# ALNS com Simulated Annealing e Shortest Path Heuristic para o Problema da Árvore de Steiner Clusterizada

**João Guilherme Guimarães e Silva**[1] **, Alexandre Ribeiro**[1]

[1]Pontifícia Universidade Católica de Goiás (PUC Goiás)
Goiânia – GO – Brasil

joaoguilherme.gsilva.pro@gmail.com, alexribeiro@pucgoias.edu.br

***Abstract.*** *This paper addresses the Clustered Steiner Tree Problem (CluSteiner), in which terminals are partitioned into clusters and local trees of different clusters must be vertex-disjoint. It proposes an Adaptive Large Neighborhood Search (ALNS) metaheuristic with Simulated Annealing (SA) acceptance and shortest-path guided repair procedures. The method combines destroy/repair operators with explicit feasibility checking to preserve the disjointness constraint. Computational results on non-Euclidean instances indicate improvements over reference methods.*

***Resumo.*** *Este trabalho aborda o Problema da Árvore de Steiner Clusterizada (CluSteiner), em que os terminais são particionados em clusters e as árvores locais de clusters distintos devem ser disjuntas em vértices. Propõe-se uma meta-heurística de Adaptive Large Neighborhood Search (ALNS) com aceitação por Simulated Annealing (SA) e procedimentos de reparo guiados por caminhos mínimos. A abordagem combina operadores de destruição e reconstrução com verificação explícita de viabilidade para preservar a restrição de disjunção. Os resultados computacionais em instâncias não euclidianas indicam melhorias frente a métodos de referência.*

## 1. Introdução

Problemas de projeto de redes frequentemente exigem conectar conjuntos de vértices com o menor custo possível. Em aplicações desse tipo, os vértices de interesse podem estar organizados em grupos que precisam manter separação estrutural entre si. Esse cenário é modelado pelo Problema da Árvore de Steiner Clusterizada (*Clustered Steiner Tree Problem* – CluSteiner), uma variação do Problema da Árvore de Steiner em Grafos (STP), introduzido originalmente por [Hakimi 1971], em que os terminais são particionados em clusters [Wu and Lin 2015].

No CluSteiner, o objetivo é encontrar uma árvore de custo mínimo que conecte todos os terminais, permitindo o uso de vértices de Steiner, mas garantindo que as árvores locais de clusters distintos sejam disjuntas em vértices. Além da restrição de disjunção entre clusters, o CluSteiner herda a dificuldade computacional do STP clássico. De fato, o problema é NP-difícil, pois o STP pode ser obtido como um caso particular do CluSteiner quando cada cluster é formado por um único terminal [Garey and Johnson 1979].

Neste trabalho, propõe-se uma meta-heurística baseada em *Adaptive Large Neighborhood Search* (ALNS) com aceitação por *Simulated Annealing* (SA), utilizando a *Shortest Path Heuristic* (SPH) em etapas de construção e reparo. A abordagem combina operadores de destruição e reconstrução com verificação explícita de viabilidade. Além disso, o

método é avaliado em instâncias não euclidianas da literatura recente [Long et al. 2024], permitindo comparação direta com métodos de referência.

## 2. Definição do Problema e Trabalhos Relacionados

### 2.1. Definição formal do CluSteiner

Seja $G = (V, E, w)$ um grafo não direcionado ponderado, em que $V$ é o conjunto de vértices, $E$ é o conjunto de arestas e $w : E \to \mathbb{R}_{\geq 0}$ associa um custo não negativo a cada aresta. Seja $R \subseteq V$ o conjunto de terminais, particionado em $h$ clusters não vazios $\mathcal{C} = \{R_1, \ldots, R_h\}$, tais que $\bigcup_{i=1}^{h} R_i = R$ e $R_i \cap R_j = \varnothing$ para todo $i \neq j$. Os vértices de $V \setminus R$ são chamados de vértices de Steiner e podem ser incluídos na solução para reduzir seu custo total.

Uma solução para o CluSteiner é uma árvore de Steiner $T = (V_T, E_T)$, com $R \subseteq V_T \subseteq V$ e $E_T \subseteq E$, que minimiza o custo total

$$\sum_{e \in E_T} w(e).$$

Para cada cluster $R_i$, define-se sua árvore local $T_i$ como a menor subárvore contida em $T$ que conecta todos os terminais de $R_i$, podendo incluir vértices de Steiner necessários para essa conexão. A restrição característica do problema impõe que as árvores locais de clusters distintos sejam disjuntas em vértices, isto é,

$$V(T_i) \cap V(T_j) = \varnothing, \qquad \forall\, 1 \leq i < j \leq h. \tag{1}$$

### 2.2. Trabalhos relacionados

O CluSteiner foi formalizado por Wu e Lin [Wu and Lin 2015], que introduzem a restrição de disjunção entre as árvores locais dos clusters. Diferentemente do STP clássico, o problema exige não apenas baixo custo, mas também separação estrutural entre clusters.

Heurísticas clássicas para STP, como a *Shortest Path Heuristic* (SPH) [Takahashi and Matsuyama 1980] e métodos baseados em MST [Kou et al. 1981], podem ser utilizadas na construção de soluções, mas não consideram de forma explícita a restrição de disjunção do problema.

Mais recentemente, Long *et al.* [Long et al. 2024] avaliaram métodos para instâncias não euclidianas, incluindo SPMST, SP-GA, SP-MFEA e SP-MFEA-II, sendo este último a principal referência experimental.

## 3. Heurística Proposta

### 3.1. Representação em dois níveis

A solução é representada por um conjunto de arestas que forma uma árvore viável. Para operar a busca, mantém-se uma decomposição em dois níveis: (1) arestas locais, pertencentes às árvores locais de cada cluster; e (2) arestas globais, responsáveis por conectar as estruturas locais. Essa organização permite perturbar a conectividade global com menor risco de destruir a estrutura interna de todos os clusters a cada iteração.

### 3.2. Solução inicial

A solução inicial é construída por um decodificador em duas camadas (R8), inspirado no procedimento de dois níveis utilizado em [Long et al. 2024]: (i) para cada cluster, constrói-se uma árvore local por SPH [Takahashi and Matsuyama 1980] em um subgrafo permitido (terminais do cluster mais um *pool* de vértices de Steiner disponíveis); e (ii) conectam-se as árvores locais por uma SPH global, usando apenas vértices de Steiner remanescentes (não utilizados nas árvores locais), o que reduz a chance de violar (1). Ao final, aplica-se uma finalização por Kruskal [Kruskal 1956].

Para contornar estagnações, um *reset* é acionado quando o algoritmo permanece sem melhorar a melhor solução por um número mínimo de iterações, respeitando um tempo de recarga e uma probabilidade; quando acionado, ele reconstrói a solução por R8 e o candidato é aceito (se viável), seguido de reaquecimento da temperatura para ao menos $T_0$.

### 3.3. Operadores *destroy/repair*

Na estrutura do ALNS, cada iteração seleciona um operador *destroy* e um *repair*. Os *destroys* incluem: remoção aleatória de $k$ arestas globais, desconexão de um cluster, quebra completa de árvore local, remoção das $k$ piores arestas, remoção de segmento no caminho global e remoção em estrela de um vértice Steiner de alto grau (atuando como um *hub* central). Os *repairs* incluem: reconexão gulosa por Dijkstra multi-fonte [Dijkstra 1959], MST entre componentes via caminhos mínimos [Prim 1957, Kruskal 1956], reconexão usando um ou dois *hubs* Steiner, pontes entre componentes dispersos e reconstrução por SPH.

Quando um *destroy* remove a árvore local de um cluster, antes do reparo global aplica-se um reparo local (R0) por SPH com vértices restritos: proíbem-se vértices alocados a outros clusters e excluem-se Steiners já presentes na estrutura global corrente.

### 3.4. Aceitação por SA, pesos adaptativos e reaquecimento

O critério de aceitação segue o SA: melhorias são sempre aceitas; pioras $\Delta > 0$ são aceitas com probabilidade $\exp(-\Delta/T)$ [Kirkpatrick et al. 1983]. A temperatura inicial $T_0$ é estimada automaticamente por amostragem de pioras típicas e calibrada para uma probabilidade-alvo $p_0$ [Ben-Ameur 2004]; o resfriamento é geométrico $T \leftarrow \alpha T$, com reaquecimento periódico após estagnação. No modo adaptativo, escolhem-se operadores por roleta e atualizam-se pesos em segmentos conforme a prática de ALNS [Ropke and Pisinger 2006, Pisinger and Ropke 2010].

## 4. Experimentos Computacionais e Resultados

### 4.1. Instâncias e métricas

Avaliam-se exclusivamente instâncias não euclidianas do benchmark adotado em [Long et al. 2024], com 30 execuções independentes por instância. Para garantir comparabilidade direta com o SP-MFEA-II (método de referência), adota-se o mesmo método de busca reportado pelos autores: 25 000 soluções avaliadas por execução [Long et al. 2024].

Além disso, cada execução termina ao atingir 25 000 avaliações ou 30 minutos, o que ocorrer primeiro. Observou-se que, frequentemente, a busca converge antes de completar os 30 minutos, especialmente nas instâncias de maior porte.

Reportam-se as métricas *Best Found* (BF), *Average* (AVG) e tempo médio por *run* (em minutos). Para comparações agregadas, emprega-se o *Relative Percentage Difference* (RPD) a partir do custo médio:

$$\mathrm{RPD}(i) = \frac{\mathrm{AVG}(i) - \mathrm{BKS}(i)}{\mathrm{BKS}(i)} \cdot 100\%, \tag{2}$$

onde

$$\mathrm{BKS}(i) = \min\{\mathrm{BF}_{\text{SPMST}}(i), \mathrm{BF}_{\text{SP-GA}}(i), \mathrm{BF}_{\text{SP-MFEA-II}}(i), \mathrm{BF}_{\text{SP-ALNS-SA}}(i)\}.$$

Para medir ganho direto sobre o SP-MFEA-II, define-se o *Improvement Percentage* (PI):

$$\mathrm{PI}_{\text{SP-ALNS-SA/MFEA-II}}(i) = \frac{\mathrm{AVG}_{\text{MFEA-II}}(i) - \mathrm{AVG}_{\text{SP-ALNS-SA}}(i)}{\mathrm{AVG}_{\text{MFEA-II}}(i)} \cdot 100\%. \tag{3}$$

### 4.2. Resultados

A Tabela 1 apresenta um subconjunto representativo das execuções em instâncias de diferentes topologias, comparando os algoritmos apresentados no trabalho de [Long et al. 2024] com o método proposto (SP-ALNS-SA). O código-fonte completo da implementação, bem como os resultados detalhados contendo a tabela integral para todas as instâncias, estão disponíveis publicamente no repositório do projeto[1].

Considerando as 100 instâncias em comum com SPMST, SP-GA e SP-MFEA-II, o SP-ALNS-SA apresenta $\mathrm{AVG}$ menor que o método de referência SP-MFEA-II em 96 instâncias, empata em 1 e é superado em 3. O PI médio do SP-ALNS-SA em relação ao SP-MFEA-II, calculado por (3), é de $19.11\%$ em todas as instâncias da tabela integral publicada no repositório[1].

Com a definição de BKS acima, o SP-ALNS-SA obtém $\mathrm{RPD}$ médio de $3.33\%$, enquanto o SP-MFEA-II apresenta $\mathrm{RPD}$ médio de $37.37\%$.

**Tabela 1. Resultados de SPMST, SP-GA, SP-MFEA-II e SP-ALNS-SA.**

| | | SPMST | | | SP-GA | | | SP-MFEA-II | | | SP-ALNS-SA | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | **Instância** | **BF** | **AVG** | **Tempo** | **BF** | **AVG** | **Tempo** | **BF** | **AVG** | **Tempo** | **BF** | **AVG** | **Tempo** |
| Type 1 Small | 15eil51 | 3004.00 | 3004.00 | 0.01 | 886.00 | 966.61 | 0.35 | 886.00 | 947.65 | 0.82 | 886.00 | **888.20** | 0.87 |
| | 15st70 | 4401.00 | 4401.00 | 0.01 | 1113.00 | 1292.90 | 0.57 | 1113.00 | 1198.13 | 0.94 | 1071.00 | **1079.30** | 1.74 |
| | 50kroA100 | 3390.00 | 3390.00 | 0.01 | 1577.00 | 1714.58 | 1.55 | 1515.00 | 1614.29 | 2.71 | 1347.00 | **1359.10** | 4.55 |
| Type 1 Large | 10lin318 | 8738.00 | 8738.00 | 0.01 | 1004.00 | 1102.77 | 4.94 | 874.00 | 972.16 | 16.79 | 717.00 | **743.60** | 30.00 |
| | 25a280 | 15851.00 | 15851.00 | 0.02 | 2107.00 | 2350.35 | 4.17 | 1813.00 | 2004.55 | 13.03 | 1052.00 | **1123.87** | 30.00 |
| | 50pcb442 | 23602.00 | 23602.00 | 0.06 | 3623.00 | 4142.00 | 9.29 | 3467.00 | 3793.61 | 15.41 | 1303.00 | **1367.03** | 30.01 |
| Type 5 Small | 10i75-22 | 3870.00 | 3870.00 | 0.01 | 854.00 | 912.23 | 0.48 | 768.00 | 835.19 | 0.75 | 778.00 | **795.03** | 1.89 |
| | 7i65-21 | 4079.00 | 4079.00 | 0.01 | 991.00 | 1101.19 | 0.35 | 989.00 | 1054.10 | 0.87 | 989.00 | **990.60** | 1.20 |
| | 7i60-21 | 2678.00 | 2678.00 | 0.01 | 636.00 | 664.77 | 0.33 | 630.00 | 644.19 | 0.87 | 612.00 | **616.03** | 1.00 |
| Type 5 Large | 15i400-207 | 14799.00 | 14799.00 | 0.01 | 1594.00 | 1836.84 | 7.72 | 1498.00 | 1643.00 | 21.34 | 928.00 | **992.90** | 30.00 |
| | 5i400-205 | 5554.00 | 5554.00 | 0.01 | 712.00 | 761.77 | 8.23 | 648.00 | 702.97 | 25.87 | 565.00 | **588.07** | 30.00 |
| | 20i500-307 | 14909.00 | 14909.00 | 0.01 | 1871.00 | 2150.55 | 9.44 | 1729.00 | 1959.35 | 12.42 | 849.00 | **931.50** | 30.01 |
| Type 6 Small | 25eil101-5x5 | 5425.00 | 5425.00 | 0.01 | 1547.00 | 1728.35 | 1.09 | 1402.00 | 1502.90 | 1.73 | 1320.00 | **1348.67** | 4.06 |
| | 35kroB100-5x5 | 5778.00 | 5778.00 | 0.01 | 1368.00 | 1536.68 | 1.01 | 1264.00 | 1391.16 | 2.44 | 1179.00 | **1188.43** | 4.29 |
| | 16eil51-4x4 | 2313.00 | 2313.00 | 0.01 | 556.00 | 649.94 | 0.33 | 556.00 | 598.84 | 1.04 | 555.00 | **557.70** | 0.87 |
| Type 6 Large | 25pcb442-5x5 | 16728.00 | 16728.00 | 0.03 | 2070.00 | 2416.71 | 8.56 | 1896.00 | 2229.42 | 13.76 | 958.00 | **1040.80** | 30.00 |
| | 36pcb442-6x6 | 17437.00 | 17437.00 | 0.03 | 2971.00 | 3284.45 | 9.02 | 2845.00 | 3046.23 | 13.76 | 1199.00 | **1285.13** | 30.01 |
| | 9pcb442-3x3 | 8612.00 | 8612.00 | 0.01 | 884.00 | 987.61 | 9.01 | 829.00 | 900.77 | 12.74 | 590.00 | **627.63** | 30.00 |

[1] `https://github.com/Jotinha7/clusteiner-alns-sa`

## Referências

Ben-Ameur, W. (2004). Computing the initial temperature of simulated annealing. *Computational Optimization and Applications*, 29:369–385.

Dijkstra, E. W. (1959). A note on two problems in connexion with graphs. *Numerische Mathematik*, 1:269–271.

Garey, M. R. and Johnson, D. S. (1979). *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman.

Hakimi, S. L. (1971). An algorithmic approach to the steiner problem in graphs. *Networks*, 1(2):113–133.

Kirkpatrick, S., Gelatt, C. D., and Vecchi, M. P. (1983). Optimization by simulated annealing. *Science*, 220(4598):671–680.

Kou, L., Markowsky, G., and Berman, L. (1981). A fast algorithm for steiner trees. *Acta Informatica*, 15(2):141–145.

Kruskal, J. B. (1956). On the shortest spanning subtree of a graph and the traveling salesman problem. *Proceedings of the American Mathematical Society*, 7(1):48–50.

Long, N. B., Anh, D. T., Ban, H., and Binh, H. T. T. (2024). An online transfer learning based multifactorial evolutionary algorithm for solving the clustered Steiner tree problem. *Knowledge-Based Systems*, 296:111870.

Pisinger, D. and Ropke, S. (2010). Large neighborhood search. In Gendreau, M. and Potvin, J., editors, *Handbook of Metaheuristics*, volume 146 of *International Series in Operations Research & Management Science*, pages 399–419. Springer.

Prim, R. C. (1957). Shortest connection networks and some generalizations. *Bell System Technical Journal*, 36(6):1389–1401.

Ropke, S. and Pisinger, D. (2006). An adaptive large neighborhood search heuristic for the pickup and delivery problem with time windows. *Transportation Science*, 40(4):455–472.

Takahashi, H. and Matsuyama, A. (1980). An approximate solution for the steiner problem in graphs. *Mathematica Japonica*, 24:573–577.

Wu, B. Y. and Lin, C. (2015). On the clustered Steiner tree problem. *Journal of Combinatorial Optimization*, 30(2):370–386.