UMA HEURÍSTICA BASEADA NA BUSCA LOCAL ITERADA PARA O PROBLEMA DE ALOCAÇÃO DE CORREDOR

Charles Abreu Santana
Universidade Federal de Viçosa - UFV
Av. P. H. Rofls, s/n, Campus universitário, 36570-900, Viçosa, MG
charles.abreu@outlook.com

André Gustavo dos Santos
Universidade Federal de Viçosa - UFV
Av. P. H. Rofls, s/n, Campus universitário, 36570-900, Viçosa, MG
andre@dpi.ufv.br

RESUMO

O Problema de Alocação de Corredor investiga como organizar facilidades em ambos os lados de um corredor ao longo de sua extensão, de forma que o custo de comunicação total entre as facilidades seja minimizado. Para esse fim, o arranjo das facilidades deve começar de uma origem em comum no corredor, sendo que não é permitido espaço vazio entre as facilidades. Este problema é aplicável em construções como hospitais, escritórios e escolas. Uma abordagem baseada na busca local iterada é apresentada para minimizar o custo entre as facilidades. O algoritmo apresentou bom desempenho e resultados satisfatórios em relação à literatura.

PALAVRAS CHAVE. Leiaute de facilidades, otimização combinatória, busca local iterada, alocação de corredor.

MH - Metaheuristicas; OC - Otimização Combinatória

ABSTRACT

The Corridor Allocation Problem deals with the organization of facilities on both sides of a corridor along its length, so that the total cost of communication is minimized. For this, the arrangement of the facilities must start from a common origin in the corridor and no space is allowed between two adjacent facilities. This problem is applicable in arrangement of rooms in buildings like hospitals, offices and schools. An approach based on iterated local search is presented to minimize the cost between facilities. The algorithm showed good performance and satisfactory objective values when compared to the one presented in the literature.

KEYWORDS. Facilities layout, combinatorial optimization, iterated local search, corridor allocation.

MH - Metaheuristics; OC - Combinatorial Optimization

1. Introdução

O Problema de Alocação de Corredor [Amaral, 2012], abreviado como CAP *(Corridor Allocation Problem)*, lida com a organização de facilidades, sem sobreposição, ao longo de um corredor. A organização física das facilidades na extensão do corredor é chamada de *leiaute*. Sabendo que existe um tráfego médio entre as facilidades, o objetivo é encontrar um leiaute que minimize o custo de comunicação total entre as mesmas, de forma que atenda a duas condições principais: (a) os arranjos devem começar a partir de um ponto comum na extremidade esquerda em ambos os lados do corredor, (b) nenhum espaço é permitido entre facilidades adjacentes. Leiautes com estas características são requisitados em construções como prédios de escritórios, hospitais e escolas, onde há um tráfego intenso entre as salas, por exemplo.

O problema de alocação de corredor foi introduzido por Amaral [2012] e tem relação com outros problemas de disposição de facilidades. Um deles é a organização de facilidades em fileira única (Single Row Facility Layout Problem, SRFLP) [Kothari e Ghosh, 2013] que trabalha somente com a disposição de facilidades em somente um dos lados do corredor. Um outro problema é o de organização de facilidades em fileira dupla (Double Row Layout Problem, DRLP), introduzido por [Chung e Tanchoco, 2010] muito similar ao CAP, porém as condições em que os arranjos precisam iniciar em um ponto em comum na extremidade à esquerda e nenhum espaço pode ser permitido entre facilidades adjacentes não precisam necessariamente ser atendidas.

Amaral [2012] apresentou uma formulação de programação inteira mista para resolução do CAP. Porém o problema CAP é NP-Difícil, o que acaba deixando ineficiente abordagens exatas. Ghosh e Kothari [2012] desenvolveram um algoritmo genético hibrido e Scather Search com path-relinking para o CAP. Ahonen et al. [2014] aplicaram Busca Tabu e Simulated Annealing para o problema. Dessa forma, neste trabalho, foi escolhida a heurística de busca local iterada, ou simplesmente ILS (Iterated Local Search) [Lourenço et al., 2010], como uma abordagem diferenciada para ser aplicada ao CAP.

Este artigo está organizado da seguinte forma: na próxima seção o problema é formalizado e é mostrado um modelo matemático que descreve o problema; na seção 3 são descritos os detalhes da heurística proposta; a seção 4 apresenta os resultados obtidos para uma séria de instâncias de diferentes tamanhos e uma comparação com os resultados da literatura; por fim, a seção 5 conclui o trabalho.

2. Formalização do Problema

CAP lida com a organização de facilidades na extensão de um corredor cujo lados são representados por duas linhas horizontais paralelas ao eixo x do sistema de coordenadas cartesiano. Cada facilidade possui um tamanho específico definido por sua extensão horizontal.

Dadas n facilidades, sua organização física é chamada de *leiaute*. Entre duas facilidades existe uma *média de tráfego diário*. Em um leiaute o *custo de comunicação* entre um par de facilidades é o produto entre sua média de tráfego diário e a distância entre elas. Essa distância é considerada como a distância entre os centros das facilidades. Usualmente, a largura do corredor, que é o espaço entre seus lados, é desprezada. Assim lidamos somente com a extensão do corredor.

Associado a um leiaute existe o *custo de comunicação total*, que é a soma de todos os custos de comunicação. O objetivo é encontrar um leiaute que minimize o custo de comunicação total entre as facilidades, atendendo a duas condições principais: (a) os arranjos devem começar a partir de um ponto comum na extremidade esquerda em ambos os lados do corredor, (b) nenhum espaço é permitido entre facilidades adjacentes.

O leiaute de uma solução do CAP pode ser representado, segundo [Amaral, 2012], por uma partição de uma permutação π em dois componentes: $[\pi_1, \pi_2] = \pi$, onde π_1 e π_2 representam o arranjo das facilidades de cada lado do corredor a partir de uma extremidade à esquerda em comum. Portanto, $[\pi_1, \pi_2] = \pi$, se e somente se $\{\pi_1, \pi_2\}$ é uma partição de π.

Como pode ser observado na Figura 1, adaptada de [Ahonen et al., 2014], o leiaute $\pi = (5, 1, 7, 2, 4, 6, 3)$ é constituído de duas partições $\pi_1 = (5, 1, 7, 2)$ e $\pi_2 = (4, 6, 3)$ que representam

os arranjos das facilidades para cada lado do corredor. Note que ambos os arranjos iniciam-se em um mesmo ponto em comum à esquerda do corredor, além disso, não há espaços nem sobreposição entre as facilidades.

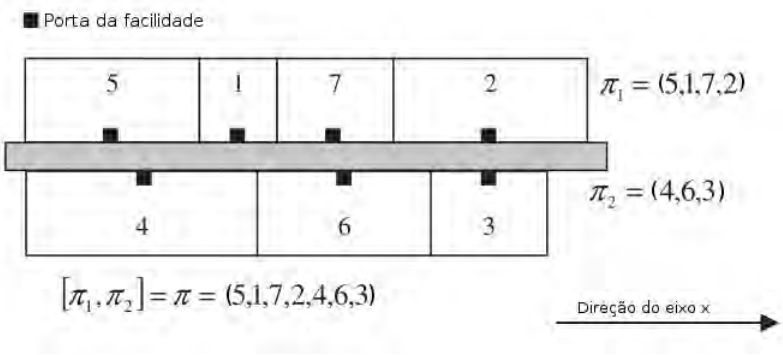

Figura 1: Exemplo ilustrativo de uma solução do CAP. Adaptado de [Ahonen et al., 2014]

Alguns valores devem ser considerados para o CAP:

- n : é o número de facilidades que devem ser organizadas ao longo do corredor.
- $C_{\pi_r(i),\pi_s(j)}$: a média diária de tráfego entre a *i-ésima* facilidade da linha r e a *j-ésima* facilidade da linha s $(1 \leq r \leq s \leq 2)$.
- $x_{\pi_r(i)}$: é a abscissa do centro da *i-ésima* facilidade na linha r $(r \in \{1, 2\})$

Seja Π_n o conjunto de todas as permutações π de $N = \{1, 2, \ldots, n\}$, então um modelo matemático para o CAP, proposto em [Amaral, 2012], é dado por:

$$\min_{[\pi_1,\pi_2]:[\pi_1,\pi_2]\in\Pi_n} \left\{ \sum_{i=1}^{|\pi_1|-1} \sum_{j=i+1}^{|\pi_1|} c_{\pi_1(i),\pi_1(j)} |x_{\pi_1(i)} - x_{\pi_1(j)}| \right.$$
$$+ \sum_{i=1}^{|\pi_2|-1} \sum_{j=i+1}^{|\pi_2|} c_{\pi_2(i),\pi_2(j)} |x_{\pi_2(i)} - x_{\pi_2(j)}|$$
$$\left. + \sum_{i=1}^{|\pi_1|} \sum_{j=1}^{|\pi_2|} c_{\pi_1(i),\pi_2(j)} |x_{\pi_1(i)} - x_{\pi_2(j)}| \right\} \tag{1}$$

onde,

$$x_{\pi_r(j)} = \frac{l_{\pi_r(j)}}{2} + \sum_{i=1}^{j-1} l_{\pi_r(i)}, r \in 1, 2; 1 \leq j \leq n. \tag{2}$$

e $l_{\pi_r(j)}$ é o tamanho da *j-ésima* facilidade na linha r.

A função objetivo na equação (1) considera os custos de comunicação entre elementos da mesma linha e entre elementos de linhas opostas. A restrição (2) não permite que exista nenhum espaço entre itens adjacentes. Além disso, garante que facilidades não se sobreponham e que o elemento mais à esquerda de cada linha tenha valor 0 de abscissa.

3. Heurísticas propostas

Nesta seção serão descritos os algoritmos implementados para o CAP. Primeiramente será descrito o procedimento de construção da solução inicial juntamente com a estrutura de vizinhança e busca local. Em seguida será apresentada a heurística proposta neste trabalho, que utiliza como solução inicial a solução produzida no procedimento de construção, que é melhorada pela busca local.

3.1. Heurística Construtiva e Busca Local

Seja um leiaute do CAP definido como (π, t), onde o parâmetro t representa o número de facilidades no arranjo 1 (ou seja, $(\pi, t) \equiv \{\pi_1, \pi_2\} \Leftrightarrow [\pi_1, \pi_2] = \pi, |\pi_1| = t, |\pi_2| = n - t$). O custo de um leiaute (π, t) é dado por $f(\pi, t)$.

O algoritmo construtivo utilizado baseia-se na ideia de que facilidades próximas devam ter um tráfego maior entre elas. O objetivo é de obter um balanceamento entre quantidade de tráfego e distância. Para esse fim, foi utilizada uma métrica baseada na relação tráfego/tamanho da facilidade. Assim, podemos aproximar tanto facilidades de alto tráfego quanto facilidades menores, proporcionando, teoricamente, um menor custo total.

O Algoritmo 1 mostra a construção da solução inicial. Começando com dois arranjos vazios (Linha 2), utiliza-se a ideia de balanceamento implementada em Ghosh e Kothari [2012], inserindo elementos na partição de menor tamanho, com o intuito de balancear os lados do corredor. Adicionalmente à ideia anterior, usando a funçao MAX (Linhas 5 e 9), escolhe-se uma facilidade, dentre as disponíveis, com a maior relação tráfego/tamanho em referência à facilidade de última posição em um dos arranjos. É inserida na partição a facilidade escolhida e o processo continua até que não haja mais itens a serem inseridos. Finalmente, na Linha 13, o parâmetro t da solução S receberá o tamanho do primeiro arranjo. Na Linha 14 a solução é construída concatenando as duas partições obtidas no processo de construção.

Algorithm 1 Algoritmo Construtivo

1: **function** CONSTRUTIVO
2: $\pi_1 \leftarrow \emptyset, \pi_2 \leftarrow \emptyset$
3: **while** $ExistirFacilidades()$ **do** ▷ Enquanto a solução não estiver completa
4: **if** $|\pi_1| \leq |\pi_2|$ **then**
5: $F \leftarrow MAX(Facilidades, \pi_1),\ F \notin \pi_1 \cup \pi_2$
6: $\pi_1 \leftarrow \pi_1 \cup F$
7: **end if**
8: **if** $|\pi_2| < |\pi_1|$ **then**
9: $F \leftarrow MAX(Facilidades, \pi_2),\ F \notin \pi_1 \cup \pi_2$
10: $\pi_2 \leftarrow \pi_2 \cup F$
11: **end if**
12: **end while**
13: $S.t \leftarrow |\pi_1|$
14: $S \leftarrow \pi_1 \oplus \pi_2$ ▷ Concatena as partições
15: **return** S
16: **end function**

Para o problema do CAP foi implementada uma estrutura de vizinhança baseada na troca de dois elementos na solução. Dada uma solução S, seu vizinho denominado S' consiste de um novo leiaute trocando-se a posição de duas facilidades do leiaute de S. O Algoritmo 2 descreve o procedimento de busca local. Durante a busca local os movimentos de vizinhança são executados sobre um valor fixo de t, até um leiaute melhor ser encontrado (Linhas 5 a 17). O parâmetro t representa o ponto em que uma solução deve ser particionada para gerar as duas linhas do leiaute, portanto t é necessário para calcular a função objetivo. Para cada valor de t uma busca é executada, e a melhor solução dentre toda a vizinhança, considerando todos os valores de t (Linha 4), é retornada (Linha 23).

Algorithm 2 Busca Local

```
function BUSCALOCAL(S_0)
    S_melhor ← S_0                                  ▷ f() corresponde ao custo da solução
    for t ← 1 to n-1 do
        while melhorou = true do
            melhorou ← false
            for i ← 1 to n do
                for j ← 1 to n do
                    S_temp ← S_melhor.Swap(i, j)
                    S_temp.t ← t
                    if f(S_temp) < f(S_melhor) then
                        S_melhor ← S_temp
                        melhorou ← true
                    end if
                end for
            end for
        end while
    end for
    return S_melhor
end function
```

3.2. Iterated Local Search - ILS

As heurística escolhida para o CAP é baseada na Busca Local Iterada ou ILS (Iterated Local Search). Esta metaheurística aplica, repetidamente, uma busca local em um conjunto de soluções obtidas a partir de perturbações feitas em ótimos locais anteriormente visitados [Lourenço et al., 2010]. O algoritmo utiliza a busca local apresentada na Seção 3.1 e inicia sua execução a partir da solução produzida pela heurística construtiva proposta.

A heurística ILS implementada é constituída de uma série de iterações com sequências de perturbações e busca local. O critério de perturbação utilizado tem característica aleatória. Dado um leiaute π e duas posições p_1 e p_2 deste leiaute, escolhidas aleatoriamente, uma perturbação em π é a inversão na sequência entre os elementos na posição p_1 e p_2 do arranjo π. Exemplificando, se temos o leiaute $\pi = (5, 1, 7, 2, 4, 6, 3)$ e as posições $p_1 = 2$ e $p_2 = 5$, teremos um novo leiaute $\pi_2 = (5, \mathbf{4}, \mathbf{2}, \mathbf{7}, \mathbf{1}, 6, 3)$.

O Algoritmo 3 mostra como foi implementado o ILS. Inicia-se com uma solução inicial (Linha 2) e uma sequência de perturbações com tamanho dinâmico é executada. O parâmetro $fraqueza$ na Linha 3 está relacionado com o tamanho da perturbação. O tamanho da perturbação é dado por $tamanho = \frac{1}{fraqueza} * n$, onde n é o número de facilidades na solução. Na Linha 5 aplica-se uma perturbação na solução corrente e realiza-se uma busca local na solução perturbada (Linha 6). O processo continua sempre conservando a melhor solução corrente, até que o critério de parada seja alcançado. À medida que a solução não melhora seu valor de função objetivo, a fraqueza é decrementada aumentando o tamanho da perturbação (Linha 11). Neste caso o critério de parada foi para $\theta = 2$, ou seja, quando a perturbação tiver metade do tamanho da solução.

4. Resultados experimentais

As instâncias utilizadas para avaliar o algoritmo foram retiradas de um *benchmark* disponível em: *http://www.miguelanjos.com/flplib*. Algumas destas instâncias foram originalmente usadas para testar o SRFLP (Single Row Facility Layout Problem), problema que tem grande semelhança com o CAP. Contudo, o SRFLP organiza os elementos somente em um lado do corredor.

As instâncias foram separadas em três grupos. O primeiro grupo são de instâncias com $n \leq 15$:

Algorithm 3 Ierated Local Search

```
function ILS(S_0, fraqueza)
    S ← S_0
    θ ← fraqueza
    repeat
        S' ← Perturbacao(S, θ)
        S'' ← BuscaLocal(S')
        if f(S'') < f(S) then
            S ← S''
            θ ← fraqueza
        else
            θ ← θ − 1
        end if
    until θ = 2
    return S
end function
```

- duas instâncias de Simmons [1969] com tamanho $n = 10$ e $n = 11$
- uma instância de Amaral [2006] com $n = 15$

O segundo grupo contém 15 instâncias de Anjos [2009] com tamanhos 42, 49 e 56 (5 instâncias de cada tamanho). O terceiro grupo é constituído por 36 instâncias de tamanho $n = 60$ geradas por Ahonen et al. [2014]. Todas as instâncias foram utilizadas em [Ahonen et al., 2014], onde foram obtidos os melhores resultados até o momento.

As soluções obtidas pela heurística proposta nete trabalho serão comparadas com resultados presentes em [Ahonen et al., 2014], onde foram implementadas as heurísticas Busca Tabu e Simulated Annealing para o CAP. Os resultados apresentados são os melhores valores de função objetivo dentre 30 execuções. Da mesma forma, será mostrado o tempo médio de execução (das 30 execuções) para resolver cada instância.

O algoritmo ILS foi implementado em C++ e compilado com g++. O algoritmo foi executado em um Intel(R) Core(TM) i5-3470 Quad CPU com 3.2 GHz e 3.6 gigabytes de RAM com um sistema operacional Ubuntu 14.04.

O parâmetro $fraqueza$ para execução da ILS foi inicializado com valor 10. Neste caso a pertubação inicial tem um décimo do tamanho da solução. Os resultados estão divididos em três classes, são elas: instâncias com $n \leq 15$, com $42 \leq n \leq 56$ e com $n = 60$. Os resultados para cada classe são mostrados nas seções a seguir.

4.1. Para $n \leq 15$

Para as instâncias menores, $n = 10$ e $n = 11$, o melhor valor encontrado pelas heurísticas é o valor do ótimo global. A verificação pôde ser feita por um método exato [Ahonen et al., 2014]. A Tabela 1 apresenta os resultados, sendo os melhores valores de função objetivo e de tempo de execução destacados em negrito. Também mostra o leiaute para cada solução respectivamente. Podemos observar que para instâncias pequenas todas as implementações tiveram bons resultados. Contudo o ILS conseguiu encontrar os valores em muito menos tempo, obtendo os resultados em 2 a 3 ordens de grandeza mais rápido que os demais.

4.2. Para $42 \leq n \leq 56$

A Tabela 2 mostra os resultados para instâncias de tamanho $42 \leq n \leq 56$. Para estes tamanhos o ILS conseguiu melhores valores de função objetivo em apenas duas das quinze instâncias. Porém nas demais instâncias o ILS conseguiu valores de função objetivo muito próximos dos melhores valores da literatura. Os valores tarjados de cinza na tabela indicam que os mesmos ficaram a menos de $0,1\%$ de distância dos melhores valores. Note que o tempo de execução foi

Tabela 1: Resultados para $n \leq 15$

Instância	n	Função Objetivo			Tempo de Execução			Leiaute ILS
		SA	TS	ILS	SA	TS	ILS	
S10	10	**1374,5**	**1374,5**	**1374,5**	12,020	4,700	**0,004**	6-2-10-1-3 8-4-5-7-9
S11	11	**3439,5**	**3439,5**	**3439,5**	4,880	1,580	**0,006**	8-6-3-10-7-9 11-5-4-1-2
A15	15	**3195,0**	**3195,0**	**3195,0**	5,970	2,020	**0,015**	10-6-3-7-11-2-9 15-5-4-14-12-8-13-1

mais uma vez menor para todas as instâncias. O tempo de execução foi pelo menos uma ordem de grandeza menor, logo bons ou melhores resultados foram obtidos mais rapidamente. Os leiautes para estas instâncias são mostrados na Tabela 3.

Tabela 2: Resultados para $42 \leq n \leq 56$

Instância	n	Função Objetivo			Tempo de Execução		
		SA	TS	ILS	SA	TS	ILS
sko42_01_n	42	**12731,0**	**12731,0**	12738,0	25,480	89,590	**2,144**
sko42_02_n		**108016,5**	108020,5	108046,0	25,130	88,140	**2,355**
sko42_03_n		**86646,5**	86657,5	86668,5	21,670	90,950	**1,945**
sko42_04_n		**68708,0**	68711,0	68739,0	21,800	91,450	**2,190**
sko42_05_n		**124017,5**	124018,5	126694,0	22,120	88,860	**2,347**
sko49_01_n	49	**20470,0**	**20470,0**	20483,0	47,740	110,510	**2,441**
sko49_02_n		**208079,0**	208157,0	208156,0	47,410	113,940	**3,728**
sko49_03_n		**162196,0**	162249,0	162287,0	47,800	106,960	**4,558**
sko49_04_n		118260,5	118302,5	118272,0	47,630	108,600	**3,833**
sko49_05_n		332853,0	332972,0	**332834,0**	47,220	109,090	**5,497**
sko56_01_n	56	**31972,0**	**31972,0**	**31972,0**	92,610	179,120	**6,698**
sko56_02_n		**248225,0**	248247,0	248302,0	92,590	175,600	**11,865**
sko56_03_n		85190,0	85209,0	85241,0	93,350	174,400	**14,458**
sko56_04_n		**156666,0**	156701,0	156676,0	128,690	175,260	**8,538**
sko56_05_n		**296176,5**	296315,5	296220,0	142,370	175,230	**11,377**

4.3. Para $n = 60$

As instâncias de maior tamanho ($n = 60$), e também em maior quantidade, foram geradas aleatoriamente em [Ahonen et al., 2014]. O conjunto de instalações dessas instâncias contém s instalações de tamanhos pequenos distribuídos uniformemente entre os tamanhos 1 e 10 e $(n - s)$ instalações de tamanhos grandes distribuídos uniformemente entre 10 e 20. Cada c_{ij} tem probabilidade $\frac{p}{100}$ de ser positivo, com valores uniformemente distribuídos entre 1 e 10.

Sendo assim, para cada $s \in \{30, 40, 50, 60\}$ e para cada $p \in \{30, 60, 90\}$ foram geradas instâncias con características $\{n, s, p\}$, totalizando 36 instâncias.

Os resultados para estas instâncias e seus leiautes são exibidos na Tabela 4 e na Tabela 5 respectivamente. Na grande maioria das instâncias o SA mostrou-se superior. Porém o ILS proposto conseguiu resultados muito próximos dos ótimos locais encontrados pelo SA e com tempos de execução inferiores. Os valores próximos também estão tarjados de cinza na tabela. Foram encontrados valores a 0,1% do melhor valor conhecido em aproximadamente 2/3 das instâncias, sendo 2 deles melhores que os da literatura.

Tabela 3: Leiautes para $42 \leq n \leq 56$

Instância	Custo	Linhas
sko42_01_n	12738	25-8-11-34-17-5-40-3-1-21-9-18-14-19-31-4-27-36-33-22-16
		41-6-29-42-7-12-15-38-37-32-24-26-20-35-10-39-30-2-28-13-23
sko42_02_n	108046	2-15-31-9-42-18-19-12-26-28-40-5-23-38-7-41-34-24-11-8
		14-17-13-35-10-4-32-39-27-20-33-30-37-29-36-16-3-21-25-6-22-1
sko42_03_n	86668,5	29-19-27-41-5-11-7-17-25-30-39-3-35-26-13-28-37-40-24-15-9-21-32
		16-10-42-2-6-34-12-36-23-18-20-31-14-38-4-33-1-8-22
sko42_04_n	68739	4-33-15-14-12-17-20-13-32-42-19-18-5-30-39-28-38-37-36-1-25-6-8
		16-9-35-7-41-34-31-11-10-26-2-27-21-3-40-23-24-22-29
sko42_05_n	126694	8-29-17-25-16-38-37-24-33-21-7-42-11-22-39-31-19-14-32-28-5-23-13
		6-40-15-9-1-3-34-41-10-20-27-12-35-18-26-36-13-2-30
sko49_01_n	20483	4-44-32-30-24-49-7-16-11-8-48-31-15-6-34-10-42-29-37-38-14-20-13-28-46
		17-19-22-21-36-26-33-5-40-25-23-3-12-2-9-35-18-41-45-1-27-43-47-39
sko49_02_n	208156	28-36-21-30-16-32-7-11-23-22-24-31-48-20-47-1-41-14-38-9-34-12-45-29-39
		25-19-42-3-15-17-5-4-8-33-40-26-44-49-13-35-18-27-6-10-43-37-46-2
sko49_03_n	162287	28-36-21-30-15-16-11-24-22-49-8-44-20-13-48-41-14-38-34-27-12-37-46-2
		25-19-42-3-17-32-5-7-4-40-26-33-31-23-47-35-1-18-9-6-10-43-45-29-39
sko49_04_n	118272	37-14-10-41-3-12-29-38-46-42-20-6-31-48-8-25-36-19-4-21-24-7-32-22-44
		13-47-9-35-15-27-45-1-34-18-43-2-28-23-39-5-26-11-16-30-49-33-40-17
sko49_05_n	332834	22-44-11-36-30-26-25-16-8-32-3-23-49-48-2-28-46-20-18-9-13-37-45-14-1-43-47
		17-4-12-19-24-21-39-5-40-31-33-7-6-35-29-34-15-10-41-27-38-42
sko56_01_n	31972	21-43-25-15-45-6-38-24-8-22-50-7-1-28-20-12-48-23-29-9-14-41-13-47-35-40-3-39
		16-31-54-26-46-2-4-55-56-53-11-34-37-32-5-27-17-33-30-36-42-51-44-18-52-10-49-19
sko56_02_n	248302	15-17-54-21-7-4-20-45-19-31-53-38-43-55-50-32-35-5-39-12-14-47-23-49-42-10-9-44-30
		26-6-25-24-46-34-37-8-16-22-27-11-1-36-29-48-56-13-41-18-51-33-28-40-52-2-3
sko56_03_n	85241	19-54-56-22-50-9-38-24-53-36-35-52-7-13-25-55-3-28-34-39-26-20-31-12-41-48-46-1-21-43
		49-51-42-10-8-11-33-30-23-14-29-40-44-18-5-47-37-6-17-45-27-4-2-15-32-16
sko56_04_n	156676	41-25-1-16-46-37-34-26-17-12-6-31-43-53-28-24-55-36-7-11-51-44-13-18-40-30-42-49-19-32-35
		15-21-54-4-2-8-48-39-27-45-20-47-50-14-10-29-5-56-52-9-23-33-22-3-38
sko56_05_n	296220	38-41-53-54-15-17-51-23-21-6-46-31-34-55-35-7-33-47-28-8-27-50-52-44-39-11-48-12-18-32-49-42
		16-26-14-24-25-36-22-45-43-19-4-29-37-20-5-10-13-40-1-9-2-56-30-3

5. Conclusão

Este trabalho abordou o Problema de Alocação de Corredor, onde foi proposto um algoritmo de busca local iterada para o problema. Mostramos a estrutura de vizinhança utilizada e introduzimos uma estratégia de perturbação para busca local iterada. Os resultados foram comparados com duas heurísticas da literatura, SA e TS implementados em [Ahonen et al., 2014], utilizando instâncias de tamanhos menores ($n \leq 15$), médios ($42 \leq n \leq 56$) e maiores ($n = 60$). O ILS implementado conseguiu resultados equiparáveis aos da literatura e em menor tempo de execução. Em algumas instâncias foram obtidos melhores valores de função objetivo e em outras muito próximo dos melhores valores apresentados na literatura. Desse modo, acredita-se que o algoritmo proposto pode servir como boa abordagem para tratar o problema de alocação de corredor. Como o tempo de execução foi bastante reduzido, há espaço para melhorias e incorporação de outras características na heurística para melhorar ainda mais sua qualidade, que deixamos como trabalhos futuros.

Agradecimentos: Os autores agradecem às instituições CAPES e FAPEMIG por proporcionarem os recursos necessários para a realização deste trabalho.

Tabela 4: Resultados para $n = 60$

		Função Objetivo			Tempo de Execução		
Instância	n	SA	TS	ILS	SA	TS	ILS
CAP_60_d_30_L_30_1	60	**204103,0**	204208,0	204260,0	136,080	281,700	**13,256**
CAP_60_d_30_L_30_2		**193218,5**	193289,5	193572,0	135,630	314,670	**11,286**
CAP_60_d_30_L_30_3		**161486,5**	161582,5	161822,0	137,580	353,980	**20,472**
CAP_60_d_30_L_40_1		**135133,5**	135189,5	135456,0	137,880	345,100	**15,146**
CAP_60_d_30_L_40_2		**159114,0**	159224,0	159221,0	135,900	344,070	**13,825**
CAP_60_d_30_L_40_3		**158976,5**	159108,5	159100,0	137,470	365,870	**21,349**
CAP_60_d_30_L_50_1		**110507,5**	110611,5	110588,0	138,170	345,280	**21,918**
CAP_60_d_30_L_50_2		**115321,0**	115396,0	115421,0	136,870	312,460	**12,563**
CAP_60_d_30_L_50_3		**114165,0**	114261,0	114531,0	139,330	291,270	**10,009**
CAP_60_d_30_L_60_1		108124,0	108158,0	**105948,0**	136,070	274,270	**18,800**
CAP_60_d_30_L_60_2		**109942,5**	109983,5	110434,0	138,000	319,850	**12,880**
CAP_60_d_30_L_60_3		**91638,0**	91753,0	92030,0	137,120	350,380	**19,337**
CAP_60_d_60_L_30_1		**445381,5**	445550,5	445602,0	135,370	300,920	**20,704**
CAP_60_d_60_L_30_2		**407892,5**	407967,5	408082,0	133,870	305,080	**19,453**
CAP_60_d_60_L_30_3		**416935,5**	417043,5	417018,0	135,130	258,020	**22,984**
CAP_60_d_60_L_40_1		**313320,0**	313467,0	313404,0	134,800	291,150	**12,733**
CAP_60_d_60_L_40_2		**320758,5**	320808,5	320908,0	136,920	276,000	**19,914**
CAP_60_d_60_L_40_3		**363020,5**	363184,5	363700,0	134,270	326,670	**22,695**
CAP_60_d_60_L_50_1		**273423,0**	273500,0	273646,0	136,610	297,620	**14,100**
CAP_60_d_60_L_50_2		**269683,5**	269798,5	269766,0	137,110	305,780	**18,989**
CAP_60_d_60_L_50_3		**295424,0**	295529,0	295712,0	134,560	271,780	**12,201**
CAP_60_d_60_L_60_1		**227898,0**	227976,0	227956,0	137,590	287,130	**16,600**
CAP_60_d_60_L_60_2		**246523,0**	246639,0	247093,0	135,660	329,510	**18,574**
CAP_60_d_60_L_60_3		**206494,5**	206586,5	206538,0	133,460	319,770	**14,341**
CAP_60_d_90_L_30_1		**628859,0**	628915,0	628922,0	134,750	275,360	**15,099**
CAP_60_d_90_L_30_2		**561170,5**	561213,5	561194,0	133,860	289,420	**9,386**
CAP_60_d_90_L_30_3		**587824,5**	587874,5	587840,0	134,370	256,790	**11,912**
CAP_60_d_90_L_40_1		**474046,0**	474094,0	474080,0	135,650	255,200	**8,741**
CAP_60_d_90_L_40_2		**479963,0**	480007,0	480111,0	135,320	282,450	**17,139**
CAP_60_d_90_L_40_3		**512453,0**	512490,0	512456,0	135,130	270,030	**21,156**
CAP_60_d_90_L_50_1		**479692,0**	479721,0	479728,0	135,360	307,110	**11,965**
CAP_60_d_90_L_50_2		**445032,0**	445140,0	445220,0	133,360	281,360	**10,116**
CAP_60_d_90_L_50_3		**495059,5**	495083,5	495596,0	133,480	296,830	**16,199**
CAP_60_d_90_L_60_1		**385430,5**	385448,5	385512,0	134,300	273,860	**12,469**
CAP_60_d_90_L_60_2		344775,0	344816,0	**331192,0**	137,160	334,600	**10,685**
CAP_60_d_90_L_60_3		**411206,0**	411262,0	411270,0	133,930	256,570	**8,960**

Tabela 5: Leiautes para $n = 60$

Instância	Custo	Linhas
CAP_60_d_30_L_30_1	204260	41-59-45-60-34-33-5-29-30-13-44-22-26-14-25-8-4-1-2-16-15-19-23-28-6-3-12-31-11-58-38-54-51-43-57 48-46-55-52-42-27-47-24-21-32-37-7-17-18-20-10-49-9-36-40-39-56-35-53-50
CAP_60_d_30_L_30_2	193572	60-46-41-40-31-54-56-20-21-26-27-13-7-17-58-22-25-15-3-4-29-23-2-5-30-11-12-16-28-49-57-44-39-34-36-45 42-52-1-47-43-33-18-59-6-24-53-8-10-9-50-14-51-55-38-37-19-32-35-48
CAP_60_d_30_L_30_3	161822	38-53-57-50-51-41-45-26-30-9-13-22-4-6-34-19-27-15-46-23-8-16-5-17-14-12-52-31-28-48-54-47-32 35-49-59-55-58-10-39-24-11-44-3-7-43-20-29-25-1-21-36-18-56-2-60-42-40-33-37
CAP_60_d_30_L_40_1	135456	58-37-49-54-10-52-12-2-27-20-16-3-11-26-21-40-19-18-33-1-7-17-4-15-24-34-8-35-14-30-22-31-29-50-28-25-44-59 45-42-46-60-57-9-53-6-43-41-32-23-36-5-39-13-48-47-51-55-38-56
CAP_60_d_30_L_40_2	159221	59-52-35-3-41-47-22-2-7-11-1-18-8-26-23-49-30-10-32-9-17-38-33-34-20-39-19-5-13-31-4-44-46-48 56-55-45-42-51-16-24-29-25-43-37-12-6-21-27-36-40-14-60-58-50-15-54-28-57-53
CAP_60_d_30_L_40_3	159100	55-50-49-57-19-25-8-4-6-39-40-27-11-13-17-21-28-36-10-7-14-16-32-9-35-30-26-18-48-15-23-46-24-45-52 54-59-29-42-20-47-22-37-2-31-34-3-5-38-51-12-56-1-33-53-41-43-44-58-60
CAP_60_d_30_L_50_1	110588	55-48-9-24-31-5-28-14-49-50-34-26-30-32-47-15-16-7-45-35-36-46-17-19-12-29-3-23-43-1-25-18-58 59-60-52-38-11-2-53-40-6-13-21-39-44-41-42-27-33-54-20-10-4-37-22-8-56-57-51
CAP_60_d_30_L_50_2	115421	54-35-16-17-58-44-10-15-25-3-34-42-37-1-49-21-51-11-12-2-30-5-28-43-55-23-52-59 60-57-26-39-56-18-20-24-45-29-19-4-46-8-50-22-47-13-48-9-7-38-32-6-14-33-31-40-27-41-36-53
CAP_60_d_30_L_50_3	114531	11-4-47-33-60-24-51-5-18-25-41-43-10-31-38-8-16-48-12-30-28-3-20-50-32-7-6-36-23-35-22-29-9-54-42 52-53-57-58-26-44-34-27-19-37-17-21-55-13-15-45-59-14-40-1-46-2-49-39-56
CAP_60_d_30_L_60_1	105948	31-4-12-45-8-9-11-59-17-20-60-28-29-51-52-39-14-54-1-37-22-38-21-5-32-13-50-47-56-6 42-15-40-58-55-23-44-34-30-27-10-24-57-53-35-49-36-3-7-33-42-19-25-16-43-18-46-26-48-41
CAP_60_d_30_L_60_2	110434	2-26-49-42-15-4-56-50-57-25-18-51-35-58-52-44-11-32-54-9-7-14-45-41-53-34-21-38-36-39-60-10-30 22-20-13-27-31-24-37-6-48-33-55-40-1-17-43-5-19-59-46-47-29-8-28-16-23-3-12
CAP_60_d_30_L_60_3	92030	10-49-29-31-42-51-13-2-43-6-45-35-12-8-17-56-1-37-52-27-25-11-59-5-54-46-20-40-39-55-34-60-7-33-21-47-9 18-26-22-19-48-14-4-3-57-30-41-28-38-53-58-32-16-50-36-44-15-23-24
CAP_60_d_60_L_30_1	445602	49-35-34-47-57-27-52-1-48-24-11-2-23-20-15-25-26-22-13-14-21-16-5-7-9-12-31-19-41-50-53-60-43-33 38-32-39-37-55-36-58-29-17-18-10-40-28-3-4-6-30-8-46-56-45-44-54-51-42-59
CAP_60_d_60_L_30_2	408082	56-32-49-21-57-47-59-11-22-16-13-28-17-26-29-5-19-23-1-2-3-8-6-9-4-20-39-24-30-41-33-31-38-46-54-50 45-35-34-48-53-14-40-60-44-7-10-27-15-12-18-25-58-51-42-36-37-55-52-43
CAP_60_d_60_L_30_3	417018	56-37-10-42-38-4-13-16-33-28-24-8-20-18-25-26-3-17-27-19-12-9-29-2-22-21-14-54-11-57-40-51-36-41-49 47-55-43-53-46-34-32-39-44-23-5-7-1-15-6-30-50-58-48-35-52-59-45-60-31
CAP_60_d_60_L_40_1	313404	56-47-45-51-38-30-37-34-20-18-22-21-40-7-26-4-31-19-5-33-29-17-15-23-12-39-8-6-13-52-16-1-55-42-46-60 48-50-41-10-9-11-2-3-14-36-24-27-25-35-28-44-32-59-43-58-53-49-54-57
CAP_60_d_60_L_40_2	320908	42-55-54-49-53-28-4-8-30-27-12-6-31-29-26-22-1-10-3-16-19-5-23-24-9-11-14-18-39-37-17-59-43-60-45 50-46-52-44-47-13-32-21-25-15-7-40-33-35-2-34-20-38-41-57-36-58-51-56-48
CAP_60_d_60_L_40_3	363700	58-45-41-47-54-40-8-21-36-37-10-6-33-2-35-28-7-12-18-25-24-27-15-11-3-17-26-14-19-30-31-57-44-42-49 56-43-59-46-60-16-20-23-29-39-38-53-13-22-34-32-1-9-5-4-51-48-50-55-52
CAP_60_d_60_L_50_1	273646	51-57-44-50-32-6-48-29-21-33-8-38-36-9-27-43-19-14-28-30-45-18-1-23-7-39-16-42-11-37-13-22-46-41-47-59-58 60-56-53-15-17-3-5-26-34-4-24-49-20-12-31-25-10-2-35-40-54-55-52
CAP_60_d_60_L_50_2	269766	58-56-55-59-43-14-28-13-3-31-41-45-50-49-2-5-38-44-39-37-27-6-21-7-1-17-18-8-20-4-60-33-9-52 51-57-30-19-12-10-23-47-11-22-36-35-24-16-29-26-42-15-46-34-40-48-32-54-25-53
CAP_60_d_60_L_50_3	295712	53-19-58-7-23-16-37-48-39-8-28-31-13-2-3-5-12-21-15-33-18-41-34-22-38-46-43-1-25-50-35-26-32-56-51 24-17-55-4-9-59-30-40-42-14-29-27-20-10-6-36-49-11-44-47-54-45-57-60-52
CAP_60_d_60_L_60_1	227956	44-25-23-32-42-28-40-52-12-37-48-36-34-7-8-29-3-58-4-1-18-16-43-41-53-30-45-60-31-2-10 27-13-33-39-15-6-26-20-17-5-50-24-19-57-11-59-21-9-22-56-55-38-46-47-35-54-51-49-14
CAP_60_d_60_L_60_2	247093	32-44-31-11-25-28-39-18-2-19-50-41-14-10-55-46-53-48-56-58-6-36-52-49-60-8-29-5-26-24-16-1-59 13-17-43-15-7-34-27-3-38-23-22-12-54-21-33-40-4-30-35-47-37-42-45-20-51-57-9
CAP_60_d_60_L_60_3	206538	15-1-8-47-14-19-40-59-18-11-30-44-24-34-25-52-12-23-37-60-21-26-13-54-45-29-2-42-57-22-10-38-4 27-36-48-51-31-53-20-55-46-17-6-28-39-50-41-43-5-35-16-56-3-9-33-32-7-58-49
CAP_60_d_90_L_30_1	628922	37-52-55-31-41-1-54-36-23-24-29-25-5-19-6-12-26-30-13-10-16-3-4-2-15-35-27-59-45-42-34-46-51 39-56-50-33-43-38-60-57-17-9-14-11-22-20-28-8-18-53-21-32-7-47-48-58-44-49-40
CAP_60_d_90_L_30_2	561194	55-49-14-59-50-33-34-12-25-5-16-9-18-19-17-30-3-1-29-20-10-22-28-11-6-46-4-32-37-54-31-53-44 56-60-41-43-52-35-51-26-40-15-23-2-13-8-24-27-21-7-36-45-42-39-58-38-57-47-48
CAP_60_d_90_L_30_3	587840	49-55-42-36-15-44-35-52-22-14-26-7-28-12-8-25-30-24-4-21-9-18-23-3-1-27-45-59-39-41-46-53-37 32-34-60-56-33-48-58-31-10-20-19-2-13-16-17-11-29-6-5-47-40-51-57-50-43-38-54
CAP_60_d_90_L_40_1	474080	52-44-50-53-59-24-19-18-26-35-17-34-28-36-6-13-14-27-7-31-32-37-11-15-20-4-3-5-22-48-45-42-46 56-58-47-60-54-43-10-21-40-1-38-23-16-12-30-25-8-39-9-33-29-2-49-57-41-51-55
CAP_60_d_90_L_40_2	480111	55-47-43-50-60-13-15-12-31-21-19-27-18-1-36-24-7-32-3-22-37-39-17-2-23-8-25-6-45-34-56-52-48 42-44-57-58-28-49-20-4-29-30-10-35-26-33-38-16-14-5-11-9-40-53-41-54-46-51-59
CAP_60_d_90_L_40_3	512456	44-60-55-41-49-32-27-25-35-40-24-3-8-5-30-11-7-31-23-17-28-10-4-15-22-13-56-57-54-42-43 51-52-46-53-59-29-9-36-1-21-37-39-12-20-33-6-19-2-26-18-38-34-16-14-47-58-45-50-48
CAP_60_d_90_L_50_1	479728	55-57-36-1-22-33-30-31-9-12-34-26-10-3-4-46-39-49-29-11-20-14-50-23-35-8-48-43-21-18-54-51 58-59-52-24-15-7-25-5-17-6-41-16-40-44-37-19-32-42-38-2-27-28-45-47-60-13-56-53
CAP_60_d_90_L_50_2	445220	59-60-21-22-47-40-35-16-5-48-6-3-42-15-25-36-17-12-38-37-26-49-20-9-11-8-34-7-13-4-52-51-57 56-54-18-14-44-1-33-29-2-10-39-43-23-31-27-46-24-50-30-19-28-45-53-32-41-58-55
CAP_60_d_90_L_50_3	495596	54-56-2-21-5-9-47-44-18-7-17-14-1-28-25-37-45-20-40-19-36-15-32-39-43-16-48-4-38-26-10-55-58 51-59-52-12-46-23-24-30-27-34-31-50-13-49-35-11-29-6-41-42-3-33-8-22-53-57-60
CAP_60_d_90_L_60_1	385512	17-30-50-16-40-10-23-37-38-15-35-3-49-36-12-2-18-26-7-19-29-58-59-54-13-39-56-5-24-51-48-28 4-45-22-1-44-52-46-11-57-47-42-34-20-31-14-53-60-21-55-8-27-6-33-32-43-9-25-41
CAP_60_d_90_L_60_2	331192	39-33-26-16-35-56-30-11-20-12-36-5-19-39-46-14-55-6-58-53-47-28-54-34-27-23-25-38-10-3-44-31-45-24-48-49 7-40-32-15-18-4-22-51-60-2-17-29-13-57-37-8-21-52-43-59-9-41-50-42
CAP_60_d_90_L_60_3	411270	15-39-41-17-55-35-7-36-59-30-25-19-22-52-5-32-56-10-37-51-26-14-23-49-21-13-46-45-24-18-31-60-12-3 6-44-33-28-43-1-42-8-54-9-16-40-4-48-20-53-29-38-47-34-58-11-57-50-27-2

Referências

Ahonen, H., de Alvarenga, A. G., e Amaral, A. (2014). Simulated annealing and tabu search approaches for the corridor allocation problem. *European Journal of Operational Research*, 232 (1):221–233.

Amaral, A. R. (2006). On the exact solution of a facility layout problem. *European Journal of Operational Research*, 173(2):508–518.

Amaral, A. R. (2012). The corridor allocation problem. *Computers & Operations Research*, 39 (12):3325–3330.

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

Chung, J. e Tanchoco, J. (2010). The double row layout problem. *International Journal of Production Research*, 48(3):709–727.

Ghosh, D. e Kothari, R. (2012). Population heuristics for the corridor allocation problem. *IIMA Working Papers WP2012-09-02, Indian Institute of Management Ahmedabad, Research and Publication Department*, <http:// EconPapers.repec.org/RePEc:iim:iimawp:11453>.

Kothari, R. e Ghosh, D. (2013). Insertion based Lin-Kernighan heuristic for single row facility layout. *Computers & Operations Research*, 40(1):129–136.

Lourenço, H. R., Martin, O. C., e Stützle, T. (2010). Iterated local search: Framework and applications. In Gendreau, M. e Potvin, J.-Y., editors, *Handbook of Metaheuristics*, p. 363–397. Springer.

Simmons, D. M. (1969). One-dimensional space allocation: an ordering algorithm. *Operations Research*, 17(5):812–826.