Proceeding Series of the Brazilian Society of Computational and Applied Mathematics, v. 11, n. 1, 2025.

Trabalho apresentado no XLIII CNMAC, Centro de Convenções do Armação Resort - Porto de Galinhas - PE, 2024

Proceeding Series of the Brazilian Society of Computational and Applied Mathematics

Um Método Heurístico para o Problema do Caixeiro Viajante Suficientemente Próximo

Paula C. R. Ertel[1]
Departamento de Matemática Aplicada - IME/USP, São Paulo, SP
Ernesto G. Birgin[2]
Departamento de Ciência da Computação - IME/USP, São Paulo, SP

Resumo. O Problema do Caixeiro Viajante (TSP, do inglês *Traveling Salesman Problem*) é um dos problemas de otimização combinatória mais estudados. Dado m cidades e seus pares de distâncias, busca-se encontrar um tour de distância mínima que passe por todas as cidades. Na variante *Close-Enough* do TSP (CETSP), não é necessário passar exatamente por cada cidade, mas por uma vizinhança de cada cidade. Esse problema pode ser modelado como um problema de otimização contínua com variáveis inteiras. Neste trabalho, apresentamos um método heurístico para encontrar soluções para o CETSP, incluindo uma heurística construtiva, uma busca local baseada no movimento 2-opt e um método de otimização contínua de busca em blocos de coordenadas. Experimentos numéricos foram realizados com 60 instâncias de pontos aleatórios e 10 instâncias da biblioteca TSPLIB adaptadas ao CETSP. O método proposto se sobressai, na maioria dos testes, em relação a qualidade de soluções e tempo de execução aos métodos que usam movimentos de realocação.

Palavras-chave. Problema do Caixeiro Viajante, CETSP, Heurísticas, Métodos de Busca em Bloco de Coordenadas

1 Introdução

O Problema do Caixeiro Viajante (TSP) consiste na seguinte questão: dado um conjunto de m cidades, com a distância de se viajar de uma cidade i para qualquer outra cidade j conhecida, deseja-se encontrar o tour de menor comprimento que passe por todas as m cidades uma única vez e retorne à cidade inicial. Apesar do TSP ser um problema que já foi intensivamente investigado, seu estudo ainda apresenta muitas possibilidades, sobretudo quando variantes do problema são consideradas. Uma delas é o Problema do Caixeiro Viajante Suficientemente Próximo (CETSP, do inglês *Close-Enough Traveling Salesman Problem* [1, 4]).

Sejam $\{p^0, c^1, \ldots, c^m\} \in \mathbb{R}^2$ $m+1$ pontos distintos e $\Omega_i \in \mathbb{R}^2$ um conjunto compacto que contém c^i, para cada $i = 1, \ldots, m$. Na variante *Close-Enough* do TSP o problema consiste em determinar os pontos $x^i \in \Omega_i$ para cada $i = 1, \ldots, m$ e uma sequência $\mathcal{S} = \{i_1, i_2, \ldots, i_m\}$, de modo que o tour que inicia em p^0 (*depósito*), passa por cada x^i na ordem definida por $\mathcal{S}$ e retorna para p^0 tenha comprimento mínimo (em relação à distância Euclidiana). Neste trabalho será considerado o CETSP em que os conjuntos visitados são dados por bolas pertencentes ao $\mathbb{R}^2$.

O CETSP apresenta ampla aplicabilidade em problemas do mundo real. Alguns exemplos são na localização de incêndios florestais por veículos aéreos não tripulados [6] e no diagnóstico de painéis solares por reconhecimento [5]. Outro exemplo é em tecnologias de identificação por radiofrequência, empregadas na leitura automatizada de medidores [8].

[1] paulaertel@ime.usp.br
[2] egbirgin@ime.usp.br

O artigo está organizado da seguinte forma. A Seção 2 contém a formulação matemática do problema. A Seção 3 descreve o método heurístico proposto. Os experimentos numéricos são apresentados na Seção 4 e a Seção 5 contém as considerações finais deste trabalho.

2 Formulação do problema

Considere um conjunto de m bolas disjuntas $\mathcal{B}_i := \mathcal{B}(c^i, r^i) \in \mathbb{R}^2$, $i = 1, \ldots, m$, em que $c^i \in \mathbb{R}^2$ e $r^i \in \mathbb{R}$ denotam o centro e raio, respectivamente. Utilizando a formulação proposta em [2], o CETSP pode ser modelado como o seguinte problema de otimização contínua com variáveis inteiras:

$$\underset{x \in \mathbb{R}^n}{\text{minimizar}} \ \|x^{i_m} - x^{i_1}\| + \sum_{\nu=1}^{m-1} \|x^{i_\nu} - x^{i_{\nu+1}}\| \quad \text{sujeito a} \quad x^{i_\nu} \in \mathcal{B}_{i_\nu} \ \text{ para } \ \nu = 1, \ldots, m, \tag{1}$$

em que $\{i_1, i_2, \ldots, i_m\}$ é uma permutação de $\{1, 2, \ldots, m\}$, $n = 2m$, $x^{i_\nu} = (x_1^{i_\nu}, x_2^{i_\nu}) \in \mathbb{R}^2$ para $\nu = 1, \ldots, m$, $x^T = ((x^1)^T, \ldots, (x^m)^T) \in \mathbb{R}^n$ e $\|\cdot\|$ corresponde à norma Euclidiana.

A resolução de (1) pode ser tratada em duas partes: (i) fixado um conjunto de pontos x o problema (1) corresponde ao TSP Euclidiano, e heurísticas já desenvolvidas para este caso podem ser empregadas para obter uma permutação $\{{i_1}^*, {i_2}^*, \ldots, {i_m}^*\}$ que minimiza o valor da função objetivo; (ii) por outro lado, fixada uma permutação $\{i_1, i_2, \ldots, i_m\}$, o problema (1) se torna contínuo com variáveis reais e, neste caso, métodos de busca em bloco de coordenadas (BCDM [9]) podem ser utilizados para encontrar uma solução x^*.

Na Seção 3 a seguir as estratégias (i) e (ii) são combinadas em um método heurístico capaz de obter soluções para o problema (1).

3 Algoritmos para o CETSP

O algoritmo proposto para resolver numericamente o problema (1) segue um esquema iterativo, no qual a determinação de cada iterando $(i_1^k, \ldots, i_m^k, x^k)$ ocorre em duas etapas. Em resumo, o algoritmo é inicializado com um conjunto inicial de m pontos $x_i \in \mathcal{B}_i$ para $i = 1, \ldots, m$. Com estes pontos fixados, uma permutação inicial $T^0 := (i_1^0, \ldots, i_m^0)$ é construída utilizando a heurística construtiva do vizinho mais próximo (NN, do inglês *Nearest Neighbor*). Em seguida, os pontos x_i são recalculados com o método de busca em bloco de coordenadas (BCDM) na ordem dada pela permutação T^0. Tais pontos irão compor o vetor $x^0 := (x_1^0, \ldots, x_m^0)$ e com isso obtemos o tour inicial (T^0, x^0). A partir disso, o algoritmo segue iterativamente, de modo que a cada iteração desejamos encontrar um par (T^k, x^k) com custo menor do que o da iteração anterior. Essa busca se faz testando permutações (obtidas com o método de busca local 2-opt) e, para cada permutação, calculando um novo x com o BCDM, para então determinar o comprimento do tour dessa permutação. Vale ressaltar que a palavra "tour" é empregada para se referir ao caminho gerado ao percorrer os pontos de $x = (x_1, \ldots, x_m)$ na ordem dada pelo vetor T, e que o comprimento total de um tour (T, x) é dado pelo custo da função objetivo de (1).

O método de busca em bloco de coordenadas proposto para atualizar os pontos visitados em um tour do CETSP é dado pelo Algoritmo 1. Neste algoritmo, fixada uma permutação T, a ideia é minimizar a função objetivo de (1) em relação a um bloco de coordenadas x^i por vez, tal que a escolha do bloco é feita na ordem cíclica dada por T. Ou seja, para cada $\nu = 1, \ldots, m$, desejamos determinar o ponto $x^{i_\nu} \in \mathcal{B}_{i_\nu}$ que minimiza $\varphi(x^{i_\nu}) := \|x^{i_{\nu-1}} - x^{i_\nu}\| + \|x^{i_\nu} - x^{i_{\nu+1}}\|$, em que $x^{i_{\nu-1}}$ e $x^{i_{\nu+1}} \in \mathbb{R}^2$ correspondem aos pontos anterior e posterior a x^{i_ν} de acordo com a permutação T. Assumimos que $x^{i_{\nu-1}} = x^{i_n}$ para $\nu = 1$ e $x^{i_{\nu+1}} = x^{i_1}$ para $\nu = n$. Se o segmento $[x^{i_{\nu-1}}, x^{i_{\nu+1}}]$ intersecta $\mathcal{B}_{i_\nu}$, o ponto x^{i_ν} recebe a projeção ortogonal do centro c^{i_ν} sobre

DOI: 10.5540/03.2025.011.01.0495

$[x^{i_{\nu-1}}, x^{i_{\nu+1}}]$. Caso contrário, o minimizador de φ pertencerá à fronteira de $\mathcal{B}_{i_\nu}$, de modo que x^{i_ν} pode ser parametrizado em função do centro c^{i_ν} e do raio r^{i_ν} de $\mathcal{B}_{i_\nu}$ e reescrito da forma $x^{i_\nu} = (c_1^{i_\nu} + r^{i_\nu}\cos\theta, c_2^{i_\nu} + r^{i_\nu}\sin\theta)$, com $\theta \in [0, 2\pi]$. Neste caso, o método de Newton globalizado é empregado para minimizar $\psi(\theta) := \varphi(c_1^{i_\nu} + r^{i_\nu}\cos\theta, c_2^{i_\nu} + r^{i_\nu}\sin\theta)$.

Cada vez que todos os pontos x^{i_ν} são atualizados, diz-se que um ciclo do método BCDM foi realizado. A contagem dos ciclos é feita utilizando o parâmetro ℓ. O Algoritmo 1 recebe como dados de entrada os parâmetros utilizados pelo método de Newton ($\alpha \in (0,1)$, $\beta > 0$, $0 < \sigma_1 \leq \sigma_2 < 1$, $\epsilon^{\mathrm{N}} > 0$ e $\ell^{\mathrm{N}}_{\max} \geq 0$), o número de bolas m, os vetores c e r contendo respectivamente os centros e raios das bolas, os parâmetros $\epsilon^{\mathrm{BC}} \in \mathbb{R}$ e $\ell^{\mathrm{BC}}_{\max}$ (número máximo de ciclos) que serão utilizados como critérios de parada, além de uma permutação T e um chute inicial x^0 para os pontos visitados.

Algoritmo 1: Método de busca em coordenadas para otimizar os pontos de uma rota.

Entrada: $\alpha \in (0,1)$, $\beta > 0$, $0 < \sigma_1 \leq \sigma_2 < 1$, $\epsilon^{\mathrm{N}} > 0$, $\ell^{\mathrm{N}}_{\max} \geq 0$, m, $c = (c_1, \ldots, c_m)$, $r = (r_1, \ldots, r_m)$, ϵ^{BC}, $\ell^{\mathrm{BC}}_{\max}$, $\theta = (\theta_1, \ldots, \theta_m)$, T, $x^0 = (x_1^0, \ldots, x_m^0)$.

Saída: $x^* = (x_1^*, \ldots, x_m^*)$.

Função BuscaEmCoordenadas(α, β, σ_1, σ_2, ϵ^{N}, $\ell^{\mathrm{N}}_{\max}$, m, c, r, ϵ^{BC}, $\ell^{\mathrm{BC}}_{\max}$, θ, T, x^0, x^*)
- $\ell \leftarrow 0$
- **faça**
  - **para** $\boldsymbol{k \leftarrow 1}$ **até** $\boldsymbol{m}$ **faça**
    - $a \leftarrow$ ponto anterior a $x^{\ell}_{T_k}$ no tour
    - $b \leftarrow$ ponto posterior a $x^{\ell}_{T_k}$ no tour
    - $r \leftarrow r_{T_k}$ e $c \leftarrow c_{T_k}$
    - ProjecaoEmAB(a, b, c, c^P)
    - **se** $\boldsymbol{\|c - c^P\|_2 \leq r}$ **então**
      - $x^{\ell+1}_{T_k} \leftarrow c^P$
    - **senão**
      - $\theta_0 \leftarrow \theta_{T_k}$
      - Newton(α, β, σ_1, σ_2, ϵ^{N}, $\ell^{\mathrm{N}}_{\max}$, θ_0, θ^*)
      - $\theta_{T_k} \leftarrow \theta^*$
      - $x^{\ell+1}_{T_k} \leftarrow c + r \begin{pmatrix} \cos\theta^* \\ \sin\theta^* \end{pmatrix}$
  - $\delta \leftarrow |f(T, x^{\ell}) - f(T, x^{\ell-1})|$
  - $\ell \leftarrow \ell + 1$
- **enquanto** $\boldsymbol{\delta > \epsilon^{\mathrm{BC}}}$ **e** $\boldsymbol{\ell < \ell^{\mathrm{BC}}_{\max}}$
- $x^* := x^{\ell}$
- **retorne** x^*

O Algoritmo 2 - 2optBest, por sua vez, descreve a heurística de busca local que utiliza movimentos do tipo 2-Opt para encontrar soluções para o CETSP. Este método, que utiliza a estratégia do *melhor vizinho*, recebe um tour inicial (T^0, x^0), avalia o comprimento de todos os seus vizinhos 2-Opt com os pontos otimizados e o atualiza com o tour que apresentar o menor comprimento, obtendo assim o tour (T^1, x^1). Este processo é repetido de maneira iterativa, até obter um tour (T^{ℓ}, x^{ℓ}) para o qual não existem vizinhos 2-Opt de comprimento menor que o de (T^{ℓ}, x^{ℓ}) ou até atingir um número máximo de iterações $\ell^{\mathrm{2opt}}_{\max}$. Além dos parâmetros que são utilizados pelo método de busca em coordenadas, o Algoritmo 2 recebe como dados de entrada uma permutação inicial T^0 e um chute inicial x^0 para os pontos visitados.

A seção seguinte é dedicada aos experimentos numéricos.

DOI: 10.5540/03.2025.011.01.0495

Algoritmo 2: Busca local que utiliza movimentos do tipo 2-opt e o método de busca em coordenadas para construir a vizinhança.

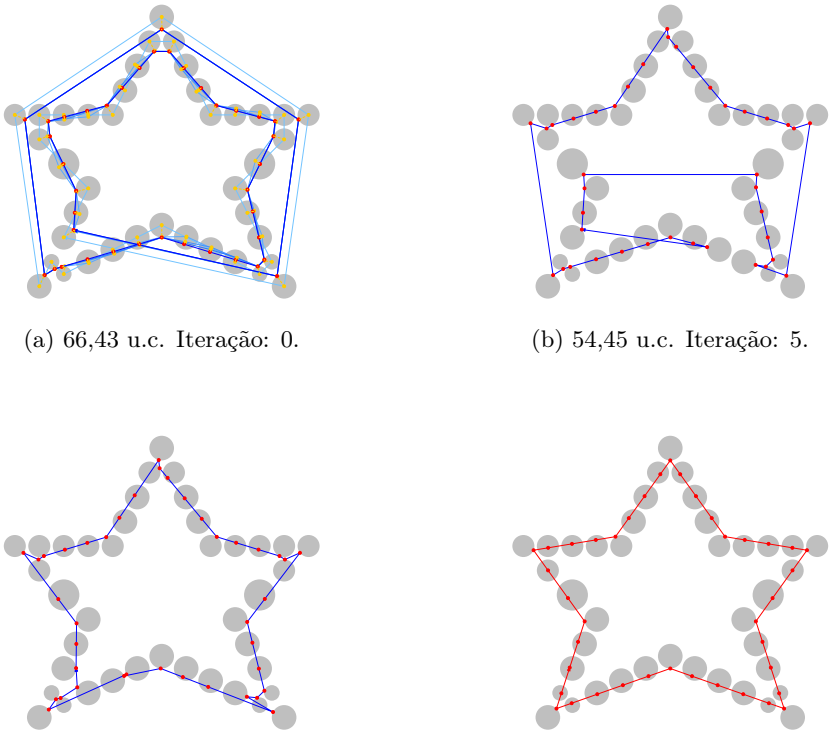
Entrada: $\ell^{\text{2opt}}_{\max} \geq 0$, T^0, $x^0 = (x^0_1, \ldots, x^0_m)$.
Saída: T^*, $x^* = (x^*_1, \ldots, x^*_m)$.
Função 2optBest(α, β, σ_1, σ_2, ϵ^{N}, $\ell^{\text{N}}_{\max}$, ϵ^{BC}, $\ell^{\text{BC}}_{\max}$, $\ell^{\text{2opt}}_{\max}$, m, c, r, T^0, x^0, θ, T^*, x^*)
  $\ell \leftarrow 0$
  faça
    $T^{\text{best}} \leftarrow T^\ell$ e $x^{\text{best}} \leftarrow x^\ell$
    para $i \leftarrow 1$ até $(m-2)$ faça
      para $j \leftarrow i+2$ até m faça
        $T^{\text{2opt}}_k \leftarrow T^\ell_k$ para $k \in \{1, \ldots, i\} \cup \{j+1, \ldots, m\}$
        $T^{\text{2opt}}_k \leftarrow T^\ell_{j-i+1-k}$ para $k \in \{i+1, \ldots, j\}$
        BuscaEmCoordenadas(α, β, σ_1, σ_2, ϵ^{N}, $\ell^{\text{N}}_{\max}$, m, c, r, ϵ^{BC}, $\ell^{\text{BC}}_{\max}$, θ, T^{2opt}, x^ℓ, x^{2opt})
        se $f(T^{\text{2opt}}, x^{\text{2opt}}) < f(T^{\text{best}}, x^{\text{best}})$ então
          $T^{\text{best}} \leftarrow T^{\text{2opt}}$ e $x^{\text{best}} \leftarrow x^{\text{2opt}}$
    $\delta \leftarrow f(T^\ell, x^\ell) - f(T^{\text{best}}, x^{\text{best}})$
    se $\delta > 0$ então
      $T^{\ell+1} := T^{\text{best}}$ e $x^{\ell+1} := x^{\text{best}}$
    $\ell \leftarrow \ell + 1$
  enquanto $\delta > 0$ e $\ell < \ell^{\text{2opt}}_{\max}$
  $T^* := T^\ell$ e $x^* := x^\ell$
  retorne T^* e x^*

4 Experimentos numéricos

Na implementação dos algoritmos utilizou-se a linguagem de programação `Fortran 90` e como compilador `GFortran`. Os experimentos foram realizados em um computador com processador Intel Core i7-8700 CPU @ 3.20GHz e 16GB de RAM. As figuras foram geradas com `MetaPost`. Nos experimentos realizados, cada instância consiste num conjunto de m pontos $c^i \in \mathbb{R}^2$, que correspondem aos centros das bolas $\mathcal{B}_i$. Os raios r^i foram determinados de modo que as bolas tivessem interseção vazia através da fórmula $r^i := \alpha\left(\frac{d_i}{2}\right)$, em que d_i corresponde à menor distância Euclidiana entre c^i e os demais centros e $\alpha \in (0{,}0, 0{,}9)$.

A Figura 1 ilustra a atuação dos métodos propostos na seção anterior em uma instância com 40 bolas. A construção do tour inicial (T^0, x^0), desenhado na cor azul escura com os pontos em vermelho, está detalhada na Figura 1a. A ordenação T^0 é construída pela heurística NN e os pontos x^0 são gerados pelo Algoritmo 1 a partir dos centros c^i. Neste caso, os pontos em amarelo calculados a cada ciclo foram condensados em uma mesma figura para possibilitar a visualização de suas trajetórias convergindo até os pontos em vermelho. As linhas em azul, por sua vez, denotam a rota obtida em cada ciclo do Algoritmo 1 ao se percorrer os pontos em amarelo com a ordem dada por T^0. A tonalidade dessas rotas se torna mais escura conforme se aproxima do tour inicial (T^0, x^0). As demais imagens da Figura 1 contêm alguns tours gerados durante o processo iterativo do Algoritmo 2 - 2optBest. As Figuras 1b e 1c apresentam os tours obtidos nas iterações 5 e 10, respectivamente, enquanto que a Figura 1d corresponde à solução encontrada pelo algoritmo na iteração 20. O comprimento de cada tour é exibido abaixo das respectivas figuras.

DOI: 10.5540/03.2025.011.01.0495



(a) 66,43 u.c. Iteração: 0.

(b) 54,45 u.c. Iteração: 5.

(c) 41,30 u.c. Iteração: 10.

(d) 38,95 u.c. Iteração: 20.

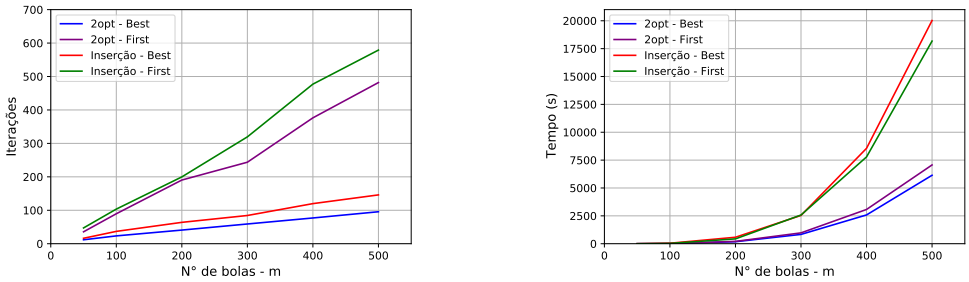
Figura 1: Amostra de tours obtidos pelo Algoritmo 2 - 2optBest para uma instância de 40 bolas. Fonte: dos autores.

O Algoritmo 2 - 2optBest também foi testado em 60 instâncias geradas com pontos aleatórios, sendo 10 instâncias para cada número de bolas $m \in \{50, 100, 200, 300, 400, 500\}$. No intuito de avaliar seu desempenho, outros três algoritmos foram implementados para compará-los: 2optFirst - que também utiliza a heurística 2-opt, porém a atualização dos iterandos segue a estratégia do *primeiro vizinho* que melhora; InsercaoBest e InsercaoFirst - que utilizam movimentos de realocação para criar vizinhanças. Em relação a atuação desses algoritmos nos experimentos com instâncias aleatórias, os gráficos da Figura 2 apresentam, respectivamente, o tempo de execução e total de iterações em função do número de bolas.

O Algoritmo 2 - 2optBest consegue obter, em média, soluções de melhor qualidade, i.e., menor comprimento, e em menos iterações. Este método também encontra estas soluções mais rapidamente que os demais métodos testados. Apesar de o número de iterações em função do número de bolas apresentar um crescimento linear para todos os métodos, o mesmo não é observado no gráfico da Figura 2b, no qual o tempo de execução em função do número de bolas apresenta um comportamento quadrático. Uma das razões para os métodos 2-Opt obterem soluções melhores está no fato destes métodos desfazerem as arestas que geram sobreposições na rota do tour, característica que não é observada nos métodos de inserção.

Por fim, o Algoritmo 2 - 2optBest foi testado em 10 instâncias Euclidianas do $\mathbb{R}^2$ da biblioteca TSPLIB [7]. Tais instâncias foram adaptadas para a formulação dada no problema (1) da seguinte forma: os pontos fornecidos pelas instâncias foram definidos como sendo os centros das bolas $\mathcal{B}_i$ e os raios foram calculados utilizando a mesma estratégia dos experimentos anteriores com $\alpha = 0{,}9$. Como exemplo a Figura 3 apresenta as respectivas soluções encontradas para as instâncias *ch150*,

DOI: 10.5540/03.2025.011.01.0495

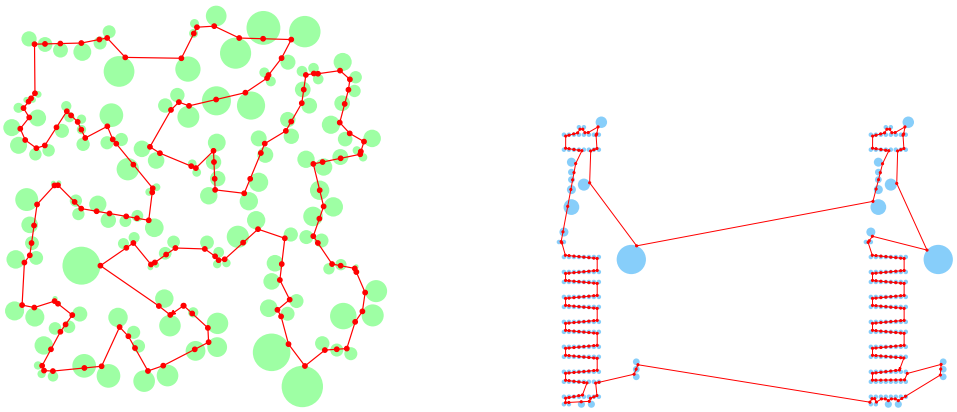


(a) Iterações em função do número de bolas.

(b) Tempo em função do número de bolas.

Figura 2: Gráficos das iterações e do tempo, respectivamente, em função do número de bolas. Fonte: dos autores.

de 150 bolas, e *pr264*, com 264 bolas.



(a) Solução com 2optBest: 4.986,02 u.c. Iteração: 21. Tempo de execução: 56,21s.

(b) Solução com 2optBest: 45.340,69 u.c. Iteração: 36. Tempo de execução: 1.160,43s.

Figura 3: Soluções obtidas pelo Algoritmo 2 - 2optBest para as instâncias *ch150* e *pr264*. Fonte: dos autores.

Os experimentos realizados com as instâncias aleatórias e da biblioteca TSPLIB demonstram que o Algoritmo 2 - 2optBest encontra soluções melhores quando comparado com os outros três algoritmos, e na maioria dos casos em menor tempo. O algoritmo 2optFirst se sobressai sobre o 2optBest no quesito tempo de execução apenas para valores menores de m. Conforme aumentamos a quantidade de bolas do problema, o algoritmo 2optFirst tende a realizar um grande número de soluções até parar, consumindo mais tempo de processamento.

5 Considerações finais

Neste trabalho apresentamos uma formulação para o CETSP e propomos um método heurístico para resolver numericamente o caso em que os conjuntos visitados são bolas disjuntas do $\mathbb{R}^2$. Experimentos numéricos demonstram que os algoritmos que utilizam movimentos do tipo 2-Opt

se sobressaem, na maioria dos exemplos, aos métodos que usam movimentos de realocação em relação a qualidade de soluções e tempo de execução. Outro detalhe observado é que, apesar dos métodos com a estratégia do *melhor vizinho* possuírem iterações mais custosas - pois avaliam todos os vizinhos antes de definir o próximo iterando - eles encontram na maioria dos testes soluções melhores que os algoritmos que utilizam a estratégia do *primeiro vizinho* que melhora. Também notamos que o número de iterações tende a aumentar conforme o número de bolas cresce. Isso foi observado tanto nos experimentos com instâncias aleatórias quanto nos testes com instâncias da biblioteca TSPLIB. O principal diferencial do método proposto neste trabalho em relação aos demais presentes na literatura está no uso de um método de busca em bloco de coordenadas para resolver os subproblemas de calcular os pontos visitados. Algumas possibilidades para trabalhos futuros são: adaptar o método para permitir que a intersecção entre as bolas seja não-vazia e testá-lo em instâncias disponíveis na literatura [1, 4]; e testar uma modelagem baseada em programação cônica de segunda ordem disponível na literatura [3] para o subproblema de determinar os pontos visitados e compará-la com o BCDM.

Agradecimentos

O presente trabalho foi realizado com apoio da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior – Brasil (CAPES).

Referências

[1] B. Behdani e J. C. Smith. "An Integer-Programming-Based Approach to the Close-Enough Traveling Salesman Problem". Em: **INFORMS Journal on Computing** 26 (2014), pp. 415–432.

[2] E. G. Birgin e J. M. Martínez. "Block Coordinate Descent for smooth nonconvex constrained minimization". Em: **Computational Optimization and Applications** 83 (2021), pp. 1–27.

[3] W. P. Coutinho, R. Q. do Nascimento, A. A. Pessoa e A. Subramanian. "A Branch-and-Bound Algorithm for the Close-Enough Traveling Salesman Problem". Em: **INFORMS Journal on Computing** 28.4 (2016), pp. 752–765.

[4] W. K. Mennell. "Heuristics for Solving Three Routing Problems: Close-Enough Traveling Salesman Problem, Close-Enough Vehicle Routing Problem, Sequence-Dependent Team Orienteering Problem". Tese de doutorado. University of Maryland (College Park, Md.), 2009.

[5] A. D. Placido, C. Archetti e C. Cerrone. "A genetic algorithm for the close-enough traveling salesman problem with application to solar panels diagnostic reconnaissance". Em: **Computers & Operations Research** 145 (2022), pp. 105–831.

[6] S. Poikonen, X. Wang e B. Golden. "The vehicle routing problem with drones: Extended models and connections". Em: **Networks** 70.1 (2017), pp. 34–43.

[7] G. Reinelt. **TSPLIB - A traveling salesman problem library**. Online. Acessado em 05/01/2023, `http://comopt.ifi.uni-heidelberg.de/`. 1991.

[8] R. Shuttleworth, B. L. Golden, S. Smith e E. Wasil. "Advances in Meter Reading: Heuristic Solution of the Close Enough Traveling Salesman Problem over a Street Network". Em: **The Vehicle Routing Problem: Latest Advances and New Challenges**. Ed. por B. Golden, S. Raghavan e E. Wasil. Boston, MA: Springer US, 2008, pp. 487–501. ISBN: 978-0-387-77778-8.

[9] S. J. Wright. "Coordinate descent algorithms". Em: **Mathematical Programming** 151 (2015), pp. 3–34.

