UM ALGORITMO SIMULATED ANNEALING E UM HÍBRIDO VNS/ILS PARA RESOLVER O PROBLEMA DE LAYOUT EM MÚLTIPLAS LINHAS

Bernardo De Polli Cellin
Programa de Pós-Graduação em Informática - Universidade Federal do Espírito Santo
Av. Fernando Ferrari, No 514, CEP: 9060-900, Vitória, Espírito Santo
bcellin@hotmail.com

André Renato Sales Amaral
Programa de Pós-Graduação em Informática - Universidade Federal do Espírito Santo
Av. Fernando Ferrari, No 514, CEP: 9060-900, Vitória, Espírito Santo
amaral@inf.ufes.br

RESUMO

Problemas de layout de facilidade são NP-difíceis e possuem diversas aplicações na indústria. O problema de layout de facilidades em múltiplas linhas, ou Multi row Facility Layout Problem (MRFLP), considera várias facilidades que serão alocadas em linhas paralelas. O objetivo do MRFLP é encontrar uma ordenação de facilidades para cada linha a fim de minimizar uma função de custo total. Neste artigo implementações da meta-heurística Simulated Annealing e de um algoritmo híbrido de VNS/ILS são propostas para resolver o MRFLP. Resultados computacionais para instâncias da literatura demonstram a eficiência dos algoritmos.

PALAVRAS CHAVE. Problema de facilidades em múltiplas linhas, Simulated Annealing, Algoritmo Híbrido VNS/ILS.

Tópicos principais: Meta-heurísticas, Otimização combinatória.

ABSTRACT

Facility layout problems are NP-hard and have many applications in industry. The Multi Row Facility Layout Problem (MRFLP) considers several facilities that will be allocated on parallel lines. The goal of the MRFLP is to find an ordering of facilities for each line so as to minimize a total cost function. In this article implementations of the meta-heuristic Simulated Annealing and a hybrid algorithm VNS / ILS are proposed to solve the MRFLP. Computational results on instances of the literature demonstrate the efficiency of the algorithms.

KEYWORDS. Multi Row Facility Layout Problem, Simulated Annealing, Hybrid Algorithm VNS/ILS.

Paper topics: Metaheuristics, Combinatorial Optimization.

Introdução

Vários Problemas de Layout de Facilidades podem ser encontrados na literatura. O Problema de Layout de Facilidades em linha única, ou Single-row Facility Layout Problem (SRFLP) busca um layout de várias facilidades em uma única linha, de modo a minimizar uma função de custo. Um caso especial do SRFLP é o Single-row Equidistant Facility Layout Problem (SREFLP), no qual as facilidades são todas de mesmo tamanho Palubeckis [2012].

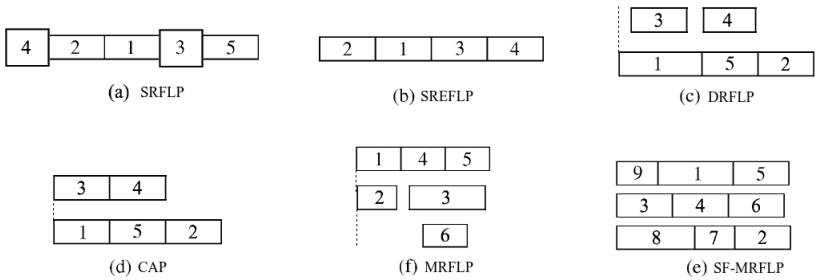
O Double-row Facility Layout Problem (DRFLP) considera o layout de facilidades em duas linhas paralelas horizontais. Como o corredor definido pelas linhas paralelas tem largura desprezível, somente as distâncias horizontais entre facilidades são consideradas. Outro problema de ordenação de facilidades em duas linhas paralelas é o Problema de Alocação em Corredor, ou Corridor Allocation Problem (CAP), também conhecido como Space-free Double-row Facility Layout Problem (SF-DRFLP). A diferença entre o CAP e o DRFLP é que no CAP não há espaços entre as facilidades dispostas em uma mesma linha e todas as alocações de facilidades nas linhas iniciam-se de um ponto comum Amaral [2012]; já no DRFLP, pode haver espaços entre as facilidades Chung e Tanchoco [2010]. Essa característica faz com que o DRFLP seja de natureza contínua, diferentemente do CAP e SRFLP. O espaço necessário entre duas facilidades adjacentes no DRFLP é chamado de *clearance*. As clearances são necessárias para problemas nos quais máquinas precisam de um espaço para manutenção.

Mais geral que os problemas de layout em linha discutidos anteriormente é o Problema de Layout de Facilidades em múltiplas linhas, ou Multi-row Facility Layout Problem (MRFLP). O MRFLP considera um conjunto de facilidades retangulares que precisam ser atribuídas a um determinado número de linhas e, em seguida, posicionadas nessas linhas a fim de minimizar uma função de custo total.

Um caso particular do MRFLP é Parallel Row Ordering Problem (k-PROP), que considera que cada facilidade está previamente alocada a uma das k linhas paralelas e o objetivo é encontrar o melhor arranjo de facilidades nas suas respectivas linhas de forma a minimizar uma função de custo Amaral [2013]. Outra variação do MRFLP é o Problema de Facilidades em Múltiplas Linhas sem Espaçamento, ou Space-free Multi-row Facility Layout Problem (SF-MRFLP), o qual considera que as facilidades devem ser arranjadas em múltiplas linhas com nenhum espaço entre elas Hungerländer e Anjos [2015]; e que a facilidade mais a esquerda em uma linha tem sua parede esquerda sobre uma linha vertical de referência. O MRFLP e o SF-MRFLP são os problemas tratados neste trabalho.

As características e diferenças de cada um dos problemas discutidos até aqui podem ser visualizadas nos exemplos da Figura 1.

Figura 1: Exemplos de Problemas de Layouts

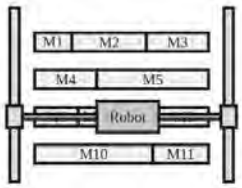


Na literatura podem-se encontrar várias aplicações para o MRFLP como o problema da disposição de componentes em computadores, ou Computer Backboard Wiring Problem Steinberg [1961], Layouts com Múltiplos Andares com Elevador (Multi-floor Layout Problem) Kochhar e Heragu [1998], planejamento de Campus Dickey e Hopkins [1972], Scheduling Geoffrion e Graves [1976], design de teclados Pollatschek et al. [1976], Layout de Hospitais Elshafei [1977], Balanceamento de Rotores Hidráulicos de Turbinas Laporte e Mercure [1988], Análise Numérica

Brusco e Stahl [2000] e Geração de Layouts de Memória em Processadores Digitais de Sinal Wess e Zeitlhofer [2004]. Assim como aplicações na indústria, como em Layout de Máquinas em Sistemas Automatizados de Manufatura Heragu e Kusiak [1988], Layout de Facilidades com áreas diferentes Lee e Lee [2002] e em Sistemas Flexíveis de Manufatura, ou Flexible Manufacturing Systems (FMS), nas quais operam Automated Guided Vehicles (AGV) ou Gantry Robots Nearchou [2006].

A Figura 2 mostra como seria um Layout de uma instância do MRFLP considerando um Gantry Robot posicionado acima das máquinas controlando o fluxo de material entre essas máquinas.

Figura 2: Gantry Robot em um FMS



Fonte: Hungerländer e Anjos [2015]; Heragu e Kusiak [1988]

Uma entrada para o MRFLP especifica um número n de facilidades a serem arranjadas no layout, o comprimento l_i de cada facilidade i, o fluxo $c_{i,j}$ entre cada par $\{i, j\}$ de facilidades e o numero k de linhas paralelas.

A formulação do MRFLP leva em conta a distância d_{ij}^{π} entre os centros de cada par distinto de facilidades i e j posicionadas em um layout π, multiplicada pelo fluxo total de materiais c_{ij} entre essas facilidades, como pode ser visto na Eq. 1.

$$\min_{\pi} \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} c_{ij} d_{ij}^{\pi} \tag{1}$$

A distância de centro a centro usada no cálculo depende da métrica usada em cada abordagem. Se for definida como a distância euclidiana, por exemplo, ela será definida pela raiz quadrada da soma das distâncias no eixo x e y ao quadrado. Neste trabalho, usa-se a mesma métrica de Hungerländer e Anjos [2015], que considera somente a distância horizontal entre os centros das facilidades.

Na próxima seção será feita uma breve revisão bibliográfica sobre o MRFLP, na seção 3 são propostos para o problema um algoritmo Simulated Annealing e um algoritmo híbrido das metaheurísticas Iterated Local Search (ILS) e Variable Neighbouhood Search (VNS). Na seção 4 serão mostrados resultados computacionais dos algoritmos e finalmente na seção 5 serão apresentadas conclusões.

Revisão Bibliográfica

Pesquisas sobre problemas de layout de facilidades em linha têm utilizado abordagens exatas ou meta-heurísticas.

O CAP já foi resolvido por Amaral [2012], com o uso de Programação Inteira Mista; por Ghosh e Kothari [2012], com uso de Algoritmos Genéticos com Buscal Local e Scatter Search com Path Relinking; por Hungerländer e Anjos [2012] com otimização semidefinida e heurísticas para obter soluções viáveis de uma relaxação de programação semidefinida e por Ahonen et al. [2014], com Simulated Annealing e Busca Tabu.

O DRFLP foi considerado por Chung e Tanchoco [2010], que propuseram um modelo de programação inteira mista para o problema. Pouco tempo depois, Zhang e Murray [2012] publicaram um trabalho corrigindo este modelo. Recentemente, Hungerländer e Anjos [2015] resolveram o SF-DRFLP com modelo de relaxação semi-definida e heurística.

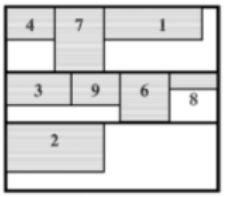
Para o k-PROP, uma formulação de programação inteira foi proposta por Amaral [2013] e uma meta-heurística Simulated Annealing foi apresentada por Cellin e Amaral [2015] para k = 2.

Hungerländer e Anjos [2015] usaram um modelo de relaxação semidefinida para o k-PROP com espaços entre as facilidades. Para a atribuição de facilidades às linhas no k-PROP, os autores fizeram um balanceamento do número de facilidades atribuídas a cada linha do layout. Em seus testes eles consideram problemas com $2 <= k <= 5$.

Para o MRFLP, muitos trabalhos já foram publicados, como pode ser visto em Surveys sobre FLPs como Drira et al. [2007], Jain et al. [2013] e Mavridou e Pardalos [1997].

Lee e Lee [2002] utilizaram uma hibridização com Algoritmos Genéticos, Simulated Annealing e Busca Tabu para resolver o problema de múltiplas linhas com facilidades de áreas heterogêneas. Um layout válido para esse trabalho pode ser visualizado na Fig. 3.

Figura 3: Layout com áreas diferentes



Fonte: Lee e Lee [2002]

Ficko et al. [2004] propuseram um Algoritmo Genético para resolver o MRFLP baseando-se em um sistema FMS com um robô AGV que circula em corredores entre as linhas. Miao e lin Xu [2009] propuseram uma hibridização de Algoritmo Genético com Busca Tabu para resolver um modelo para o MRFLP. Ficko et al. [2010] fizeram uma implementação eficiente da meta-heurística Particle Swarm Optimization (PSO) para encontrar soluções para o problema de layout irrestrito em um FMS.

Jankovits et al. [2011] resolveram o Problema de Layout com áreas desiguais. Tasadduq et al. [2011] utilizaram a meta-heurística Algoritmo Memético, que envolve buscas locais no procedimento do Algoritmo Genético.

Hungerländer e Anjos [2012] consideraram o SF-MRFLP e propuseram relaxações semi-definidas para encontrar limitantes inferiores e uma heurística para obter layouts viáveis a partir das soluções dessas relaxações. Já Anjos et al. [2015] consideraram o Multirow Equidistant Layout Problem. Eles propuseram dois modelos para o problema: um de programação linear inteira e um de otimização semi-definida. Eles mostraram resultados com até 5 linhas paralelas.

Continuando os estudos anteriores, Hungerländer e Anjos [2015] fizeram uma extensão do modelo criado para o SF-MRFLP e propuseram uma nova modelagem para o MRFLP, tratando-o como um problema discreto. Eles consideram que cada facilidade pode ser atribuída a qualquer linha, que todas as linhas possuem a mesma altura, que todas as distâncias entre linhas são iguais e que todos as facilidades possuem a mesma altura, igual à da linha. Usaram uma relaxação semi-definida e uma heurística para tornar as soluções viáveis para resolver não só o SF-MRFLP mas também problemas relacionados como o MRFLP, PROP e DRFLP.

Abordagem Proposta

Neste trabalho são apresentados algoritmos para resolver o SF-MRFLP e MRFLP considerando que tem-se um gantry robot para manipular materiais entre máquinas em um FMS, como discutido em Hungerländer e Anjos [2015]. Nesta abordagem, a distância de centro a centro entre um par de facilidades distinto pode ser calculada considerando-se altura da linha l_k para a linha k, e tamanho de corredor w arbitrários. Como a abordagem de Hungerländer e Anjos [2015] não considera espaços entre as linhas, não considerou-se também neste trabalho.

Como se trata de um problema NP-hard, aqui serão implementados algoritmos que utilizam meta-heurística para encontrar boas soluções em tempo computacional viável. As meta-heurísticas

que foram utilizadas neste trabalho para comparação são: Simulated Annealing e um algoritmo híbrido VNS/ILS.

Heurística Construtiva

Antes de aplicar as meta-heurísticas, uma solução viável inicial é construída com um procedimento heurístico. Foram usadas três tipos diferentes de heurísticas construtivas: uma heurística construtiva determinística e gulosa em Algoritmo 1, outra aleatória e gulosa em Algoritmo 2 e outra totalmente aleatória em Algoritmo 3. O *comprimento de uma linha*, utilizado no Algoritmo 1, consiste da posição mais a direita alcançada pela alocação de facilidades na linha, ou seja, é a soma de todos comprimentos das facilidades que ocupam aquela linha e dos espaços entre elas.

Algoritmo 1 Heurística Construtiva Determinística e Gulosa

inicializa a lista L com todas as facilidades que podem entrar no layout
escolher uma facilidade i com menor comprimento
inserir facilidade i no layout
$L \leftarrow L - \{i\}$
enquanto $L \neq \emptyset$ **faça**
    para para cada facilidade j em L **faça**
        calcular a contribuição para o valor objetivo caso j fosse adicionado à linha l de menor comprimento ou de menor rótulo em caso de empate
    fim para
    seja $j*$ a facilidade com a menor contribuição para o valor objetivo
    inserir $j*$ no layout na linha l após a facilidade anterior
    $L \leftarrow L - \{j*\}$
fim enquanto

Algoritmo 2 Heurística Construtiva Aleatória e Gulosa

inicializa a lista L com todas as facilidades que podem entrar no layout
enquanto $L \neq \emptyset$ **faça**
    escolher uma facilidade i aleatória em L
    inserir i na primeira linha do layout
    $L \leftarrow L - \{i\}$
    ordenar L por ordem decrescente de fluxo c_{ij} entre i e cada facilidade j de L
    para para cada linha k no layout, a começar da segunda linha **faça**
        inserir a primeira facilidade j da lista L na linha k
        $L \leftarrow L - \{j\}$
    fim para
fim enquanto

Algoritmo 3 Heurística Construtiva Aleatória

inicializa a lista L com todas as facilidades que podem entrar no layout
enquanto $L \neq \emptyset$ **faça**
    escolher uma facilidade i aleatória em L
    encontrar a linha k com menor comprimento
    inserir i na linha k
    $L \leftarrow L - \{i\}$
fim enquanto

Estruturas de Vizinhança

Neste trabalho foram utilizadas alguns tipos de estrutura de vizinhança. Para o SF-MRFLP, foram utilizadas dois tipos: a troca simples e a inserção. Já no MRFLP, acrescenta-se mais 2 estruturas de vizinhança: a movimentação de uma facilidade e o deslocamento de linha.

A *troca simples* consiste em trocar duas facilidades de posição. Após a troca, no caso do SF-MRFLP muitas vezes desloca-se muitas facilidades ou a linha inteira porque não deve haver sobreposições nem espaços entre facilidades. Já para o MRFLP, após a troca simples somente há a verificação de que não há sobreposição; se houver alguma, apenas as facilidades sobrepostas serão deslocadas de modo a tornar o layout viável.

A *inserção* consiste em inserir uma facilidade que está em uma linha em outra linha, mantendo sua posição relativa e também respeitando-se as restrições de que não pode haver espaços entre facilidades no SR-MRFLP.

A estrutura de vizinhança de *movimentação de uma facilidade* muda aleatoriamente a posição em relação ao eixo horizontal de uma facilidade sem alterar sua posição relativa na linha. Por exemplo, se a facilidade for a segunda em sua linha, esse método irá sortear uma nova posição entre a primeira e a terceira facilidade de sua linha. Como o MRFLP é um problema de natureza contínua, a variável recebe um número real aleatório para identificar a nova posição.

O *deslocamento de linha* primeiramente sorteia uma facilidade aleatória em uma linha. Após isso, desloca todas as outras facilidades ordenadas a direita da de referência, aproximando-as da facilidade imediatamente a esquerda de cada uma, de modo que o espaço entre as facilidades a partir da de referência seja reduzido a zero. É um movimento que leva a soluções com menores gaps entre facilidades.

Simulated Annealing

A primeira meta-heurística utilizada, Simulated Annealing, baseia-se na teoria da termodinâmica de resfriamento dos materiais para otimizar uma solução de um problema de otimização combinatória. Este método faz pequenas mudanças na solução iterativamente, buscando melhores soluções vizinhas e recebe os seguintes parâmetros de entrada: temperatura inicial T_0, número de perturbações por iteração de temperatura *P*, número máximo de iterações SA_{max} e taxa de resfriamento α. Neste trabalho, caso nenhuma solução melhor fosse aceita em uma iteração de temperatura, a mesma é incrementada de um pequeno fator β. A perturbação usada escolhe aleatoriamente um entre dois movimentos de troca: a troca simples ou a inserção. Já no caso contínuo do MRFLP, outros dois movimentos foram adicionados, que são o da movimentação da facilidade em sua linha e o deslocamento de linha. O pseudocódigo utilizado pode ser observado no Algoritmo 4.

Foi feita uma paralelização em termos de threads de processamento, iniciando-o com as três heurísticas construtivas. Cada uma delas gerou uma solução inicial em uma thread de programação diferente. Para cada uma das soluções iniciais diferentes foram feitas perturbações e quedas de temperatura independentes. Apesar de haver um custo para controlar a concorrência de memória compartilhada, o uso de paralelização faz com que mais soluções sejam exploradas. A biblioteca utilizada para essa paralelização foi a openMP. Os parâmetros utilizados foram: T_0 = 3500; α = 0.975; $\beta = 1/\alpha$; $SA_{max} = 650$.

Algoritmo Híbrido

O VNS, ou Variable Neighborhood Search, foi proposto por Mladenovic e Hansen [1997]. O método consiste em fazer buscas locais no espaço de soluções e a ideia principal é fazer mudanças nas estruturas de vizinhanças para tentar explorar mais soluções e escapar de ótimos locais. Normalmente as meta-heurísticas aceitam a degradação da solução corrente para fugir de mínimos (ou máximos) locais mas o VNS não trabalha com tal procedimento. Consiste apenas em realizar a busca, alterando a estrutura de vizinhança quando for necessário.

Iterated Local Search, ou ILS, é uma meta-heurística que tem o objetivo de, iterativamente, aplicar buscas locais e degradar a solução com perturbações Lourenço et al. [2003]. A busca local

Algoritmo 4 Implementação do Simulated Annealing

obtém uma solução inicial s
$T \leftarrow T_0$
para $i \leftarrow 1$ **to** SA_{max} **faça**
  para $j \leftarrow 1$ **to** P **faça**
    perturba solução gerando solução vizinha s'
    $\Delta \leftarrow$ f(s') - f(s)
    se $\Delta < 0$ **então**
      $s \leftarrow s'$
    senão
      faça $s \leftarrow s'$ de acordo com probabilidade $e^{\frac{-\Delta}{T}}$
    fim se
  fim para
  se nenhuma solução melhor foi aceita **então**
    $T \leftarrow \beta * T$
  fim se
  $T \leftarrow \alpha * T$
fim para

leva a um ótimo local e, assim como outras meta-heurísticas, o ILS tenta escapar destruindo a solução corrente com uma perturbação que seja capaz de levar a outras soluções no espaço de busca.

Neste trabalho fez-se uma hibridização entre as duas meta-heurísticas, utilizando elementos do ILS, como a perturbação, e do VNS como a mudança de estrutura de vizinhança. A cada iteração o algoritmo proposto faz buscas locais utilizando diferentes estruturas de vizinhança, já que um ótimo local para uma estrutura de vizinhança pode não ser um ótimo local em relação a outra. Após serem aplicadas as buscas, a solução é degradada com uma perturbação, para poder explorar outras soluções no espaços de soluções. A força da perturbação varia de acordo com a variável p, que conta o número de iterações executadas sem encontrar soluções melhores. Assim como no Simulated Annealing, as buscas locais utilizam as estruturas de vizinhanças citadas anteriormente. O pseudocódigo utilizado pode ser visualizado no Algoritmo 5.

Algoritmo 5 Algoritmo Híbrido

obtém uma solução inicial s
para $i \leftarrow 1$ **to** i_{max} **faça**
  $p \leftarrow 1$
  enquanto $p \leq p_{max}$ **faça**
    perturba solução gerando solução vizinha s' com intensidade p
    efetua busca local utilizando cada uma das vizinhanças
    se a solução s' for melhor **então**
      $s \leftarrow s'$
      $p \leftarrow 1$
    senão
      $p \leftarrow p + 1$
    fim se
  fim enquanto
fim para

Assim como o Simulated Annealing, o Algoritmo híbrido também foi iniciado com as três heurísticas construtivas em threads diferentes utilizando a biblioteca openMP. O parâmetro p_{max}

utilizado no algoritmo foi igual ao número de facilidades n.

Resultados Computacionais

Os algoritmos meta-heurísticos foram implementados em linguagem C e os testes foram feitos em máquina com processador Intel core i7 2.0GHz com 8GB de memória RAM, utilizando o sistema operacional Ubuntu 14.04.

Os experimentos computacionais foram realizados baseados em instâncias usadas para o PROP, SF-MRFLP (CAP) e MRFLP em Hungerländer e Anjos [2015] e Ahonen et al. [2014]. Todas elas podem ser encontradas na FLPLIB (http://www.miguelanjos.com/flplib). Para cada instância, os algoritmos foram executado 10 vezes. A coluna DP mostra o desvio padrão da solução, a coluna Min o menor valor de solução encontrado nas execuções e a coluna Tempo mostra o tempo médio obtido.

Resultados obtidos para o CAP

Primeiramente resolveu-se o CAP e comparou-se os resultados dos algoritmos com os valores obtidos em Hungerländer e Anjos [2015], que utilizam uma heurística baseada em programação semi-definida (SDP), e Ahonen et al. [2014], que utilizaram uma eficiente implementação do Simulated Annealing (SA Ahonen et al., 2014). Os resultados podem ser observados na Tabela 1.

Tabela 1: Resultados para o CAP

Instância	SDP		SA (Ahonen et al., 2014)		SA			VNS/ILS		
	Min	Tempo (s)	Min	Tempo (s)	Min	DP (%)	Tempo (s)	Min	DP (%)	Tempo (s)
H_5	**450**	4,80	-	-	**450**	0,00	0,283	**450**	0,00	0,016
Rand_5	**52,5**	4,50	-	-	**52,5**	0,00	0,269	**52,5**	0,00	0,013
H_6	**720**	11,10	-	-	**720**	0,00	0,292	**720**	0,00	0,022
Rand_6	**190,5**	13,80	-	-	**190,5**	0,00	0,288	**190,5**	0,00	0,023
H_7	**1700**	25,20	-	-	**1700**	0,00	0,307	**1700**	0,00	0,036
Rand_7	**166**	31,70	-	-	**166**	0,00	0,313	**166**	0,00	0,037
H_8	**2385**	91,30	-	-	**2385**	0,00	0,330	**2385**	0,00	0,056
S_8	**408**	91,60	-	-	**408**	0,00	0,331	**408**	0,00	0,054
SH_8	**1135,5**	87,40	-	-	**1135,5**	0,00	0,332	**1135,5**	0,00	0,057
Rand_8	**205**	82,20	-	-	**205**	0,00	0,332	**205**	0,00	0,054
S_9	**1181,5**	253,40	**1181,5**	1,14	**1181,5**	0,00	0,358	**1181,5**	0,00	0,089
SH_9	**2294,5**	251,90	**2294,5**	1,15	**2294,5**	0,00	0,367	**2294,5**	0,00	0,081
Rand_9	**492,5**	252,60	-	-	**492,5**	0,00	0,362	**492,5**	0,00	0,079
S_10	**1374,5**	713,00	**1374,5**	1,58	**1374,5**	0,00	0,396	**1374,5**	0,00	0,115
Rand_10	**838**	698,20	-	-	**838**	0,00	0,407	**838**	0,00	0,118
S_11	**3439,5**	2127,00	**3439,5**	2,02	**3439,5**	0,00	0,451	**3439,5**	0,00	0,172
Am12a	-	-	**1529**	2,43	**1529**	0,00	0,481	**1529**	0,00	0,250
Am12b	-	-	**1609,5**	2,43	**1609,5**	0,00	0,463	**1609,5**	0,00	0,242
Am13a	-	-	**2467,5**	2,99	**2467,5**	0,00	0,513	**2467,5**	0,00	0,338
Am13b	-	-	**2870**	3,01	**2870**	0,02	0,534	**2870**	0,00	0,324
Am15	-	-	**3195**	4,70	**3195**	0,00	0,610	**3195**	0,03	0,618

Pode-se observar que os resultados dos algoritmos baseados em Simulated Annealing e VNS/ILS mostraram excelentes soluções para o CAP nas instâncias testadas, obtendo os mesmos valores produzidos pelas heurísticas SDP e SA Ahonen et al., em pelo menos uma das execuções. Os desvios obtidos por ambos algoritmos foram baixos ou iguais a zero na maioria das instâncias. Em relação às meta-heurísticas implementadas, o desempenho do algoritmo híbrido foi melhor do que o do SA, obtendo as mesmas soluções em menor tempo computacional.

Resultados obtidos para o SF-MRFLP

Posteriormente foram testadas as mesmas instâncias para o SF-MRFLP de 3 a 5 linhas paralelas. Os resultados podem ser observados na Tabela 2.

Tabela 2: Resultados do SF-MRFLP para 3 a 5 linhas paralelas

Instância		Heurística SDP		SA			VNS/ILS		
	k	Valor	Tempo (s)	Min	DP	Tempo (s)	Min	DP	Tempo (s)
H_5	3	**290**	8,0	**290**	0,0	0,281	**290**	0,0	0,013
Rand_5		**34,5**	7,6	**34,5**	0,0	0,254	**34,5**	0,0	0,011
H_6		**560**	32,8	**560**	0,0	0,281	**560**	0,0	0,021
Rand_6		**110,5**	36,4	**110,5**	0,0	0,283	**110,5**	0,0	0,023
H_7		**1190**	139,2	**1190**	0,0	0,308	**1190**	0,0	0,036
Rand_7		**108**	157,0	**108**	0,0	0,298	**108**	0,0	0,035
H_8		**1515**	638,0	**1515**	0,0	0,327	**1515**	0,0	0,053
S_8		**260**	709,1	**260**	0,0	0,324	**260**	0,0	0,050
SH_8		**740,5**	633,6	**740,5**	0,0	0,347	**740,5**	0,0	0,058
Rand_8		**138**	653,3	**138**	0,0	0,320	**138**	0,0	0,049
S_9		**771,5**	3074,5	**771,5**	0,0	0,355	**771,5**	0,0	0,086
SH_9		**1431,5**	2883,7	**1431,5**	0,0	0,366	**1431,5**	0,0	0,095
Rand_9		**317,5**	3055,0	**317,5**	0,0	0,358	**317,5**	0,0	0,076
H_5	4	**140**	3,5	**140**	0,0	0,242	**140**	0,0	0,013
Rand_5		48,5	4,5	**34,5**	0,0	0,263	**34,5**	0,0	0,017
H_6		**410**	34,9	**410**	0,0	0,228	**410**	0,0	0,024
Rand_6		**98,5**	40,0	**98,5**	0,0	0,254	**98,5**	0,0	0,025
H_7		**820**	232,2	**820**	0,0	0,281	**820**	0,0	0,034
Rand_7		**77**	190,1	**77**	0,0	0,283	**77**	0,0	0,031
H_8		**1135**	1211,1	**1135**	0,0	0,308	**1135**	0,0	0,049
S_8		**188**	1461,4	**188**	0,0	0,308	**188**	0,0	0,051
SH_8		**491,5**	1292,7	**491,5**	0,0	0,295	**491,5**	0,0	0,055
Rand_8		**98**	1411,1	**98**	0,0	0,279	**98**	0,0	0,055
H_5	5	200	0,2	**140**	0,0	0,257	**140**	0,0	0,017
Rand_5		44,5	0,2	**34,5**	0,0	0,280	**34,5**	0,0	0,011
H_6		**260**	7,0	**260**	0,0	0,249	**260**	0,0	0,021
Rand_6		**94,5**	6,7	**94,5**	0,0	0,262	**94,5**	0,0	0,021
H_7		**650**	87,6	**650**	0,0	0,258	**650**	0,0	0,030
Rand_7		**61**	83,3	**61**	0,0	0,277	**61**	0,0	0,036
H_8		**875**	981,6	**875**	0,0	0,311	**875**	0,0	0,048
S_8		**146**	1194,8	**146**	0,0	0,295	**146**	0,0	0,052
SH_8		**413,5**	911,8	**413,5**	0,0	0,271	**413,5**	0,0	0,056
Rand_8		**82**	967,2	**82**	0,0	0,287	**82**	0,0	0,049

Os algoritmos SA e VNS/ILS conseguiram alcançar os valores obtidos por Hungerländer e Anjos [2015]. Em relação às instâncias Rand_5 (para k igual a 4 e 5) e H_5 (para k igual a 5), os valores encontrados são menores devido ao fato de que os algoritmos propostos não obrigaram a alocação de todas as k linhas, permitindo uma linha sem nenhuma facilidade a fim de obter melhor resposta.

Resultados obtidos para o MRFLP

Além do SF-MRFLP, o MRFLP também foi explorado neste trabalho. O MRFLP é mais difícil de se resolver com meta-heurística do que o SF-MRFLP devido a sua natureza contínua. Os

algoritmos foram executados para instâncias da literatura considerando-se de duas a cinco linhas paralelas. Os resultados, mostrados em 3, foram comparados com os obtidos em Hungerländer e Anjos [2015] para resolver o DRFLP, para k igual a 2 e MRFLP para k de 3 até 5 linhas.

Tabela 3: Resultados do MRFLP para 2 a 5 linhas paralelas

Instância	k	Heurística SDP		SA			VNS/ILS		
		Valor	Tempo (s)	Min	DP (%)	Tempo (s)	Min	DP (%)	Tempo (s)
H_5	2	**350**	4,8	**350,0**	0,00	0,306	**350,0**	0,00	0,018
Rand_5		**52,5**	4,5	**52,5**	0,00	0,324	**52,5**	1,44	0,016
H_6		**640**	11,1	**640,0**	0,77	0,318	**640,0**	0,00	0,029
Rand_6		**190,5**	13,8	**190,5**	0,09	0,314	**190,5**	0,08	0,028
H_7		**1660**	25,2	1660,0	1,18	0,335	**1660,0**	0,00	0,043
Rand_7		**159**	31,7	162,7	3,03	0,337	159,3	0,87	0,042
H_8		**2265**	91,3	2325,0	2,32	0,356	**2265,0**	1,09	0,073
S_8		**396**	91,6	396,1	1,74	0,343	**396,0**	0,15	0,073
SH_8		1125,5	87,4	1130,1	0,32	0,372	**1123,0**	0,00	0,072
Rand_8		**189,5**	82,2	197,2	1,96	0,377	**189,5**	0,27	0,074
S_9		**1179**	253,4	1188,8	2,23	0,379	**1179,0**	0,08	0,106
SH_9		2294,5	251,9	2302,0	0,44	0,371	**2293,0**	0,06	0,109
Rand_9		**486,5**	252,6	509,5	1,03	0,380	**486,5**	0,31	0,108
S_10		1353,5	713	1385,2	2,46	0,388	**1351,0**	0,45	0,181
Rand_10		**821**	698,2	864,9	1,38	0,389	**821,0**	0,95	0,168
S_11		**3424,5**	2127	3616,2	2,11	0,401	**3424,5**	0,56	0,254
Rand_11		**697**	2048,5	845,7	2,92	0,395	773,5	0,44	0,263
H_12		**8875**	5943,3	9538,6	2,05	0,404	8899,3	0,49	0,375
Rand_12		**788**	6389,5	1126,0	1,46	0,401	1026,5	0,24	0,369
H_5	3	**175**	680,6	**175,0**	3,41	0,310	**175,0**	0,00	0,022
Rand_5		**18**	893,2	**18,0**	0,00	0,349	**18,0**	0,00	0,014
H_5	4	**105**	2287,4	105,8	3,29	0,331	105,1	1,29	0,024
Rand_5		**10**	4360,1	**10,0**	4,56	0,342	**10,0**	0,00	0,017
H_5	5	**0**	271,3	**0,0**	0,00	0,338	**0,0**	0,00	0,019
Rand_5		**0**	6252,4	**0,0**	0,00	0,346	**0,0**	0,00	0,016

O algoritmo híbrido mostrou-se eficiente para o MRFLP para as instâncias testadas. O algoritmo conseguiu alcançar os resultados da abordagem semi-definida para várias instâncias ou atingiu valores próximos. O algoritmo híbrido conseguiu melhorar as soluções conhecidas para três instâncias: SH_8, SH_9 e S_10, todas três com k igual a 2. Os tempos continuaram baixos.

Agradecimentos

Bernardo Cellin agradece a bolsa de mestrado da FAPES.

Conclusões

Neste trabalho foi feito um estudo sobre o problema de ordenação de Facilidades em múltiplas linhas, ou MRFLP. O problema é computacionalmente difícil e possui diversas aplicações na indústria. Foram implementados algoritmos que utilizam as meta-heurísticas Simulated Annealing e uma hibridização de VNS e ILS para resolver o MRFLP, assim como o SF-MRFLP, o qual não possui espaços entre facilidades.

A comparação dos resultados obtidos com os encontrados na literatura mostrou que os algoritmos baseados nas meta-heurísticas Simulated Annealing, VNS e ILS são eficientes na resolução do MRFLP e SF-MRFLP.

Como trabalhos futuros, pretende-se fazer a aplicação de outras meta-heurísticas na resolução de problemas de layout, assim como a aplicação de abordagens exatas.

Referências

Ahonen, H., De Alvarenga, A., e Amaral, A. (2014). Simulated annealing and tabu search approaches for the corridor allocation problem. *European Journal of Operational Research*, 232(1):221 – 233. ISSN 0377-2217.

Amaral, A. R. S. (2012). The corridor allocation problem. *Computers and Operations Research*, 39.

Amaral, A. R. S. (2013). A parallel ordering problem in facilities layout. *Computers and Operations Research*, 40.

Anjos, M. F., Fischer, A., e Hungerländer, P. (2015). Solution approaches for equidistant double- and multi-row facility layout problems. *Group for Research in Decision Analysis*.

Brusco, M. J. e Stahl, S. (2000). Using quadratic assignment methods to generate initial permutations for least-squares unidimensional scaling of symmetric proximity matrices. *Journal of Classification*, 17(2):197–223.

Cellin, B. D. P. e Amaral, A. R. S. (2015). Simulated annealing aplicado ao problema de ordenação em linhas paralelas. *XLVII Simposio Brasileiro de Pesquisa Operacional*.

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

Dickey, J. W. e Hopkins, J. W. (1972). Campus building arrangement using topaz. *Transportation Research*, 6(1):59–68.

Drira, A., Pierreval, H., e Hajri-Gabouj, S. (2007). Facility layout problems: A survey. *Annual Reviews in Control*, 31(2):255 – 267. ISSN 1367-5788.

Elshafei, A. N. (1977). Hospital layout as a quadratic assignment problem. *Operational Research Quarterly*, 28(1):167–179.

Ficko, M., Brezocnik, M., e Balic, J. (2004). Designing the layout of single and multiple-rows flexible manufacturing system by genetic algorithms. *Journal of Materials Processing Technology*, 157-158:150–158. ISSN 0924-0136. Achievements in Mechanical and Materials Engineering Conference.

Ficko, M., Brezovnik, S., Klancnik, S., Balic, J., Brezocnik, M., e Pahole, I. (2010). Intelligent design of an unconstrained layout for a flexible manufacturing system. *Neurocomputing*, 73(4-6): 639 – 647. ISSN 0925-2312.

Geoffrion, A. e Graves, G. (1976). Scheduling parallel production lines with changeover costs: Practical applications of a quadratic assignment/lp approach. *Operations Re- search*, 24:595–610.

Ghosh, D. e Kothari, R. (2012). Population heuristics for the corridor allocation problem. *Indian Institute of Management*, p. 1–13.

Heragu, S. S. e Kusiak, A. (1988). Machine layout problem in flexible manufacturing systems. *Operations Research*, 36(2):258–268.

Hungerländer, P. e Anjos, M. F. (2012). A semidefinite optimization approach to space-free multi-row facility layout. *Group for Research in Decision Analysis*.

Hungerländer, P. e Anjos, M. F. (2015). A semidefinite optimization-based approach for global optimization of multi-row facility layout. *European Journal of Operational Research*, 245(1):46 – 61. ISSN 0377-2217.

Jain, A. K., Khare, V. K., e Mishra, P. M. (2013). Facility planning and associated problems: A survey. *Innovative Systems Design and Engineering*, 4(6):1–8.

Jankovits, I., Luo, C., Anjos, M. F., e Vannelli, A. (2011). A convex optimisation framework for the unequal-areas facility layout problem. *European Journal of Operational Research*, 214(2):199 – 215. ISSN 0377-2217.

Kochhar, J. S. e Heragu, S. S. (1998). Multi-hope: A tool for multiple floor layout problems. *International Journal of Production Research*, 36(12):3421–3435.

Laporte, G. e Mercure, H. (1988). Balancing hydraulic turbine runners: A quadratic assignment problem. *European Journal of Operational Research*, 35(3):378–381.

Lee, Y. H. e Lee, M. H. (2002). A shape-based block layout approach to facility layout problems using hybrid genetic algorithm. *Computers and Industrial Engineering*, 42(2-4):237 – 248. ISSN 0360-8352.

Lourenço, H. R., Martin, O. C., e Stützle, T. (2003). *Handbook of Metaheuristics*, chapter Iterated Local Search, p. 320–353. Springer US, Boston, MA. ISBN 978-0-306-48056-0.

Mavridou, T. D. e Pardalos, P. M. (1997). Simulated annealing and genetic algorithms for the facility layout problem: A survey. *Computational Optimization and Applications*, 7(1):111–126. ISSN 0926-6003.

Miao, Z. e lin Xu, K. (2009). Research of multi-rows facility layout based on hybrid algorithm. In *Information Management, Innovation Management and Industrial Engineering, 2009 International Conference on*, volume 2, p. 553–556.

Mladenovic, N. e Hansen, P. (1997). Variable neighborhood search. *Computers and Operations Research*, 24(11):1097 – 1100. ISSN 0305-0548.

Nearchou, A. C. (2006). Meta-heuristics from nature for the loop layout design problem. *International Journal of Production Economics*, 101(2):312–328.

Palubeckis, G. (2012). A branch-and-bound algorithm for the single-row equidistant facility layout problem. *OR Spectrum*, 334:1–21.

Pollatschek, M., Gershoni, N., e Radday, Y. (1976). Optimization of the typewriter keyboard by computer simulation. *Angewandte Informatik*, 10:438–439.

Steinberg, L. (1961). The backboard wiring problem: A placement algorithm. *SIAM Review*, 3(1): 37–50.

Tasadduq, I. A., Imam, M., e Ahmad, A.-R. (2011). A novel metasearch algorithm for facility layout optimization. *International Conference on Computers & Industrial Engineering*, 41:854–859.

Wess, B. e Zeitlhofer, T. (2004). On the phase coupling problem between data memory layout generation and address pointer assignment. In Schepers, H., editor, *Software and Compilers for Embedded Systems*, volume 3199 of *Lecture Notes in Computer Science*, p. 152–166. Springer Berlin Heidelberg. ISBN 978-3-540-23035-9.

Zhang, Z. e Murray, C. C. (2012). A corrected formulation for the double row layout problem. *International Journal of Production Research*, 50(15):4220–4223.