in C \quad (2)$$

$$\sum_{(j,i)\in A} x_{ji} + \sum_{\langle i,j,k\rangle \in F} y_{ijk} = 1, \text{para } j \in C \quad (3)$$

$$\sum_{j\in N_+} x_{0j} = \sum_{i\in N_0} x_{i,c+1} = 1 \quad (4)$$

$$\sum_{(i,j)\in A} x_{ij} = \sum_{(j,i)\in A} x_{ji} \quad (5)$$

Foi estabelecida uma série de restrições, de maneira a garantir as relações de tempo. Nestas restrições, a constante M representa um valor bem grande. (6) garante que, se o arco (i,j) for percorrido pelo caminhão, então o tempo no nó j será maior ou igual ao tempo gasto até o nó i, acrescido do tempo gasto no próprio arco. A restrição (7) define que, se houver uma surtida $\langle i,k,j\rangle$, então o tempo t_j deve ser maior ou igual ao tempo gasto pelo drone na surtida. As restrições (6) e (7) também previnem que o nó j seja visitado pelo caminhão antes do nó i. (8) estabelece o tempo de espera do caminhão, caso ele chegue a um nó de reencontro j antes do drone. Caso o drone chegue ao nó j antes do caminhão, a espera será realizada em voo pairado, e esse tempo será absorvido por t_j na restrição (6), com o tempo incluído na função-objetivo.

$$t_j \geq t_i + \tau_{ij}^{C} - M(1 - x_{ij}), \text{para } (i,j) \in A \quad (6)$$

$$t_j \geq t_i + \tau_{ik}^{D} + \tau_{kj}^{D} - M(1 - \sum_{\langle i,k,j\rangle \in F} y_{ikj}), (i,j) \in A \quad (7)$$

$$w_j \geq t_j - t_i - \tau_{ij}^{C} - M(1 - x_{ij}), \text{para } (i,j) \in A \quad (8)$$

Em (9), caso uma surtida $\langle i,j,k\rangle$ seja realizada, o tempo gasto desde o lançamento do drone até a sua recuperação deve respeitar a autonomia E do aparelho.

$$t_k - t_i + \sigma^{R} - M(1 - \sum_{\langle i,j,k\rangle \in F} y_{ijk}) \leq E, \text{para } (i,k) \in A \quad (9)$$

As restrições (10) a (12) compreendem relações entre a variável z_i, que define se um drone está ou não embarcado no caminhão, e x_{ij} e y_{ijk}. (10) mostra que o drone pode estar no caminhão no nó i. (11) atesta que uma surtida pode ter início no nó i caso a variável z_i seja igual a 1, enquanto (12) regula a variável z_i, definindo a presença do drone no caminhão ao longo do percurso desse veículo. Portanto, dado $z_i = 1$ por causa de (11), (12) impõe $z_i = 0$, enquanto que, caso a surtida iniciada em i retorne em j, z_j poderá assumir o valor 1, o que significa que uma nova surtida poderia começar em j. Voltando ao caso geral em que a surtida não é finalizada em j, as variáveis z ao longo do caminho (após o lançamento da surtida em i e antes do reencontro em l) permanecem com o valor 0, impondo que nenhuma outra surtida possa começar até que o caminhão alcance o nó de reencontro l. Por fim, (13) define o tempo de início do percurso, no depósito, como 0.

$$z_i \leq \sum_{(j,i)\in A} x_{ji}, \text{para } i \in N_+ \quad (10)$$

$$\sum_{\langle i,j,k\rangle \in F} y_{ijk} \leq z_i, \text{para } i \in N_0 \quad (11)$$

$$z_j \leq z_i - x_{ij} + \sum_{\langle l,j,k\rangle \in F} y_{ljk} - \sum_{\langle i,k,l\rangle \in F} y_{ikl} + 1, \quad (i,j) \in A \quad (11)$$

$$t_0 = 0 \quad (13)$$

IV. META-HEURÍSTICA HÍBRIDA

A utilização de algoritmos de otimização baseados em meta-heurísticas para problemas combinatórios como o TSP e suas variações é comum. No entanto, pode ser observado em algoritmos no estado da arte a necessidade de adaptações nas meta-heurísticas tradicionais, incorporando elementos puramente heurísticos específicos para o problema em questão. Estas soluções, por vezes identificadas como híbridas [19], tendem a ser menos susceptíveis a mínimos locais, apresentando um custo computacional maior que de soluções puramente heurísticas, mas que não chega a ser tão elevado quanto o de soluções exatas [20]. Nesse contexto, o estudo de soluções híbridas de meta-heurísticas adaptadas para problemas combinatórios como o FSTSP, justifica-se como uma busca por alternativas intermediárias entre a solução exata e heurística, tanto em relação ao desempenho computacional, quanto à otimalidade das soluções que podem ser encontradas.

Assim, este trabalho apresenta uma proposta de solução para o FSTSP baseada em um Algoritmo Genético, com passos personalizados para o problema e que foi chamada de *Hybrid Genetic to Flying Sidekick* (HGenFS). A proposta tem similaridade com o trabalho apresentado em [17], no entanto, foram adotadas soluções diferentes para as fases de busca local

e reconstrução, além de uma nova modelagem para o cromossomo.

A. *Definição do Cromossomo*

Neste trabalho, o cromossomo proposto para representar o indivíduo básico do FSTSP foi definido para incluir tanto a sequência de entregas (equivalente a um TSP regular), quanto a definição do meio a ser utilizado. Esse modelo de cromossomo difere do proposto em [17] por considerar o meio que faz a entrega, drone ou caminhão, como parte do indivíduo. Em [17] os autores definem o indivíduo como a sequência completa de entregas (sem especificação do veículo que a realizou), sendo esse fator recalculado por uma função específica cada vez que um novo indivíduo é criado. O modelo proposto busca definir um indivíduo mais específico para o FSTSP, reduzindo o custo computacional para processar cada um deles, à medida que os meios escolhidos já fazem parte do indivíduo e não precisam ser selecionados a cada modificação. Desse modo, apenas é demandada uma análise da viabilidade para os pontos de decolagem e pouso do drone para atender os consumidores já especificados, o que é feito durante o processo de criação de novas populações.

Cada indivíduo foi definido como dois vetores, um com os números de todos os clientes, ordenados na sequência de atendimento, e outro, de mesmo tamanho, com valores binários que indicam se a entrega para o cliente de mesmo índice é atendida por drone ou caminhão. Uma representação desse modelo está apresentada na Fig. 2, na qual o vetor superior representa a sequência de entregas com 0 e 11 representando o depósito, e os demais números representando cada cliente. O vetor inferior representa a definição de qual meio seria utilizado para cada cliente, com 1 para entregas por drones e 0 para entrega por caminhão. Adicionalmente, para cada entrega por drone é definido um par indicando o índice relativo à decolagem e pouso.

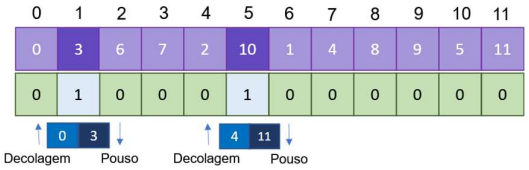

Fig. 2. Exemplo de Cromossomo que define o indivíduo.

B. *Crossover*

Para a criação de novos indivíduos a partir da população corrente, os pais são selecionados e mesclados de alguma maneira a fim de gerar descendentes possivelmente melhores. Esse processo é dividido entre a seleção e a mesclagem em si, que é chamada de *crossover* [17]. Para a seleção, foi utilizado um método conhecido como roleta com aceitação estocástica, que, conforme discutido em [21], apresenta resultados similares aos da roleta tradicional, mas com um custo computacional mais baixo, O(1) contra O(log n).

Foram definidas duas propostas de *crossover*. A primeira consiste na troca entre os diferentes vetores dentre dois parentes, ou seja, cada descendente é gerado adotando a sequência de entregas de um parente e a definição de meios do outro. Essa proposta, bastante simples, foi chamada de SWAP e avaliada pela facilidade de implementação e o baixo custo computacional, dado o modelo de cromossomo.

A segunda proposta de *crossover* consistiu em uma solução próxima ao método consagrado na literatura para o TSP, conhecido como *OX (Ordered Crossover)* [22]. No entanto, pela própria característica do cromossomo, a solução acaba também incorporando as informações das entregas por drones, tornando-se uma solução específica que foi chamada de DX2. Neste modelo, são definidos dois pontos de corte representando índices dos dois vetores do cromossomo, delimitando uma região central. Os descendentes são criados complementando-se os extremos de um dos pais selecionados, sequencialmente com os clientes faltantes, conforme ordem do outro. Desse modo, espera-se que as sequências de entregas, incluindo a escolha do meio (drone ou caminhão), seja parcialmente mantida, possibilitando o aproveitamento das características boas dos dois pais. Vale ressaltar que, após a criação dos descendentes, é necessário recriar a rota do drone, definindo pontos de decolagem e pouso, já que pontos anteriormente definidos podem não estar mais disponíveis.

C. *Mutação*

Como forma de diversificação da população, foram adotados dois métodos de mutação, com probabilidade igual de ocorrência de um ou outro quando uma mutação é realizada. O primeiro método consistiu na troca da posição de dois clientes, independente do meio utilizado para entrega. Assim, a sequência de entregas é modificada, podendo modificar também para qual cliente uma entrega definida como drone é realizada, pois o vetor dos meios selecionados não é modificado.

O segundo método consiste na alternância do meio de entrega, ou seja, se a entrega de um cliente, selecionado aleatoriamente, era realizada por drone passa para caminhão e vice-versa. Na prática, essa mutação corresponde a uma troca do valor do vetor de meios de 1 para 0, ou o contrário. Nesse caso, também é necessário reconstruir a rota do drone, caso a troca seja para adicionar uma nova entrega por esse meio.

D. *Descrição do Hybrid Genetic to Flying Sidekick* (HGenFS)

Em geral, o HGenFS segue os passos de um algoritmo genético padrão conforme apresentado em [23], com fases e características adaptadas para o FSTSP que resultaram no algoritmo apresentado no Quadro I.

QUADRO I. PSEUDO-CÓDIGO DO ALGORITMO 1.

Algoritmo 1 - HGenFS
1: Inicializa População
2: **Para** todo *IND* na População
3: **Avalia**(*IND*)
4: **Enquanto** critério de parada não for atingido:
5: **Para** todo *N* < *nElite* **então**:
6: Adiciona o indivíduo *N* na nova população
7: **Para** todo *N* >= a *nElite* e *N* < que *nCrossOver/2* **então**:
8: Seleciona parentes P1 e P2
9: Obtém indivíduos *IND1* e *IND2* através do CrossOver de P1 e P2
10: **CriarRota**(*IND1*)
11: **CriarRota**(*IND2*)
12: Adiciona IND1 e IND2 na nova população
13: **Para** todo *n* maior ou igual a *nCrossOver* **então**:
14: Cria novo indivíduo aleatório *IND_R*
15: **CriarRota**(*IND_R*)

```
16:  Para todo IND na População
17:      Avalia(IND)
18:  Para todo IND na população avaliada
19:      Enquanto número de iterações < limite de nBuscaSwap:
20:          Troca aleatória de posições na rota do IND
21:          CriarRota(IND)
22:          Avalia(IND)
23:          Enquanto número de iterações < limite de nBuscaDrone:
24:              Armazena IND em IND_BACK
25:              Alterna o meio de entrega de algum cliente
26:              CriarRota(IND)
27:              Avalia IND
28:              Caso a qualidade do IND for inferior ao IND_BACK
29:                  IND volta ao estado de IND_BACK
```

Durante a execução, a função Avalia é responsável por calcular a qualidade do indivíduo, representada pelo tempo total até finalizar as entregas (função objetivo), adicionada por penalizações por extrapolar a autonomia do drone. Ainda, a função CriarRota é responsável por buscar pontos de decolagem e pouso para viabilizar a entrega com drone para um cliente, em pontos que conforme o FSTSP, devem corresponder a clientes atendidos por caminhão. Nessa função, a seleção pode ser simplesmente aleatória dentre as possíveis, ou por meio de uma busca local adicional (selecionando a melhor rota possível), a um custo computacional mais alto e podendo reduzir a diversidade, devendo ser adotado conforme o problema em estudo. Caso não existam pontos satisfatórios, a entrega por drone é cancelada e o consumidor atendido por caminhão.

O HGenFS inicializa com a criação de uma nova população (linha 1) que, para os testes realizados, foi criada aleatoriamente. A criação é feita inicializando um vetor com uma sequência aleatória de entregas, além de um meio também aleatório, 1 ou 0 no segundo vetor. Obtém-se então a qualidade de cada indivíduo chamando a função Avalia. Em seguida, o algoritmo entra no ciclo de evolução até que seja atingido algum critério de parada.

O primeiro passo no processo é a manutenção de um grupo chamado de elite para a próxima geração, definido por *nElite* (linha 5). Em seguida são criados os descendentes, dois para cada par de pais *P1* e *P2* selecionados (linhas 8 e 9). Para esses novos indivíduos, são criadas as rotas para o drone (linhas 10 e 11), ou seja, são selecionados pontos de decolagem e pouso através da função CriarRota.

Após criados o número de descendentes equivalente ao *nCrossOver*, caso o tamanho da população ainda não tenha sido atingido (foi adotado um tamanho fixo para todas as gerações), são criados indivíduos aleatórios para completar (linha 14). A nova população é então avaliada e procede-se a busca local que se estende de acordo com os parâmetros *nBuscaSwap* e *nBuscaDrone.* Inicialmente, faz-se uma inversão de posição entre dois clientes, cria-se nova rota para o drone e o indivíduo é avaliado para a obtenção da nova qualidade. Partindo desse indivíduo, por *nBuscaDrone* vezes modifica-se aleatoriamente o modo de entrega de um dos clientes, cria-se nova rota e avalia-se novamente (linhas 20 a 22). Caso um indivíduo melhor seja obtido, este é mantido; caso contrário, o que já existia antes da modificação é restaurado. Este processo é repetido por *nBuscaSwap* vezes.

Após a busca local, com indivíduos que conseguiram melhora substituindo os originais na população, o processo é reiniciado, checando o critério de parada e assim por diante.

V. RESULTADOS E DISCUSSÃO

Os modelos matemáticos em PIM foram resolvidos no ambiente AMPL [24], mediante a utilização do *solver* Gurobi (versão 9.1.1), em um *notebook* com um processador Intel Core i7-10750H de 2,60 GHz e memória RAM de 8,00 Gb. O algoritmo genético-híbrido foi implementado na linguagem de programação C++, sendo utilizado uma máquina virtual com disponibilidade 40 *cores* de Xeon Gold 6230R de 2,10Ghz e memória RAM de 128Gb.

Para ambos os métodos, foram utilizadas nove instâncias (ou conjuntos de dados) de um total de 36 propostas em [11], cada uma contendo dez clientes. Este grupo foi selecionado por representar os cenários com maior diferença entre os principais trabalhos avaliados nestes dados [11], [17], [18], que supostamente seriam os casos mais difíceis. Nesses cenários, os clientes foram distribuídos aleatoriamente em uma região de cerca de treze quilômetros quadrados, enquanto o depósito estava localizado próximo ao centro de gravidade dos clientes. Os tempos definidos para o drone foram baseados na distância Euclidiana entre os nós, enquanto os tempos do caminhão conforme a distância de Manhattan [25].

Na PIM, foi definido um valor máximo de dez minutos para cada uma das instâncias, de maneira que o modelo realizasse a busca pela solução ideal, enquanto a razão $|C'|/|C|$ foi mantida entre 80% e 90% do total de clientes, e os seguintes parâmetros foram adotados: E = 20 minutos; e $\sigma^L = \sigma^R = 1$ minuto. No HGenFS, foram realizadas dez inicializações em paralelo, com um limite de 30 segundos. Foram definidos seis grupos de parâmetros com a finalidade de demonstrar o efeito de configurações específicas do algoritmo proposto, conforme apresentado na Tabela I.

TABELA I. CASOS DE TESTE PARA O HGENFS.

	Caso1	***Caso2***	***Caso3***	***Caso4***	***Caso5***	***Caso6***
CrossOver	***DX2***	***DX2***	***DX2***	***SWAP***	***SWAP***	***SWAP***
nBuscaSwap	10	1	20	10	1	20
nBuscaDrone	10	20	1	10	20	1

A Tabela II apresenta, para cada uma das instâncias utilizadas (coluna Instância), os tempos de conclusão das rotas (em minutos) obtidos em [18] e neste trabalho, por meio dos métodos exato (coluna Exato) e meta-heurístico (coluna Ótimo). Para o HGenFS, também é mostrado o tempo em segundos que cada Caso de Teste levou para encontrar o valor ótimo, ou 30,0, caso o limite de tempo tenha sido atingido sem a obtenção desse valor.

TABELA II. COMPARAÇÃO ENTRE OS RESULTADOS OBTIDOS E EM [18].

Teste	***Ref [18] (min)***	***Exato (min)***	***HGenFS***						
			Ótimo (min)	***C1 (s)***	***C2 (s)***	***C3 (s)***	***C4 (s)***	***C5 (s)***	***C6 (s)***
437v6	48,604	48,604	48,604	3,5	1,9	0,9	4,1	0,7	3,4
437v12	56,849	56,849	56,849	5,0	0,6	5,8	14,7	1,1	13,1
440v6	44,506	44,506	44,506	22,1	9,7	8,2	4,6	5,5	9,2
440v7	49,900	49,900	49,900	0,7	4,3	9,9	16,4	4,5	7,0
440v8	62,700	62,700	62,700	11,5	13,4	2,7	27,7	30,0	16,2
440v9	42,533	42,533	42,533	6,5	2,6	30,0	30,0	7,5	30,0
443v7	65,523	65,523	65,523	25,9	17,6	14,9	30,0	15,3	30,0
443v10	47,935	47,935	47,935	2,5	3,6	28,5	10,4	11,7	17,2
443v11	57,382	57,382	57,382	17,2	2,8	1,8	10,4	1,5	14,3
Média para a obtenção do ótimo (s)				10,5	6,3	11,4	16,5	8,6	14,3

Pelos resultados apresentados na Tabela II, é possível observar que os valores encontrados pelo método exato coincidiram com os valores de Dell'Amico, Montemanni e Novelanni [18], tendo, em todos os casos, sido obtida a solução ótima com um tempo de processamento máximo de 98 segundos e uma média de 40,6 segundos. Esses resultados confirmaram que a proposta apresentada pelos autores, e implementada para este trabalho, consegue encontrar soluções ótimas para o FSTSP com dez clientes em um tempo satisfatório, mesmo com um computador pessoal comum.

O HGgenFS, com as configurações dos casos 1 e 2, foi capaz de encontrar os resultados ótimos para todos as instâncias dentro do limite de 30 segundos. Para os demais casos, as exceções estão destacadas em vermelho na tabela, tendo sido atingido esse limite sem a obtenção do ótimo. Comparando as diferentes configurações para o algoritmo proposto, quanto ao método de *crossover*, pode-se observar que o DX2 foi mais eficiente (Casos de 1 a 3), sempre chegando na solução ótima em tempo menor que o SWAP (Casos 4 a 6) para as mesmas configurações de busca local. Vale destacar que nas três configurações utilizando o SWAP ocorreram casos que não se chegou ao ótimo dentro dos trinta segundos.

Quanto à busca local, é possível observar que priorizar a busca na seleção dos meios (*nBuscaDrone*) melhorou significativamente os resultados, independentemente do *crossover* utilizado. Por exemplo, no Caso 1, utilizando o *crossover* DX2 e uma busca equilibrada (*nBuscaDrone=10, nBuscaSwap=10*), o tempo médio até a obtenção da solução ótima foi de 10,5 segundos, enquanto no Caso 2, modificando *nBuscaDrone* para 20 e *nBuscaSwap* para 1, obteve-se 6,3 segundos. Ganho similar também pode ser observado com o *crossover* SWAP no Caso 5. No sentido contrário, nos Casos 3 e 6, observa-se que a priorização da busca por melhores sequenciais de entregas tornou o HGenFS menos eficiente, inclusive gerando o único caso em que não se obteve a solução ótima com o crossover DX2 (teste 440v9 no Caso 3).

VI. CONCLUSÕES

Por meio da solução exata implementada, foi possível confirmar os resultados encontrados em [18], demonstrando a viabilidade para solucionar a proposta para FSTSP de até dez clientes em tempo da ordem de segundos, enquanto a solução apresentada por [11] não havia sido capaz de solucionar mesmo após trinta minutos de processamento.

O algoritmo híbrido proposto, chamado HGenFS, também se mostrou bastante eficiente para solucionar o FSTSP, encontrando as soluções ótimas em todos os testes selecionados em um tempo médio de 6,27s (com o modelo de crossover DX2) e priorizando a busca local para encontrar os melhores meios de entrega ao final da criação de cada geração.

Apesar dos resultados já demonstrarem que o HGenFS pode superar significativamente soluções exatas no estado da arte para dez clientes, o algoritmo precisa ser testado com problemas mais complexos, permitindo a comparação mais direta com outras soluções híbridas e/ou heurísticas. De qualquer maneira, os resultados encontrados já demonstraram que a solução proposta é eficaz e se apresenta como uma alternativa ao existente na literatura para resolver o FSTSP.

A abordagem apresentada neste trabalho pode ser aplicada à otimização da logística militar de distribuição de itens necessários a unidades desdobradas no terreno.

REFERÊNCIAS

[1] G. C. Crişan and E. Nechita, "On a cooperative truck-and-drone delivery system," in *Procedia Computer Science*, 2019, vol. 159, pp. 38–47.

[2] J. C. de Freitas and P. H. V. Penna, "A variable neighborhood search for flying sidekick traveling salesman problem," *International Transactions in Operational Research*, vol. 27, no. 1, pp. 267–290, 2020.

[3] B. Rao, A. G. Gopi, and R. Maione, "The societal impact of commercial drones," *Technology in Society*, vol. 45, 2016.

[4] Machine, "Last Mile Delivery," *https://machine.global/last-mile-delivery/*, May 22, 2020. https://machine.global/last-mile-delivery/ (accessed Apr. 15, 2021).

[5] D. Bamburry, "Drones: Designed for Product Delivery," *Design Management Review*, vol. 26, no. 1, 2015.

[6] A. Goel and L. Kok, "Truck driver scheduling in the United States," *Transportation Science*, vol. 46, no. 3, 2012.

[7] S. et el. Brar, "Drones for deliveries," 2015. http://scet.berkeley.edu/wp-content/uploads/ConnCarProjectReport-1.pdf (accessed Apr. 15, 2021).

[8] D. Schermer, M. Moeini, and O. Wendt, "A branch-and-cut approach and alternative formulations for the traveling salesman problem with drone," *Networks*, vol. 76, no. 2, 2020.

[9] "Drones to be used to deliver supplies to Marines in combat | Intelligent Aerospace." https://www.intelligent-aerospace.com/unmanned/article/14168635/drone-delivery-us-military-marines (accessed Sep. 07, 2021).

[10]Autonomous delivery drones revolutionise military logistics." https://www.army-technology.com/features/autonomous-delivery-drones-military-logistics/ (accessed Sep. 07, 2021).

[11] C. C. Murray and A. G. Chu, "The flying sidekick traveling salesman problem: Optimization of drone-assisted parcel delivery," Transportation Research Part C: Emerging Technologies, vol. 54, pp. 86–109, May 2015.

[12] A. Otto, N. Agatz, J. Campbell, B. Golden, and E. Pesch, "Optimization approaches for civil applications of unmanned aerial vehicles (UAVs) or aerial drones: A survey," *Networks*, vol. 72, no. 4. 2018.

[13] R. Karp, "Reducibility among Combinatorial Problems (1972)," in *Ideas That Created the Future*, 2021.

[14] G. C. Crişan and E. Nechita, "On a cooperative truck-and-drone delivery system," in *Procedia Computer Science*, 2019, vol. 159, pp. 38–47.

[15] N. Agatz, P. Bouman, and M. Schmidt, "Optimization approaches for the traveling salesman problem with drone," *Transportation Science*, vol. 52, no. 4, 2018.

[16] Q. M. Ha, Y. Deville, Q. D. Pham, and M. H. Hà, "Heuristic methods for the Traveling Salesman Problem with Drone," *Technical Report, September 2015. ICTEAM/INGI/EPL*, vol. 86, no. September, 2015.

[17] Q. M. Ha, Y. Deville, Q. D. Pham, and M. H. Hà, "A hybrid genetic algorithm for the traveling salesman problem with drone," *Journal of Heuristics*, vol. 26, no. 2, 2020.

[18] M. Dell'Amico, R. Montemanni, and S. Novellani, "Models and algorithms for the Flying Sidekick Traveling Salesman Problem," Oct. 2019, [Online]. Available: http://arxiv.org/abs/1910.02559

[19] A. H. Halim and I. Ismail, "Combinatorial Optimization: Comparison of Heuristic Algorithms in Travelling Salesman Problem," *Archives of Computational Methods in Engineering 2017 26:2*, vol. 26, no. 2, pp. 367–380, Nov. 2017.

[20] K. Yao, W. Sun, Y. Cui, L. He, and Y. Shao, "A Genetic Algorithm With Projection Operator for the Traveling Salesman Problem," *2021 IEEE International Conference on Artificial Intelligence and Industrial Design (AIID)*, pp. 194–197, May 2021.

[21] A. Lipowski and D. Lipowska, "Roulette-wheel selection via stochastic acceptance," Sep. 2011, Accessed: Jul. 18, 2021. [Online]. Available: http://arxiv.org/abs/1109.3627

[22] J.-Y. Potvin, "Genetic algorithms for the traveling salesman problem," *Annals of Operations Research 1996 63:3*, vol. 63, no. 3, pp. 337–370, 1996.

[23] L. Haldurai, T. Madhubala, and R. Rajalakshmi, "A Study on Genetic Algorithm and its Applications," *International Journal of Computer Sciences and Engineering International Journal of Computer Sciences and Engineering*, 2016, Accessed: Jul. 18, 2021. [Online]. Available: www.ijcseonline.org

[24] AMPL, "version 3.6.6.202102082257." 2019.

M. Dell'Amico, R. Montemanni, and S. Novellani, "Exact models for the flying sidekick traveling salesman problem," *International Transactions in Operational Research*, p. itor.13030, Jul. 2021.