Heurística para o Problema do Caixeiro Viajante com Visitas Próximas o Suficiente

Lettiery D'Lamare Portela Procópio
Universidade Federal do Rio Grande do Norte - UFRN
Campus Universitário Lagoa Nova, 59078-970, Natal, RN
lettiery@ppgsc.ufrn.br

Marco Cesar Goldbarg
Universidade Federal do Rio Grande do Norte - UFRN
Campus Universitário Lagoa Nova, 59078-970, Natal, RN
gold@dimap.ufrn.br

Elizabeth Ferreira Gouvêa Goldbarg
Universidade Federal do Rio Grande do Norte - UFRN
Campus Universitário Lagoa Nova, 59078-970, Natal, RN
beth@dimap.ufrn.br

RESUMO

Este trabalho apresenta um algoritmo baseado em clusterização para o Problema do Caixeiro Viajante com Visitas Próximas o Suficiente, que utiliza o planejamento de percursos para atender demandas a uma certa distância do alvo. São utilizadas técnicas para melhorar a eficácia do modelo em relação a sua solução em um *solver* e confrontar seus resultados com os obtidos pela heurística proposta. A comparação entre os resultados do *solver* e da heurística em 240 instâncias, mostra uma aproximação considerável entre os valores dos custos das soluções, e uma diferença significativa entre os tempos de processamento.

PALAVRAS CHAVE. Caixeiro viajante, Próximo o Suficiente, Heurística de Agrupamento

ABSTRACT

This work presents an algorithm based on clustering for the Close-Enough Traveling Salesman Problem, which uses route planning to carry out demands at a distance from the target. We also use techniques to improve the effectiveness of the model concerning its solution by a solver and compare its results with those obtained by the proposed heuristic. The comparison between the results from the solver and the heuristic on 240 instances, shows a considerable approximation concerning cost values and a large difference concerning the processing times.

KEYWORDS. Traveling Salesman, Close-Enough, Heuristic of Grouping

https://proceedings.science/p/85488?lang=pt-br

1. Introdução

O Problema do Caixeiro Viajante (PCV) é um dos problemas mais abordados na literatura de otimização combinatória, devido a sua grande aplicação prática e dificuldade de solução. Por ser um problema NP-Difícil [Karp, 1972], a expectativa de soluções exatas computacionalmente eficientes para os casos de grande porte é pequena ou limitada a situações particulares.

Os problemas de atendimento de demandas através de veículos aéreos não tripulados (VANTs), também denominados muitas vezes como "drones", são naturalmente passíveis de modelagem através do PCV ou de suas variantes por, caracterizadamente, cumprirem visitas em ciclos hamiltonianos. Uma revisão da literatura abordando as variantes do PCV foi apresentada por Goldbarg et al. [2016].

O presente trabalho propõe um algoritmo baseado na heurística de Gillett e Miller [1974] com método de busca local *Variable Neighborhood Descent*(VND) [Mladenović e Hansen, 1997], para o Problema do Caixeiro Viajante com Visitas Próximas o Suficiente (Close-Enough Travel Salesman Problem - CETSP). Este problema, foi apresentado por Gulczynski et al. [2007] e tem por objetivo criar um ciclo hamiltoniano mínimo para a coleta de dados em sensores de rádio frequência. O caixeiro pode executar a coleta de dados em uma certa distância do alvo.

Mennell [2009] apresentou um modelo não linear do CETSP para criar rotas de VANTs atendendo a necessidade de uma aplicação de monitoramento envolvendo áreas específicas e afastadas entre si. Nesta formulação, Mennell fixa um ponto base para o VANT iniciar e finalizar seu passeio de monitoramento ao longo dos círculos que representam as regiões de demanda.

Com a popularização dos VANTs, o CETSP vem ganhando cada vez mais utilidade e torna-se de importância crescente na pesquisa de boas soluções para o modelo. Aplicações como monitoramento de áreas [Bandeira et al., 2015], coleta de dados [Gulczynski et al., 2007], roteamento de robôs [Yuan et al., 2007], e aplicações bélicas [Mennell, 2009], entram na lista de possíveis cenários do problema.

O algoritmo proposto neste trabalho comprovou alcançar um desempenho significativamente superior ao de um *solver*, encontrando 24 soluções ótimas para as instâncias de um banco de casos teste em tempo até 99,99% menor que o utilizado pelo *solver*. A heurística também foi capaz de apresentar 11 novos limites para o problema em instâncias de até 20 pontos de demanda.

Este trabalho é dividido em cinco seções além da atual. Na seção 2 é apresentado os trabalhos com problemas semelhantes ao CETSP. A seção 3 descreve a formulação do problema. Na seção 4 é explicado a heurística proposta. A seção 5 apresenta os resultados dos experimentos realizados e por fim a conclusão do trabalho é feita na seção 6.

2. Trabalhos relacionados

O CETSP foi originalmente criado como uma variante do *Traveling Salesman Problem with Neighborhood* (TSPN) apresentado por Arkin e Hassin [1994]. Neste problema, o caixeiro precisa completar seu passeio mas considera-se que o veículo aéreo visita a localidade e atende a demanda desse ponto quando sobrevoa qualquer ponto pertencente a uma área convenientemente delimitada ao redor da localidade de demanda. Limites para o TSPN foram apresentados por de Berg et al. [2005], Mitchell [2007] e Elbassioni et al. [2009]. A diferença entre o TSPN e o CETSP é a padronização das áreas de demanda dos clientes. No TSPN, as áreas correspondentes aos pontos são figuras com formas distintas. No caso do CETSP, tais áreas são círculos com variação nos raios.

Gulczynski et al. [2007] apresentaram o CETSP para coleta de dados. O caixeiro/coletor necessitava aproximar-se à distância r dos sensores localizados em pontos distintos em um mapa de demanda. A tarefa de leitura dos dados não necessita de ordem pre-definida ou de tempo para realizá-la. Gulczynski et al. [2007] apresentaram diversas heurísticas para solucionar o problema, mas a *Steiner Zone Heuristic* se mostrou eficaz para o conjunto de instâncias criado.

O trabalho de Gulczynski et al. [2007], serviu de base para Mennell [2009] reformular o CETSP como um problema de roteamento de VANTs. Mennell [2009] formulou o problema

https://proceedings.science/p/85488?lang=pt-br

através de um modelo de programação não linear inteira mista considerando a distância euclidiana. Para solucionar o problema, foi proposto um aprimoramento da *Steiner Zone Heuristic*. As áreas de sobrevoo ao redor dos pontos de demanda são modeladas através de círculos denominados círculos de demanda. O centro de um círculo de demanda é a coordenada da localidade de demanda e o raio é o valor de afastamento considerado aceitável para que o sobrevoo atenda a demanda da localidade. Como há a possibilidade de que os círculos de atendimento se interceptem, as suas áreas de interseção são classificadas como: 0 se o círculo não intercepta outro, 1 se existe área comum entre duas circunferências, 2 se existe uma zona comum entre três círculos, etc. Após a classificação das intercessões, são criados pontos nas áreas comuns entre os círculos, e a rota do veículo aéreo é construída priorizando os pontos que estão em áreas de maior grau. A heurística apresentou bons resultados em instâncias criadas com auto grau de intercessão entre as circunferências de demanda, o que nem sempre está presente em cenários reais.

No trabalho de Mennell [2009], além do CETSP foi formulado uma versão de múltiplos veículos nomeado de Problema de Roteamento de Veículos Suficientemente Próximos (*Close-Enough Vehicle Routing Problem* - CEVRP). No CEVRP, k veículos partem de uma base para realizarem demandas à distância r do alvo. Mennell [2009] também utilizou a *Steiner Zone Heuristc* como solução para o problema de múltiplos veículos.

O trabalho de Yuan et al. [2007] modelou o roteamento de robôs em uma rede de sensores sem fio com variação do raio de alcance de cada cliente. O CETSP foi apresentado como modelo para o problema e foi proposta uma heurística dividida em duas fases. No inicio é determinada a rota sobre os centros dos círculos com um método de permutação. Depois, um algoritmo genético determina os pontos onde a demanda será atendida dentro do círculo, minimizando o comprimento da rota.

Duas discretizações para o CETSP foram propostas por Behdani e Smith [2014]. Na primeira, a região da demanda é substituída por pequenos quadrados, criando-se uma grade no círculo original. Os possíveis pontos de visita são definidos pelos vértices da grade. Essa discretização é denominada *Grid-Based Partition* e sua aplicação pode ser estendida para outros problemas com as características do CETSP e TSPN. A segunda discretização, denominada *Arc-Based Partition*, utiliza uma heurística de secção dos círculos para espalhar de forma arbitrária pontos sobre algumas regiões da circunferência. Esses pontos são as bases para criação da rota final.

Coutinho et al. [2016] apresentaram um algoritmo *branch-and-bound* para solucionar o CETSP. Foi realizada uma transformação no modelo de programação não linear proposto por Mennell [2009] para um modelo cônico de segunda ordem. Foram realizados testes com 824 instâncias em cenários de duas e três dimensões. Os resultados do método são melhores em instâncias com alto grau de interseção entre os círculos.

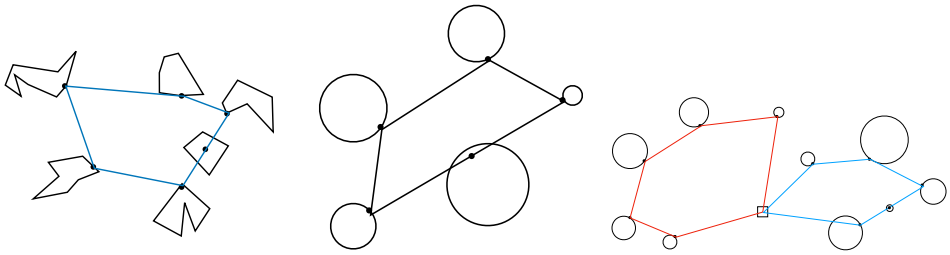
Outra proposta de discretização foi apresentada por Carrabs et al. [2017], onde o círculo do cliente é subdividido em regiões selecionadas por um algoritmo que analisa possíveis rotas. A aplicação da discretização em um *solver* tanto apresenta limites inferiores como superiores para o problema. Além disso, aplica-se um algoritmo de redução do grafo que restringe o tamanho do problema e acelera a computação dos limites.

Para melhor compreensão das diferentes abordagens dos trabalhos citados, pode-se observar na Figura 1 soluções para os problemas TSPN, CETSP e CEVRP e suas particularidades. A Figura 1 (a) apresenta uma solução para o TSPN onde a demanda pode ser considerada atendida em qualquer ponto dos polígonos de demanda. No CETSP, Figura 1 (b), as áreas de demanda são apresentadas como círculos de diferentes raios. No CEVRP, Figura 1 (c), é incluído o depósito (quadrado) no mapa como início e fim das rotas dos veículos.

3. Descrição e Formulação do Problema

O PCV é um dos problemas mais estudados da Pesquisa Operacional. O problema consiste em determinar um ciclo hamiltoniano mínimo sobre um grafo ponderado em arestas. Este é

https://proceedings.science/p/85488?lang=pt-br



(a) TSPN (b) CETSP (c) CEVRP

Figura 1: Problemas suficientemente próximos

um problema da classe NP-Difícil [Applegate et al., 2006], e sua solução para instâncias com mais de 85 mil vértices não é trivial.

Algumas particularidades na aplicação do PCV dão origem a variantes do problema com, no mínimo, a mesma dificuldade de resolução. O Problema do Caixeiro Viajante Euclidiano (*Euclidean Traveling Salesman Problem* - ETSP) é um exemplo de variante do PCV onde os vértices são dados de coordenadas em duas e três dimensões e é considerada a distância euclidiana. Uma recente revisão acompanhada de uma taxonomia para variantes do PCV foi proposta por Goldbarg et al. [2016].

Derivado do ETSP e do TSPN podemos compreender as particularidades do problema tratado neste trabalho. No CETSP, assim como no ETSP, o passeio é realizado no espaço euclidiano e as localidades de demanda são representadas por suas coordenadas nesse espaço. Contudo, a demanda não necessariamente deve ser atendida na coordenada do ponto do cliente, mas pode ser atendida mesmo com a rota passando afastada da coordenada de demanda de um valor máximo conhecido e igual a r.

3.1. Formulação

A formulação do CETSP apresentada por Mennell [2009] não pode ser resolvida por *solvers* de programação linear, pois sua modelagem envolve o cálculo da distância euclidiana com a utilização de raiz quadrada, o que classifica sua formulação como um modelo de programação não linear com funções não convexas. Essa característica restringe a solução do modelo aos *solvers* de programação não-linear.

Uma alternativa para solucionar problemas dessa classe foi proposta por Coutinho et al. [2016] com uma formulação cônica de segunda ordem para substituir a distância euclidiana. Essa proposta serviu de base para implementarmos o modelo proposto por Mennell [2009] em um *solver* linear que suporta equações cônicas.

Mennell [2009] construiu uma formulação para o CETSP baseada na programação não linear inteira mista, com as coordenadas $(x_i, y_i) \in R$ do ponto i como variáveis de decisão.

Os dados de entrada são as coordenadas dos pontos de demanda (a_i, b_i), $i = 0, ..., n$ referenciadas a um eixo cartesiano x, y, bem como o raio r_i associado ao afastamento aceitável da passagem do veículo aéreo em relação ao ponto de demanda i. Mennell [2009] considera os raios r_i iguais ($r_i = r$) para qualquer dos pontos de demanda i. O formulação do CETSP utiliza as seguintes variáveis de decisão:

- $e_{ij} = \begin{cases} 1, & \text{se o ponto } i \text{ é predecessor imediato para o ponto } j. \\ 0, & \text{caso contrário.} \end{cases}$
- x_i coordenada cartesiana x de um ponto suficientemente próximo de i.
- y_i coordenada cartesiana y de um ponto suficientemente próximo de i.

A lista de dados de entrada para o modelo do CETSP é dada a seguir:

https://proceedings.science/p/85488?lang=pt-br

- n quantidade de círculos de demanda.
- a_i coordenada cartesiana x de um ponto de demanda i.
- b_i coordenada cartesiana y de um ponto de demanda i.
- r distância máxima para o caixeiro realizar a demanda do ponto de demanda.

O modelo matemático proposto por Mennell [2009] é apresentado nas expressões (1)-(7).

$$Min \qquad \sum_{i=0}^{n}\sum_{j=0}^{n} e_{ij}\ \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} \tag{1}$$

Sujeito a:

$$(x_i - a_i)^2 + (y_i - b_i)^2 \leqslant r^2 \qquad \forall i = 0, \cdots, n \tag{2}$$

$$\sum_{i=0}^{n} e_{ij} = 1 \qquad \forall j = 0, \cdots, n \tag{3}$$

$$\sum_{j=0}^{n} e_{ij} = 1 \qquad \forall i = 0, \cdots, n \tag{4}$$

$$\sum_{i,j \in S}^{n} e_{ij} \leqslant |S| - 1 \qquad \forall\ S \subseteq \{1, \cdots, n\}, 2 \leqslant |S| \leqslant n \tag{5}$$

$$\sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} \geqslant \epsilon \qquad \forall i, j = 0, \cdots, n \tag{6}$$

$$e_{ij} \in \{0, 1\};\ x_i, y_i \in \mathbb{R} \qquad \forall i, j = 0, \cdots, n \tag{7}$$

A função objetivo (1) minimiza a rota realizada pelos pontos representativos. O custo da aresta que liga os pontos de coordenadas (x_i, y_i) a (x_j, y_j) é dado pelo pela distância euclidiana entre os pontos i e j, o que torna a função objetivo não linear. A restrição (2) garante que o ponto representativo (x_i, y_i) esteja dentro da circunferência de centro (a_i, b_i) de raio r. A restrições (3), (4) e (5) são as restrições clássicas do PCV.

A restrição (6) foi inserida no modelo para que toda aresta tenha um comprimento mínimo. Mennell [2009] justificou essa restrição como uma forma de otimizar o trabalho do *solver*. Por fim, a restrição (7) define a variável e_{ij} binária, e as coordenadas x_i, y_i como valores reais. Algumas linearizações e outras modelagens para o CETSP foram propostas por Behdani e Smith [2014], Coutinho et al. [2016],Carrabs et al. [2017].

Para a implementação do modelo descrito acima, a pesquisa utilizou a abordagem de Coutinho et al. [2016] para transformar a função objetivo em equação cônica de segunda ordem. Insere-se a variável de decisão d_{ij} para auxiliar o cálculo da distância euclidiana entre dois pontos $d_{ij}^2 = (x_i - x_j)^2 + (y_i - y_j)^2$. Essa equação remove do modelo a função da raiz quadrada (não linear), possibilitando a implementação em *solvers* lineares.

https://proceedings.science/p/85488?lang=pt-br

4. Algoritmo Clock a Round

A literatura fornece um grande números de métodos exatos para a solução do PCV. Contudo, para problemas de grande porte, a expectativa de obter a solução exata do PCV em tempo de processamento razoável é pequena, justificando-se o uso de abordagens aproximativas.

A Heurística de Gillett e Miller [1974] mostrou-se eficaz para problemas derivados do PCV e com pontos marcados por coordenadas cartesianas ou polares. Em Prins et al. [2014], pode-se encontrar mais de 70 referências de utilização desta heurística em trabalhos derivados do PCV. Nesta Heurística o problema é dividido em duas partes diferentes. No primeiro, os vértices são agrupados seguindo algum critério de proximidade (*cluster-first*). Posteriormente, cada grupo é ordenado independentemente (*route-second*).

A etapa de agrupamento tem como coordenada base do plano polar (a_0, b_0), o ponto de partida do caixeiro. A partir dele são calculadas todas as outras coordenadas polares (r, θ), onde r é a distância entre o ponto e a origem $r = \sqrt{(a_0 - x_i)^2 + (b_0 - y_i)^2}$ e θ é o ângulo formado entre o segmento de reta que une o ponto à origem e o eixo x. Podemos encontrar esse valor com a função trigonométrica $\theta = arctan(\frac{y}{x})$, onde x e y são as diferenças das coordenadas de origem até o ponto. Logo depois desta transformação, os pontos polares são ordenados por seus valores de θ, e agrupados por proximidade de ângulos. Essa etapa pode ser melhor compreendida com a Figura 2.

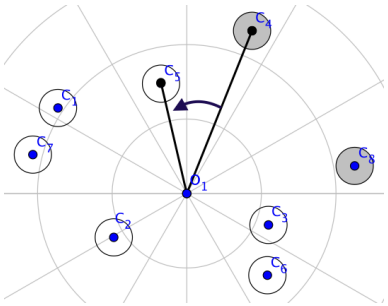


Figura 2: Agrupamento segundo Gillett e Miller [1974]

Na Figura 2, a clusterização iniciou em C_8, ao incrementar o valor de θ achamos C_4 e logo em seguida C_5. Ao fim desta etapa do algoritmo, a solução inicial terá a configuração $\{0, C_8, C_4, C_5, C_1, C_7, C_2, C_6, C_3, 0\}$, esse processo é reiniciado com todos valores de θ. A rota de custo mínimo criará a solução base para a escolhas dos pontos representativos para o refinamento desta solução.

A solução inicial dada pela melhor configuração seguindo a regra heurística de Gillett e Miller [1974], roteia o caixeiro pelo centro das circunferências $(x_i, y_i) = (a_i, b_i)$. O que desconsidera o raio de tolerância da visita, eliminando a possibilidade de explorar trajetórias afastadas dos pontos de demanda.

Para representar a ordem de visitas no percurso do caixeiro, a solução s é constituída por um vetor ordenado de pontos p_i de duas dimensões (x e y) com índice i dos respectivos círculos de demanda C_i. A solução inicial dada pelo agrupamento da Figura 2 pode ser representado como $s = \{p_8, p_4, p_5, p_1, p_7, p_2, p_6, p_3\}$. Para calcular o custo da solução, além das distâncias entre os pontos de demanda, deve-se incluir a distância de partida e chegada no ponto inicial da rota do caixeiro.

4.1. Geometric Neighborhoods

Para otimizar a solução criada pela heurística de Gillett e Miller [1974], são utilizadas algumas propriedades geométricas do problema para reduzir o percurso efetuado pelo caixeiro. Essas propriedades dão origem a 5 estruturas de vizinhanças que modificam as coordenadas dos pontos de demanda do caixeiro. Ocasionalmente, essa tarefa pode proporcionar uma redução no percurso escolhido.

https://proceedings.science/p/85488?lang=pt-br

Para o CETSP a região de demanda é demarcada por círculos, o que permite utilizar sua característica "próximo o suficiente"(atender a demanda passando suficientemente próximo do ponto). Com isso, para construir as vizinhanças geométricas, partimos da definição básica do círculo em um espaço euclidiano $\{\forall(x, y), r > 0 \mid (x - x_c)^2 + (y - y_c)^2 \leq r^2\}$ definindo um conjunto de pontos que estão, no máximo, à distância r do ponto central (x_c, y_c). A superfície deste objeto é formada pelos pontos cuja distância é igual a r, $s_c = \{(x, y) \mid (x-x_c)^2+(y-y_c)^2 = r^2\}$.

4.2. Estrutura de Vizinhança 1

Na primeira estrutura de vizinhança, adotaremos a premissa que para toda semi-reta $\overline{u}$ que passa pelo círculo C, existem até dois pontos que pertencem a superfície do círculo S_c e a semi-reta $\overline{u}$, $\forall(\overline{u}, C) \rightarrow \{\overline{u} \in C, 1 \leq \| \ S_c \cap \overline{u} \ \| \leq 2\}$. A estrutura de vizinhança seleciona uma aresta que passa pelo centro de C, depois calcula o ponto de intercessão da aresta com a superfície da circunferência.

As coordenadas dos pontos de interseção entre a semi-reta $\overline{u}$ e a superfície a circunferência S_c são dadas pela solução do sistema de equação do segundo grau constituído com a equação geral da reta $ax + by + c = 0$, e com a equação reduzida da circunferência $(x - x_c)^2 + (y - x_c)^2 = r^2$.

4.3. Estrutura de Vizinhança 2

Para esta estrutura de vizinhança, são criadas retas tangentes às circunferências de demanda a partir de um ponto externo a figura. As coordenadas desse ponto externo são obtidas pelos pontos de demanda de outra circunferência. Para encontrar os dois pontos tangentes t_1 e t_2 presentes na circunferência de demanda C e que partem do ponto externo p, criamos uma circunferência fictícia CV centrada no ponto médio entre p e o centro de C. O raio dessa nova circunferência CV é exatamente a metade da distância de p e o centro de C. A circunferência fictícia passa pelo ponto externo p e o centro de CV, assim como os dois pontos tangentes t_1 e t_2. Com isso, é formado um novo sistema de equações de segundo grau com as equações paramétricas das duas circunferências $(x - x_{ci})^2 + (y - x_{ci})^2 = r^2$ e $(x - x_{cv})^2 + (y - x_{cv})^2 = r_v^2$. A solução deste sistema sistema nos dará as coordenadas dos pontos tangentes t_1 e t_2.

4.4. Estrutura de Vizinhança 3

A terceira estrutura de vizinhança usa da intercessão de duas circunferências de demanda para gerar pontos pertencentes aos dois círculos simultaneamente. Essa possível ação é identificada quando as distâncias entre os centros de dois círculos c_i e c_j, é menor do que a soma dos seus raios. Para esse caso é dito que as circunferências c_i e c_j são secantes. E da mesma forma do cenário descrito acima, a solução do sistema de equações de segundo grau formado pelas equações das duas circunferências $(x - x_{ci})^2 + (y - x_{ci})^2 = r^2$ e $(x - x_{cj})^2 + (y - x_{cj})^2 = r^2$, dará as coordenadas dos dois pontos. Ainda existe uma caso específico onde as circunferências c_i e c_j são tangentes, ou seja, elas se tocam em apenas um ponto, e o sistema de equação admite apenas uma solução viável.

4.5. Estrutura de Vizinhança 4

Para criar pontos ao redor da superfície do círculo, esta estrutura de vizinhança cria retas secantes a circunferência de demanda para gerar pontos presentes nas retas e na supercie do círculo. Os pontos que interceptam a superfície do círculo e as retas são utilizadas para substituir as coordenadas de demanda do cliente. Caso o novo ponto reduza o percurso do caixeiro, ele prevalecerá como ponto representativo. Mas se não reduzir o comprimento da rota, o antigo ponto retorna a sua posição.

4.6. Estrutura de Vizinhança 5

Para melhorar a ordem dos pontos ao longo do passeio, utiliza-se o método 2-opt. Este método foi apresentado por Gendreau et al. [1992] para otimizar problemas derivados do PCV. O 2-opt parte de um círculo hamiltoniano $c_1, c_2, \cdots, c_n$, e escolhe duas arestas para ser desfeitas $(c_i - c_j), (c_k - c_l)$. Logo em seguida testa-se a nova combinação que não cria um *subtour* $(c_i - c_l), (c_k - c_j)$. Se esse novo conjunto de arestas melhorar o comprimento do passeio, prevalecerá como nova sequencia de pontos, caso contrário, é testado novos conjuntos buscando a redução do *tour*.

https://proceedings.science/p/85488?lang=pt-br

4.7. Variable Neighborhood Descent

Todas as estruturas de vizinhanças descritas nessa seção buscam a otimização da solução criada pelo algoritmo de Gillett e Miller [1974]. Para melhor explorar o potencial das estruturas, é utilizada a heurística de busca local *Variable Neighborhood Descent* (VND) apresentada por Mladenović e Hansen [1997]. Ela utiliza uma troca sistemática de vizinhanças para explorar melhores locais do problema. Este método se diferencia de outras buscas locais pois não é formada apenas por uma estrutura de vizinhança, mas experimenta distintas estruturas para modificar a solução analisada.

O VND parte de uma solução inicial *s* para encontrar um melhor local. Isso acontece com a utilização da estrutura V_i que cria uma nova solução vizinha *s"*, a qual torna-se a nova base para o VND e reinicia o *loop* de utilização das estruturas se seu custo for menor do que a solução original *s'*. Caso a solução vizinha *s"* não for melhor do que *s'*, então será utilizada uma nova estrutura de vizinhança V_{i+1} para aplicar modificações diferentes à *s'*. Esse método preserva a característica da solução atual e também verifica soluções próximas e promissoras. O Algoritmo 1 demostra a utilização do VND com as estruturas de vizinhanças descritas anteriormente.

Algorithm 1 *Variable Neighborhood Descent*

Require: s - solução inical
V_i - estruturas de vizinhanças
Ensure: s' - solução vizinha

```
1: s' ⇐ s
2: i ⇐ 1
3: while i ≤ |V_i| do
4:    s'' ⇐ V_i(s')
5:    if s'' < s' then
6:       s' ⇐ s''
7:       i ⇐ 1
8:    else
9:       i ⇐ i + 1
10:   end if
11: end while
12: return s'
```

5. Experimentos

Os modelo descrito na seção 3 foi implementado no software *IBM ILOG CPLEX Optimization Studio 12.4*. Todo código deste trabalho foi desenvolvido na linguagem de programação C++, e os testes executados em um computador com processador Intel Core i5 2,3 GHz, com 8 GB de memória RAM.

A ordem de chamada das estruturas de vizinhanças é essencial para encontrar uma melhor solução em um tempo de processamento reduzido. No presente problema utilizamos as estruturas de 2-opt primeiro e logo em seguida recalculamos as coordenadas dos pontos representativos com as estruturas geométricas. Essa ordem foi obtida por testes com diferentes instâncias e configurações. Todas as estruturas descritas na seção 4 aplicam modificações e redução no custo da solução.

As instâncias utilizadas nos experimentos deste trabalho foram disponibilizadas por Behdani e Smith [2014]. Seguimos metodologia análoga ao seu trabalho. As instâncias testes são divididas em 8 grupos, com 6, 8, 10, 12, 14, 16, 18, e 20 pontos. Cada conjunto tem 30 problemas testes, totalizando 240 instâncias. Os resultados apresentados pela heurística de *Clock a Round* são identificadas pelas colunas demarcadas com *CR*. Como a heurística apresentada na seção 4 é determinística, os valores apresentados na coluna *CR* são oriundos de apenas uma execução independente do algoritmo *Clock a Round*. Esses valores são comparados com as instâncias em que o *Solver* consegue finalizar sua execução em um tempo limite de 4000 segundos.

https://proceedings.science/p/85488?lang=pt-br

Há casos onde o limite de tempo não foi suficiente para sua finalização, os valores de *lower* e *ubber bonds* obtidos pelo *Solver* no instante de sua interrupção, e são apresentados nas colunas *LB* e *UB* respectivamente. A média do custo de cada conjunto de instância é apresentada pelo identificador *Cost* nas colunas. Já o tempo de processamento para obter esses valores, é dado em segundo pelas colunas *Time*.

Inst	*Solver*		*CR*	
	Cost	*Time (s)*	*Cost*	*Time (s)*
6	29,06	0,80	29,26	0,05
8	31,85	2,21	32,05	0,08
10	34,30	17,49	34,51	0,10
12	36,28	341,91	36,54	0,18

Tabela 1: Resultados apresentados pelo *Solver* e *CR* nas instâncias com 6, 8, 10 e 12 pontos

A Tabela 1 apresenta os resultados dos experimentos computacionais realizados nos conjuntos com 6, 8, 10 e 12 pontos. No conjunto com 6 clientes, é possível observar um afastamento médio dos resultados obtidos pelo *Solver, Cost* e *CR, Cost* de 0,68% na média do custo das soluções. No conjunto de 8 clientes, esse valor é reduzido para 0,65%. Já no conjunto de 10 clientes, o afastamento médio entre a melhor solução do *CG* e o *Solver* é de 0,60%.

Quando são confrontados os tempos de processamento em segundos dado pelas colunas *Time* entre o *Solver* e *CR*, encontramos um afastamento médio de 94,16%, 96,47%, 94,16%, 99,45% e 99,95% para o conjunto de instâncias 6, 8, 10 e 12 respectivamente. Isso representa que a heurística finaliza sua execução primeiro do que o *Solver* em todos os conjuntos de instâncias.

Inst	*Solver*			*CR*	
	LB	*UB*	*Time (s)*	*Cost*	*Time (s)*
14	32,78	37,15	2478,46	37,33	0,27
16	19,14	39,17	4000,00	39,31	0,47
18	10,97	40,48	4000,00	40,76	0,47
20	27,37	42,69	4000,00	42,78	0,48

Tabela 2: Resultados apresentados pelo *Solver* e *CR* nas instâncias com 14, 16, 18 e 20 pontos

A Tabela apresenta os valores obtidos pelos conjuntos 14, 16, 18 e 20 das instâncias. Nela é possível identificar um afastamento médio entre os limites superiores do *Solver UB*, e os valores de *CR* de 0,48%, 0,37%, 0,70% e 0,21% para as instâncias com 14, 16, 18 e 20 pontos respectivamente. Nesta tabela também é possível observar um afastamento médio de 13,35%, 51,13%, 72,90% e 56,00% entre os limites *LB* e *UB* do *Solver*. A heurística também foi capaz de criar 24 soluções ótimas, e 11 novos limites para instâncias de até 20 pontos.

6. Conclusão

A combinação entre o método de Gillett e Miller [1974] e as estruturas de vizinhanças que compõem a heurística *Clock a Round (CR)* apresentada na seção 4, se mostraram eficazes ao encontrar soluções com valores bem próximos ao encontrados pelo *Solver* em um tempo consideravelmente menor. A necessidade de redução do tempo de processamento pode ser encontrada em cenários que os dados do problema são obtidos no instante em que a solução é requerida. Isto pode ser compreendido em casos que o algoritmo de roteamento está sendo executando quando o veículo já está disponível na base aguardando apenas as coordenadas de demanda. Portanto, para aplicações em que o tempo de processamento é um fator restrito e soluções de até 0,7% longe do ótimo são aceitáveis, a heurística *CR* apresenta-se como boa alternativa para o problema.

https://proceedings.science/p/85488?lang=pt-br

Sugere-se que para uma melhor adequação do modelo proposto à sua aplicação real seria positivo a consideração de forças que podem interferir no deslocamento do drone como, por exemplo, a influência do vento. Sugere-se, outrossim, o exame de métodos meta-heurísticos construtivos para a solução do modelo como, por exemplo *Ant Colony Optimization* e GRASP, especialmente devido ao bom resultado obtido pela composição de heurísticas *ad hoc* construtivas.

Referências

Applegate, D. L., Bixby, R. E., Chvátal, V., e Cook, W. J. (2006). *The traveling salesman problem: a computational study*. Princeton university press, Princeton, New Jersey, USA. ISBN 978069112993.

Arkin, E. M. e Hassin, R. (1994). Approximation algorithms for the geometric covering salesman problem. *Discrete Applied Mathematics*, 55(3):197–218.

Bandeira, T. W., Coutinho, W. P., Brito, A. V., e Subramanian, A. (2015). Analysis of path planning algorithms based on travelling salesman problem embedded in uavs. In *2015 Brazilian Symposium on Computing Systems Engineering (SBESC)*, p. 70–75.

Behdani, B. e Smith, J. C. (2014). An integer-programming-based approach to the close-enough traveling salesman problem. *INFORMS Journal on Computing*, 26(3):415–432.

Carrabs, F., Cerrone, C., Cerulli, R., e Gaudioso, M. (2017). A novel discretization scheme for the close enough traveling salesman problem. *Computers & Operations Research*, 78:163–171.

Coutinho, W. P., Nascimento, R. Q. d., Pessoa, A. A., e Subramanian, A. (2016). A branch-and-bound algorithm for the close-enough traveling salesman problem. *INFORMS Journal on Computing*, 28(4):752–765.

de Berg, M., Gudmundsson, J., Katz, M. J., Levcopoulos, C., Overmars, M. H., e van der Stappen, A. F. (2005). Tsp with neighborhoods of varying size. *Journal of Algorithms*, 57(1):22–36.

Elbassioni, K., Fishkin, A. V., e Sitters, R. (2009). Approximation algorithms for the euclidean traveling salesman problem with discrete and continuous neighborhoods. *International Journal of Computational Geometry & Applications*, 19(02):173–193.

Gendreau, M., Hertz, A., e Laporte, G. (1992). New insertion and postoptimization procedures for the traveling salesman problem. *Operations Research*, 40(6):1086–1094.

Gillett, B. E. e Miller, L. R. (1974). A heuristic algorithm for the vehicle-dispatch problem. *Operations research*, 22(2):340–349.

Goldbarg, M. C., Goldbarg, E. G., e Luna, H. P. L. (2016). *Otimização Combinatória e Metaheurísticas: Algoritmos e Apliacações*. Editora Elsevier, Rio de Janeiro - RJ, Brasil. ISBN 9788535278125.

Gulczynski, D. J., Heath, J. W., e Price, C. C. (2007). The close enough traveling salesman problem: A discussion of several heuristics. *Perspectives in Operations Research*, p. 271–283.

Karp, R. M. (1972). Reducibility among combinatorial problems. In *Complexity of computer computations*, p. 85–103. Springer.

Mennell, W. K. (2009). *Heuristics for solving three routing problems: Close-enough traveling salesman problem, close-enough vehicle routing problem, sequence-dependent team orienteering problem*. PhD thesis.

https://proceedings.science/p/85488?lang=pt-br

Mitchell, J. S. (2007). A ptas for tsp with neighborhoods among fat regions in the plane. In *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*, p. 11–18. Society for Industrial and Applied Mathematics.

Mladenović, N. e Hansen, P. (1997). Variable neighborhood search. *Computers & operations research*, 24(11):1097–1100.

Prins, C., Lacomme, P., e Prodhon, C. (2014). Order-first split-second methods for vehicle routing problems: A review. *Transportation Research Part C: Emerging Technologies*, 40:179–200.

Yuan, B., Orlowska, M., e Sadiq, S. (2007). On the optimal robot routing problem in wireless sensor networks. *IEEE Transactions on Knowledge and Data Engineering*, 19(9):1252–1261.

https://proceedings.science/p/85488?lang=pt-br