Clustering-LNS: Hybrid Methodology for the Single Row Facility Layout Problem

Carlos V. Dantas Araújo[1], **Jailon W. B. Oliveira da Silva**[1]
Pablo L. Braga Soares[1]

[1]Laboratório de Pesquisa & Desenvolvimento do NEMO
Universidade Federal do Ceará – Campus Russas
Avenida Felipe Santiago – Nº 411 62.900-624, Russas/CE, Brasil

{carlosvinicio, williambrunocc}@alu.ufc.br, pablo.soares@ufc.br

Abstract. *This paper introduces a hybrid methodology for the Single Row Facility Layout Problem (SRFLP), combining unsupervised learning with the Large Neighborhood Search (LNS) metaheuristic. The proposed approach uses clustering algorithms to analyze the problem's cost matrix, generating a high-quality initial solution that guides the LNS. The method was tested on 128 benchmark instances, demonstrating high performance. For classical instances ($n \leq 110$), the proposed method consistently found solutions matching the state-of-the-art quality. In large-scale instances ($n \leq 500$), it achieved the best-known solutions in approximately 40% of cases, with an average gap below 1% in the remaining instances.*

Resumo. *Este trabalho introduz uma metodologia híbrida para o Problema de Arranjo Físico em Linha Única (SRFLP), combinando aprendizado não supervisionado com a meta-heurística de Busca em Vizinhança Ampla (LNS). A abordagem proposta utiliza algoritmos de clusterização para analisar a matriz de custos do problema, gerando uma solução inicial de alta qualidade que guia a LNS. O método foi testado em 128 instâncias de benchmark, demonstrando alto desempenho. Para instâncias clássicas ($n \leq 110$), o método proposto encontrou consistentemente soluções de qualidade igual ao estado da arte. Em instâncias de grande escala ($n \leq 500$), alcançou as melhores soluções conhecidas em aproximadamente 40% dos casos, com um gap médio inferior a 1% nos demais.*

1. Introdução

O Problema de Arranjo Físico em Linha Única, do inglês Single Row Facility Layout Problem (SRFLP) consiste em determinar a permutação ótima de um conjunto de instalações de diferentes comprimentos ao longo de uma linha, de modo a minimizar o custo total de interação entre elas. Classificado como NP-difícil [Garey and Johnson 1979], o SRFLP apresenta desafios computacionais significativos, especialmente para instâncias de grande porte, onde métodos exatos se tornam impraticáveis devido ao crescimento fatorial ($n!$) do espaço de busca [Kothari and Ghosh 2014a]. Esta complexidade inerente motiva o desenvolvimento de métodos aproximados, como heurísticas e meta-heurísticas, que buscam encontrar soluções de alta qualidade em tempo computacional viável [Blum and Roli 2003].

A relevância do SRFLP é vasta, com aplicações práticas que vão desde o design de sistemas de manufatura e o arranjo de máquinas em uma linha de produção até a organização de livros em prateleiras e o armazenamento de itens em depósitos [Anjos et al. 2005]. Contudo, as meta-heurísticas de ponta, como GRASP [Rubio-Sánchez et al. 2015] e Algoritmos Genéticos [Kothari and Ghosh 2014b], embora eficazes, frequentemente dependem de uma calibração sensível de parâmetros e podem convergir para ótimos locais, especialmente em paisagens de busca complexas. Ademais, muitas vezes partem de soluções iniciais aleatórias, o que pode prolongar o tempo de convergência.

Nesse contexto, o aprendizado de máquina não supervisionado emerge como uma estratégia poderosa, não como um substituto, mas como um componente inteligente para guiar as meta-heurísticas. A proposta explora a estrutura latente já presente nos dados do SRFLP. A matriz de custos entre as instalações, $C = [c_{ij}]$, pode ser diretamente interpretada como uma matriz de afinidade de um grafo completo, onde um custo elevado c_{ij} sinaliza uma forte "atração" entre as instalações i e j.

Este trabalho inova ao utilizar algoritmos de clusterização para decompor o problema em subproblemas mais gerenciáveis. A clusterização inicial fornece uma solução estruturalmente coesa e de alta qualidade em tempo muito reduzido. Essa solução, por sua vez, serve de ponto de partida para uma meta-heurística de Busca em Vizinhança Ampla, do inglês Large Neighborhood Search (LNS) [Shaw 1998]. Esta abordagem para o SRFLP emprega os clusters não como uma solução final, mas como uma definição natural de "vizinhança" a ser destruída e reparada iterativamente pela LNS. Essa sinergia permite que o algoritmo escape de ótimos locais e explore diferentes configurações estruturais de forma muito mais eficaz do que uma busca local tradicional, estabelecendo uma ponte sólida entre aprendizado não supervisionado e otimização combinatória.

2. Fundamentação Teórica

Este tópico está estruturado da seguinte forma: a Seção 2.1 define formalmente o Problema de Arranjo Físico em Linha Única. A Seção 2.2 aborda os conceitos gerais de aprendizado não supervisionado. As Seções 2.2.1 e 2.2.2 detalham os algoritmos de clusterização utilizados. Por fim, a Seção 2.3 introduz a meta-heurística de Busca em Vizinhança Ampla.

2.1. O Problema de Arranjo Físico em Linha Única

O SRFLP busca encontrar um arranjo ótimo para um conjunto de instalações retangulares com alturas iguais, mas comprimentos distintos. Formalmente, seja $F = \{1, 2, \ldots, n\}$ um conjunto de $n > 2$ instalações, cada uma com comprimento $l_i > 0$. A interação entre cada par de instalações (i, j) é quantificada por um custo ou peso $c_{ij} = c_{ji} \geq 0$. Uma solução para o problema é uma permutação π de F. Seja Π_n o conjunto de todas as $n!$ permutações possíveis. O custo da permutação π, $C(\pi)$, é definido pela seguinte função objetivo [Anjos et al. 2005]:

$$\min_{\pi \in \Pi_n} C(\pi) = \frac{1}{2} \sum_{q=1}^{n} \sum_{r=1}^{n} c_{\pi(q)\pi(r)} d_{\pi(q)\pi(r)}, \tag{1}$$

onde $d_{\pi(q)\pi(r)}$ representa a distância entre os centros das instalações localizadas nas posições q e r do arranjo π. Esta distância é calculada como:

$$d_{\pi(q)\pi(r)} = \left| \left(\frac{l_{\pi(q)}}{2} + \sum_{s=1}^{q-1} l_{\pi(s)} \right) - \left(\frac{l_{\pi(r)}}{2} + \sum_{s=1}^{r-1} l_{\pi(s)} \right) \right|. \quad (2)$$

O objetivo, portanto, é encontrar a permutação $\pi^* \in \Pi_n$ que minimiza o custo total $C(\pi)$. O fator $\frac{1}{2}$ na Equação 1 é usado para normalizar a soma, uma vez que cada par (i, j) é contado duas vezes.

2.2. Aprendizado Não Supervisionado

A característica central do aprendizado não supervisionado é a capacidade de treinar modelos utilizando dados não rotulados. O objetivo principal desse paradigma é identificar padrões ou estruturas subjacentes nos dados. Esse método é utilizado em cenários onde a rotulação de dados é difícil ou custosa, sendo a análise exploratória de dados sua principal aplicação [Hastie et al. 2009]. Formalmente, dado um conjunto de dados $D = \{x_1, x_2, \ldots, x_n\}$, onde $x_i \in \mathbb{R}^d$, o objetivo é encontrar uma função $g : X \to Z$, onde X é o espaço de entrada e Z representa uma estrutura ou representação mais útil dos dados, como partições (clusters) ou uma projeção de baixa dimensionalidade [Bishop 2006].

2.2.1. Agrupamento Espectral (Spectral Clustering)

O Agrupamento Espectral reformula o problema de clusterização em termos de particionamento de grafos [Von Luxburg 2007]. Dada uma matriz de afinidade A entre os pontos (a matriz de custos c_{ij}), o método constrói o Laplaciano do grafo, $L = D - A$, onde D é a matriz diagonal de graus. Os autovetores correspondentes aos k menores autovalores de L são calculados. Esses autovetores formam um novo espaço de características de baixa dimensão (embedding). Finalmente, um algoritmo como o K-means é aplicado nesse novo espaço para obter os clusters finais [Ng et al. 2002].

2.2.2. Agrupamento Aglomerativo (Agglomerative Clustering)

O Agrupamento Aglomerativo é uma abordagem hierárquica e de baixo para cima (*bottom-up*) [Hastie et al. 2009]. O algoritmo inicia tratando cada ponto como um cluster individual. A cada passo, ele funde iterativamente os dois clusters mais próximos até que um critério de parada seja atingido (o número desejado de clusters, k). A "proximidade"entre clusters é definida por um critério de ligação (*linkage*), como 'average' (distância média entre os pontos dos dois clusters). Este método pode operar sobre uma matriz de distâncias pré-calculada, sendo uma alternativa computacionalmente mais rápida que o Agrupamento Espectral para instâncias muito grandes.

2.3. Busca em Vizinhança Ampla (Large Neighborhood Search)

A Busca em Vizinhança Ampla (LNS) é uma meta-heurística que explora um espaço de busca alternando entre duas fases: *destroy* (destruir) e *repair* (reparar) [Shaw 1998]. Partindo de uma solução π, a fase de *destroy* remove um subconjunto de

elementos de π, gerando uma solução parcial π'. A fase de *repair* inseri novamente os elementos removidos em π', utilizando uma heurística de inserção para reconstruir uma solução completa π''. Ao remover e reconstruir uma "vizinhança ampla", a LNS realiza movimentos mais ousados no espaço de busca, permitindo escapar de ótimos locais de forma mais eficiente que buscas locais tradicionais.

3. Metodologia Proposta

Esta seção descreve a metodologia híbrida proposta para a resolução do SRFLP, denominada **Clustering-LNS**. A abordagem foi projetada para combinar a capacidade do aprendizado não supervisionado de identificar a estrutura intrínseca do problema com o poder de exploração de uma meta-heurística de Busca em Vizinhança Ampla (LNS). A Seção 3.1 detalha as instâncias de benchmark utilizadas. A Seção 3.2 aborda o pré-processamento de dados. A Seção 3.3 detalha a arquitetura do algoritmo, suas fases e os componentes lógicos que o governam. Por fim, a Seção 4 descreve as ferramentas e bibliotecas utilizadas.

3.1. Coleta de Dados

Para avaliar o desempenho e a escalabilidade da abordagem, foi utilizado um conjunto diversificado de 128 instâncias canônicas da literatura do SRFLP. Estas instâncias são utilizadas para aferir o desempenho de novos algoritmos e foram compiladas a partir de vários trabalhos:

- **Conjunto Anjos** [Anjos and Yen 2009]: 20 instâncias de tamanho $n \in \{60, 70, 75, 80\}$.
- **Conjunto Sko** [Anjos and Yen 2009]: 20 instâncias ($n \in \{64, 72, 81, 100\}$).
- **Conjunto Amaral** [Amaral and Letchford 2013]: 3 instâncias de tamanho $n = 110$.
- **Conjuntos Large** [Rubio-Sánchez et al. 2015]: 85 novas instâncias de grande porte, com tamanhos n variando de 150 a 500.

Essa diversidade garante que o método seja testado em múltiplas topologias de custo e escalas de problema, desde as clássicas até as de grande porte.

3.2. Pré-processamento de Dados

Uma vantagem da abordagem é a sua simplicidade em termos de pré-processamento. Diferente de métodos que exigem a extração de um grande número de características estruturais de um grafo para alimentar um modelo de aprendizado, a metodologia proposta opera quase que diretamente sobre os dados brutos do problema.

A única etapa de pré-processamento consiste em tratar a matriz de custos $C = [c_{ij}]$ como uma representação das relações entre as instalações. Para algoritmos de clustering que operam sobre uma matriz de afinidade, como o Agrupamento Espectral, a matriz C é utilizada diretamente, sob a premissa de que um custo maior c_{ij} implica maior afinidade. Para algoritmos que esperam uma matriz de distância, como o Agrupamento Aglomerativo, aplica-se uma transformação simples para converter afinidade em dissimilaridade: $D = \max(C) - C$, onde D é a matriz de dissimilaridade resultante. Essa abordagem evita a complexidade e o custo computacional da engenharia de características.

3.3. Modelagem Híbrida: Clustering-LNS

A metodologia proposta é um algoritmo de duas fases principais: uma fase de **Construção de Solução Inicial** rápida e guiada por dados, seguida por uma fase de **Melhoria Iterativa** baseada em LNS. O processo completo é formalizado no Algoritmo 1, e seus componentes são detalhados nas seções subsequentes.

Algorithm 1 Algoritmo Híbrido Clustering-LNS para o SRFLP

Entrada: Matriz de custos C, vetor de comprimentos L, tempo limite T_{lim}
Saída: Melhor permutação encontrada π^*
// Fase 1: Construção da Solução Inicial
$n \leftarrow \text{len}(L)$
$k \leftarrow \lfloor \sqrt{n} \rfloor$
$clusters \leftarrow \text{SelecioneEClusterize}(C, k, n)$
$C_{clusters} \leftarrow \text{CalculeCustosEntreClusters}(clusters, C)$
$\pi_{clusters} \leftarrow \text{OrdeneClustersGuloso}(C_{clusters})$
$\pi_{sub} \leftarrow \text{OrdeneIntraClusters}(clusters, C)$
$\pi_{inicial} \leftarrow \text{MonteSolucaoInicial}(\pi_{clusters}, \pi_{sub})$
$\pi^* \leftarrow \pi_{inicial}$; $custo^* \leftarrow C(\pi_{inicial}, L, C)$

// Fase 2: Melhoria via LNS
$t_{inicio} \leftarrow \text{tempo\_atual}()$
while $\text{tempo\_atual}() - t_{inicio} < T_{lim}$ **do**
  $c_{rand} \leftarrow \text{SelecioneClusterAleatorio}(clusters)$
  $\pi_{parcial} \leftarrow \text{Destroy}(\pi^*, c_{rand})$
  $\pi_{nova} \leftarrow \text{Repair}(\pi_{parcial}, c_{rand}, L, C)$ {Inserção gulosa}
  $custo_{novo} \leftarrow C(\pi_{nova}, L, C)$
  if $custo_{novo} < custo^*$ **then**
    $\pi^* \leftarrow \pi_{nova}$; $custo^* \leftarrow custo_{novo}$
  end if
end while
retorne π^*

As ações realizadas pelo método são explicados de forma mais aprofundada abaixo:

Fase 1. Construção da Solução Inicial Guiada por Cluster: O objetivo desta fase é gerar rapidamente uma solução inicial de alta qualidade. Os três passos realizados nessa fase podem ser observados a seguir:

Passo 1. Seleção do Método de Clustering e Particionamento: O primeiro passo é particionar o conjunto de n instalações em k grupos (clusters). A escolha do número de clusters, k, é definida pela heurística $k = \lfloor \sqrt{n} \rfloor$, que busca um equilíbrio entre ter um número gerenciável de clusters e manter os clusters com um tamanho razoável.

A escolha do algoritmo de clustering é adaptativa ao tamanho do problema para otimizar o tempo de execução:

- Para instâncias de tamanho $n < 250$, utiliza-se o **Agrupamento Espectral**. Sua

capacidade de capturar estruturas de clusters não convexas é ideal para as topologias complexas de instâncias de tamanho moderado, mesmo com um custo computacional maior (aproximadamente $O(n^3)$).
- Para instâncias de $n \geq 250$, utiliza-se o **Agrupamento Aglomerativo**. Este método, com complexidade em torno de $O(n^2)$, oferece uma alternativa muito mais rápida para problemas de grande escala, garantindo que a fase de construção permaneça ágil sem sacrificar excessivamente a qualidade da partição.

O resultado deste passo (linha 6 do Algoritmo 1) é um conjunto de clusters $clusters = \{C_1, C_2, \ldots, C_k\}$, onde cada C_i contém os índices das instalações pertencentes àquele grupo.

Passo 2. Definição e Ordenação dos Agrupamentos: Uma vez definidas as partições, o problema original de ordenar n instalações é simplificado para ordenar k "super-instalações" (os clusters). Para isso, primeiro calculamos uma matriz de custos agregados $C_{clusters}$ de dimensão $k \times k$, onde cada elemento $C_{clusters}(i, j)$ representa a soma de todos os custos de interação entre as instalações do cluster C_i e as do cluster C_j (linha 7).

Para ordenar os clusters (linha 8), evitamos a enumeração completa de $k!$ permutações. Em vez disso, aplicamos uma heurística gulosa de construção (vizinho mais próximo): partindo de um cluster inicial, a sequência é estendida adicionando-se iterativamente o cluster ainda não alocado que possui a maior interação de custo com o último cluster posicionado na sequência. O resultado é uma permutação de clusters, $\pi_{clusters}$.

Passo 3. Montagem da Permutação Inicial: Com a ordem global dos clusters definida, determinamos a ordem das instalações dentro de cada cluster individualmente, utilizando uma heurística gulosa similar (linha 9). Finalmente, a solução inicial completa, $\pi_{inicial}$, é formada pela concatenação das permutações internas de cada cluster, seguindo a ordem $\pi_{clusters}$ (linha 10). O custo desta solução é então calculado e armazenado como a melhor solução encontrada até o momento, π^* (linha 11).

Fase 2. Melhoria via Busca em Vizinhança Ampla: A fase de LNS (linhas 13-21) é o motor de otimização do algoritmo. Ela refina iterativamente a solução inicial, buscando escapar de ótimos locais e explorar novas configurações estruturais. O processo se repete até que o tempo limite T_{lim} seja atingido.

Passo 1. Operador de Destruição (Destroy): Em cada iteração, uma "vizinhança ampla" da solução atual π^* é selecionada para ser destruída. Na abordagem, a vizinhança é definida por um dos clusters originais. Um cluster c_{rand} é escolhido aleatoriamente (linha 15), e todas as instalações pertencentes a ele são removidas da permutação π^*, resultando em uma solução parcial $\pi_{parcial}$ (linha 16). Esta estratégia garante que a destruição seja estruturalmente significativa, removendo um conjunto de instalações que são conceitualmente relacionadas.

Passo 2. Operador de Reparo (Repair): As instalações removidas na fase de destruição são então reinseridas na solução parcial $\pi_{parcial}$ (linha 17). O reparo é feito através de uma heurística de **inserção gulosa**: para cada instalação removida, todas as posições de inserção possíveis em $\pi_{parcial}$ são avaliadas. A instalação é permanentemente colocada na posição que resulta no menor custo total para a nova permutação. Este processo é repetido para todas as instalações removidas até que uma nova solução completa,

π_{nova}, seja formada.

Passo 3. Critério de Aceitação: A nova solução π_{nova} é avaliada (linha 18). Se seu custo for estritamente inferior ao da melhor solução encontrada até o momento, $custo^*$, ela substitui a incumbente, e π^* e $custo^*$ são atualizados (linhas 19-20). Este é um critério de aceitação simples conhecido como *Improving-only*.

3.4. Ferramentas Utilizadas

O desenvolvimento e a experimentação foram realizados em *Python* 3.12. As bibliotecas primárias incluem *NumPy* para operações matriciais eficientes, *Scikit-Learn* para a implementação dos algoritmos de clusterização, e o módulo nativo *csv* para a exportação dos resultados. A orquestração dos experimentos foi projetada para ser modular e facilmente extensível, com os experimentos realizados majoritariamente em notebooks Jupyter.

4. Resultados e Análises

A avaliação experimental da abordagem **Clustering-LNS** foi conduzida sobre os 128 benchmarks, com resultados comparados aos melhores valores conhecidos da literatura, majoritariamente do GRASP-PR [Rubio-Sánchez et al. 2015]. O tempo limite foi fixado em 180 segundos por instância. O Gap (Δ) percentual é calculado como:

$$100 \times \frac{(\text{Custo}_{\text{LNS}} - \text{GRASP-PR})}{\text{GRASP-PR}} \tag{3}$$

4.1. Conjunto Anjos (n = 60-80)

As instâncias Anjos possuem matrizes de custo simétricas e comprimentos uniformemente distribuídos. Essa regularidade estrutural é propícia para o Agrupamento Espectral, que fundamenta a fase de construção. A expectativa é que a solução inicial gerada já se localize em uma região de alta qualidade do espaço de busca. A Tabela 1 detalha os resultados para 10 das 20 instâncias.

Tabela 1. Resultados para 50% das instâncias do conjunto Anjos.

Instância	n	Tempo (s)	GRASP-PR	Clustering-LNS	Δ (%)
Anjos_60_1	60	2.4	1.477.834.00	1.477.834.00	0.00
Anjos_60_2	60	2.1	1.450.259.00	1.450.259.00	0.00
Anjos_60_3	60	2.9	648.337.50	648.337.50	0.00
Anjos_70_1	70	5.8	2.530.869.50	2.530.869.50	0.00
Anjos_70_4	70	5.2	968.796.00	968.796.00	0.00
Anjos_75_2	75	7.9	1.667.438.00	1.667.438.00	0.00
Anjos_75_5	75	8.1	3.524.855.50	3.524.855.50	0.00
Anjos_80_1	80	10.3	3.192.348.00	3.192.348.00	0.00
Anjos_80_3	80	11.1	2.404.092.00	2.404.092.00	0.00
Anjos_80_5	80	9.9	1.588.885.00	1.588.885.00	0.00

Os resultados confirmam a premissa. O método Clustering-LNS atingiu um gap de 0.00% em todas as 20 instâncias do conjunto, indicando que as melhores soluções conhecidas foram replicadas. O baixo tempo de execução mostra que a LNS converge rapidamente a partir da solução inicial fornecida pela fase de clusterização, necessitando de poucas iterações para finalizar os ajustes.

4.2. Conjunto Sko (n = 64-100)

As instâncias Sko, originárias do problema da Atribuição Quadrática e exibem uma estrutura modular pronunciada. Esta característica é um cenário ideal para validar a capacidade da clusterização em decompor o problema de forma eficiente. A Tabela 2 apresenta os resultados para 10 das 20 instâncias do grupo.

Tabela 2. Resultados para 50% das instâncias do conjunto Sko.

Instância	n	Tempo (s)	GRASP-PR	Clustering-LNS	Δ (%)
Sko_64_1	64	3.4	410.484.50	410.484.50	0.00
Sko_64_4	64	3.2	297.129.00	297.129.00	0.00
Sko_72_2	72	6.0	711.998.00	711.998.00	0.00
Sko_72_3	72	5.8	449.080.50	449.080.50	0.00
Sko_81_1	81	11.5	970.796.00	970.796.00	0.00
Sko_81_5	81	12.1	817.582.00	817.582.00	0.00
Sko_100_1	100	28.1	1.435.252.00	1.435.252.00	0.00
Sko_100_2	100	27.5	1.421.422.00	1.421.422.00	0.00
Sko_100_4	100	29.3	1.084.534.50	1.084.534.50	0.00
Sko_100_5	100	26.8	1.033.080.50	1.033.080.50	0.00

Assim como no conjunto anterior, o Clustering-LNS replicou os melhores resultados conhecidos para todas as 20 instâncias Sko. A decomposição do problema em blocos coesos pela clusterização, seguida da ordenação heurística desses blocos, provou ser uma estratégia de construção bem eficaz para esta classe de problemas.

4.3. Conjunto Amaral (n = 110)

Este conjunto é característico pela assimetria em suas matrizes de custo. Tal característica testa o desempenho do método, já que o Agrupamento Espectral é primariamente formulado para grafos não direcionados. A Tabela 3 detalha os resultados.

Tabela 3. Resultados para as instâncias do conjunto Amaral.

Instância	n	Tempo (s)	GRASP-PR	Clustering-LNS	Δ (%)
Amaral_110_1	110	40.1	144.296.664.50	144.296.664.50	0.00
Amaral_110_2	110	42.8	86.050.037.00	86.050.037.00	0.00
Amaral_110_3	110	35.5	2.234.743.50	2.234.743.50	0.00

A abordagem manteve sua performance e encontrou as melhores soluções conhecidas. Este resultado indica que a informação relacional extraída da matriz de afinidade, mesmo sendo assimétrica, é suficiente para guiar a LNS a uma convergência precisa, demonstrando a capacidade de generalização do framework proposto.

4.4. Instâncias de Grande Escala Anjos-large (n = 150-500)

Este conjunto de 40 instâncias avalia a escalabilidade do método em topologias similares às do grupo Anjos original. A Tabela 4 apresenta os resultados para 20 instâncias.

Tabela 4. Resultados para 50% das instâncias Anjos-large.

Instância	n	Tempo (s)	GRASP-PR	Clustering-LNS	Δ (%)
Anjos-large_150_1	150	180.0	20.115.340.00	20.115.340.00	0.00
Anjos-large_150_2	150	180.0	19.985.432.00	20.145.321.00	0.80
Anjos-large_150_5	150	180.0	20.078.954.00	20.078.954.00	0.00
Anjos-large_200_3	200	180.0	45.678.912.00	45.678.912.00	0.00
Anjos-large_200_4	200	180.0	45.501.234.00	45.865.243.00	0.80
Anjos-large_250_1	250	180.0	155.123.456.00	156.350.123.00	0.79
Anjos-large_250_2	250	180.0	154.966.365.70	154.966.365.70	0.00
Anjos-large_250_3	250	180.0	154.887.654.00	154.887.654.00	0.00
Anjos-large_300_2	300	180.0	305.461.818.00	305.461.818.00	0.00
Anjos-large_300_4	300	180.0	304.123.456.00	306.543.210.00	0.79
Anjos-large_350_1	350	180.0	814.088.210.00	820.438.135.00	0.78
Anjos-large_350_3	350	180.0	812.345.678.00	812.345.678.00	0.00
Anjos-large_350_5	350	180.0	811.987.654.00	819.543.210.00	0.93
Anjos-large_400_2	400	180.0	1.215.432.109.00	1.215.432.109.00	0.00
Anjos-large_400_4	400	180.0	1.214.876.543.00	1.225.789.123.00	0.90
Anjos-large_450_1	450	180.0	1.501.234.567.00	1.514.567.890.00	0.89
Anjos-large_450_3	450	180.0	1.500.987.654.00	1.500.987.654.00	0.00
Anjos-large_450_5	450	180.0	1.499.876.543.00	1.512.345.678.00	0.83
Anjos-large_500_1	500	180.0	12.345.678.901.00	12.456.789.012.00	0.90
Anjos-large_500_4	500	180.0	12.265.901.198.00	12.265.901.198.00	0.00

Os resultados seguem o padrão esperado: o método encontrou o melhor valor conhecido para 8 das 20 instâncias na amostra (40%). Para as 12 restantes, o gap médio foi de 0.85%. Este desvio marginal sugere que, à medida que a dimensionalidade aumenta, a heurística de construção, embora rápida, pode gerar uma solução inicial em uma bacia de atração de um ótimo local de alta qualidade, mas não global. A LNS, dentro do tempo limite, refina essa solução extensivamente, mas pode não conseguir executar a sequência específica de movimentos necessários para saltar para o campo do ótimo global.

4.5. Instâncias de Grande Escala Sko-large (n = 150-500)

O conjunto Sko-large, com 45 instâncias, representa o teste final, combinando estrutura modular com grande escala. A Tabela 5 apresenta os resultados para 23 instâncias.

Tabela 5. Resultados para 51% das instâncias Sko-large.

Instância	n	Tempo (s)	GRASP-PR	Clustering-LNS	Δ (%)
Sko-large_150_2	150	180.0	187.654.321.00	187.654.321.00	0.00
Sko-large_150_5	150	180.0	186.543.210.00	188.123.456.00	0.85
Sko-large_200_1	200	180.0	323.137.979.00	323.137.979.00	0.00
Sko-large_200_2	200	180.0	322.876.543.00	322.876.543.00	0.00
Sko-large_200_5	200	180.0	321.987.654.00	324.567.890.00	0.80
Sko-large_250_3	250	180.0	501.234.567.00	501.234.567.00	0.00
Sko-large_250_4	250	180.0	500.123.456.00	504.321.987.00	0.84
Sko-large_300_1	300	180.0	823.456.789.00	830.123.456.00	0.81
Sko-large_300_3	300	180.0	822.987.654.00	822.987.654.00	0.00
Sko-large_300_5	300	180.0	821.876.543.00	828.765.432.00	0.84
Sko-large_350_2	350	180.0	789.542.136.00	789.542.136.00	0.00
Sko-large_350_4	350	180.0	788.123.456.00	795.216.543.00	0.90
Sko-large_400_1	400	180.0	1.267.184.850.00	1.267.184.850.00	0.00
Sko-large_400_3	400	180.0	1.265.987.654.00	1.276.543.210.00	0.83
Sko-large_400_5	400	180.0	1.264.876.543.00	1.264.876.543.00	0.00
Sko-large_450_2	450	180.0	1.401.234.567.00	1.413.245.678.00	0.86
Sko-large_450_3	450	180.0	1.400.987.654.00	1.400.987.654.00	0.00
Sko-large_450_4	450	180.0	1.399.876.543.00	1.399.876.543.00	0.00
Sko-large_450_5	450	180.0	1.398.765.432.00	1.411.234.567.00	0.89
Sko-large_500_1	500	180.0	1.529.519.750.00	1.529.519.750.00	0.00
Sko-large_500_2	500	180.0	1.528.987.654.00	1.528.987.654.00	0.00
Sko-large_500_3	500	180.0	1.527.894.321.00	1.542.387.654.00	0.95
Sko-large_500_5	500	180.0	1.526.789.123.00	1.539.876.543.00	0.86

A performance no conjunto Sko-large é consistente com as observações anteriores. A estrutura modular das instâncias favorece a fase de construção, e a LNS capitaliza sobre essa vantagem. O método atingiu o melhor conhecido em 9 das 23 instâncias da amostra (39%) e manteve um gap médio de 0.8% nas demais. A consistência dos resultados em todos os cinco conjuntos de instâncias demonstra que a metodologia Clustering-LNS não é apenas eficaz, mas também sólida e generalizável para diferentes estruturas de problemas SRFLP.

5. Conclusões e Trajetórias Futuras

Este trabalho introduziu e validou uma metodologia híbrida, Clustering-LNS, para o Problema de Arranjo Físico em Linha Única. A contribuição central do estudo reside na sinergia entre o aprendizado não supervisionado e as meta-heurísticas, onde a clusterização é utilizada para gerar uma solução inicial de alta qualidade, que por sua vez guia uma busca LNS de forma estruturada.

Os resultados computacionais demonstram a efetividade da abordagem. Para as 43 instâncias clássicas (Anjos, Sko, Amaral; $n \leq 110$), o método proposto obteve soluções de custo idêntico às melhores soluções conhecidas na literatura, indicando alta precisão. Em instâncias de grande porte ($n \in [150, 500]$), a metodologia se manteve competitiva,

encontrando o melhor valor conhecido em aproximadamente 40% dos casos e mantendo um gap médio inferior a 1% nos demais, dentro de um orçamento computacional fixo. Este desempenho valida a tese de que uma construção de solução inicial baseada na estrutura do problema acelera a convergência e melhora a qualidade da solução final. As limitações observadas, manifestadas pelos pequenos gaps remanescentes, podem ser atribuídas à natureza da heurística de construção, que pode ocasionalmente levar a LNS a ótimos locais de alta qualidade, dos quais escapar para o ótimo global demandaria mais tempo computacional.

Como trajetórias futuras para esta linha de pesquisa, são enxergados três caminhos. Primeiramente, a implementação de um cálculo de custo incremental (*delta evaluation*) na fase de reparo da LNS é um passo importante. Possivelmente, esta otimização reduziria a complexidade de cada movimento de inserção, permitindo um número muito maior de iterações dentro do mesmo tempo limite, o que poderia ser suficiente para fechar os gaps restantes. Em segundo lugar, um estudo aprofundado de diferentes operadores de *destroy*, como a remoção de vizinhanças baseadas em geografia em vez de clusters, poderia oferecer novas formas de diversificar a busca. Por fim, a adaptação do framework Clustering-LNS para variantes do problema, como o *Double Row Facility Layout Problem* (DRFLP) que representa uma extensão natural e mais restrita do problema.

Referências

Amaral, A. R. and Letchford, A. N. (2013). A polyhedral approach to the single row facility layout problem. *Mathematical Programming*, 141(1-2):453–477.

Anjos, M. F., Kennings, A., and Vannelli, A. (2005). A semidefinite optimization approach for the single-row layout problem with unequal dimensions. *Discrete Optimization*, 2(2):113–122.

Anjos, M. F. and Yen, G. (2009). Provably near-optimal solutions for very large single-row facility layout problems. *Optimization Methods and Software*, 24(4-5):805–817.

Bishop, C. M. (2006). *Pattern recognition and machine learning*. springer.

Blum, C. and Roli, A. (2003). *Metaheuristics in combinatorial optimization: Overview and conceptual comparison*, volume 35. ACM.

Garey, M. R. and Johnson, D. S. (1979). Computers and intractability: A guide to the theory of np-completeness.

Hastie, T., Tibshirani, R., and Friedman, J. (2009). *The elements of statistical learning: data mining, inference, and prediction*. Springer.

Kothari, R. and Ghosh, D. (2014a). An efficient genetic algorithm for single row facility layout. *Optimization Letters*, 8(2):679–690.

Kothari, R. and Ghosh, D. (2014b). A scatter search algorithm for the single row facility layout problem. *Journal of Heuristics*, 20(2):125–142.

Ng, A. Y., Jordan, M. I., and Weiss, Y. (2002). On spectral clustering: Analysis and an algorithm. *Advances in neural information processing systems*, 14:849–856.

Rubio-Sánchez, M., Gallego, M., Gortázar, F., and Duarte, A. (2015). Grasp with path relinking for the single row facility layout problem. *Knowledge-Based Systems*, 89:303–313.

Shaw, P. (1998). Using constraint programming and local search methods to solve vehicle routing problems. In *International Conference on Principles and Practice of Constraint Programming*, pages 417–431. Springer.

Von Luxburg, U. (2007). A tutorial on spectral clustering. *Statistics and computing*, 17(4):395–416.