PROGRAMA DE PÓS-GRADUAÇÃO EM SISTEMAS E PROCESSOS
INDUSTRIAIS – MESTRADO
ÁREA DE CONCENTRAÇÃO EM CONTROLE E OTIMIZAÇÃO DE PROCESSOS
INDUSTRIAIS

Jônatas Inácio de Freitas

INVESTIGAÇÃO E ANÁLISE DE UMA MODELAGEM PARA O USO DO ENXAME DE PARTÍCULAS NA OTIMIZAÇÃO DO PROBLEMA DE *LAYOUT* DE FACILIDADES

Santa Cruz do Sul
Fevereiro de 2016

Jônatas Inácio de Freitas

INVESTIGAÇÃO E ANÁLISE DE UMA MODELAGEM PARA O USO DO ENXAME DE PARTÍCULAS NA OTIMIZAÇÃO DO PROBLEMA DE *LAYOUT* DE FACILIDADES

Dissertação apresentada ao Programa de Pós-Graduação em Sistemas e Processos Industriais – Mestrado, Universidade de Santa Cruz do Sul – UNISC, Área de Concentração em Controle e Otimização de Processos, Linha de Pesquisa Monitoramento, Simulação e Otimização de Sistemas e Processos, como requisito parcial para obtenção do título de Mestre em Sistemas e Processos Industriais.

Orientador(a): Prof. Dr. João Carlos Furtado

Santa Cruz do Sul, fevereiro de 2016

Jônatas Inácio de Freitas

INVESTIGAÇÃO E ANÁLISE DE UMA MODELAGEM PARA O USO DO ENXAME DE PARTÍCULAS NA OTIMIZAÇÃO DO PROBLEMA DE *LAYOUT* DE FACILIDADES

Dissertação submetida ao Programa de Pós-Graduação em Sistemas e Processos Industriais – Mestrado, Universidade de Santa Cruz do Sul – UNISC, Área de Concentração em Controle e Otimização de Processos, Linha de Pesquisa Monitoramento, Simulação e Otimização de Sistemas e Processos, como requisito parcial para obtenção do título de Mestre em Sistemas e Processos Industriais.

Dr. João Carlos Furtado
Professor Orientador

Dr. Leonel Pablo Tedesco
Avaliador (UNISC)

Dr. Julio Cezar Mairesse Siluk
Avaliador (UFSM)

A meu pai José Antonio, que saiu antes de eu pagar a conta.

AGRADECIMENTOS

À Kely, por seu amor e por seu apoio, desde aquele inquietante domingo até então.

À minha irmã José, por insistir que eu passasse por isso.

À minha sobrinha Laura, por espontaneamente me questionar.

Aos meus sogros Neusa e Arcely, por serem suporte.

Ao meu sobrinho que vem aí, por trazer luz para esse mundo tão escuro, e a seus, pais, Emílio e Priscila, meus irmãos que a vida trouxe.

Aos meus outros irmãos, por tornarem a vida mais fácil.

Ao Prof. João Carlos Furtado, por sua paciência e disponibilidade em me acompanhar do zero.

Aos demais professores do Programa de Pós-Graduação em Sistemas e Processos Industriais, pelas sábias palavras.

Aos meus colegas pelos bons ensinamentos.

À UNISC e à CAPES, por possibilitarem novos rumos.

À minha mãe Eva, por tudo.

RESUMO

Esta pesquisa teve como proposta a investigação de uma modelagem para o problema de *layout* de facilidades, pela utilização do algoritmo enxame de partículas, visando à obtenção de soluções competitivas. Neste problema de otimização é discutida a alocação de facilidades, tais como máquinas, estações de trabalho, escritórios e departamentos diversos, em uma aplicação física. A maioria das abordagens do problema trata de minimizar o custo de transporte e manuseio de material, item que corresponde de 20 a 50% do custo operacional e de 15% a 70% do custo total de fabricação de um produto. Enxame de partículas é uma meta-heurística de inteligência coletiva em que se pretende a troca de informações entre os indivíduos de uma população, para otimização de uma determinada variável. Uma pesquisa bibliométrica realizada neste trabalho revelou um potencial considerável de exploração da aplicação deste algoritmo na resolução do problema de *layout*. Foram desenvolvidos dois métodos de duplo estágio baseados em enxame de partículas para abordagem do problema de *layout* de facilidades: o primeiro construindo o *layout* com árvores binárias e o segundo, com matrizes de particionamento. Dez problemas-teste consolidados na literatura científica foram utilizados para validação dos métodos e os resultados foram comparados com os melhores trabalhos publicados. Enquanto o primeiro método obteve soluções competitivas para problemas de até dez instâncias, mas infactíveis para problemas a partir de 14 instâncias, o segundo método obteve boas soluções para problemas de até 14 instâncias, mas razoáveis para todos os problemas testados.

Palavras-chave: otimização, problema de *layout* de facilidades, enxame de partículas, árvores binárias, matriz de particionamento.

ABSTRACT

This research aimed to investigate a model for the facility layout problem by using particle swarm optmization algorithm, in order to obtain competitive solutions. Facility layout optimization problem discusses allocation of facilities, such as machines, workstations, offices and ordinary departments, in a physical application. Most of the layout problem approaches deals with minimize transport and handling costs, which corresponds from 20 to 50% of operational costs and from 15 to 70% of total manufacturing costs. Particle swarm optimization is a swarm intelligence meta-heuristic which works by exchanging information between individuals from a population, aiming to optimize a specified variable. A survey presented in this work revealed potentiality on solving the facility layout problem using a particle swarm optimization approach. Two double-stage particle swarm optimization methods were developed to adress the problem: first one used slicing trees and second one used the space partitiong method for flexible bay structure. Tem benchmark datasets were used in order to validate methods, and the results were compared with the best outcomes ever published. While first method reached competitive solutions for problems up to ten instances but infeasible solutions for problems bigger or equal to 14 instances, second method was good for problems up to 14 instances, but acceptable for all tested problems.

Keywords: optimization, facility layout problem, particle swarm optimization, slicing trees, flexible bay.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE SIGLAS, SÍMBOLOS E ABREVIATURAS

$\pm v_{max}$	Velocidade máxima positiva e negativa preestabelecida
α	Componente inercial em PSO, coeficiente da distância-alvo em AR, modificador do faixa admitida para uma dimensão em ABSMODEL, fator de penalidades da modelagem proposta
β	Razão de aspecto
δ	Coeficiente de dispersão
μ	Coeficiente da função de penalidades
ρ	Probabilidade de mutação
θ	Modificador do faixa admitida para uma dimensão em ABSMODEL
Φ_1	Componente cognitivo
Φ_2	Componente social
χ	Fator de constrição, coeficiente inercial
A	Área, soma das áreas da facilidades para configuração do algoritmo
A_i	Área da facilidade
$A_{i \cap j}$	Área de sobreposição entre as facilidades *i* e *j*
A_S	Soma das áreas de sobreposição $A_{i \cap j}$
A_T	Área total do *layout*
a	Dimensão do retângulo
b	Dimensão do retângulo
c_{ij}	Custo de transporte de material da facilidade *i* para a facilidade *j*
d	Dimensão ou número de variáveis da função objetivo, distância
d_{ij}	Distância entre as facilidades *i* e *j*
F	Função de penalidades
$gbest_i$	Melhor posição global encontrada
h	Altura da facilidade, corte horizontal em uma árvore de corte
H_{ij}	Espaço mínimo horizontal entre as facilidades i e j
i	Partícula, facilidade
j	Facilidade
l	Comprimento da facilidade, linha da matriz que denota um *layout*, número de partículas na configuração do algoritmo
L	Quantidade total de linhas *l*

$pbest_i$	Melhor posição encontrada pela partícula
p_g	$pbest_i$
p_i	$pbest_i$
P	Perímetro
P_i	Perímetro da facilidade i
S	Área de sobreposição
U	Função aleatória uniforme
v	Corte vertical em uma árvore de corte
v_i	Velocidade da partícula
$\mathbf{v}_i$	Vetor velocidade
V_{ij}	Espaço mínimo vertical entre as facilidades i e j
w	Coluna da matriz que denota um *layout*, largura da facilidade
W	Quantidade total de colunas w
x_i	Posição da partícula, coordenada horizontal da facilidade i
$\mathbf{x}_i$	Vetor posição
x_j	Coordenada horizontal da facilidade j
y_i	Coordenada vertical da facilidade i
y_j	Coordenada vertical da facilidade j
ABSMODEL	Modelagem de Heragu e Kusiak (1991)
AR	Modelagem *Attractor-Repeller*
AUF	Fator Utilização de Área *(area utilization factor)*
CLP	Problema de *layout* de facilidades contínuo *(continual layout problem)*
DISCON	Modelagem *Dispersion-Concentration*
DLP	Problema de *layout* de facilidades discreto *(discrete layout problem)*
FLP	Problema de *layout* de facilidades *(facility layout problem)*
FIPSO	*Full Informed Particle Swarm Optimization*
FPSO	*Fuzzy Particle Swarm Optimization*
IPSO	*Improved Particle Swarm Optimization*
MIP	Programação Inteira Mista (*mixed integer program)*
MFFC	Fator Custo de Fluxo de Material *(material flow factor cost)*
NLT	Modelagem *Nonlinear Optimization Layout Technique*
PO	Pesquisa Operacional
PSO	Enxame de Partículas *(particle swarm optimization)*
QAP	Problema Quadrático de Alocação *(quadratic assignment problem)*

RABSMODEL	Modelagem ABSMODEL Robusto
RAF	Força Aérea Britânica *(Royal Air Force)*
SRF	Fator Razão de Aspecto *(Shape Ratio Factor)*
SPM	Método de Particionamento Espacial *(space partitioning method)*
TBA	Área não ocupada total *(total blank area)*
TLC	Custo total do *layout (total layout cost)*

SUMÁRIO

1 INTRODUÇÃO

Pesquisa operacional (PO) é a aplicação de métodos científicos em problemas complexos para auxiliar no processo de tomada de decisão, em situações em que há escassez de recursos. Seu surgimento está relacionado à pesquisa com finalidades militares, no período imediatamente anterior à Segunda Guerra Mundial, pela Força Aérea Britânica (*Royal Air Force – RAF)*. O termo *pesquisa operacional* é atribuído a A. P. Rowe, superintendente da Estação de Pesquisa Manor Bawdsey, em Suffolk (Reino Unido), quando coordenava equipes que examinavam a eficiência de técnicas de operações provenientes de experimentos com interceptação de radar (ARENALES *et al*., 2007).

Nas décadas de 1950 e 1960, da aplicação em problemas de natureza logística na Segunda Guerra, a PO passou a evoluir rapidamente nos setores público e privado, devido à credibilidade e sucesso da abordagem científica, principalmente nos Estados Unidos e na Inglaterra. Suas contribuições estenderam-se, desta forma, por praticamente todos os domínios da atividade humana, da Medicina à Administração, passando por outras áreas como a Economia e Educação. Em Engenharia de Produção, a PO é tradicionalmente utilizada na resolução de problemas de produção e logística (MORABITO, 2008).

PO trata de resolver problemas ligados a fenômenos reais. A resolução desses problemas passa pela observação dos cenários que os envolvem e por sua conseguinte descrição. Neste contexto a matemática tem importância fundamental. Quando as regras que descrevem os cenários são relações matemáticas, temos o que se chama *modelo matemático*, objeto abstrato que visa emular as principais características de um objeto real (ARENALES *et al*., 2007).

Os modelos de programação matemática (otimização matemática) têm um papel destacado em PO (MORABITO, 2008). Problemas modelados matematicamente são, via de regra, resolvidos por algoritmos de otimização, que consistem de métodos de busca, em que o objetivo é encontrar a solução do problema de otimização, de modo que um determinado valor seja otimizado, possivelmente sujeito a um conjunto de restrições

(ENGELBRECHT, 2004).

De modo geral, cada problema de otimização consiste de uma função objetivo *f* que representa a quantidade a ser otimizada; um conjunto de variáveis *x*, que afeta o valor da função *f(x);* e um conjunto de limitações que restringe os valores que podem ser assumidos pelas variáveis (ENGELBRECHT, 2004).

O problema de *layout* de facilidades *(facility layout problem – FLP)* é um dos tradicionais problemas de otimização em que se discute a alocação de facilidades de diferentes tipos, tais como máquinas, estações de trabalho, áreas de atendimento ao cliente, escritórios e departamentos diversos (NEGHABI *et al.*, 2014). O principal fator de determinação de eficiência, quando se trata de planejamento do *layout* industrial, é o custo de transporte e manuseio de materiais entre as diferentes facilidades, que corresponde de 20 a 50% do custo operacional e de 15 a 70% do custo total de fabricação de um produto (TOMPKINS *et al.*, 2010).

Diferentes aproximações têm sido utilizadas para abordar as variações de FLPs presentes na literatura científica, ora através de métodos baseados em heurísticas, ora através de algoritmos de otimização (DRIRA *et al.*, 2007). Métodos exatos como *branch and bound* (KOUVELIS e KIM, 1992; MELLER *et al.*, 1998; KIM e KIM, 1998) e programação dinâmica (ROSENBLATT, 1986) resolvem de forma ótima apenas problemas com um pequeno número de instâncias. O FLP é do tipo *np-hard*, conforme classificação proposta por Garey e Johnson (1979), o que significa que não há método exato que o resolva em tempo computacional aceitável para uma grande quantidade de instâncias.

Por outro lado, métodos de resultado aproximado têm se mostrado eficientes, como os baseados em meta-heurísticas, das quais é possível evidenciar Busca Tabu (GLOVER, 1989), *Simulated Annealing* (KIRCKPATRICK *et al.*, 1983), Algoritmos Genéticos (HOLLAND, 1975), Colônia de Formigas (DORIGO, 1992) e Enxame de Partículas (KENNEDY e EBERHART, 1995).

Enxame de partículas *(Particle Swarm Optimization* – PSO*)*, assim como outras abordagens de inteligência coletiva, é baseado em uma população de indivíduos capazes de interagir com o meio e entre si, aprendendo com sua experiência e com a experiência de outras partículas que constituem o enxame, tendo como consequência um comportamento global (SERAPIÃO, 2009).

Müller *et al.* (2006) consideram PSO uma meta-heurística robusta e eficiente computacionalmente, o que motiva a buscar inovações em sua modelagem para empregá-lo nesta pesquisa.

Neste trabalho se pretendeu analisar as modelagens matemáticas existentes para o FLP e buscar aperfeiçoá-las de forma a permitir o adequado uso do método enxame de partículas e consequentemente obter soluções de melhor qualidade para o problema.

1.1 Justificativa

O ambiente empresarial, nativamente dinâmico, deve permitir a adequada execução das atividades finalísticas e não-finalísticas das organizações, respeitadas as restrições de recursos e a necessidade de redução de custos. A inserção das indústrias em um ambiente competitivo exige o aprimoramento de práticas na produção, destacando o arranjo físico como um grande desafio na gestão industrial (ROSA *et al.*, 2014).

Um *layout* inadequadamente configurado pode ocasionar aumento dos custos de transporte de recursos entre seus departamentos, prejudicando ou inviabilizando o desenvolvimento organizacional. Tompkins *et al*. (2010) indicam que de 20% a 50% das despesas operacionais têm ligação com o custo de manuseio e transporte de recursos materiais, de modo que um *layout* de facilidades eficiente possa significar a redução destes custos para 10% a 30%. Nesse contexto, a decisão sobre como configurar as facilidades em uma planta é um fator crítico para o sucesso da atividade industrial.

Para auxiliar no processo de solução de problemas difíceis para tomada de decisão, PSO demonstra ser um método muito promissor, principalmente pela eficiência no que se refere ao tempo de desempenho computacional (MÜLLER *et al.*, 2006; ENGELBRECHT, 2005). Diferentes meta-heurísticas tem sido utilizadas para solução do FLP.

Por outro lado, PSO ainda é um método pouco explorado para aplicação ao problema. Uma pesquisa nas bases de publicações científicas ScienceDirect, da Elsevier e SpringerLink, da Springer, realizada em 18/10/2015 pelos termos *particle swarm optimization* e *facility layout*, no resumo, título ou palavras-chave, retornou apenas 30 publicações desde 2010 que tratam do FLP utilizando-se do algoritmo PSO. Em se tratando de uma meta-heurística, esta informação é um indicativo de maior necessidade de investigação. À Tabela 1, um apanhado com a quantidade de artigos pesquisados.

Isto posto, esta pesquisava foi motivada pela insipiência do assunto na pesquisa nacional, o indicativo de maior necessidade de investigação da utilização do PSO na solução do FLP, a relevância da solução do problema de *layout* para os interesses empresariais e as contribuições ao meio acadêmico que podem ser obtidas pela pesquisa do método enxame de partículas.

Tabela 1 – Pesquisa nas bases de periódicos de Springer e Elsevier

Base	Palavra-chave	2010	2011	2012	2013	2014	2015	Total
Springer	*Layout problem*	30	38	40	51	55	50	264
	Layout Problem + Particle Swarm Optimization	0	6	6	12	15	15	54
	PSO como método de solução *	0	1	6	3	1	2	13
Elsevier	*Layout Problem*	14	10	13	17	16	24	94
	Layout Problem + Particle Swarm Optimization	2	2	4	4	4	3	17

*A partir dos resultados para "*Layout Problem*" e "*Particle Swarm Optimization*", na base Springer, foi realizada uma verificação artigo por artigo para certificar-se que o resultado referia-se ao assunto desejado.

1.2 Objetivos

Nesta Seção apresentamos os objetivos deste trabalho de pesquisa.

1.2.1 Objetivo geral

Com esta pesquisa objetivou-se investigar uma modelagem para o problema de *layout* facilidades de forma a permitir o uso do método enxame de partículas com o propósito de obter soluções de melhor qualidade para o problema.

1.2.1 Objetivos específicos

Para atender ao objetivo geral, esta pesquisa compreendeu os seguintes objetivos específicos:

a) Produzir referencial teórico consistente para o problema de *layout* de facilidades e para o método de enxame de partículas na solução de problemas de otimização;

b) Identificar as principais modelagens para problemas de *layout* disponíveis na literatura científica;

c) Aplicar o método de enxame de partículas para solução dos problemas de *layout* disponíveis na literatura científica;

d) Validar os resultados obtidos pelo modelo desenvolvido, comparando-os com outros descritos na literatura.

1.3 Metodologia

Esta pesquisa acadêmica caracteriza-se como uma pesquisa operacional (MORABITO, 2008). Foi desenvolvida em três fases distintas, visando a anteder seus objetivos. Segue a classificação das fases, conforme Santos (2000):

a) A fase de pesquisa bibliográfica foi exploratória, quando houve uma aproximação e familiarização com o tema;

b) A partir dos recursos teóricos, a pesquisa seguiu para a fase descritiva, quando foram definidos os problemas-teste balizadores da pesquisa e propostas as modelagens do FLP e o método de enxame de partículas utilizado para solucioná-lo;

c) A etapa experimental e explicativa caracterizou-se pela implementação de uma variante do algoritmo enxame de partículas para solução dos problemas propostos.

1.3.1 Procedimentos metodológicos

Para que fossem atingidos os objetivos específicos e, por consequência, o objetivo geral desta pesquisa, o trabalho foi organizado de acordo com o fluxograma executivo à figura 1.

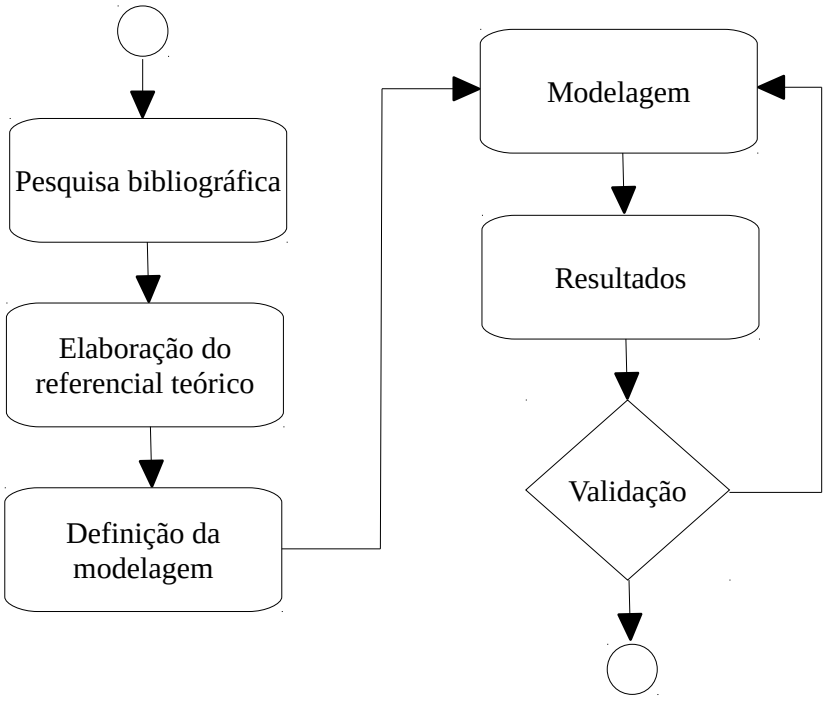


Figura 1 – fluxograma dos procedimentos metodológicos

A primeira etapa consiste da pesquisa bibliográfica exaustiva, focada em periódicos internacionais, face à insipiência do tema na literatura científica nacional. Da pesquisa bibliográfica foram elaborados os próximos dois capítulos desta dissertação, referencial teórico base para os procedimentos metodológicos subsequentes.

A partir da pesquisa bibliográfica, foi definido que o trabalho abordaria a solução de problemas de *layout* de facilidades de áreas desiguais e construídas as modelagens do problema e de sua solução, descritas aos capítulos 4 e 5 desta dissertação. A programação do protótipo foi implementada em MATLAB, uma linguagem de alto nível associada a um ambiente de desenvolvimento, destinados, entre outras finalidades, à computação numérica, visualização e desenvolvimento de aplicações (HIGHAM e HIGHAM, 2000).

O MATLAB foi escolhido considerando-se principalmente quatro fatores: simplicidade na programação envolvendo vetores e matrizes, vasto banco de funções matemáticas, usabilidade no armazenamento das variáveis de saída e facilidade de obtenção de resultados gráficos. A partir das dificuldades e restrições encontradas nos testes iniciais, foram propostas alterações às modelagens detalhadas nos capítulos 4 e 5.

2 O PROBLEMA DE *LAYOUT* DE FACILIDADES

Facilidades são máquinas, departamentos, postos de trabalho ou quaisquer entidades que facilitem a *performance* de um trabalho (HERAGU, 1997). A primeira concepção do problema, por Koopmans e Beckmann (1957), define o FLP como problema industrial em que o objetivo é configurar as facilidades, de modo a minimizar o custo entre elas. Na concepção de Meller *et al.* (1998), o FLP consiste em encontrar um arranjo não-sobreposto planar ortogonal de *n* facilidades retangulares dentro de uma planta retangular, de modo a diminuir a medida baseada na distância entre as alocações. Lee e Lee (2002) definem o FLP como um arranjo de facilidades com áreas desiguais em um dado espaço, de modo a minimizar os custos de transporte de material e de áreas ociosas. O problema de *layout* de facilidades pode também consistir na otimização de multiobjetivos através de multi-atributos, conforme proposto por Farahani *et al.* (2010).

Em classificação proposta por Drira *et al.* (2007), adaptada de Yang *et al.* (2005), quanto à configuração das facilidades no *layout*, têm-se os seguintes tipos*:*

a) *single-row* (única linha), quando as facilidades são alocadas ao longo de uma só linha de produção (NEMATIAN, 2014; KOTHARI e GHOSH, 2012a; KOTHARI e GHOSH, 2012b; AZADEH *et al.*, 2013);

b) *multi-rows* (múltiplas linhas), quando as facilidades são alocadas ao longo de mais de uma linha de produção (AZARBONYAD e BABAZADEH, 2014; RAHBARI, 2014);

c) *loop* (circuito), quando as facilidades são alocadas sob a forma de circuito (SARAVANAN e KUMAR, 2013);

d) *open field* (campo aberto), quando as facilidades podem ser alocadas sem as limitações de arranjos em linha ou circuito (ANJOS e VANELLI, 2002; MÜLLER *et al.*, 2006).

O FLP pode ainda envolver a alocação das facilidades em edificações de mais de um andar *(multi-floor),* como em Önut *et al.* (2008), Rodrigues *et al.* (2013), e Jiang *et al.* (2014) e Kia *et al.* (2014). À figura 2, uma aproximação das diferentes configurações

abordadas.

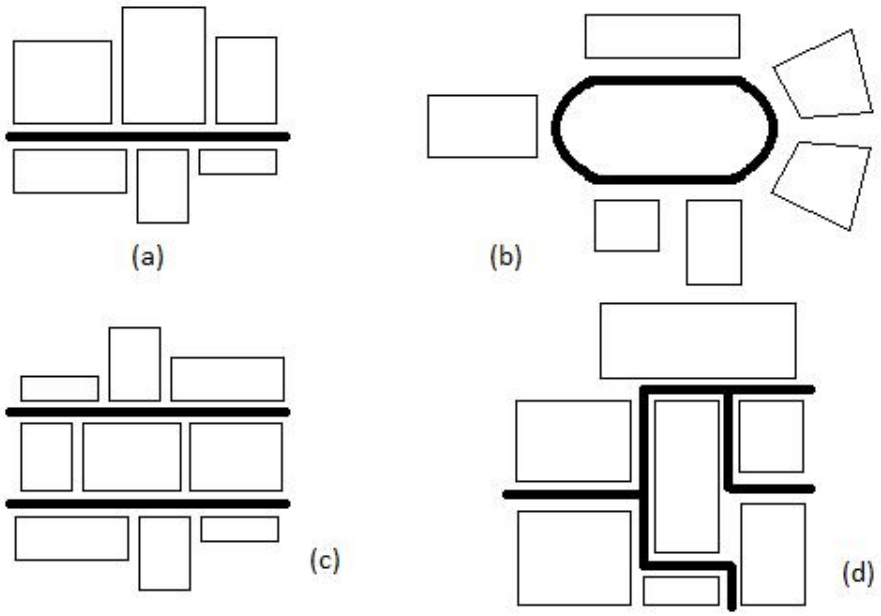


Figura 2 – configurações do *layout*: (a) *single row*, (b) *loop*, (c) *multi-row*, (d) *open-field*
Fonte: Drira *et al.* (2007).

Além da classificação quanto à disposição dos departamentos na planta, o FLP pode ser classificado quanto às áreas das facilidades (iguais ou desiguais), e ainda pode ser classificado quanto à restrição das dimensões da planta. Quando são fixas e tomadas como dados de entrada do problema, o FLP é dito restrito. Quando não fixas, mas definidas durante o processamento do algoritmo de otimização, o FLP é dito irrestrito.

Este trabalho se propõe a investigar problemas de *layout* do tipo restrito, *open-field*, com áreas desiguais.

2.1 Modelagem matemática do FLP

Há, na literatura científica, diversos meios de formular matematicamente os FLPs, de modo que possam ser resolvidos. De acordo com Drira *et al.* (2007), as modelagens encontradas na literatura apontam comumente para problemas de programação inteira mista *(mixed integer programming* – MIP) ou para problemas quadráticos de alocação *(quadratic assignment problem* – QAP). Nesta seção serão apresentadas algumas das formulações matemáticas mais comuns.

2.1.1 Problema quadrático de alocação

O QAP foi concebido por Koopmans e Beckmann (1957), a fim de modelar o problema de alocação de facilidades de áreas e formatos iguais, respeitadas as restrições de que a cada facilidade seja atribuída uma possível localização e que a cada localização associe-se apenas uma facilidade. As modelagens discretas do FLP geralmente são formuladas como QAPs. Uma definição típica para a função objetivo, em que se busca minimizar o custo total de transporte de material *(z)*, em função da distância e o custo de transporte de material entre as facilidades é a seguinte:

$$Minimizar\ z = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} c_{ij} f_{ij} d_{ij} \ , \qquad (1)$$

em que *i* e *j* são as facilidades de posição (x_i, y_i) e (x_j, y_j); *n* é o número de facilidades do problema; c_{ij} é o custo de transporte entre as facilidades *i* e *j;* f_{ij} é o fluxo de material entre as facilidades *i* e *j;* e d_{ij} é a distância euclidiana entras as facilidades *i* e *j*.

À figura 3 apresenta-se uma concepção gráfica de uma solução para um QPA discreto, em que seis facilidades com áreas desiguais são alocadas em um *layout* com 16 unidades de área.

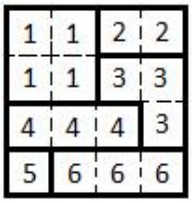


Figura 3 – concepção gráfica de um QPA com facilidades de áreas desiguais

Do objetivo proposto no QPA derivam as modelagens matemáticas da maioria das abordagens para o FLP, incluindo as propostas neste trabalho.

2.1.2 ABSMODEL e RABSMODEL

Vários modelos e algoritmos foram utilizados para propor e resolver o FLP. Neghabi *et al.* (2014) consideram ABSMODEL, de Heragu e Kusiak (1991), a mais conhecida das modelagens contínuas. O modelo assume que a as facilidades são retangulares e que a distância entre os departamentos é a táxi-distância ou distância na forma retilínea. As vantagens apontadas, pelos autores, em relação ao modelo discreto, consistem no fato de as alocações dos departamentos não precisarem, *a priori*, ser conhecidas, além da

possibilidade de lidar com facilidades de áreas desiguais. Conceitualmente, a função objetivo é a mesma da equação 1, exceto pelo que se define em

$$d_{ij} = |x_i - x_j| - |y_i - y_j|. \quad (2)$$

A função objetivo 1 é sujeita às restrições:

$$|x_i - x_j| \geq \frac{1}{2}(l_i + l_j) + H_{ij} \quad (3)$$

$$|y_i - y_j| \geq \frac{1}{2}(w_i + w_j) + V_{ij} \quad (4)$$

em que $l_{i,j}$ e $w_{i,j}$ são, respectivamente, o comprimento e a largura de cada facilidade, e H_{ij} e V_{ij} são, respectivamente, as distâncias mínimas de separação entre as facilidades *i* e *j*, horizontalmente e verticalmente.

Os problemas foram resolvidos com um algoritmo de otimização sem restrições, que proveu soluções sub-ótimas em um tempo computacional relativamente baixo.

A partir do modelo de Heragu e Kusiak, Neghabi *et al.* (2014) apresentaram o que chamaram de ABSMODEL Robusto, ou RABSMODEL. Segundo os autores, é cediço que um *layout* robusto pode ser definido por diferentes abordagens. No modelo proposto por eles, um *layout* robusto *é* definido como o *layout* que permite ao tomador de decisão alterar as dimensões dos departamentos dentro de uma faixa *(range)* preestabelecida. Desta forma, as variáveis l_i e w_i são flexíveis, onde $l_i \in [lmin_i, lmax_i]$ e $w_i \in [wmin_i, wmax_i]$. Foi ainda considerado um coeficientes α que modifica as faixas limitadoras de largura e comprimento.

A partir destas premissas, as restrições do ABSMODEL foram modificadas, de modo que:

$$|x_i - x_j| \geq \frac{1}{2}(lmin_i - lmin_j) + \frac{1}{2}\alpha_{ij}\left((lmax_i - lmin_i) + (lmax_j - lmin_j)\right) \quad (5)$$

$$|y_i - y_j| \geq \frac{1}{2}(wmin_i - wmin_j) + \frac{1}{2}\alpha_{ij}\left((wmax_i - wmin_i) + (wmax_j - wmin_j)\right). \quad (6)$$

A eficiência deste modelo é fortemente ligada à interferência do tomador de decisão, uma vez que, dentro do algoritmo de solução do problema, os autores estabelecem uma fase em que o *stakeholder* precisa definir se os limites superiores e inferiores da função objetivo são factíveis com a aplicação relacionada ao *layout*.

2.1.3 *DISpersion-CONstruction, NLT, Attractor-Repeller* e Jankovits *et al.*

Drezner (1980, 1987) modelou o problema de *layout* considerando que cada facilidade *i* fosse representada por um círculo, de posição determinada pelo seu centro *(x, y)*, e que a função objetivo *z* (a ser minimizada) fosse calculada como o somatório das distâncias euclidianas entre esses centros, multiplicado pelo custo do tráfego de material entre elas, da seguinte forma:

$$Minimizar\ z = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} c_{ij} d_{ij} \ . \tag{7}$$

A única restrição que o modelo considera é que não haja sobreposição entre os círculos, isto é a soma de seus raios tem de ser menor ou igual à distância entre seus centros:

$$r_i + r_j \leq d_{ij} \forall 1 \leq i < j \leq n \tag{8}$$

O problema foi resolvido em duas fases, a primeira com todos os círculos sendo dispersos da origem (centro) de todo o *layout*, a segunda com os círculos sendo colocados da forma mais concentrada possível, caracterizando, desse modo, as fases de dispersão e concentração, que nomearam a modelagem. Para Anjos e Vanelli (2002), autores do modelo *Attractor-Repeller,* inspirado no modelo DISCON, o modelo original não dá ao usuário o controle sobre as dimensões do *layout* resultante.

Para lidar com tal desvantagem, van Camp *et al.* (1992) introduziram ao seu modelo as seguintes restrições:

$$|x_i - x_j| \geq \frac{1}{2}(w_i + w_j), se\ |y_i - y_j| < \frac{1}{2}(h_i + h_j) \tag{9}$$

$$|y_i - y_j| \geq \frac{1}{2}(h_i + h_j), se\ |x_i - x_j| < \frac{1}{2}(w_i + w_j) \tag{10}$$

$$\frac{1}{2} w_T \geq x_i + \frac{1}{2} w_i \tag{11}$$

$$\frac{1}{2} w_T \geq \frac{1}{2} w_i - x_i \tag{12}$$

$$\frac{1}{2} h_T \geq y_i + \frac{1}{2} h_i \tag{13}$$

$$\frac{1}{2} h_T \geq \frac{1}{2} h_i - y_i \tag{14}$$

$$min\{w_i, h_i\} \geq lmin_i \tag{15}$$

$$lmax_i \geq min\{w_i, h_i\} \tag{16}$$

$$min\{w_T, h_T\} \geq lmin_T \tag{17}$$

$$lmax_T \geq \quad min\{w_T, h_T\} \tag{18}$$

em que w_i e h_i são a largura e a altura do módulo (facilidades) *i*; $lmin_i$ e $lmax_i$ são o menor e o maior comprimento do menor lado do módulo *i*; w_T e h_T são a largura e a altura do *layout;* e $lmin_T$ e $lmax_T$ são o menor o maior comprimento do lado mais curto do *layout.*

As restrições 9 e 10 impedem a sobreposição entre os módulos, enquanto as restrições 11 a 14 impedem que os módulos ultrapassem os limites do *layout.* Por outro lado, as restrições 15 a 18 tratam de colocar o menor lado de cada módulo, bem como o menor lado de todo o *layout*, entre *lmin e lmax.* Os autores chamaram o modelo de NLT (*Nonlinear Optmization Layout Technique*), em virtude de terem formulado o FLP com uma modelagem não-linear.

Assim como em outras proposições, incluindo a modelagem que será apresentada nesta pesquisa, o problema sujeito a restrições foi transformado em um problema irrestrito, de modo que o conjunto de condições fica transformado em uma função de penalidades F_{ij}. Na função de penalidade quadrática, a penalização adicionada à função objetivo é uma constante μ multiplicada por uma medida que evidencia o quanto uma restrição foi violada. No modelo NLT, *F* é dada conforme segue:

$$F_{ij} = \mu min\left\{\left(\left|x_i - x_j\right| - \frac{1}{2}\left(w_i + w_j\right)\right)^2, \left(\left|y_i - y_j\right| - \frac{1}{2}\left(h_i + h_j\right)\right)^2\right\}. \tag{19}$$

Seguindo os estudos de Vanelli, que participou da concepção do modelo NLT, em conjunto com Anjos (2002), e ainda buscando aperfeiçoar o modelo DISCON, foi apresentado o modelo *Attractor-Repeller* (AR). Assim como em Drezner (1980, 1987), a função objetivo é a equação 7, as facilidades são concebidas como círculos, tendo suas posições denotadas pelas coordenadas centrais, e a restrição principal é a inequação 8, que impede a sobreposição entre as facilidades.

Analogamente às restrições 11 a 18 do modelo NLT, ou seja, para manter as facilidades dentro dos limites do *layout* e estabelecer os limites de largura e altura total do *layout*, AR apresenta as seguintes restrições:

$$\frac{1}{2} w_T \geq x_i + r_i \tag{20}$$

$$\frac{1}{2} w_T \geq r_i - x_i \tag{21}$$

$$\frac{1}{2} h_T \geq y_i + r_i \tag{22}$$

$$\frac{1}{2} h_T \geq r_i - y_i \tag{23}$$

$$wmin_T \leq w_T \leq wmax_T \tag{24}$$

$$hmin_T \leq h_T \leq hmax_T \tag{25}$$

Os autores de AR interpretam a função objetivo como um atrator *(attractor)* das facilidades, isto é, uma função que procura manter as distâncias d_{ij} tão pequenas quanto possível. Uma consequência desta abordagem é a concentração de todas as facilidades na origem, com distância igual a zero. Para prevenir este fenômeno, os autores criaram uma função de penalidades que reforça as restrições, o que chamaram de força repulsiva *(repeller).*

A função de penalidades apoia-se no conceito de distância-alvo $\sqrt{t_{ij}}$, de modo que:

$$t_{ij} = \alpha\left(r_i + r_j\right)^2 \tag{26}$$

Outro fator definido para a função de penalidades é $D_{ij} = d_{ij}^2$, de modo que quanto mais a relação D_{ij} / t_{ij} aproxima-se de 1, mais a função aproxima-se do ótimo. O parâmetro $\alpha > 0$, como se verifica, permite que se flexibilize o quanto se deseja reforçar as restrições do problema. Na prática, α é escolhido empiricamente, de modo que se consiga uma separação razoável entre os pares de círculos. Escolhendo $0 < \alpha < 1$, permite-se sobreposição. Escolhendo $\alpha = 1$, não é permitida sobreposição.

Transformando as restrições em uma função de penalidades *F*, Anjos e Vanelli (2002) seguiram o conceito proposto no modelo NLT. Mais precisamente, *F* definiu-se pelo produto do somatório de uma função $f(z) = (1/z) - 1$, multiplicada pela relação D_{ij} / t_{ij}, e a constante de penalidade μ, de modo que a função objetivo *z* fosse definida:

$$Minimizar\ z = \sum_{i=1}^{n=-1} \sum_{j=1}^{n} c_{ij} d_{ij} + \mu \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} f\left(\frac{D_{ij}}{t_{ij}}\right). \tag{27}$$

Trabalhos semelhantes ao AR são verificados em Müller *et al.* (2006), Müller (2007) e Castillo e Sim (2003). Em Jankovits *et al.* (2011), são apresentadas melhorias em relação aos trabalhos citados, com um aprofundamento no que diz respeito à razão de aspecto (razão entra a maior e menor dimensão de uma facilidade).

Restrições quanto a razões de aspecto máximas ou limites mínimos de dimensão são frequentemente usadas em métodos de organização de *layout* em blocos, de maneira que se restrinja a ocorrência de facilidades excessivamente longas e estreitas. A abordagem de tais restrições é considerada um desafio por Jankovits *et al.* (2011), na medida que departamentos de razões de aspecto baixas são mais práticas em aplicações do mundo real,

mas isto torna o problema de *layout* mais difícil.

Por outro lado, desconsiderar tais restrições significa tornar o problema infactível, uma vez que, para a solução do problema, bastaria, por exemplo, alinhar os departamentos ou blocos em uma única fila (Jankovits *et al.*, 2011), de modo que a preocupação seria apenas resolver a melhor ordem para os diferentes blocos, independentemente de sua forma. À figura 4 temos uma ilustração de uma possível solução para o FLP se abordado sem restrições de forma, o que é incompatível com situações do mundo real.

As restrições de aspecto em Jankovits *et al.* (2011) foram transformadas pelos autores em um fator que age sobre o raio concebido nos modelos predecessores, de modo que às facilidades fosse resguardado suficiente espaço para a alocação na planta, com uma razoável razão entre suas dimensões. À equação 28, temos:

$$r_i = \sqrt{\frac{a_i}{\pi}} \log_2\left(1 + \frac{a_i}{\varphi^2}\right), \tag{28}$$

em que r_i e a_i são, respectivamente, o raio e área da facilidade i e φ é um parâmetro para controlar a desejada menor dimensão de cada departamento.

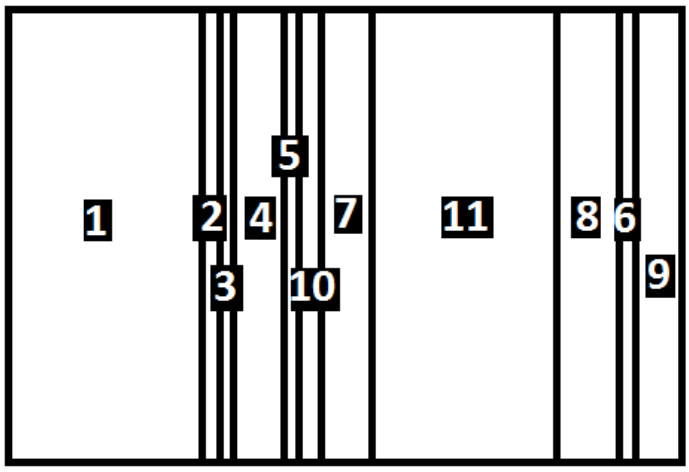


Figura 4 – possível solução para o problema de *layout* sem restrições de razão de aspecto

2.1.4 A razão de aspecto proposta por Wang *et al.*

Wang *et al.* (2005) estudaram o FLP com facilidades de áreas desiguais e propuseram uma modelagem baseada na minimização do **custo total do *layout*** (*total layout cost* – TLC). Em que pese tratar-se de uma abordagem discreta, os conceitos apresentados na modelagem da função multi-objetivo interessam a esta pesquisa.

Como se vê dos estudos anteriormente relatados, e segundo os autores, é comum a

modelagem do FLP com vias a minimizar o **custo do fluxo de material** entre as facilidades (*material flow factor cost* – MFFC), na forma das equações 1 ou 7. Apesar da maioria dos FLPs considerar apenas este fator para a função objetivo, outros fatores, como utilização da área, formas dos departamentos e formato de todo o *layout*, podem influenciar fortemente a função objetivo e deveriam ser considerados. Wang *et al.* (2005) propuseram inserir na função objetivo os fatores de **razão de aspecto** (*shape ratio factor* – SRF) e **utilização da área** (*area utilization factor* – AUF), sendo:

$$SRF = \left(\prod_{i=1}^{n} SR_i \right)^{\frac{1}{n}} = \left(\prod_{i=1}^{n} \frac{P_i}{4\sqrt{A_i}} \right)^{\frac{1}{n}} \tag{29}$$

a média geométrica das razões de aspecto *(shape ratio – SR)* de todas as facilidades, em que a razão de aspecto de cada facilidade é o quociente entre o perímetro P_i da facilidade retangular e quatro vezes a raiz quadrada de sua área A_i*;*

$$AUF = \frac{\sum A_i}{\sum A_i + TBA} \tag{30}$$

a razão entre a área total ocupada do *layout e* sua área total, composta de um somatório de todas as áreas ocupadas e as áreas não ocupadas (*total blank area* – TBA).

A razão de aspecto ideal de uma facilidade, igual a 1, é um quadrado, e a taxa de ocupação ideal de todo *layout* é 100%. Modificando o custo de fluxo de material pelo acréscimo do termo $\frac{SRF}{AUF}$, Wang *et al.* (2005) apresentam a função objetivo

$$Minimizar \; TLC = MFFC \, \frac{SRF}{AUF} \tag{31}$$

a ser minimizada, e sujeita às restrições

$$\sum_{i=1}^{n} a_{iwl} \leq 1 \forall \, w, l \tag{32}$$

$$\sum_{w=1}^{W} \sum_{l=1}^{L} a_{iwl} \leq A_i \forall \, i \tag{33}$$

$$\sum_{w=1}^{W} \sum_{l=1}^{L} \sum_{i=1}^{n} a_{iwl} \leq LW \tag{34}$$

em que a_{iwl} = 1, se ao departamento *i* for associada a alocação na *w*-ésima linha e *l*-ésima coluna da matriz que representa ou o *layout*, ou 0, caso contrário; A_i, a área requerida do departamento *i;* e *L* e *W,* respectivamente, o comprimento máximo e a largura máxima de toda a instalação, isto é, o número de elementos das linhas e colunas da matriz que

representa o *layout*.

O problema de Wang et *al.* (2005) foi resolvido utilizando-se uma variante da meta-heurística Algoritmos Genéticos (*genetic algorithm* – GA) e evoluiu o FLP de forma importante no que se refere a inserir um novo conceito para a abordagem da razão de aspecto nos problemas de *layout.* Os autores consideraram que a proposição de uma função objetivo multi-critério, incluindo os fatores MFFC, AUF e SRF, foi ao encontro das necessidades atuais em termos de aplicações práticas.

2.1.5 Método de particionamento espacial

Kim e Kim (1998) modelaram o FLP de maneira que o problema fosse codificado como uma matriz que tem a informação sobre as posições relativas entre as facilidades na planta. No método sugerido, denominado SPM (*space partitioning method* – método de particionamento espacial), a matriz que codifica todo o *layout* é decomposta recursivamente em cortes horizontais ou verticais de acordo com submatrizes geradas a partir da matriz principal.

Por exemplo, em uma planta com quatro departamentos, de áreas desiguais, uma matriz que represente o *layout* poderia ser:

$$\left|\begin{array}{c|cc} 1 & 4 & 0 \\ 2 & 0 & 3 \end{array}\right| \tag{35}$$

em que os valores 1 a 4 identificam as facilidades alocadas, e os elementos com valores 0 não tem área, consequentemente não aparecendo no *layout*. Vide que o corte vertical entre as colunas 1 e 2 divide a matriz principal em duas outras. Na representação, este corte divide a planta em duas outras áreas, contendo, em uma, as facilidades 1 e 2, e em outra, as facilidades 3 e 4, na forma da figura 5.

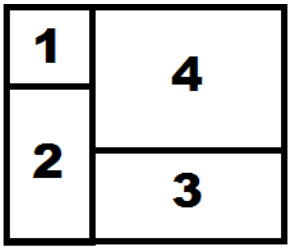


Figura 5 – layout gerado por matriz (35)

Um *layout* gerado pelo SPM sempre satisfaz as restrições de sobreposição do FLP, mas não as restrições da razão de aspecto. Para resolver a questão, os autores propuseram um algoritmo *simulated annealing* em que as restrições foram convertidas em função de penalidades, de modo que as soluções factíveis fossem obtidas mais facilmente. A Equação 36 mostra a função objetivo modificada.

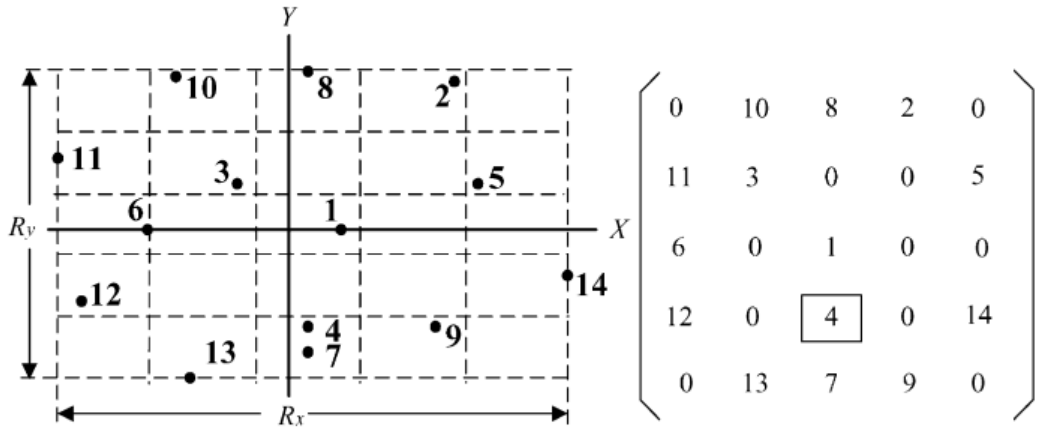
$$Minimizar \quad f = T_D(1 + \alpha N) \tag{36}$$

No modelo, T_D é o custo do fluxo de material no *layout*, conforme Equação 7, N é o número de facilidades as quais as restrições de forma não são satisfeitas e α é um parâmetro ajustado empiricamente de modo que sejam reforçadas ou relaxadas as restrições. Se α é muito alto, a probabilidade de estagnação em mínimos locais do algoritmo aumenta. Se α é muito baixa, o algoritmo pode não convergir para uma solução factível. Após uma série de experimentos em que α foi testado com valores entre 0,01 e 2, o melhor resultado obtido foi ao ajustar o peso para 0,3.

Modelos similares ao de Kim e Kim (1998) podem ser encontrados em Tate e Smith (1994) e Norman e Smith (1997). No SPM foram abordados três diferentes métodos de particionamento. No primeiro, a matriz é dividida somente em vetores-linha ou vetores-coluna, e em seguida em elementos. Este método é o mais próximo do empregado nos trabalhos antecessores citados. No segundo método, a matriz pode ser subdividida recursivamente nas orientações vertical ou horizontal, restringindo-se os cortes à primeira ou à última fila da submatriz gerada. No terceiro método, qualquer possível corte para decomposição foi considerado.

Para a abordagem do FLP, no trabalho de Chen *et al.* (2000), foi utilizado o primeiro método sugerido pelos autores do SPM. A solução final ficou também por conta do algoritmo *simulated annealing*. No entanto, para gerar um *layout* inicial em que já se levasse em conta a proximidade entre as facilidades, os autores utilizaram o escalonamento multidimensional (*multidimensional scaling – MDS*), de modo que as posições relativas dos centroides das facilidades no plano cartesiano fossem resolvidas por esta técnica.

Para decodificar as posições geradas pelo MDS, Chen *et al.* (2000) particionaram o gráfico gerado, de modo que cada partição correspondesse a um elemento da matriz de alocações proposta pelos autores do SPM, conforme se verifica à figura 6. Cada facilidade representada por um ponto gerado pelo MDS é associada à partição (região retangular) que a contém. Caso uma região retangular contenha mais de um ponto (facilidade) o conflito é resolvido associando a facilidade mais interna à partição e associando as demais facilidades a partições livres adjacentes.



(a) **(b)**

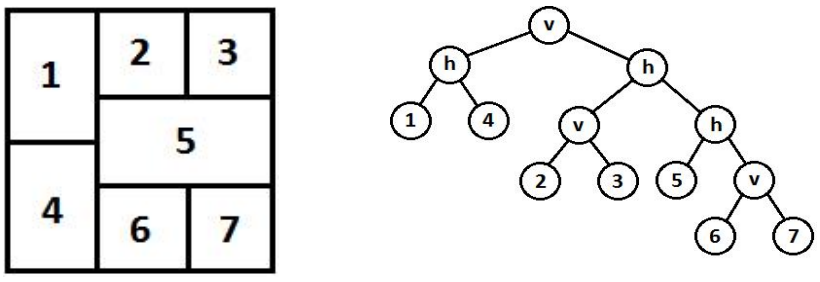
Figura 6 – particionamento da planta e os pontos estabelecendo correspondências entre as facilidades e sua localização na matriz (a) e a matriz de alocações correspondente (b).

Fonte: Chen *et al.* (2000)

No segundo estágio do algoritmo ocorre a conversão da matriz de alocações em um *layout*, assim como proposto em Tate e Smith (1994) e Kim e Kim (1998). Ao final, o *layout* gerado é otimizado por *simulated annealing*, através do ajuste de parâmetros de rotação de todos os pontos obtidos pelo MDS.

2.1.6 Árvores de corte

O FLP também pode ser abordado através da utilização de árvores binárias ou árvores de corte (*slicing trees*). De acordo com Drira *et al.* (2007), uma árvore de corte é composta de nós internos, que particionam o *layout*, e de nós externos, representando as facilidades. A cada nó interno pode ser associada uma orientação de divisão (corte) na planta, na direção vertical ou horizontal (0 = *v* ou 1 = *h*). Ao final, cada partição retangular corresponde a uma facilidade alocada em um espaço. Vide figura 7.



(a) **(b)**

Figura 7 – um *layout* (a) e sua árvore binária geradora (b)

A árvore binária pode ser utilizada para gerar um layout inicial factível, que pode ser aperfeiçoado por uma técnica de solução, como a heurística empregada nesta pesquisa. Como exemplos de trabalhos que utilizam árvores binárias para construção do *layout*, cita-se Tam (1992), Furtado e Lorena (1997), Shayan e Chittilappilly (2004), Scholz *et al.* (2009) e Komarudin e Wong (2010), entre outros. As Figuras 7.a e 7.b mostram, respectivamente, um *layout* e sua árvore binária geradora.

A árvore binária inicial pode ser gerada aleatoriamente, para em seguida ser aperfeiçoada por uma heurística, ou pode ser gerada de maneira tendenciosa, de modo que se aplique alguma técnica de agrupamento para as facilidades. Tam (1992), mensurou a dissimilaridade Γ_{ij} entre os departamentos *i* e *j* com fluxo de material c através da fórmula:

$$\Gamma_{ij} = \frac{1}{1+(c_i + c_j)} \tag{37}$$

Em seguida, uma matriz simétrica de distâncias (dissimilaridades) é gerada. A matriz é então submetida à análise hierárquica de agrupamentos (*hierarchical clustering analisys* – HCA*)* pela média das distâncias, para que seja gerado um dendrograma, do qual se obtém a árvore de corte. Tam (1992) utiliza-se de algoritmos genéticos para a otimização das árvores geradas através do agrupamento hierárquico.

2.2 Técnicas de solução do FLP

Tratando-se de técnicas de solução exatas para o FLP, podem ser destacados os métodos de *branch and bound* e de programação dinâmica. Porém estes métodos não são eficientes para problemas de alocação as partir de 7 facilidades, como observa-se nos resultados de Xie e Sahinidis (2007) e Solimanpur e Jafari (2007). Nestas situações, métodos de aproximação tem obtidos soluções sub-ótimas competitivas em tempo computacional aceitável. Neste sentido, há aplicações de heurísticas de construção (LEE e MOORE, 1967; TOMPKINS e REED, 1976), que *constroem* novas soluções a partir do zero, ou de melhoria (ARMOUR e BUFFA, 1963; DREZNER, 1987), que constroem novas soluções a partir de outra encontrada anteriormente. Recentemente, tem crescido a utilização de meta-heurísticas e abordagens híbridas (LIEN e CHENG, 2014). Dentre as meta-heurísticas destaque-se *simulated annealing* (CHWIF *et al*, 1998*;* ALVARENGA *et al.*, 2000; MCKENDALL *et al.*, 2006), busca tabu (CHIANG e KOUVELIS, 1996; FURTADO e LORENA, 1997a; MARTINS *et al.*, 2003) algoritmos genéticos (FURTADO e LORENA, 1997b; DUNKER *et al.*, 2005; WANG *et al.*, 2005; AZARBONYAD e

BABAZADEH, 2014), colônia de formigas (SOLIMANPUR *et al.*, 2005; BAYKASOGLU *et al.*, 2006) e, mais recentemente, enxame de partículas (MÜLLER *et al.*, 2006; ÖNUT *et al.*, 2008; SAMARGHANDI *et al.*, 2010), que será abordado neste trabalho de pesquisa.

2.3 Conjuntos de dados teste e *benchmarks*

Para mensurar a eficiência dos métodos propostos para solução do FLP, a literatura científica considera alguns conjuntos de problemas-teste como *benchmarks* a fim de comparação dos resultados obtidos. À tabela 2 constam os problemas utilizados para validação dos resultados de Komarudin e Wong (2010), também tidos como referência em muitos outros trabalhos, dos quais se pode destacar Jankovits *et al.* (2011) e Gonçalves e Resende (2015).

Jankovits *et al.* (2011) é o trabalho publicado mais recente com participação de Miguel Anjos e Anthony Vanelli, autores de outros importantes estudos envolvendo o FLP, como os já citados à Seção 2.1.3 desta dissertação. Gonçalves e Resende (2015) é o mais novo trabalho publicado abordando o problema de *layout* de forma compreensiva, com resultados superiores a outras pesquisas em 19 dos 28 problemas-teste utilizados. Por outro lado, Komarudin e Wong (2010) é o único dos três trabalhos citados que disponibiliza em sua publicação as instâncias (matrizes de dados) para os problemas alvo desta pesquisa.

Tabela 2 – fontes dos problemas-teste utilizados por Komarudin (2010)

Conjunto de dados	Fonte	Número de instâncias
O7	Meller *et al.* (1998)	7
O8	Meller *et al.* (1998)	8
O9	Meller *et al.* (1998)	9
vC10	van Camp *et al.* (1992)	10
Ba12	Bazaraa (1975)	12
Ba14	Bazaraa (1975)	14
AB20	Armour e Buffa (1963)	20
SC30	Liu e Meller (2007)	30
SC35	Liu e Meller (2007)	35
Du62	Dunker *et al.* (2003)	62

Outras pesquisas que também se utilizaram de conjuntos de dados expostos à tabela 2 foram: Tate e Smith (1995), Gau e Meller (1999), Castillo *et al.* (2005), Liu e Meller (2007) e Scholz *et al.* (2009).

3 ENXAME DE PARTÍCULAS

O algoritmo enxame de partículas é baseado em uma teoria sócio-cognitiva. Assim como em outras abordagens de inteligência coletiva, PSO consiste de uma população de indivíduos capazes de interagir entre si e com o ambiente. Cada indivíduo de uma população possui sua experiência e mede a qualidade dela. Como os indivíduos são sociais, as informações advindas dos outros membros do grupo também são conhecidas (SERAPIÃO, 2009).

Em PSO, o importante é o comportamento global, que é resultado das iterações entre os indivíduos da população. As propriedades de autoavaliação, comparação e imitação (KENNEDY *et al.*, 2001) é que emulam este comportamento.

A formação de equipe é observada em diversas espécies animais. Alguns grupos animais são controlados por líderes e tem seu comportamento regrado predominantemente por hierarquia, como os leões e lobos, por exemplo. No entanto, há formações em que predomina a auto-organização sem uma liderança, o que se observa nos movimentos de cardume, revoadas de pássaros ou rebanho de ovelhas (ENGELBRECHT, 2004).

O algoritmo PSO foi criado por Kennedy, psicólogo social, e Eberhart, engenheiro eletricista (1995), influenciados por um trabalho de Heppner e Grenander (1990), e envolve analogias ao comportamento de bando dos pássaros, na busca por alimento.

3.1 O algoritmo PSO original

No algoritmo PSO, cada indivíduo é representado por pontos (partículas) i que percorrem o espaço de busca R^d, sendo d a dimensão do espaço de busca. A ideia inspirada em sistemas cognitivos faz com que essas partículas movam-se de forma convergente, influenciando-se mutuamente (SERAPIÃO, 2009; ENGELBRECHT, 2005). O objetivo da coletividade é, obviamente, encontrar a melhor solução para um problema.

O algoritmo original considera que cada partícula é composta de três vetores *d*-

dimensionais: o primeiro é a posição $\vec{x}_i$ da partícula no espaço de busca, que correspondente à solução para a função objetivo; o segundo, a velocidade $\vec{v}_i$, responsável por atualizar as coordenadas da posição $\vec{x}_i$; o último, $\vec{p}_i$ ou *pbest*$_i$ (*previous best*), é o melhor resultado encontrado para a função objetivo, ou a melhor posição $\vec{x}_i$ encontrada até o momento. O valor de *pbest*$_i$ é compartilhado com as demais partículas durante cada iteração. O melhor valor encontrado pelo enxame é denominado p_g ou *gbest*$_i$ (*global best*) (POLI *et al.*, 2007).

A seguir, um pseudocódigo do PSO original (KENNEDY e EBERHART, 1995):

1. Iniciar uma população de partículas com posições x_i e velocidades v_i aleatórias, com d dimensões no espaço de busca.
2. Repetir até que um critério de parada seja atendido (por exemplo, uma função objetivo suficientemente boa ou um número de iterações definido)

Para cada partícula:

Calcular a função objetivo de d variáveis;
Comparar p_i com *pbest* e atualizar *pbest* com o melhor valor entre os dois;
Comparar *pbest* com *gbest* e atualizar *gbest* com o melhor valor;
Atualizar x_i e v_i de acordo com as seguintes equações:

$$\vec{v}_i \leftarrow \vec{v}_i + \vec{U}(0, \Phi_1) \cdot (\vec{p}_i - \vec{x}_i) + \vec{U}(0, \Phi_2) \cdot (\vec{p}_g - \vec{x}_i) \quad (38)$$

$$\vec{x}_i \leftarrow \vec{x}_i + \vec{v}_i \quad (39)$$

Sendo:
$\vec{U}$ um vetor de números randômicos entre 0 e Φ;
$\Phi_1 e \Phi_2$ constantes denominadas componentes cognitivo e social.

3.2 Modificações no algoritmo PSO

A partir do algoritmo original, muitos *esquemas* (SERAPIÃO, 2009) promoveram atualizações no PSO, visando evitar os fenômenos de convergência antecipada ou divergência entre as partículas. Para controlar a magnitude das velocidades, pode se estabelecer um limitador do tipo $\pm v_{max}$. Outra proposição nesse sentido é o fator de constrição χ (CLERC, 1999; EBERHART e SHI, 2000; CLERC e KENNEDY, 2002). A constante é calculada em função dos parâmetros cognitivo e social Φ_1 e Φ_2:

$$\chi = \frac{2}{\left| 2 - \varphi - \sqrt{\varphi^2 - 4} \right|} \quad (40)$$

em que

$$\varphi = \Phi_1 + \Phi_2, \quad \varphi > 4 \quad (41)$$

e por fim:

$$v_{i(t+1)} = \chi \cdot \left[\vec{v}_{i(t)} + \vec{U}(0,\Phi_1)\cdot\left(\vec{p}_{i(t)} - \vec{x}_i\right) + \vec{U}(0,\Phi_2)\cdot\left(\vec{p}_g - \vec{x}_i\right)\right] \quad (42)$$

Eberhart e Shi (1998) introduziram o componente inercial α à equação que atualiza a nova velocidade das partículas, de modo que

$$v_{i(t+1)} = \alpha\vec{v}_{i(t)} + \vec{U}(0,\Phi_1)\cdot\left(\vec{p}_{i(t)} - \vec{x}_i\right) + \vec{U}(0,\Phi_2)\cdot\left(\vec{p}_g - \vec{x}_i\right) \quad (43)$$

O valor de α é atualizado a cada iteração, de maneira que quanto maior o componente inercial, mais global o comportamento do enxame (SHI e EBERHART, 1998).

A partir destas modificações, inúmeras outras foram inseridas, à medida que os pesquisadores têm aprendido sobre a técnica PSO. As alterações visam otimizar, além da convergência prematura do algoritmo, questões relativas à *performance* dependente do problema a ser resolvido. É dizer que, configurando parâmetros diferentes para o PSO, o resultado poderá apresentar uma alta variação na *performance* (ESLAMI, 2012).

Mendes *et al.* (2004) introduziram o algoritmo *Full Informed* PSO (FIPSO). A diferença desta versão para a original é que as partículas usam informações de todos os seus vizinhos, e não apenas do que obtiver o melhor resultado para a função objetivo.

Uma versão com o componente inercial α dinâmico foi proposta por Jiao *et al.* (2008), e chamada *Improved* PSO (IPSO). O algoritmo usa um fator para decrementar α à medida que as iterações são realizadas. Os resultados do estudo apontaram uma melhoria significativa no resultado das funções objetivo referenciais *(benchmarks)* utilizadas no experimento em comparação com o tradicional algoritmo PSO.

Carvalho e Bastos-Filho (2009) propuseram o algoritmo *Clan* PSO. Este algoritmo é baseado no conceito de clãs, que são grupos de indivíduos, ou tribos, formados por laços de parentesco ou *linhagem,* dentro de uma sociedade maior, como um cacicado *(chiefdom).* *Clan* PSO é baseado no conceito de liderança de uma sociedade governada por clãs. Inicialmente as partículas são divididas em grupos *(clusters),* os clãs. Após uma iteração no espaço de busca, a melhor partícula dentro do clã é delegada líder. Em seguida, as partículas líderes fazem uma nova busca PSO, baseadas na informação do melhor líder. Este procedimento é denominado pelos autores de conferência dos líderes. A informação obtida pela conferência, por fim, é disseminada pelos líderes em seus clãs. O algoritmo foi testado em cinco *benchmarks* e obteve resultados melhores em comparação ao modelo PSO *global best* tradicional.

Recentemente, Su e Fan (2013) publicaram um estudo com uma nova versão do PSO baseada na lógica difusa. Em *Improved Fuzzy* PSO (IFPSO), cada partícula utiliza

informações de outras partículas, baseadas em um mecanismo *fuzzy*. O algoritmo atrasa a atração das partículas para o melhor global. No entanto, por postergar a convergência entre as partículas, IFPSO é melhor para resolução de problemas combinatoriais mais complexos.

De outra banda, a hibridização é uma área crescente na pesquisa em meta-heurísticas. O objetivo é mitigar limitações, combinando propriedades desejáveis de cada técnica. Como exemplo, tem-se o trabalho de Valdez *et al.* (2010), que combina PSO com Algoritmos Genéticos, utilizando lógica difusa para integrar os resultados de ambos os métodos. Eslami (2012) aponta o refinamento da aproximação e integração com outras técnicas como uma tendência para a pesquisa em Enxame de Partículas, além de um movimento no sentido de suas aplicações irem do laboratório para a indústria e o comércio.

3.3 Algoritmos genéticos e operadores

Algoritmo genético é um dos algoritmos paradigmas de computação evolucionária. Sua origem remonta a Holland (1975). Trata-se de uma técnica de busca global randomizada que resolve problemas imitando processos observados da evolução natural. GA envolve uma população de soluções candidatas. Cada solução é usualmente codificada como um vetor ou *string* denominado cromossomo. Em cada iteração, a função de adaptação de cada cromossomo é avaliada. Após completa a avaliação, os cromossomos adaptados são ranqueados de acordo com sua adaptação, de modo que os mais adaptados permaneçam e/ou se reproduzam na população.

Os indivíduos selecionados passam por operações de cruzamento para dar origem a novos indivíduos filhos. O processo de evolução admite ainda a probabilidade de mutação aleatória de alguns cromossomos, de modo que se mantenha razoável diversidade populacional e exploração global do espaço de busca. Por fim, os indivíduos menos adaptados podem ser retirados da população (ENGELBRECHT, 2005).

Assim como em PSO, a parametrização do GA é dependente do problema abordado. Isto significa que o ajuste de seus parâmetros tem sido realizado de forma empírica pelo modelador do problema (EIBEN *et al.*, 1999). Por exemplo, quando se quer um comportamento mais global em GA, a probabilidade de mutação é aumentada. Desejando uma convergência mais rápida, a probabilidade de mutação deve ser baixa.

A modelagem do operador varia também conforme a modelagem do problema. À Seção 4.2.7 será explicitado como foi inserido o operador de mutação de GA dentro do

algoritmo PSO proposto neste trabalho, com a finalidade de controlar a manutenção da diversidade do enxame.

3.4 Aplicações do PSO

PSO tem sido utilizado, com sucesso, como método de solução em uma ampla gama de problemas, de várias áreas de conhecimento. Poli *et al.* (2007) categorizou as principais aplicações do PSO, baseado em um levantamento das publicações disponíveis na base de dados de periódicos IEEE Xplore: *design* de redes elétricas, aplicações de controle, geração de energia e sistemas elétricos, programação (da utilização de máquinas), aplicações em eletrônica, *design* de antena, otimização de redes de comunicação, aplicações na área da saúde, mineração de dados e *clustering*, sistemas *fuzzy*, processamento de sinais, problemas de otimização combinatorial, robótica, predição, e finanças. A tendência de utilização do PSO para aplicações nestas áreas têm se mantido, conforme pode se observar pelos levantamentos de Serapião (2009) e Eslami (2012).

4 ABORDAGEM DUPLO-ESTÁGIO POR ENXAME DE PARTÍCULAS PARA O PROBLEMA DE *LAYOUT* CONSTRUÍDO COM ÁRVORES BINÁRIAS

Abordagens multi-estágio são comuns na resolução de problemas de *layout*. Como exemplo, temos os trabalhos derivados da proposta DISCON, de Drezner (1980). Tais proposições (VAN CAMP *et al.*, 1992; ANJOS e VANELLI, 2002; JANKOVITS *et al.*, 2011) representam as diferentes facilidades por círculos, conjecturando que sua acomodação na planta seja mais fácil do que a acomodação de formas retangulares. Os problemas são abordados em três estágios. Inicialmente é proposto um posicionamento das facilidades na planta, representadas apenas por seus centroides, de modo que se minimize o custo de transporte entre elas. Esta fase, denominada Estágio Um, não cuida de fornecer dimensões aos departamentos. A solução do primeiro estágio é referência para construção do próximo.

Na fase seguinte, o Estágio Dois, as facilidades são representadas por círculos e busca-se, por otimização não-linear, posicioná-las de maneira não sobreposta na planta. Por fim, tem-se o Estágio Três, em que às posições dos círculos são dispostos quadrados de área igual à requerida para a facilidade. A otimização desta fase cuida de alterar as posições e dimensões dos departamentos de modo que não se sobreponham, gerando a solução (*layout)* final. À figura 8, uma ilustração conceitual da abordagem citada.

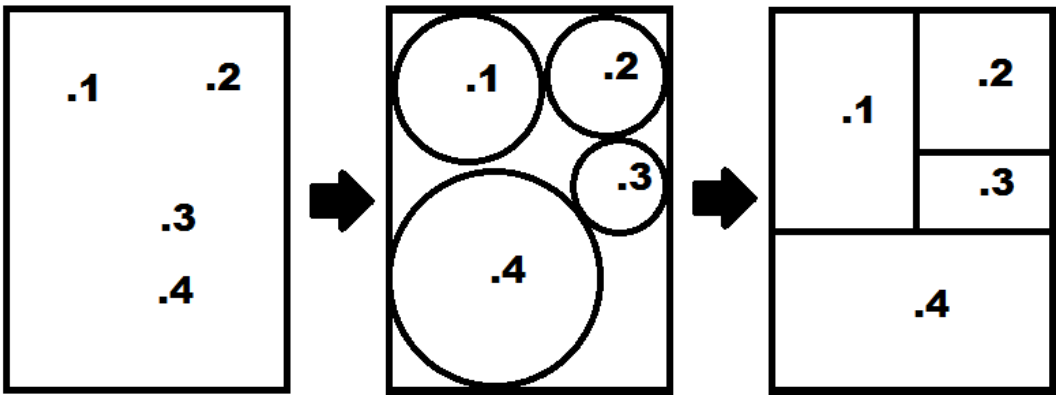


Figura 8 – otimização do FLP em três estágios

Outro trabalho que resolve o problema de maneira multi-estágio é o de Chen *et al.* (2000). Conforme já explicitado à Seção 2.1.5, o primeiro estágio constitui-se na disposição das facilidades, representadas pelo seu centroide, no R^2, a partir da solução de um problema de distância entre as facilidades de acordo com seu custo. A partir desta disposição, é realizado um mapeamento do plano cartesiano, de maneira a gerar uma matriz de particionamento da planta. O *layout* final é então otimizado através do algoritmo *simulated annealing.*

A abordagem multi-estágio é utilizada na formulação do método de solução proposto nesta pesquisa. No entanto, ora diferencia-se dos demais métodos expostos pelo fato dos estágios serem realizados de forma sucessiva. A cada iteração, um *layout* denominado auxiliar é otimizado, dele obtendo-se a representação de um *layout* factível. O *layout* factível é avaliado como solução para o problema. No entanto, para a próxima iteração, o *layout* auxiliar é que é modificado, dele novamente obtendo-se um *layout* factível.

4.1 Definição da função objetivo

Um *layout* gerado por matriz de particionamento é sempre factível no que se refere às restrições de tamanho e sobreposição das facilidades (KIM e KIM, 1998). Isto é facilmente observado, uma vez que a partir de uma planta fornecida são distribuídas as áreas das facilidades, de modo que não há como haver sobreposição entre elas, o que leva a concluir que as soluções geradas também não violam as restrições de posicionamento dentro dos limites da planta. Analogamente, podemos estender esta interpretação ao *layout* gerado por uma árvore de cortes.

Os FLPs referidos à tabela 2 cuidam de minimizar *z*, o somatório dos custos de transporte de material *c* entre as facilidades *i* e *j*, que é diretamente proporcional à distância *d* entre elas, ou seja:

$$Minimizar\ \ z = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} c_{ij} d_{ij} \tag{44}$$

Como as restrições de área, sobreposição e posicionamento dentro dos limites da planta (inequações 9 a 18) nunca são violadas se o problema for representado como uma árvore de cortes, resta ser considerada a seguinte restrição:

$$\beta i_i \leq \frac{B_i}{b_i} \tag{45}$$

em que β é a razão de aspecto entre a maior dimensão (B) e a menor dimensão (b) de cada

facilidade *i*. A razão de aspecto e as dimensões das plantas são especificadas para cada problema e podem ser verificadas à tabela 3.

Tabela 3 – dimensões das plantas e restrições de forma para os FLPs pesquisados

Conjunto	Instâncias	Dimensão máxima	Requisito	Fonte
o7	7	13 × 8,54	$\beta_i < 4$	Meller *et al.* (1998)
o8	8	13 × 11,31	$\beta_i < 4$	Meller *et al.* (1998)
o9	9	13 × 12	$\beta_i < 4$	Meller *et al.* (1998)
vC10	10	51 × 25	$w_i, h_i < 5$	van Camp *et al.* (1992)
Ba12	12	10 × 6	$w_i, h_i < 1$	Bazaraa (1975)
Ba14	14	9 × 7	$w_i, h_i < 1$	Bazaraa (1975)
AB20	20	3 × 2	$\beta_i < 4$	Armour e Buffa (1963)
SC30	30	15 × 12	$\beta_i < 5$	Liu e Meller (2007)
SC35	35	16 × 15	$\beta_i < 4$	Liu e Meller (2007)
Du62	62	100 × 137,18	$\beta_i < 4$	Dunker *et al.* (2003)

***β*: razão de aspecto; *w*: largura; *h*: altura; *i*: facilidade**

Meta-heurísticas como GA, PSO, ACO, *Simulated Annealing*, entre outras, visam encontrar soluções ótimas para problemas de otimização irrestrita, de modo que sua utilização em problemas restritos, como os FLPs deste trabalho, depende de adaptações (ENGELBRECHT, 2005). As adaptações mais comuns a heurísticas envolvem a inclusão de um fator penalizante à função objetivo, denominado função de penalidades, transformando o problema restrito em irrestrito.

A função de penalidades deve ser concebida de modo que represente da melhor forma as restrições a que se refere. Um *layout* gerado por árvore de decisão sempre satisfaz às restrições do problema, excetuando as de forma. Neste trabalho, tais restrições são convertidas em uma função de penalidades de tal forma que soluções factíveis são obtidas mais facilmente pelo algoritmo.

O fator que é multiplicado à função objetivo foi proposto por Kim e Kim (1998), e é dado por $(1 + \alpha N_S)$, em que N_S é o número de facilidades para as quais as restrições de forma não são satisfeitas e α é o peso da penalidade sobre a função objetivo. Se o peso é muito alto, a qualidade da solução final pode ser afetada, eis que a factibilidade do *layout* é evidenciada em detrimento do custo. Ao contrário, se o peso é muito baixo, o algoritmo pode não convergir para uma solução factível.

O fator de penalidades foi obtido empiricamente pelos autores do modelo original. Testado em uma amplitude de 0,01 a 2, o melhor fator para otimização do FLP por *Simulated Annealing* foi 0,3. Os melhores fatores obtidos por este trabalho são os expostos à tabela 7.

Desta forma, a função objetivo adaptada ao propósito desta pesquisa resta desenvolvida

da seguinte forma:

$$Minimizar\ z = (1 + \alpha N_S) \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} c_{ij} d_{ij} \tag{46}$$

4.2 O algoritmo proposto

A otimização proposta é baseada no algoritmo enxame de partículas e funciona da seguinte maneira:

a) Um enxame com *n* partículas é inicializado aleatoriamente. Cada partícula *i* é representada por uma matriz **posição** X_i que carrega as informações do *layout* auxiliar, utilizado para gerar o *layout* factível.

b) A seguir são realizadas as iterações do algoritmo, iniciando pela construção de uma matriz de distâncias entre as facilidades.

c) A partir da matriz de distâncias, o próximo passo na iteração consiste em gerar uma árvore de cortes, que será utilizada na construção do *layout*.

d) O *layout* factível então é construído pela alocação das facilidades na planta, de acordo com o estabelecido na árvore de cortes.

e) A seguir, para cada *layout* factível é calculada a função objetivo de acordo com a equação 46 e escolhido o melhor *layout*, isto é, aquele que tiver a menor função objetivo.

f) O enxame então é atualizado influenciado pela posição da partícula correspondente ao melhor *layout.*

g) Após *k* iterações, se satisfeito o critério, ocorre a parada do algoritmo.

Às próximas seções, as fases mencionadas, bem como as variáveis nela envolvidas, serão detalhadas. À figura 9, um fluxograma do algoritmo proposto.

4.2.1 Inicialização *pseudo*-aleatória

Para o algoritmo proposto, o primeiro passo consiste da *leitura* dos dados dos FLPs fornecidos. Os custos de transporte entre as facilidades *i* e *j* são determinados por matrizes. À tabela 4, os custos fornecidos no problema O7, de Meller *et al.* (1998).

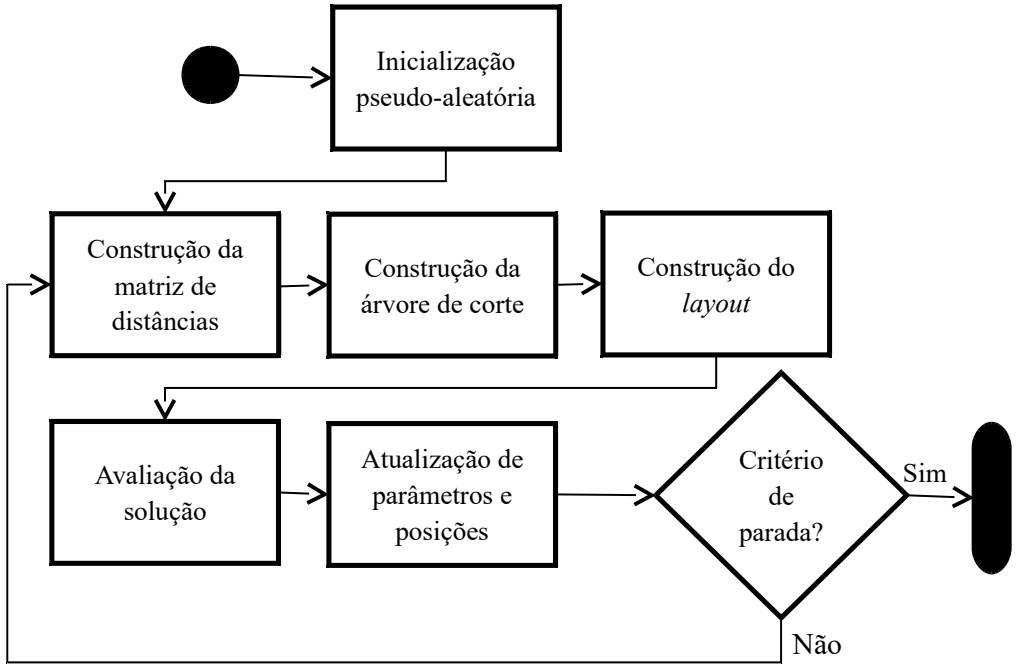


Figura 9 – fluxograma do algoritmo proposto - abordagem com árvores binárias

Tabela 4 – matriz de custos do problema O7 (Meller *et al.*, 1998).

	1	**2**	**3**	**4**	**5**	**6**	**7**
1	-	0	0	5	0	0	1
2	-	-	0	3	0	0	1
3	-	-	-	2	0	0	1
4	-	-	-	-	4	4	0
5	-	-	-	-	-	0	2
6	-	-	-	-	-	-	1
7	-	-	-	-	-	-	-

No início do algoritmo também são definidos os parâmetros utilizados pelo PSO para guiar o procedimento de otimização. Tais variáveis serão detalhadas à Seção 4.2.5. Em seguida são inicializadas as partículas.

Cada partícula, que representa um *layout* (ou solução), é uma matriz composta de *n* vetores-linha $x_i =< x_{1i}, x_{2i}, w_i, h_i, A_i >$, cujas componentes são, respectivamente, a abscissa e a ordenada do centroide, a dimensão horizontal (largura), a dimensão vertical (altura) e a área da facilidade retangular *i*. A_i é preestabelecida, x_{1i}, x_{2i} e w_i são variáveis otimizadas durante a execução do algoritmo e $h_i=A_i/w_i$.

O algoritmo é inicializado com o posicionamento aleatório uniforme das facilidades,

distribuídas a partir da origem do plano cartesiano (R^2), na forma da equação:

$$(x_{1i}, x_{2i}) = (U(0,\delta), U(0,\delta)), \qquad \delta = \sqrt{\sum_{i=1}^{n} A_i} \tag{47}$$

em que x_{1i} e x_{2i} são a abscissa e a ordenada do centroide da facilidade retangular i, U é a distribuição uniforme de mínimo e máximo indicados, δ é a constante de dispersão, que determina o afastamento da facilidade da origem do R^2, n é o número de facilidades e A_i é a área requerida para a facilidade i. Quanto maior δ, mais separadas ficam as facilidades.

Para inicialização dos componentes w_i e h_i é estabelecida aleatoriamente uma razão de aspecto β_i para cada facilidade. As razões de aspecto são propositalmente distribuídas normalmente, com média μ e desvio padrão ς, a fim de aproximar a razão de aspecto da desejada, controlada sua diversidade.

Uma concepção ilustrativa do *layout* gerado aleatoriamente pode ser verificada à Figura 10.a.

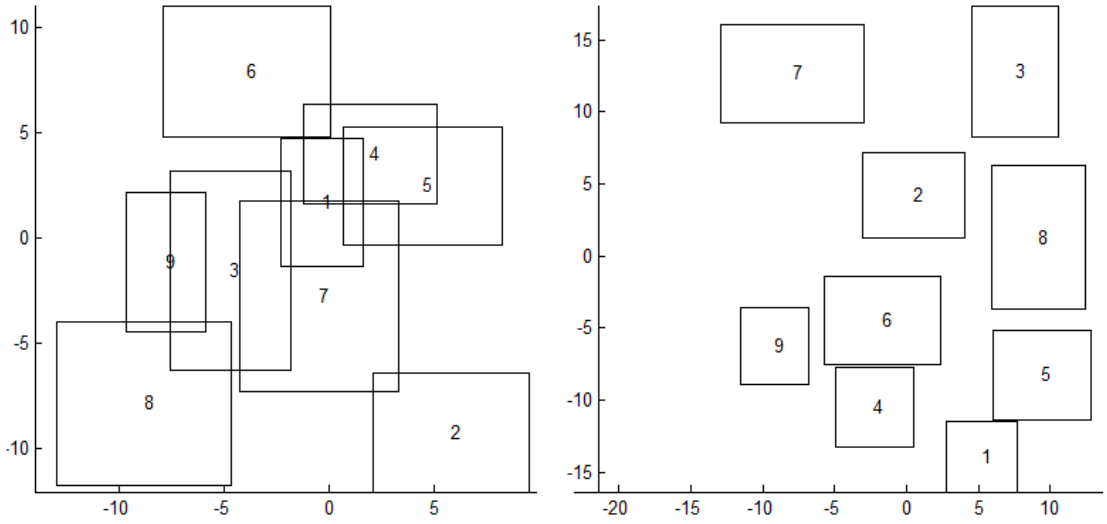
4.2.2 Construção de uma matriz de distâncias

A partir da solução inicial, construída aleatoriamente, é razoável aduzir que seria possível realizar a otimização do FLP empregando o PSO diretamente sobre o vetor x_i, tomando-o como a variável posição para o desenvolvimento do algoritmo. Tal tendência foi testada e verificada neste trabalho de pesquisa. Por exemplo, à figura 10, temos a representação de um *layout* otimizado por enxame de partículas após 100 iterações (b) e seu *layout* inicial correspondente (a), com áreas, custos, posições e dimensões aleatórias.

Para realização dos testes foi elaborada a seguinte função objetivo:

$$\text{Minimizar } f = (1+\alpha_1 N_S)(1+\alpha_2 S)\sum_{i=1}^{n-1}\sum_{j=i+1}^{n} c_{ij} d_{ij} \tag{48}$$

em que o custo de transporte de material é penalizado por dois fatores, o primeiro já explicitado à equação 46. No segundo fator temos S, o percentual de sobreposição entre as facilidades em relação à área total. As constantes α_1 e α_2 são pesos sobre as penalidades. A adição do segundo fator deve-se ao problema ser modelado como irrestrito, isto é, não há limites para posicionamento na planta.



(a) **(b)**

Figura 10 – um *layout* inicial aleatório (a) e a solução da otimização por PSO após 100 iterações (b)

No exemplo mencionado, os custos foram minorados em 50,76%, a sobreposição foi suprimida e as facilidades tiveram sua razão de aspecto abaixo da máxima estabelecida (β_i=3). No entanto, o que se observa é uma solução fora do paradigma normalmente encontrado na concepção de *layout*. Como se verifica à figura 10.b, as facilidades se dispersam, não se posicionando de modo a gerar um todo retangular, estética e funcionalmente viável, restando na planta muitas áreas ociosas entre as facilidades. Em outras palavras, embora seja verificada a otimização do *layout* se adotados os procedimentos citados, a solução apresentada não é factível, o que levou esta pesquisa a buscar novos procedimentos para abordar esta dificuldade.

Desta forma, assim como nos trabalhos citados previamente neste capítulo, uma abordagem multi-estágio é proposta com vistas a gerar soluções factíveis para o FLP. A contribuição desta pesquisa se dá no fato dos estágios serem repetidos a cada iteração. O primeiro estágio é denominado *layout* auxiliar neste trabalho e corresponde ao *layout* inicial e às alterações de posição nele realizadas durante o processamento do algoritmo.

Para fazer a transição do *layout* auxiliar para o *layout* final de cada partícula, a cada iteração, são tomadas as distâncias ponderadas entre as facilidades com a finalidade de gerar uma matriz de distâncias. A distância ponderada é dada por:

$$\eta_{ij} = \frac{\sqrt{(x_{1i} - x_{1j})^2 + (x_{2i} - x_{2j})^2}}{\sqrt{w_{i^2} + h_{i^2}} + \sqrt{w_{j^2} + h_{j^2}}} \tag{49}$$

e corresponde à razão entre a distância euclidiana dos centroides das facilidades e a soma

de suas diagonais. A distância ponderada é outra contribuição desta pesquisa e foi obtida experimentalmente. Foram testadas outras variáveis associadas à distância euclidiana, como, por exemplo, a soma da área das facilidades e a soma da raiz quadrada das áreas. Porém, a utilização da soma das diagonais no denominador da distância ponderada gerou melhores resultados preliminares. A necessidade de ponderar a distância deve-se ao fato de que os centroides das facilidades com maiores áreas sempre guardam maior distâncias das facilidades adjacentes.

Após a medição, as distâncias ponderadas são organizadas na forma de matriz. À figura 11, um exemplo da função da distância ponderada: apesar da distância da facilidade 1 para a facilidade 3 ser menor que para a facilidade 2, a distância ponderada é maior.

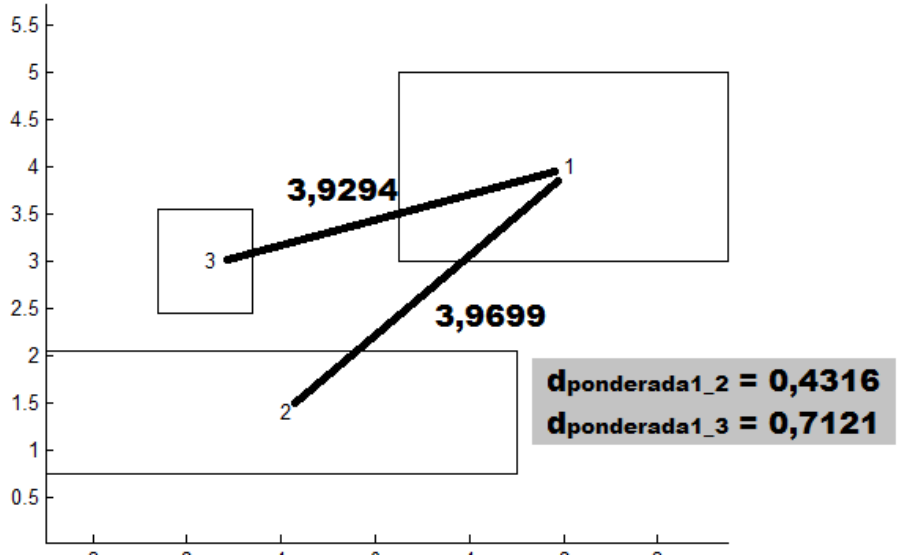


Figura 11 – exemplo do cálculo das distâncias euclidianas e das distâncias ponderadas entre uma facilidade e outras duas adjacentes

A matriz de distâncias (ponderadas) entre as três facilidades ilustradas à figura 11 é discriminada à tabela 5.

Tabela 5 – matriz de distâncias ponderadas entre as facilidades do *layout* representado à figura 11.

	1	2	3
1	-	0,4316	0,7121
2	0,4316	-	0,2689
3	0,7121	0,2689	-

4.2.3 Construção de uma árvore de cortes

A partir da matriz de distâncias de cada partícula, é possível a construção de uma árvore binária, em que se pretende, recursivamente, a associação de facilidades em função de sua proximidade, de forma que, quando da construção do *layout* definitivo, seja preservada esta relação de proximidade. O algoritmo de construção da árvore de cortes é derivado da proposta de Furtado e Lorena (1998), que, por sua vez, faz uso do algoritmo de clusterização de Anderberg (1973), no qual facilidades com alta proximidade formam um aglomerado. Este aglomerado é tomado como uma facilidade por si só para ser comparado às demais facilidades, com as quais poderá formar novos aglomerados, e assim sucessivamente até todas as facilidades restarem associadas. O processo é descrito da seguinte maneira:

1. Inicialização da matriz de distâncias Θ_n, de n aglomerados formados pelas facilidades individuais;
2. Geração de um novo aglomerado combinando outros dois para os quais $\Theta_{ij}=min(\Theta_n)$, isto é, a distância ponderada escolhida é o menor elemento da matriz.
3. Atualização da matriz de tráfego, substituindo as linhas e colunas correspondentes aos aglomerados selecionados pela distância ponderada entre os aglomerados restantes e o novo aglomerado gerado.
4. Se restar apenas um aglomerado, parar. Caso contrário, voltar ao passo 2.

Para estabelecimento das distâncias ponderadas entre o novo aglomerado e os demais (passo 3), faz-se a média das distâncias entre os aglomerados que o geraram e os demais.

Por fim, cada aglomerado gerado corresponde a um nodo pai da árvore, enquanto os nodos filhos são as facilidades ou outros aglomerados geradores.

4.2.4 Construção do *layout*

Para construção do *layout* final de cada partícula, para cada iteração, percorre-se a árvore binária do nodo pai para os nodos filhos, sempre priorizando os nodos hierarquicamente superiores. A partir do nodo pai são realizados recursivos cortes na planta, dividindo-a. As facilidades são representadas pelas folhas da árvore. A área de cada subdivisão da planta corresponde à soma das áreas das facilidades (folhas) a ela subordinadas.

A orientação do corte (vertical ou horizontal) foi definida da seguinte forma: caso a

altura for maior que a largura, faz-se um corte horizontal; caso contrário, faz-se um corte vertical. Alternativamente, pode-se adicionar uma probabilidade de mutação, fazendo com o que em algumas iterações, para algumas partículas, a lógica da orientação do corte seja alterada. À figura 12 temos um esquema exemplificativo de como é formado um *layout* a partir de uma árvore binária.

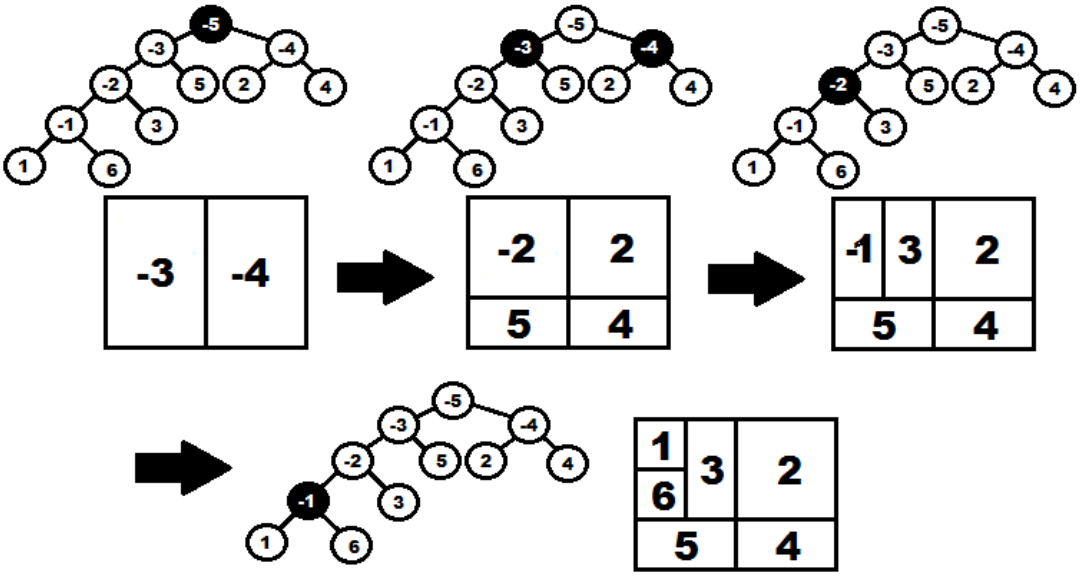


Figura 12 – construção de um *layout* de facilidades através da árvore binária

4.2.5 Avaliação das soluções

Após a geração da solução *(layout)* para cada partícula, é calculada a função objetivo na forma da Equação 46, isto é, o custo de transporte de material, diretamente proporcional ao fator de penalidades. A versão do algoritmo enxame de partículas utilizada nesta pesquisa foi a global, concebida por Kennedy e Eberhart (1995).

Realizado o cálculo do custo da função objetivo para cada partícula *i* em cada iteração, é determinada a matriz posição que implica a melhor solução para cada partícula ($pbest_i$), da seguinte forma:

a) Em se tratando da primeira iteração, $pbest_i$ é a posição X_i da partícula *i*.
b) Não se tratando da primeira iteração, X_i é comparado com $pbest_i$ e, se tiver menor custo, passa a ser o novo $pbest_i$.

Após o cálculo de todos os $pbest_i$, passa-se à determinação da melhor solução do enxame (*gbest*), isto é, a matriz posição da partícula com menor custo.

Para determinação do *gbest:*

a) Em se tratando da primeira iteração, *gbest* é a posição da partícula com menor

custo;

b) Não se tratando da primeira iteração, a posição da partícula com menor custo na iteração é comparada com *gbest* e, se tiver menor custo, passa a ser o novo *gbest*.

Gbest é a posição com o menor valor para a função objetivo. No entanto, não é assegurada a factibilidade do *layout* gerado por *gbest*, isto é, o *layout* representado por *gbest* pode conter facilidades com razão de aspecto violada. É que o fator de penalização não assegura a factibilidade, mas apenas aumenta a probabilidade de sua ocorrência.

Desta forma, para efeito da finalidade desta pesquisa, é necessário o registro da melhor solução factível obtida, mesmo que esta solução não seja *gbest*. Para registro, e apenas para este efeito, a solução factível de menor custo de cada iteração *k* é denominada $fbest_k$. *Gbest*, no entanto, é a posição utilizada como referência pelo enxame de partículas para realização da otimização. O custo de transporte de material, denotado pela equação 7, é também registrado, a fim de comparação com os trabalhos escolhidos da literatura científica.

Ademais, de fato, $fbest_k$ não é uma matriz posição em sentido estrito, pois na realidade seus valores são derivados de uma matriz posição X_i, que foi transformada em uma matriz de distâncias, em seguida em uma árvore de cortes, dando origem a uma solução na forma de *layout.* No entanto $fbest_k$ e seu custo são as variáveis mais importantes da pesquisa, eis que $min(fbest_k)$ é o *layout* factível com menor custo entre todas as iterações realizadas, caracterizando-se como solução para o FLP.

Ao final do algoritmo, espera-se que a otimização de *gbest* utilizando-se da função objetivo de equação 46 reflita na otimização de *fbest* e do custo de transporte de material, uma vez que *fbest* pode ser considerada a transformação de uma das soluções obtidas com a otimização de *gbest*.

4.2.6 Atualização de posições e velocidades

Uma vez determinadas $pbest_i$ e $gbest_k$, é realizada a alteração da posição das partículas no espaço de busca. A atualização das posições no algoritmo PSO utilizado neste trabalho é realizada pela adição da matriz velocidade V_i à matriz posição X_i. Note-se que a diferença básica entre este trabalho e outras pesquisas que abordam PSO recai no fato de a avaliação ser realizada levando-se em conta uma transformação das posições (*layout* final de cada iteração) das partículas no espaço de busca. No entanto, a atualização das posições é realizada na variável original, que viaja no espaço de busca – neste trabalho, a matriz

posição X_i (*layout* auxiliar).

Para tanto, foi testada a utilização do fator de constrição e do componente inercial, obtendo-se experimentalmente melhores resultados com o componente inercial χ variando linearmente de 0,5 a 0,9 da primeira à última iteração. Tal amplitude para o componente inercial é sugerida por Banks *et al.* (2007), baseado em experimentos de Eberhart e Shi (2000).

Assim sendo, a cada iteração *k*, a matriz de posições X_i de cada partícula é atualizada conforme a seguinte equação:

$$X_{ik} = X_{i(k-1)_{ik}} \tag{50}$$

em que:

$$V_{ik} = \chi V_{i(k+1)} + U(0,\Phi_1)\cdot\left(pbest_i - X_{i(k-1)}\right) + U(0,\Phi_2)\cdot\left(gbest - X_{i(k-1)}\right) \tag{51}$$

sendo X_{ik} e V_{ik} a posição e a velocidade da partícula *i* na *k*-ésima iteração e *U* um escalar gerado aleatoriamente com distribuição uniforme variando entre 0 e o valor dos componentes cognitivos (também denominados parâmetros de confiança) individual Φ_1 ou social Φ_2.

A fim de evitar o fenômeno de dispersão das partículas, é fixada a amplitude da velocidade, com limite mínimo *vmin* e limite máximo *vmax*, de maneira que *vmin* é o oposto de *vmax*. Se os elementos de V_i se tornarem menores que *vmin*, são necessariamente fixados neste valor. O análogo ocorre caso se tornarem maiores que *vmax*. Quando da primeira iteração, os elementos da matriz de velocidades são gerados aleatoriamente com uma distribuição uniforme de mínimo *vmin* e máximo *vmax*. Uma sugestão de Banks *et al.* (2007) para o controle da velocidade máxima é fixa-la em valor que varia de 0,1 a 1 vezes o máximo valor possível (desejável) no espaço de busca.

Ainda com a finalidade de evitar o fenômeno de dispersão das partículas, foi estabelecido experimentalmente um limite no espaço de busca (ENGELBRECHT, 2005), de modo que:

$$-1{,}5\delta < x_{1i} < 1{,}5\delta\,, \tag{52}$$

$$-1{,}5\delta < x_{2i} < 1{,}5\delta\,, \tag{53}$$

$$(1{,}25\beta)^{-1} h_i < w_i < 1{,}25\beta h_i\,, \tag{54}$$

isto é, nem a abscissa tampouco a ordenada do centroide de cada facilidade podem ser posicionadas em local superior 1,5 vezes a constante de dispersão δ. Também não é permitido que se estabeleça uma das dimensões da facilidade de forma que seja superior

1,25 vezes a razão de aspecto β em relação à outra dimensão.

Caso uma das equações 52 a 53 seja violada, o vetor linha x_i, que correspondente à facilidade fora do espaço de busca, é reposicionado da seguinte forma:

$$< x_{1ik}, x_{2ik}, w_{ik}, h_{ik}, A_i >=< \kappa x_{1i(k-1)}, \kappa x_{2i(k-1)}, \tau w_{i(k-1)}, A_i / \tau w_{i(k-1)}, A_i > \quad (55)$$

$$0 < \kappa < 1 \quad (56)$$

$$0 < \tau < 1 \quad (57)$$

Foram testados valores entre 0,1 e 0,99 para κ e τ, bem como valores uniformemente aleatórios. Após os testes, os melhores resultados deram-se fixando κ = 0,95 e τ = 0,9.

Atualizadas as posições no *layout* inicial, o algoritmo retorna ao início dos procedimentos descritos nesta seção, repetindo-os até o critério de parada ser preenchido. No entanto, com a finalidade de melhorar o procedimento de cobertura do espaço de busca pelas partículas, evitando o fenômeno de convergência, fenômenos de mutação das partículas podem ser emulados durante o processamento do algoritmo, o que será tratado na Seção 4.2.7.

4.2.7 Mutação das partículas

De acordo com Engelbrecht (2005), PSO é geralmente classificado como um paradigma em computação evolucionária. No entanto, o autor sustenta que embora existam semelhanças entre PSO e algoritmos evolucionários *(evolutionary algorithms* – EAs), PSO é inserido no campo de inteligência coletiva, separado da computação evolucionária.

Para Engelbrecht (2005), embora PSO e EAs sejam algoritmos estocásticos, com otimização baseada no conceito de população, as transformações em EAs ocorrem de forma discreta, enquanto as partículas se movem em trajetórias contínuas. Por outro lado, há elementos que, embora não funcionem de forma análoga à computação evolucionária no campo da inteligência coletiva, podem ser objeto de comparação.

Ainda de acordo com o autor, tem-se, em PSO, os componentes estocásticos $U(0,\Phi_1)$ e $U(0,\Phi_2)$, que são randômicos e alteram a magnitude das atualizações de posição das partículas. O propósito da mutação em EAs é introduzir novo material genético na população com a finalidade de aumentar sua diversidade, servindo como um mecanismo de balanço entre exploração global e local do espaço de busca. Em PSO, este controle é originalmente feito pelo controle da velocidade por limitação ou constrição (BANKS *et al.*, 2007).

No entanto, o que se observou nos testes realizados é que somente controlando a

velocidade das partículas não é possível a obtenção de resultados factíveis e com baixo custo para o FLP. Vide, por exemplo, o gráfico à figura 13, com a variação da função objetivo (equação 46) na otimização do problema o7 (MELLER *et al.,* 1998). Foram realizadas 100 iterações com 30 partículas.

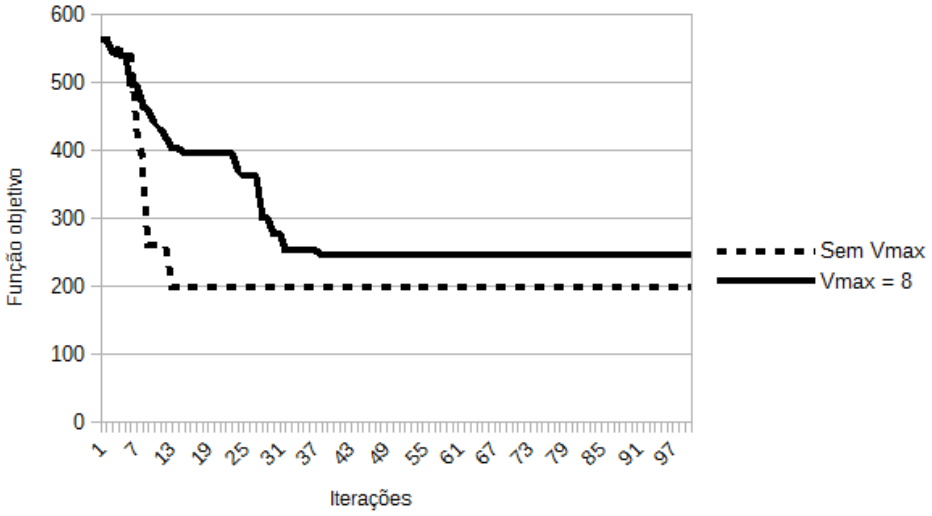


Figura 13 – variação na função objetivo com e sem controle de velocidade

Note-se pelo gráfico que o algoritmo passa a maior parte das iterações sem ganho significativo na função objetivo caso não haja limitação da velocidade, pois a conversão das partículas para um dos mínimos locais da função é rápida. Com a limitação da velocidade há uma otimização mais gradual da função objetivo, indicando um maior aproveitamento das soluções locais no espaço de busca. No entanto, ainda assim as partículas convergem, indicando que as alterações de posição das partículas no espaço de busca são prejudicadas pelas baixas velocidades. Em resumo, sem a limitação da velocidade, o comportamento global de busca é verificado. Caso contrário, é verificado um comportamento mais local.

Para abordar a dificuldade de balanço entre exploração global ou local, esta pesquisa utiliza-se do conceito de mutação próprio dos algoritmos genéticos, isto é, além da influência de umas partículas sobre as outras, são inseridas perturbações aleatórias externas (ENGELBRECHT, 2005), o que ocasiona uma maior diversidade populacional gerada tendenciosamente.

Neste trabalho, foram definidas as seguintes alterações por mutação no processamento do algoritmo: durante a atualização das posições, a troca de centroides entre facilidades mutuamente; durante a construção da árvore binária, a troca de ordem entre dois nodos

filhos ou duas folhas subordinadas ao mesmo nodo pai; durante a construção do *layout*, a troca de orientação do corte de vertical para horizontal ou vice-versa.

Para as duas últimas formas de mutação, após o sorteio de um número aleatório gerado pela função uniforme, com mínimo 0 e máximo 1, a mutação ocorre se a probabilidade for menor que o número sorteado.

Para a primeira forma de mutação (troca de centroides na matriz de posição X_i), após o procedimento descrito supra, é definido aleatoriamente o número de facilidades a trocar de posição. A distribuição experimentada para geração de números aleatórios foi a exponencial, com média igual a 20% do número de facilidades. Definidas quantas facilidades terão seus centroides permutados, é realizada a escolha aleatória das facilidades, através da distribuição uniforme.

As figuras 14 e 15 apresentam as três formas possíveis de mutação. Os valores para a probabilidade de mutação foram obtidos experimentalmente. Serão detalhados na seção "Resultados" deste capítulo.

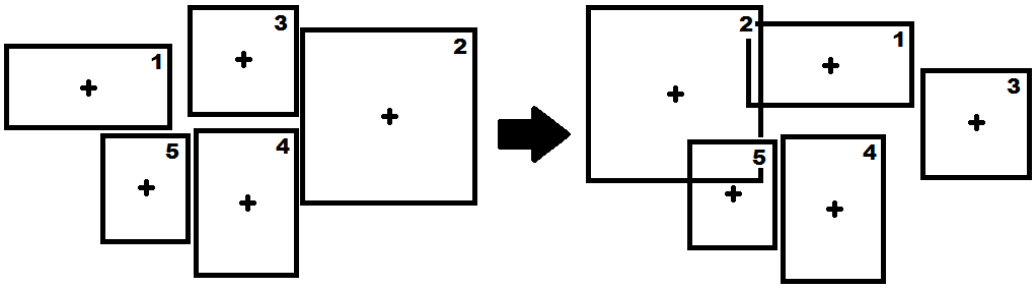


Figura 14 – primeira forma de mutação: troca de centroides entre as facilidades 1 a 3

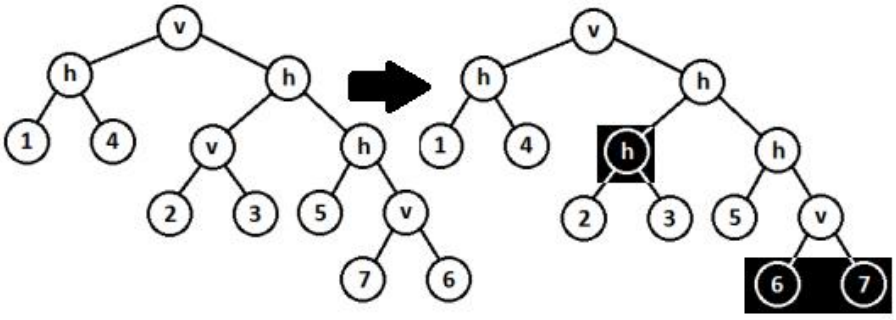


Figura 15 – segunda e terceira formas de mutação: alteração na ordem de prioridade para a árvore de cortes e na orientação do corte do *layout*

A quarta forma de mutação apresentada neste algoritmo é apenas a realização de permutações entre as facilidades que tenham a mesma área, caso existirem, ao final do processo de otimização, com a finalidade última de obter a configuração com o menor custo possível.

4.2.8 Finalização do algoritmo

Devido ao fato da variável χ (componente inercial) ter sido modelada com uma variação linear obtida em função do número de iterações, foi necessário fixar o critério de parada em um número *k* de iterações. O número de iterações realizado para cada experimento está apresentado à seção “Resultados” deste capítulo.

4.3 Resultados

Para avaliar o desempenho e as capacidades do algoritmo proposto, foram realizadas séries de testes computacionais. Os experimentos numéricos foram conduzidos em computador dotado de processador Intel® Core i5-3377U de 1,8 GHz, com memória RAM de 4 GB, rodando o Matlab versão R2012a no sistema operacional Windows 8.

Foram investigados os problemas constantes à tabela 3, do tipo restrito, em que se consideram as dimensões da planta, já fornecidas, e as diferentes áreas de facilidades retangulares. Seus dados detalhados constam anexos a esta dissertação.

Os parâmetros empregados na otimização pelo algoritmo PSO foram obtidos experimentalmente, tendo como referência inicial para teste as amplitudes sugeridas pela bibliometria de Poli *et al*. (2007). A tabela 6 apresenta uma lista com tais variáveis e a tabela 7 apresenta as demais variáveis necessárias para inicialização do algoritmo proposto.

Tabela 6 – parâmetros empregados na otimização por PSO com árvores binárias

FLP	*n*	*k*	Φ_1	Φ_2	χ	*V*	*l*
O7	7	400	1,5	1,4	0,4 – 0,9	-7 – 7	400
O8	8	500	1,5	1,4	0,4 – 0,9	-8 – 8	450
O9	9	600	1,5	1,4	0,4 – 0,9	-10 – 10	500
vC10	10	600	1,5	1,4	0,4 – 0,9	-25 – 25	600
Ba12	12	800	1,5	1,4	0,4 – 0,9	-6 – 6	800
Ba14	14	1000	1,5	1,4	0,4 – 0,9	-6 – 6	1000
AB20	20	1000	1,5	1,4	0,4 – 0,9	-2 – 2	1000
SC30	30	1000	1,5	1,4	0,4 – 0,9	-10 – 10	1000
SC35	35	1000	1,5	1,4	0,4 – 0,9	-7 – 7	1000

n: instâncias, k: iterações, Φ_1 e Φ_2: parâmetros de confiança;
χ: coeficiente inercial; V: velocidade; *l*: partículas

Tabela 7 – parâmetros específicos do algoritmo proposto

FLP	δ	ρ	α	x_d	w_i	w_0	β_0
o7	$A^{½}$	0,3	0,25	$-1{,}5\delta - 1{,}5\delta$	$0{,}2h_i - 5h_i$		$N(1;0{,}3)$
o8	$A^{½}$	0,4	0,25	$-1{,}5\delta - 1{,}5\delta$	$0{,}2h_i - 5h_i$		$N(1;0{,}3)$
o9	$A^{½}$	0,45	0,25	$-1{,}5\delta - 1{,}5\delta$	$0{,}2h_i - 5h_i$		$N(1;0{,}3)$
vC10	$A^{½}$	0,45	0,25	$-1{,}5\delta - 1{,}5\delta$	$0{,}2h_i - 5h_i$		$N(1;0{,}3)$
Ba12	$A^{½}$	0,48	0,4	$-1{,}5\delta - 1{,}5\delta$	$0{,}2h_i - 5h_i$	$N(1;0{,}3)$	
Ba14	$A^{½}$	0,48	0,4	$-1{,}5\delta - 1{,}5\delta$	$0{,}2h_i - 5h_i$	$N(1;0{,}1)$	
AB20	$A^{½}$	0,5	0,4	$-1{,}5\delta - 1{,}5\delta$	$0{,}2h_i - 5h_i$		$N(1;0{,}3)$
SC30	$A^{½}$	0,5	0,4	$-1{,}5\delta - 1{,}5\delta$	$0{,}2h_i - 5h_i$		$N(1;0{,}3)$
SC35	$A^{½}$	0,5	0,4	$-1{,}5\delta - 1{,}5\delta$	$0{,}2h_i - 5h_i$		$N(1;0{,}3)$
Du62	$A^{½}$	0,5	0,4	$-1{,}5\delta - 1{,}5\delta$	$0{,}2h_i - 5h_i$		$N(1;0{,}3)$

δ: dispersão inicial; A: somatório das áreas das facilidades; ρ: probabilidade de mutação; α: constante de penalidades; x_d: posicionamento no R^2; w_i: largura da facilidade; h_i: altura da facilidade; w_0: largura inicial; βo: razão de aspecto inicial; $N(\mu,\varsigma)$: distribuição normal de média μ e desvio padrão ς.

No que se refere à melhoria em relação ao *layout* inicial, o método obteve resultados razoáveis para os problemas de até 12 instâncias. Em algumas das simulações, mesmo com nenhum *layout* inicial factível, o algoritmo trouxe os *layouts* para a factibilidade e os melhorou significativamente no que se refere ao custo. O índice de melhoria em relação ao *layout* inicial apresentou-se em níveis entre 12,28 e 45,01% (vide tabela 8). A diferença considerável de melhoria das menores para as maiores instâncias deve-se à queda na qualidade do *layout* inicial gerado aleatoriamente à medida que se incrementa o número de facilidades dos problemas.

Tabela 8 – percentual de melhoria obtida da primeira solução factível encontrada pelo algoritmo até a solução final – abordagem com árvores binárias

Problema	Número de facilidades	Total de iterações	Melhor iteração (média[1])	1ª solução factível (custo médio[1])	Solução final (custo médio[1])	Melhoria (%)
o7	7	400	245	155,73	136,61	12,28
o8	8	500	209	312,41	253,51	18,85
o9	9	600	263	318,46	248,08	22,10
vC10	10	600	354	43560,86	23954,12	45,01
Ba12	12	600	219	19462,19	14876,90	23,56
Ba14-SC35	14-35	RESULTADO INFACTÍVEL: RAZÃO DE ASPECTO VIOLADA				

[1] Média após dez iterações

Gonçalves e Resende (2015) utilizaram-se de uma heurística baseada em um algoritmo

genético para abordagem de uma série de FLPs teste, do tipo irrestrito e do tipo restrito, entre eles os problemas abordados nesta pesquisa. Os autores apresentaram os melhores resultados da literatura pesquisada até o momento, obtendo melhorias para uma parte significativa dos problemas. Os resultados apresentados por Gonçalves e Resende (2015) e pelos autores por eles citados são utilizados para efeito de comparação e mensuração da eficiência do método apresentado nesta pesquisa.

As tabelas 9 a 11 apresentam o resultado médio e os melhores resultados obtidos por este trabalho, comparando-os com os trabalhos descritos em Gonçalves e Resende (2015) como referência. Acompanham os tempos de processamento, em segundos.

Tabela 9 – comparação com os melhores resultados da literatura científica para os problemas de 7 a 9 instâncias – abordagem com árvores binárias

Conjunto / Abordagem	o7	o8	o9
	Melhores Resultados		
I-MIP	131,64	242,89	235,95
MIP-ε	131,57	242,77	**235,87**
MIP-MINLP	131,64	**242,73**	236,14
GA-SP-MIP	131,63	242,88	235,95
STaTs	132	243,16	239,07
AS-FBS	131,68	243,12	236,14
BRKGA-LP	**131,56**	242,92	236,57
Esta pesquisa	**131,56**	247,46	245,52
Benchmark	131,56	242,73	235,87
Diferença (%)	**0**	**1,94**	**4,09**

Tabela 10 – comparação com os melhores resultados da literatura científica para os problemas de 10 e 12 instâncias – abordagem com árvores binárias

Conjunto / Abordagem	vC10	Ba12
MIP-MINLP	21297,98	8180
STaTs	19994,1	8264
AS-ST	19967	8252,67
PSO-RFBS	22889,65	8129
BRKGA-LP	**19951,17**	**8020,97**
Esta pesquisa	21418,42	13404,07
Melhor resultado	19951,17	**8020,97**
Diferença (%)	**7,35**	**67,11**

Tabela 11 – abordagens do FLP restrito utilizadas para comparação

Abordagem	Método	Fonte
I-MIP	Programação inteira mista melhorada	Sherali *et al*. (2003)
MIP-ε	Programação inteira mista com erro reduzido	Castillo e Westerlund (2005)
MIP-MINLP	Programação inteira mista linear e não-linear	Castillo *et al.* (2005)
GA-SP-MIP	Algoritmos genéticos e programação inteira mista	Liu e Meller (2007)
STaTs	Busca-tabu com árvore binária	Scholz *et al.* (2009)
AS-FBS	*Ant system* com matriz de particionamento	Wong e Komarudin (2010)
AS-ST	*Ant system* com árvore binária	Komarudin e Wong (2010)
PSO-RFBS	Enxame de partículas com matriz de particionamento	Kulturel-Konak e Konak (2011)
BRKGA-LP	Algoritmo genético de chaves aleatórias viciadas	Gonçalves e Resende (2015)

A partir das tabelas 9 e 10 é possível verificar que a diferença de desempenho entre o método apresentado por esta pesquisa e os melhores métodos empregados aumenta gradativamente, à medida que se aumenta o número de instâncias do problema. No entanto, do problema com 10 instâncias para o problema com 12 instâncias há um incremento considerável na diferença de resultados desta proposição, em comparação com outras abordagens do FLP. Embora haja uma melhoria a partir do *layout* inicial na ordem de 50%, o *layout* final tem um custo 67,11% superior ao melhor resultado encontrado na literatura.

Cabe ressaltar que, embora existam efetivamente 12 e 14 instâncias para os conjuntos Ba12 e Ba14, há um acréscimo de *pseudo*-facilidades em sua modelagem, em número de sete e quatro, respectivamente, que fazem as vezes de espaços vazios no *layout.* A essas facilidades não é atribuído custo algum ou mesmo restrição de medidas. Desta forma, os conjuntos Ba12 e Ba14 restam construídos com 19 e 18 instâncias, respectivamente. À figura 16, apresenta-se o *layout* final obtido com o conjunto Ba12, do qual é possível identificar as *pseudo*-facilidades no espaço ocioso da planta.

Realizados os experimentos, verifica-se que o método sugerido é eficiente para problemas com pequena quantidade de instâncias (até dez instâncias). Para o FLP com menor tamanho (sete facilidades - o7) foi obtido um resultado igual ao melhor já publicado. Para o problema de oito facilidades, o custo obtido é aproximadamente 1,9% pior do que o melhor resultado encontrado. Já para o problema de nove facilidades, foram obtidos resultados 4,09% inferiores ao melhor já encontrado. Para o problema de dez instâncias (vC10), o resultado é 7,53% inferior.

À medida que são acrescentadas facilidades aos problemas, fica mais difícil para o algoritmo encontrar soluções factíveis e de baixo custo, uma vez que o espaço de busca ganha dimensões e mais facilidades precisam ter sua razão de aspecto estabelecida dentro dos limites de factibilidade. A partir do problema com 14 instâncias (Ba14), verificou-se a ausência de factibilidade no resultado final das simulações. Isto é, foi possível a redução do

custo da função objetivo proposta. No entanto, nenhum dos *layouts* manteve todas suas facilidades dentro das restrições de forma.

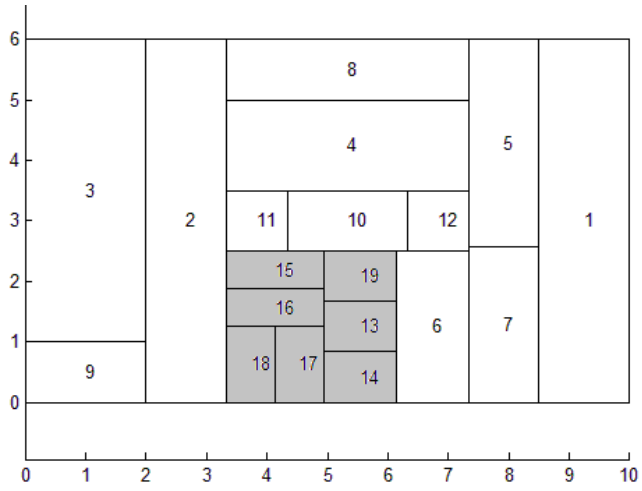


Figura 16 – melhor *layout* encontrado para o conjunto Ba12 – em destaque as facilidades falsas, acrescidas com a finalidade de ocupação de espaços ociosos na planta.

Em resumo: para FLPs de até 10 instâncias, o algoritmo proposto possibilitou bons resultados em comparação com o *layout* inicial e resultados iguais ou inferiores, porém competitivos, em comparação com outras abordagens da literatura científica. Para o problema de 12 instâncias, embora se mantenha a factibilidade dos resultados, o custo final de transporte de material, variável objeto deste trabalho, é elevado (67,11% mais alto que o melhor resultado apresentado). Para problemas a partir de 14 instâncias, o algoritmo proposto não cumpre seu objetivo. As soluções finais factíveis obtidas para os FLPs o7, o8, o9, vC10 e Ba12 encontram-se às figuras 16 a 20.

No que se refere ao tempo de processamento, esta pesquisa obteve resultados similares a outros em que se utilizaram meta-heurísticas de inteligência coletiva (vide, por exemplo, KOMARUDIN e WONG, 2010). Não há, na literatura científica, uma padronização quanto ao ideal em termos de tempo de processamento computacional, embora, obviamente, quanto menor o tempo, menor o custo de processamento. No entanto, há que se ressaltar que o problema de *layout* é bem conhecido como np-*hard* (GONÇALVES e RESENDE, 2015).

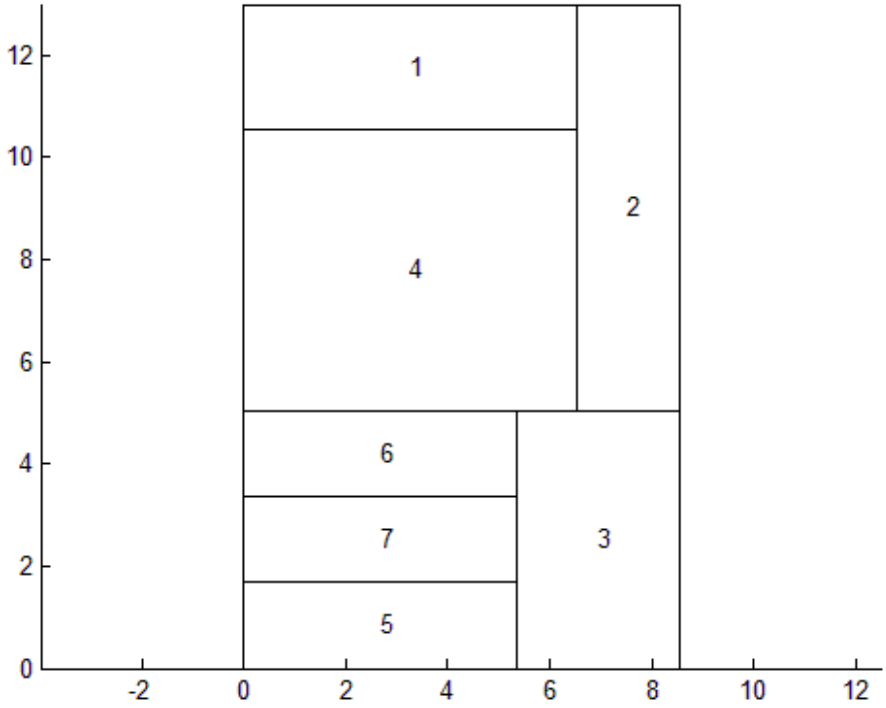


Figura 17 – *layout* de menor custo para o problema o7, com MFFC = 131,56.

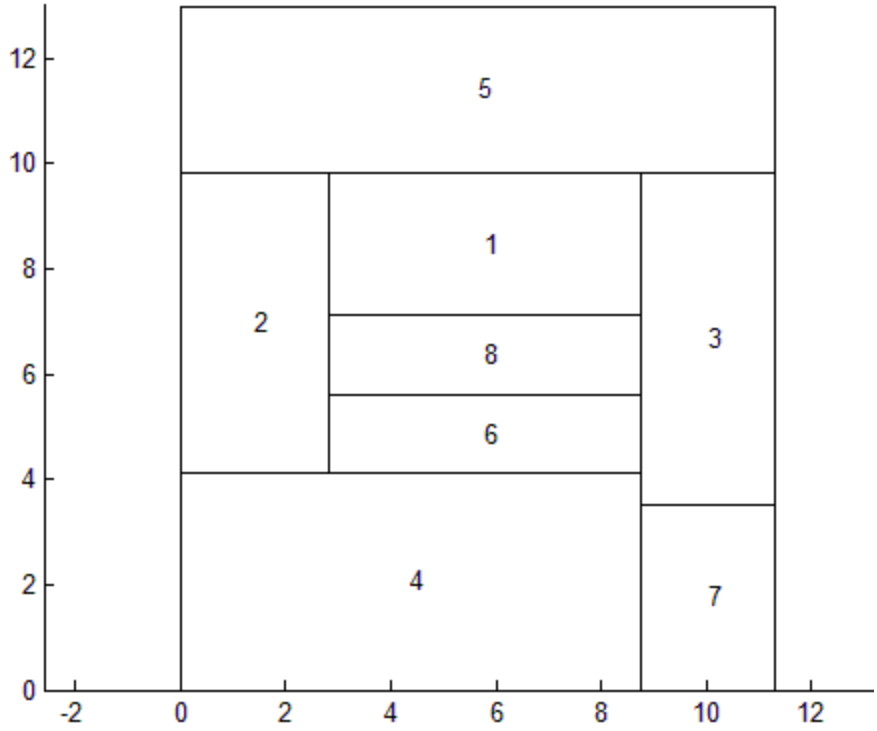


Figura 18 – *layout* de menor custo para o problema o8, com MFFC = 247,46.

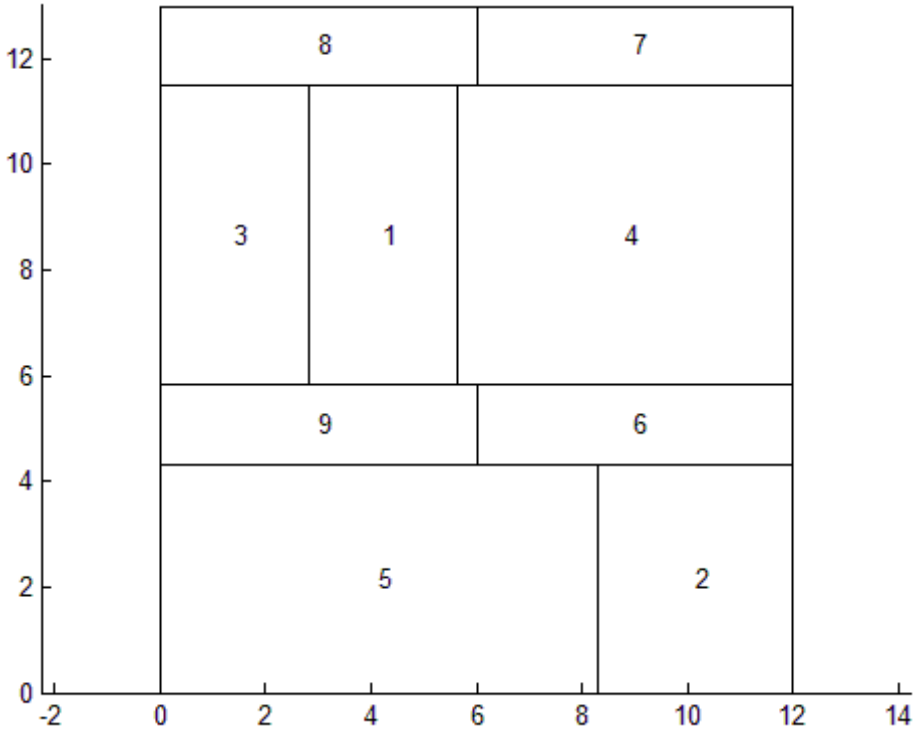


Figura 19 – *layout* de menor custo para o problema o9, com MFFC = 245,52.

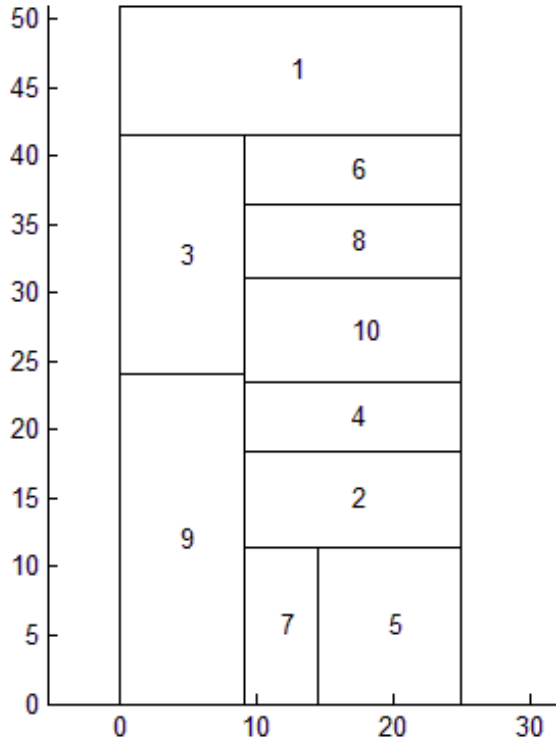


Figura 20 – *layout* de menor custo para o problema vC10, com MFFC = 21418,42.

Em se tratando de meta-heurísticas de inteligência de enxame, temos dois trabalhos recentes que podem ser utilizados como comparativos em relação a este. Komarudin e Wong (2010) resolveram o mesmo grupo de problemas-teste desta pesquisa, utilizando-se do algoritmo *Ant System*, uma variante da meta-heurística Colônia de Formigas. Os tempos de processamento variaram de 0,21 a 24 horas, considerando a solução dos FLPs de 7 a 62 facilidades (o7 e Du62, respectivamente). Asl e Wong (2015) desenvolveram um algoritmo baseado no PSO e, para problemas de instâncias de 8 a 20 departamentos, os tempos de processamento variaram de 205 a 2250 segundos (0,057 a 0,625 horas).

Por outro lado, métodos de otimização convexa, como por exemplo, Jankovits *et al.* (2011), obtiveram resultados em minutos (os autores do referido trabalho não explicitaram os tempos obtidos, apenas informando que para o primeiro estágio a solução se deu em segundos, enquanto para o segundo estágio a solução se deu de 3 a 5 minutos).

Embora o tempo para tomada de decisão seja um fator crítico, de acordo com See e Wong (2008), o *design* de *layouts* não é urgente em questão de tempo. A tabela 12 apresenta um resumo dos tempos de processamento para os problemas o7 a Ba12 e os demonstra compatíveis com os obtidos por Komarudin e Wong (2010), modelagem que também se utiliza de uma heurística de inteligência de enxame (*Ant System*).

Tabela 12 – tempos de processamento para abordagem dos FLPs o7 a Ba12, utilizando Enxame de Partículas com árvores binárias

Problema	Tempo (horas)		Diferença (%)
	AS-ST	Esta pesquisa	
o7	**0,21**	**0,24**	**14,28**
o8	**0,22**	**0,26**	**18,18**
o9	**0,25**	**0,26**	**4**
vC10	**0,28**	**0,40**	**42,86**
Ba12	**1,41**	**1,56**	**10,64**

Como foi possível verificar nesta seção, o método é competitivo para solução de FLPs de até 10 instâncias. No entanto, a partir de 12 instâncias, os resultados não compensam todo o processamento dispensado. E a partir de 14 instâncias, não são gerados resultados satisfatórios. O problema teste Du62 não foi testado nesta proposição, considerando o tempo que seria dispensado para seu processamento. Komarudin e Wong (2010) o processaram com critério de paradas em 24 horas de processamento. Uma vez que a partir do problema Ba14 já havia sido identificada a falta de factibilidade nos resultados, optou-se por não mencionar os resultados sobre este problema nos testes desta abordagem, o mantendo apenas na proposição descrita ao capítulo seguinte.

Visando tratar as adversidades encontradas, mais uma forma de construção do *layout* foi abordada nesta pesquisa. Enquanto este capítulo cuidou de relatar os resultados obtidos por plantas geradas a partir de árvores binárias (ou árvores *de corte*), o próximo se desenrola sobre a otimização de *layouts* através de matrizes de particionamento.

5 ABORDAGEM DUPLO-ESTÁGIO POR ENXAME DE PARTÍCULAS PARA O PROBLEMA DE *LAYOUT* CONSTRUÍDO COM MATRIZ DE PARTICIONAMENTO

Com a finalidade de tratar as dificuldades apresentadas no processo de solução das maiores instâncias, o custo mais alto em relação aos melhores resultados da literatura científica e a escassez de soluções factíveis, conjecturou-se que as soluções geradas através de matrizes de particionamento pudessem resolver naturalmente tais adversidades.

Como será possível verificar da explanação neste capítulo, a forma com que a matriz é construída guarda uma distribuição dos departamentos mais uniforme na planta. Corroborando para que isto aconteça, o algoritmo contém um dispositivo para que sejam posicionadas menos facilidades nas filas em que haja alocação das facilidades com maior área.

Da mesma maneira que o *layout* gerado pela árvore de cortes, o gerado pela matriz de posicionamento dispensa que se leve em conta a restrição de sobreposição das facilidades. Inicialmente tentou-se apenas a alteração na forma de construção do *layout*, mantendo-se todas as demais características da modelagem proposta no capítulo anterior, mas tal condição não se verificou na prática.

A maneira com que a função objetivo foi modelada para otimização do FLP gerado por árvore de cortes não pôde ser aproveitada para a modelagem através de matriz de particionamento. É que, para as instâncias maiores, ao utilizar-se da equação 46 como função objetivo, não se conseguiu experimentalmente encontrar um parâmetro α eficiente. Com α menor que 0,3, não foram gerados *layouts* factíveis, ou seja, para todos os resultados obtidos foram encontradas facilidades acima da razão de aspecto. Já com α fixado em um valor igual ou superior a 0,3, embora fossem gerados *layouts* factíveis, a baixa diversidade de boas partículas colocava o enxame em estagnação rapidamente.

Kim e Kim (1998) obtiveram resultado diverso do encontrado por esta pesquisa, ou seja, a otimização foi possível com a função objetivo descrita. No entanto, é imperioso

ressaltar que a heurística criada pelos autores foi baseada no algoritmo *Simulated Annealing* e que, dos conjuntos de dados teste utilizados nesta pesquisa, constou apenas o AB20 (ARMOUR e BUFFA, 1963).

A fim de abordar a dificuldade exposta, isto é, diminuir o custo de transporte de material enquanto respeitada a razão de aspecto, foi utilizado o fator razão de aspecto (*shape ratio factor* – SRF) de Wang *et al*. (2005). Os autores do SRF o propuseram em conjunto com a taxa de utilização de área (*area utilization factor* – AUF), que visa manter o menor número possível de área ociosa na planta. No entanto, devido ao modo de construção por matriz de particionamento, que evita áreas ociosas no *layout*, essa última variável não teve aplicação neste trabalho.

No sentido de contabilizar custos de investimento em imobilizados, isto é, na construção da planta industrial, é necessário levar em consideração também a forma das facilidades ou departamentos. Quanto mais regulares forem as formas geométricas dos departamentos, menor o custo de arranjo e construção do *layout*. É dizer que, em se tratando de facilidades retangulares, quanto mais próximo do quadrado for sua forma geométrica, menor o custo de produção.

Por exemplo, ao pretendermos construir um departamento de uma unidade de área (u.a.), suas dimensões de menor custo devem ter uma unidade de comprimento (u.c.). Por consequência, o menor perímetro será alcançado, com 4 u.c. Por outro lado, se diminuirmos uma das dimensões para, por exemplo, 0,5 u.c., e quisermos manter a área em 1 u.a., precisaremos da outra dimensão valendo 2 u.c. e, consequentemente, o perímetro valerá 5 u.c.

Considerando P o perímetro de um retângulo, A a sua área e a e b suas dimensões:

$$P = 2(a+b) \tag{58}$$

$$A = ab \tag{59}$$

$$b = \frac{A}{a} \tag{60}$$

$$P = 2\left(a + \frac{A}{a}\right) \tag{61}$$

Derivando a função P de $R^+ \rightarrow R^+$ em relação a a, sendo A uma constante, e considerando que em $P'=0$ têm-se um ponto crítico da função, de acordo com o teste da primeira derivada:

$$0 = \frac{d}{da}\left[2\left(a + \frac{A}{a}\right)\right] \tag{62}$$

$$0 = 2\left(1 - \frac{A}{a^2}\right) \tag{63}$$

$$A = a^2 \Rightarrow a = \sqrt{A} \Rightarrow a = b \tag{64}$$

Da equação 46 extrai-se que, quando a área é igual a um quadrado, temos um ponto crítico na função *P*, ou seja, o valor da dimensão é igual à raiz quadrada da área. Já verificamos empiricamente que se trata de um ponto de mínimo para a função *P*, mas podemos seguir realizando o teste da segunda derivada *P''*, definida por:

$$P'' = 2\frac{A}{a^3} = 2\frac{A}{\left(\sqrt{A}\right)^3} = 2\frac{\sqrt{A}}{A}. \tag{65}$$

Como *A* é uma constante não negativa, *P'' > 0*, o que significa que o ponto crítico é de mínimo, de acordo com o teste da segunda derivada, de modo que quando $A=a^2$, temos o mínimo da função *P*, ou seja, o menor perímetro para um retângulo, mantendo-se sua área, é o perímetro de um quadrado, de maneira que:

$$P = 2(a+b) = 2\left(\sqrt{A} + \frac{A}{\sqrt{A}}\right) = 4\sqrt{A}. \tag{66}$$

Este conceito foi agregado à função objetivo por meio do SRF. Sejam w_i e h_i as dimensões da facilidade *i*, $A_i = w_i h_i$ a área de *i* e $P_i = 2(w_i + h_i)$ o perímetro de *i*, a razão de aspecto SR_i é dada por:

$$SR_i = \frac{P_i}{4\sqrt{A_i}} = \frac{w_i + h_i}{2\sqrt{w_i h_i}}, \tag{67}$$

de modo que quando P_i aproxima-se de $4\sqrt{A}$ temos o valor mínimo ideal $SR_i = 1$.

É possível verificar que uma facilidade quadrada apresentará a melhor razão de aspecto, uma vez que $h_i=w_i$ e, por conseguinte:

$$SR_i = \frac{h_i + h_i}{2\sqrt{h_i^2}} = \frac{2h_i}{2h_i} = 1. \tag{68}$$

A partir de SR é definido SRF (WANG *et al.*, 2005), que foi agregado à função objetivo e define-se pela média geométrica das razões de aspecto de todas as facilidades, na forma da equação 69. É dizer que, quanto mais departamentos com formatos próximos de quadrados, menor será o custo final da formatação do *layout*.

$$SRF = \left(\prod_{i=1}^{n} SR_i\right)^{\frac{1}{n}} \tag{69}$$

O SRF foi agregado à função objetivo e obteve *layouts* factíveis logo nas primeiras

iterações do algoritmo, diferentemente do fator α. Desta forma, a função objetivo foi alterada para a expressão:

$$Minimizar\ TLC = SRF\sum_{i=1}^{n-1}\sum_{j=i+1}^{n} c_{ij}d_{ij} \tag{70}$$

em que TLC é o custo total do *layout* (*total layout cost*).

Como o SRF ≥ 1, quanto maior o SRF, maior o TLC. A otimização do TLC influencia diretamente a proposta de otimizar simultaneamente o custo de transporte de material e o SRF, que são variáveis diretamente proporcionais. Desta forma, o SRF assume o lugar de função de penalidades da função objetivo.

5.1 A construção do *layout* a partir da matriz de particionamento

As diferenças entre esta modelagem e a proposta do capítulo anterior recaem sobre a função objetivo utilizada e a forma de construção do *layout.* A função objetivo já fora abordada na introdução deste capítulo, de modo que esta seção dedique-se à construção do *layout* por matriz de particionamento. À figura 21, um fluxograma adaptado a essa aproximação.

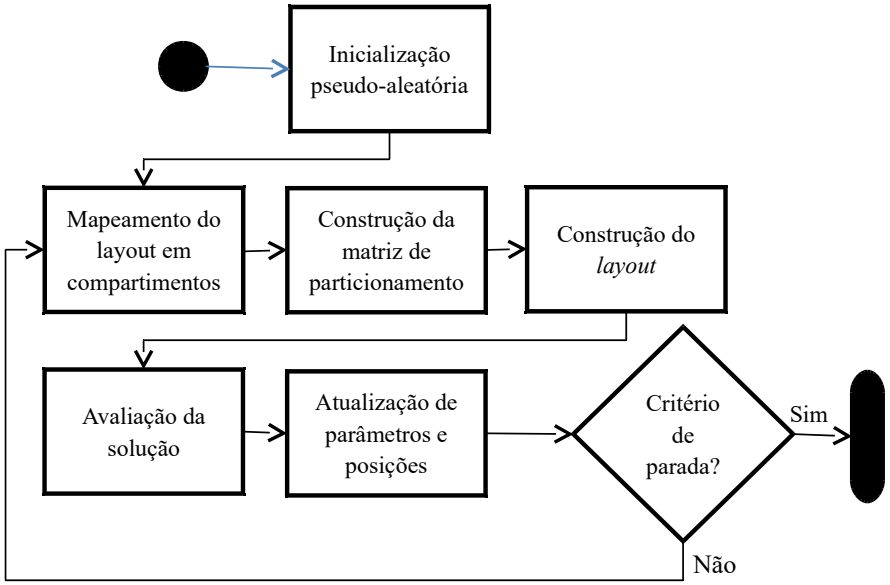


Figura 21 – fluxograma da abordagem proposta utilizando matrizes de particionamento

Diferentemente das propostas semelhantes já citadas, nesta pesquisa a matriz de particionamento não é gerada pela permutação das diversas células da matriz (KIM e KIM, 1998) ou por algum método de proximidades ou similaridade (Chen *et al.*, 2000). Nesta versão, assim como na proposta utilizando árvores de corte, é o algoritmo PSO o

responsável pela alocação das facilidades.

Inicialmente, para cada partícula do enxame, são gerados *layouts* auxiliares, cujas facilidades têm posições aleatórias X_i. O algoritmo PSO cuida de otimizar estas posições, tendo como base a função objetivo de expressão 70.

Para cada *layout* auxiliar ocorre um mapeamento das posições das facilidades, assim como em Chen *et al.*, 2000. A cada iteração do algoritmo é atualizado o conjunto de posições X_i, observadas as equações 50 a 57.

Para fazer a *tradução* do *layout* auxiliar para o *layout* final de cada iteração, uma malha compostas de intervalos do R^2 divide a planta em compartimentos menores, a cada compartimento sendo associado um elemento da matriz de particionamento, assim como proposto em Chen *et al.*, 2000. A figura 6 traz uma ilustração do procedimento de particionamento e formação da matriz. Neste trabalho foi implementada apenas a versão de particionamento no sentido horizontal (na direção das colunas, percorrendo linhas da matriz).

Após vários experimentos, obteve-se uma quantidade ideal *m* de linhas e colunas para a matriz de particionamento, conforme a expressão:

$$m = teto(\sqrt{n}) + 1 \tag{71}$$

ou seja, a raiz quadrada da quantidade de facilidades do problema, arredondada ao nível de seu primeiro inteiro igual ou subsequente, acrescida de uma unidade.

No entanto, pode ocorrer de duas ou mais facilidades estarem associadas ao mesmo compartimento. Outra dificuldade ocorre quando, na mesma fila da matriz de particionamento existem facilidades com áreas em proporções díspares – neste caso ocorre necessariamente um aumento da probabilidade da restrição da razão de aspecto máxima ser violada nas facilidades menores. Vide figura 22 para uma aproximação desta característica.

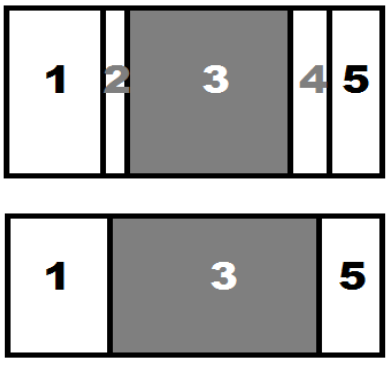


Figura 22 – fila de facilidades com uma delas em área muito superior a das demais: quando se exclui parte das facilidades da fila, a razão de aspecto tende a diminuir.

Note-se que, quando há muitas facilidades alocadas na mesma fila e uma delas tem área muito maior que as demais, as outras facilidades tendem a possuir razões de aspecto mais altas. Para lidar com tal característica, uma estratégia de posicionamento foi estabelecida, conforme os seguintes passos:

1. A prioridade de alocação na matriz de particionamento é da facilidade com maior área para a facilidade com menor área.
2. Caso já exista uma facilidade associada ao compartimento em que está posicionada a facilidade a ser alocada, segue-se o seguinte procedimento:
   a) Busca-se o compartimento adjacente livre mais próximo (figura 23);
   b) Busca-se na próxima vizinhança um compartimento livre, seguindo o sentido horário, a partir do compartimento mais à esquerda e acima (figura 24)
   c) Busca-se na vizinhança seguinte um compartimento livre, da mesma forma que em “b”.

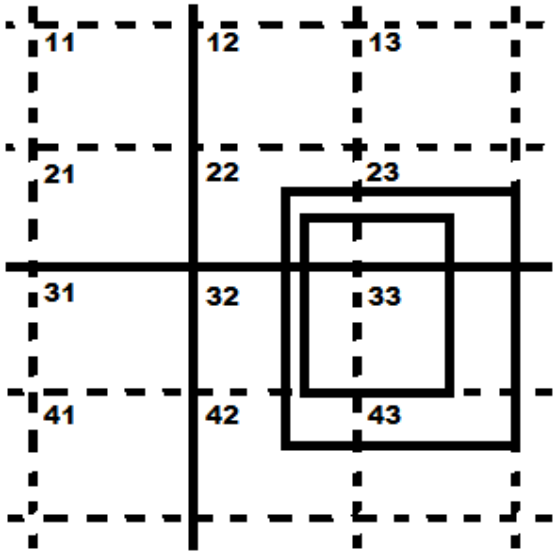


Figura 23 – regra de alocação na matriz de particionamento: a facilidade maior tem prioridade para alocar-se no compartimento 33 da matriz de particionamento, o que leva à facilidade menor alocar-se no compartimento 32, mais próximo de seu centroide, embora não o contenha.

Ao final dos procedimentos, caso não estejam alocadas todas as facilidades, as facilidades não alocadas são inseridas de forma randômica na matriz de particionamento.

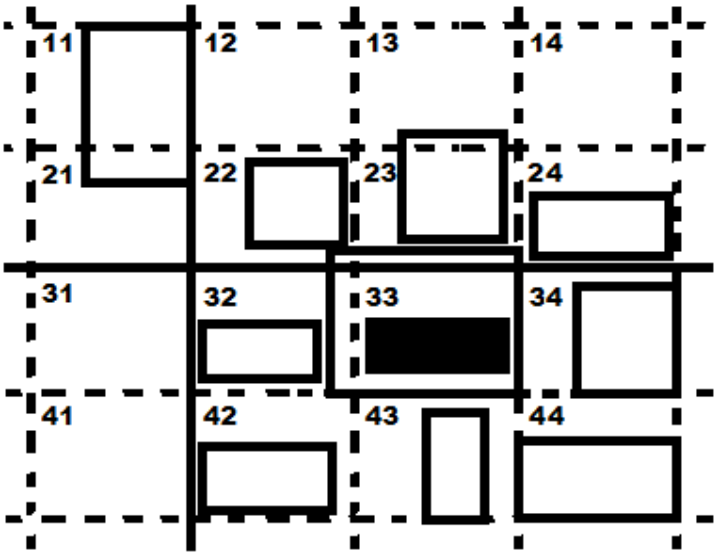


Figura 24 – regra de alocação na matriz de particionamento: o compartimento onde está posicionada a facilidade (33), bem como todos os compartimentos vizinhos, está ocupado, o que leva à necessidade de buscar sua alocação na próxima vizinhança, a partir do compartimento disponível mais à esquerda e acima (12), no sentido horário.

5.2 Resultados

Assim como na modelagem proposta no capítulo anterior, foram realizados testes utilizando-se dos conjuntos de dados provenientes dos problemas à tabela 3.

Em comparação aos fenômenos de dispersão e convergência do enxame, observados na aplicação do método sugerido anteriormente, este método tem como vantagem a desnecessidade de utilização de artifícios para manutenção da diversidade populacional no enxame. Para balancear o comportamento de busca global e local, um componente inercial χ variável, não linear, é sugerido neste trabalho, de modo que:

$$\chi_k = \chi_{\min}\left(\frac{k}{k_{final}} - 1\right)^2 + \chi_{\max} \tag{72}$$

em que χ_k é o componente inercial na k-ésima iteração.

Como χ_k é uma função quadrática, no início das iterações o componente inercial está em χ_{max}, quando é admitido um comportamento mais global ao enxame, em virtude das velocidades mais altas. Gradualmente χ_k é minorado até chegar a χ_{min}, na metade das iterações, quando seu comportamento será mais local, com velocidades mais baixas. A partir de então, o componente inercial gradualmente retorna a χ_{max}, no final das iterações, desta forma, permitindo velocidades pouco maiores, evitando a estagnação precoce do

enxame.

À figura 25 apresenta-se um gráfico da função objetivo ao longo das iterações, ao realizar uma simulação com o conjunto de testes SC30 (LIU e MELLER, 2007). Ressalte-se que não se trata do melhor resultado obtido pelo enxame até o momento (*fbest*), mas o melhor resultado obtido pelo enxame para cada iteração. É possível verificar o comportamento de busca global no início, com grandes variações na função objetivo, permitindo movimentos *uphill*, isto é, movimentos de fuga de mínimos locais. A variação diminui gradualmente, de modo que a busca passe a ser mais local, até que a melhoria não se dê de forma significativa.

Os parâmetros utilizados pelo algoritmo enxame de partículas, assim como na proposição explanada no capitulo anterior, foram obtidos empiricamente e seguem diretrizes de amplitude para teste conforme os valores mencionados pela pesquisa bibliométrica de Poli *et al.* (2007).

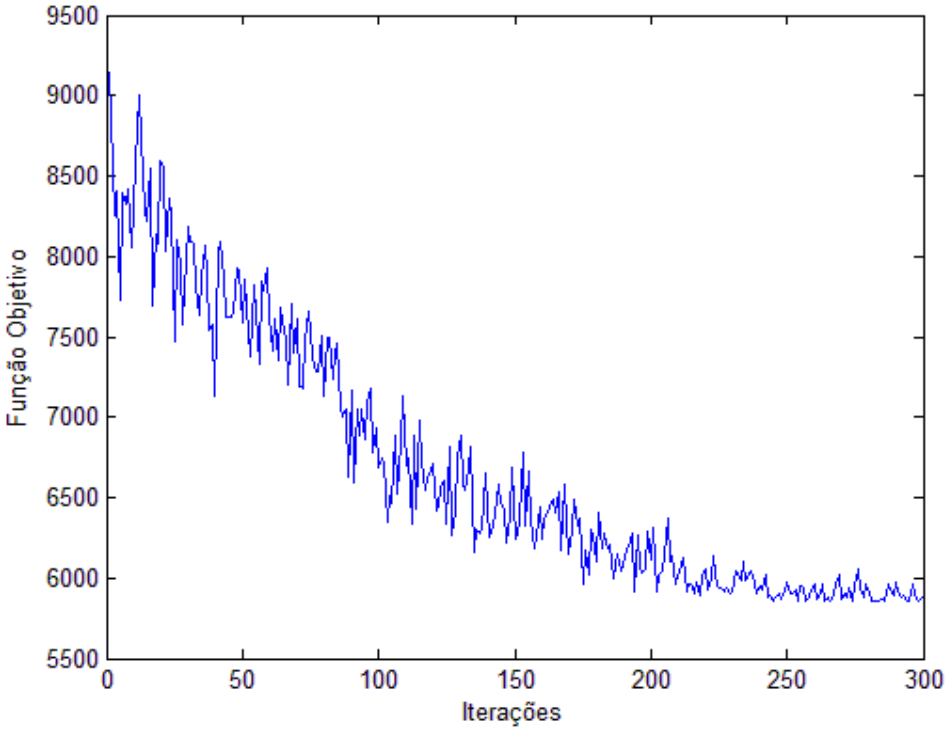


Figura 25 – comportamento da função objetivo para um teste realizado no conjunto SC30

Para a otimização, os parâmetros originais empregados no algoritmo PSO foram os descritos à tabela 13. Quanto à constante de dispersão, ao somatório das áreas das facilidades, aos limites de posicionamento no R e à razão de aspecto inicial, os valores são

coincidentes aos explicitados à tabela 7.

Tabela 13 – parâmetros empregados na otimização por PSO com matriz de particionamento

FLP	*n*	*k*	Φ_1	Φ_2	χ	*V*	*l*
O7	7	75	0,8	0,97	0,4 – 0,9	-7 – 7	50
O8	8	150	0,8	0,97	0,4 – 0,9	-8 – 8	100
O9	9	200	0,8	0,97	0,4 – 0,9	-10 – 10	150
vC10	10	400	0,8	0,97	0,4 – 0,9	-25 – 25	500
Ba12	12	600	0,8	0,97	0,4 – 0,9	-6 – 6	600
Ba14	14	1000	0,8	0,97	0,4 – 0,9	-6 – 6	1000
AB20	20	600	0,8	0,97	0,4 – 0,9	-2 – 2	600
SC30	30	600	0,8	0,97	0,4 – 0,9	-10 – 10	600
SC35	35	1000	0,8	0,97	0,4 – 0,9	-7 – 7	1000
Du62	62	2987[1]	0,8	0,97	0,4 – 0,9	-120 – 120	1000

n: instâncias, k: iterações, Φ_1 e Φ_2: parâmetros de confiança; χ: coeficiente inercial; V: velocidade; *l*: partículas

[1] Para Du62, o limite de iterações foi fixado em tantas quantas possíveis em 24 horas.

É possível verificar, em comparação entre as tabelas 6 e 13, que a abordagem com matriz de particionamento utilizou-se de menos partículas e iterações que o modelo com árvore binária. Desta forma é reforçada a conjectura de que a construção do *layout* por esta abordagem aumenta a ocorrência de resultados naturalmente factíveis.

Numericamente tal fenômeno foi verificado a partir da observação dos *layouts* iniciais, gerados aleatoriamente. À tabela 14 apresenta-se uma comparação entre o número de *layouts* factíveis na primeira iteração utilizando as duas abordagens. Para este teste foi rodado o algoritmo com 1000 partículas em apenas uma iteração. O experimento foi repetido 10 vezes. Os conjuntos vc10, Ba12 e Ba14 não foram incluídos, pois sua restrição de forma é relacionada à altura e largura mínimas.

Tabela 14 – taxa de *layouts* factíveis gerados aleatoriamente

FLP	***Layouts* factíveis (‰) na primeira iteração (média após 10 experimentos)**	
	Árvores binárias	**Matriz de particionamento**
o7	207,5	154,8
o8	318,4	355,2
o9	212,7	455,3
AB20	5,3	92,8
SC30	1,1	39,7
SC35	0	10,5
Du62	0	294,1

Ao contrário do observado nos experimentos que envolveram a abordagem do FLP por árvores binárias, nesta aproximação foi possível obter resultados factíveis em todos os problemas-teste. Uma característica importante é que a solução inicial aleatória foi de

qualidade superior à solução inicial gerada com árvores binárias, o que impacta diretamente a possibilidade de melhoria do *layout*, uma vez que *layouts* iniciais de qualidade são mais difíceis de serem otimizados (FURTADO e LORENA, 1997). A tabela 15 apresenta os percentuais de melhoria para os problemas-teste estudados.

Tabela 15 – percentual de melhoria obtida da primeira solução factível encontrada pelo algoritmo até a solução final – abordagem com matriz de particionamento

Problema	Número de facilidades	Total de iterações	Melhor iteração (média[1])	1ª solução factível (custo médio[1])	Solução final (custo médio[1])	Melhoria (%)
o7	7	75	60	156,89	141,18	10,01
o8	8	150	85	290,32	252,44	13,05
o9	9	200	167	287,32	257,48	10,38
vC10	10	400	354	34004,11	22185,37	34,76
Ba12	12	600	548	18756,73	9732,36	48,11
Ba14	14	600	535	11721,13	5971,03	49,05
AB20	20	600	395	10091,95	6537,12	35,22
SC30	30	600	442	8893,28	5005,13	43,72
SC35	35	1000	912	14815,19	5649,03	61,87
Du62	62	3005[2]	1678	5367541,32	4313765,04	19,63

[1] Média após dez iterações, exceto para Du62 (5 iterações) [2] Média de iterações em 24 horas

As tabelas 16 a 18 apresentam os menores custos obtidos em simulações processadas neste trabalho, em comparação com as principais abordagens da literatura, de acordo com Gonçalves e Resende (2015). Acrescente-se às comparações o método SA-MIP, de Kulturel-Konak e Konak (2014), que utiliza *Simulated Annealing* em hibridização com programação inteira mista para solução dos FLPs.

Dos resultados infere-se que o método obteve soluções competitivas para FLPs de até 14 instâncias, com destaque para os problemas de tamanho intermediário (vC10 e Ba12), que obtiveram diferenças de custo não superiores a 2,4%. A partir do problema AB20, com vinte instâncias, a diferença do MFFC obtido por esta abordagem, em comparação aos *benchmarks* da literatura científica, aumenta consideravelmente.

Divergindo da tendência apresentada para os problemas de número considerável de instâncias, para o problema com maior número de facilidades, Du62, foi obtido um resultado (MFC = 4216523,78) 13,3% maior que o único paradigma encontrado (KOMARUDIN e WONG, 2010), de custo igual a 3720521,16. Outras abordagens de Du62 o tomam como problema irrestrito (vide, por exemplo, Gonçalves e Resende, 2015).

Tabela 16 – comparação com os melhores resultados da literatura científica para os problemas de 7 a 9 instâncias – abordagem com matriz de particionamento

Conjunto / Abordagem	o7	o8	o9
	Melhores Resultados		
I-MIP	131,64	242,89	**235,95**
MIP-ε	131,57	242,77	235,87
MIP-MINLP	131,64	**242,73**	236,14
GA-SP-MIP	131,63	242,88	**235,95**
STaTs	132	243,16	239,07
AS-FBS	131,68	243,12	236,14
BRKGA-LP	**131,56**	242,92	236,57
Esta pesquisa	139,22	251,37	257,48
Benchmark	131,56	242,73	235,95
Diferença (%)	**5,82**	**3,56**	**9,12**

Tabela 17 – comparação com os melhores resultados da literatura científica para os problemas de 10 a 14 instâncias – abordagem com matriz de particionamento

Conjunto / Abordagem	vC10	Ba12	Ba14
MIP-MINLP	21297,98	8180	–
STaTs	19994,1	8264	**4712,33**
AS-ST	19967	8252,67	4724,68
PSO-RFBS	22889,65	8129	4913,22
BRKGA-LP	**19951,17**	**8020,97**	**4628,79**
GA-LP	–	8021	4686,81
Esta pesquisa	20425,81	8098	4919,19
Melhor resultado	19951,17	8020,97	4628,79
Diferença (%)	**2,38**	**0,96**	**6,28**

Tabela 18 – comparação com os melhores resultados da literatura científica para os problemas de 20 a 35 instâncias – abordagem com matriz de particionamento

Conjunto / Abordagem	AB20	SC30	SC35
STaTs	5225,96	3707	3604
AS-ST	5073,82	3868,55	4132,36
PSO-RFBS	5336,36	3443,34	3700,75
BRKGA-LP	5021,17	3367,87	**3316,77**
SA-MIP	**5016,93**	**3318,76**	3469,40
Esta pesquisa	6295,43	4896,34	5593,66
Melhor resultado	5016,93	3318,76	**3316,77**
Diferença (%)	**25,48**	**47,53**	**68,65**

Sendo necessárias menos partículas e iterações para implementação do algoritmo na forma desta abordagem, o tempo de processamento acabou por mostrar-se menor do que o

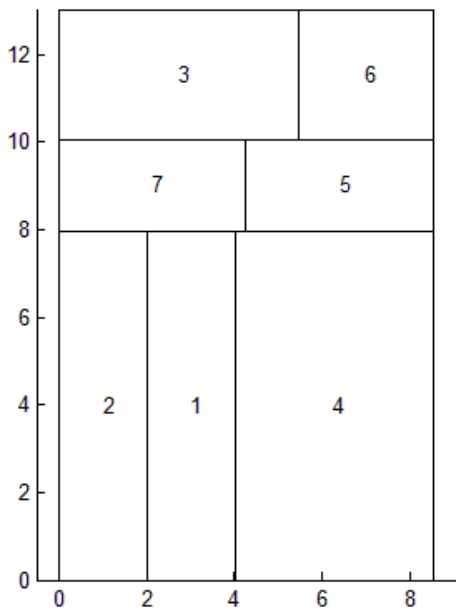
dispensado para a abordagem do FLP com árvores binárias, conforme é possível extrair da tabela 19.

Tabela 19 – tempos de processamento utilizando Enxame de Partículas com matriz de particionamento

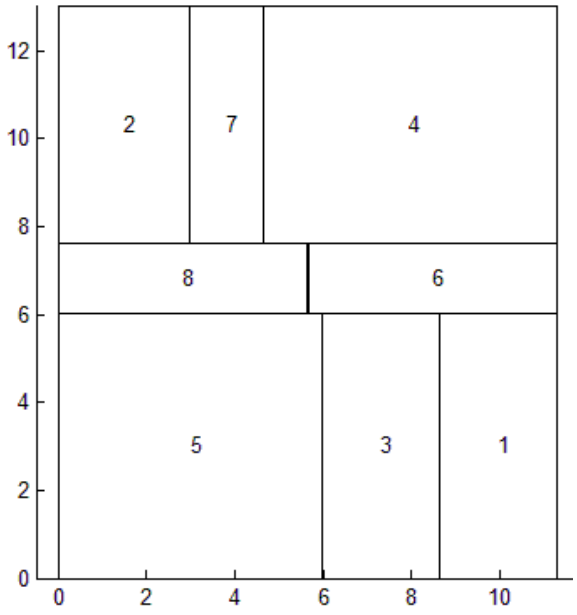
Problema	Tempo (horas)		Diferença (%)
	AS-ST	Esta pesquisa	
o7	0,21	0,0036	-98,28
o8	0,22	0,0217	-90,14
o9	0,25	0,15	-40
vC10	0,28	0,29	-3,57
Ba12	1,41	1,21	-14,18
Ba14	2,38	1,95	-18,07
AB20	4,95	1,38	-72,12
SC30	6,54	1,76	-73,09
SC35	23,33	12,61	-45,95
Du62	24	24	0

Desta vez os tempos de processamento dispensados pelo algoritmo são menores que os apresentados no trabalho de Komarudin e Wong (2010). Imprescindível ressaltar, no entanto, que esses autores inseriram, nos conjuntos SC30 e SC35, 17 e 14 facilidades falsas, respectivamente, a fim de ocupar áreas ociosas na planta. Desta forma, é de se esperar considerável diferença de tempo de processamento, conforme se infere da tabela 19. Outro aspecto a ser observado é que, no MATLAB, a codificação do algoritmo com matrizes exigiu menos tempo de processamento que a abordagem utilizando árvores binárias.

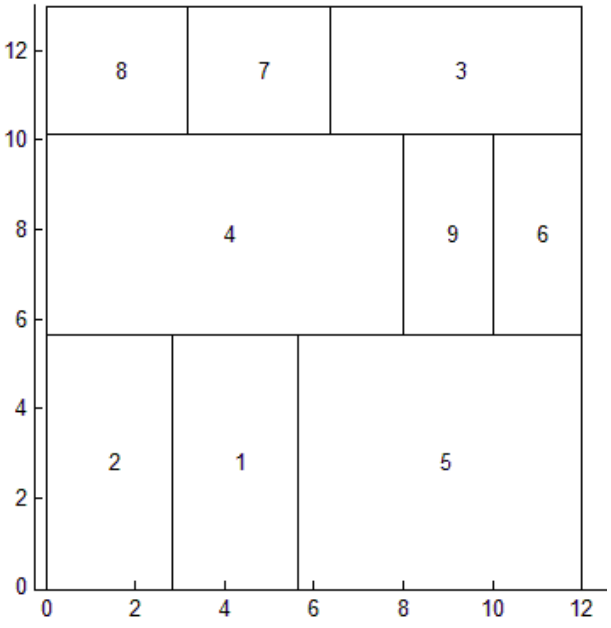
Às figuras 26.a a 26.j são apresentados os melhores resultados obtidos pela proposição de resolução dos FLPs discutida neste capítulo. A seguir, serão apresentadas as conclusões obtidas neste trabalho.


3
6
7
5
2
1
4
0
2
4
6
8
0
2
4
6
8
10
12

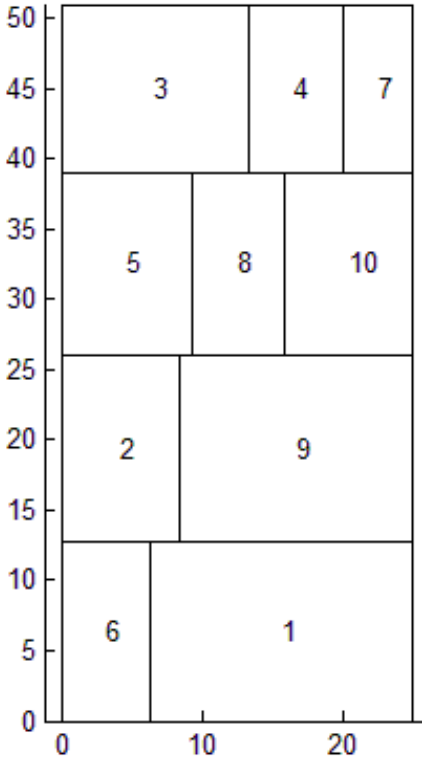
(a)


2
7
4
8
6
5
3
1
0
2
4
6
8
10
12

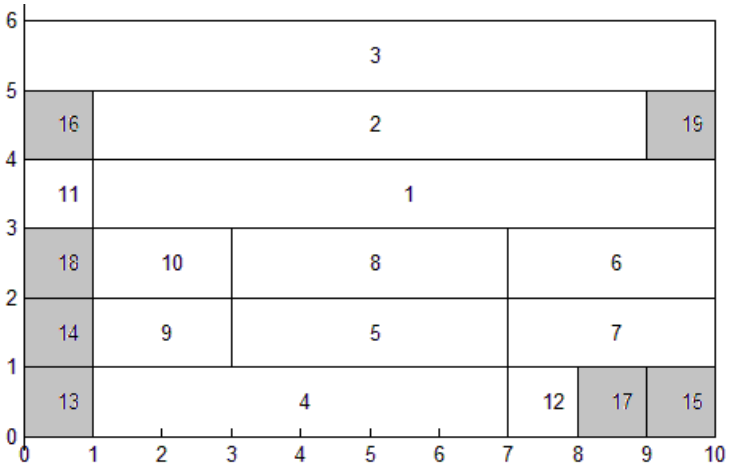
(b)


8
7
3
4
9
6
2
1
5
0
2
4
6
8
10
12

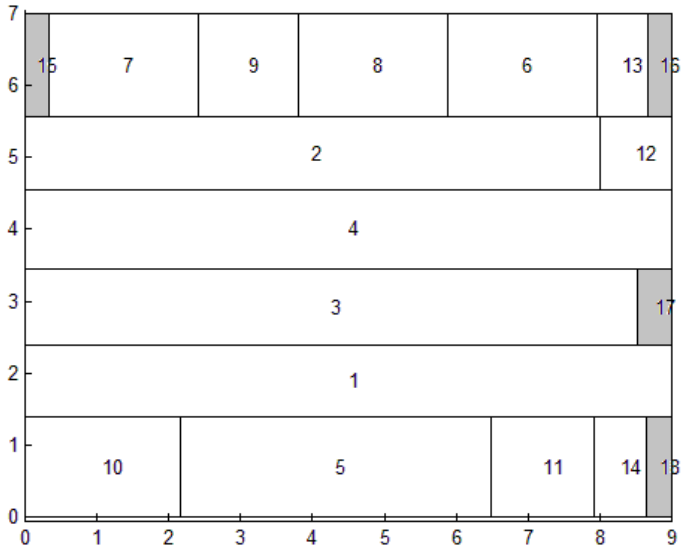
(c)


3
4
7
5
8
10
2
9
6
1
0
5
10
15
20
25
30
35
40
45
50
0
10
20

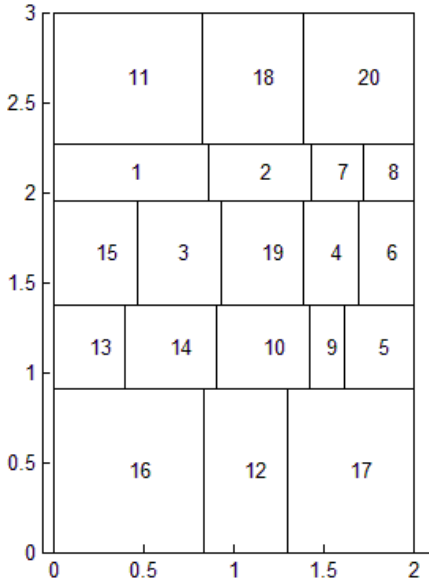
(d)


3
16
2
19
11
1
18
10
8
6
14
9
5
7
13
4
12
17
15
0 1 2 3 4 5 6 7 8 9 10
0 1 2 3 4 5 6

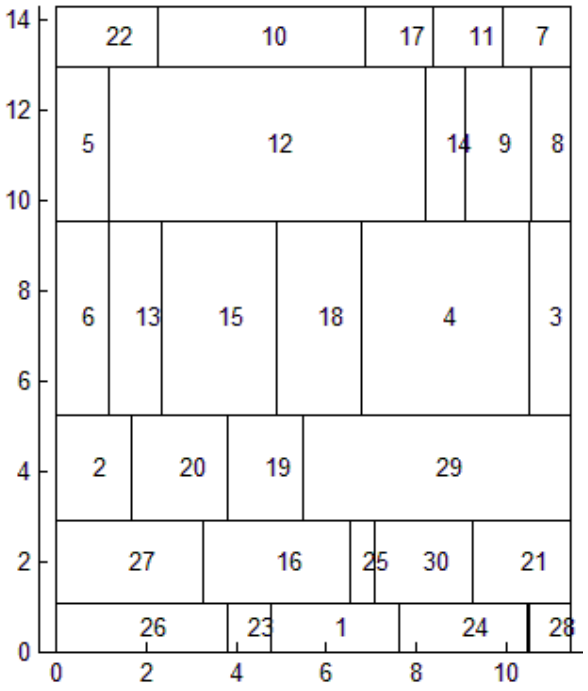
(e)


15
7
9
8
6
13
15
2
12
4
3
17
1
10
5
11
14
13
0 1 2 3 4 5 6 7 8 9
0 1 2 3 4 5 6 7

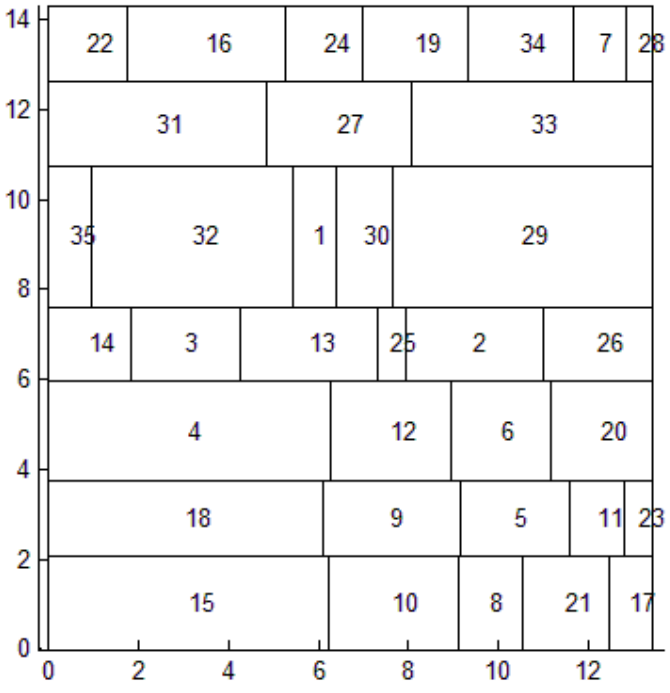
(f)


11
18
20
1
2
7
8
15
3
19
4
6
13
14
10
9
5
16
12
17
0 0.5 1 1.5 2
0 0.5 1 1.5 2 2.5 3

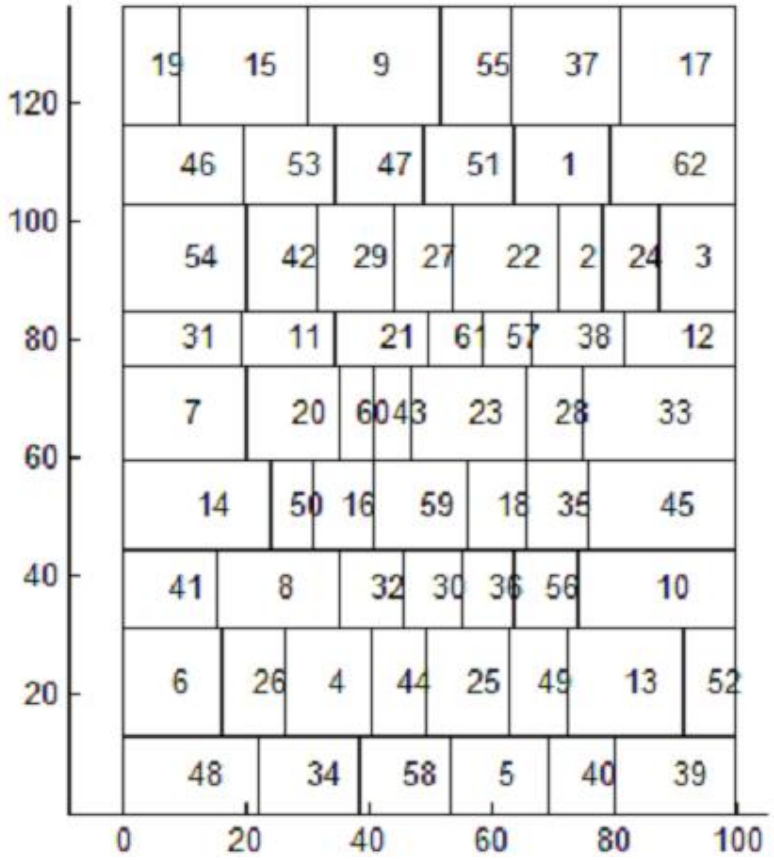
(g)


22
10
17
11
7
5
12
14
9
8
6
13
15
18
4
3
2
20
19
29
27
16
25
30
21
26
23
1
24
28
0
2
4
6
8
10
12
14

(h)


22
16
24
19
34
7
28
31
27
33
35
32
1
30
29
14
3
13
25
2
26
4
12
6
20
18
9
5
11
23
15
10
8
21
17
0
2
4
6
8
10
12
14

(i)



(j)

Figura 26: melhores *layouts* obtidos por matriz de particionamento – conjuntos o7 (a), o8 (b), o9 (c), vC10 (d), Ba12 (e), Ba14 (f), AB20 (g), SC30 (h), SC35 (i) e Du62 (j).

CONCLUSÃO

Esta pesquisa investigou a modelagem do problema de *layout* de facilidades por árvores binárias e matriz de particionamento, utilizando um algoritmo de duplo estágio baseado no enxame de partículas. Foram identificados os *benchmarks* no que se refere aos problemas de *layout* do tipo restrito, com áreas de facilidades desiguais, disponíveis na literatura científica. As restrições consideradas foram a razão de aspecto, as dimensões da planta e a área das facilidades.

O enxame de partículas de duplo estágio com árvores binárias revelou-se competitivo na resolução de problemas de até dez facilidades, apresentando uma melhoria a partir do *layout* inicial, gerado aleatoriamente, na ordem de 12,28% a 45,01%. Em comparação com os *benchmarks* da literatura científica, apresentou desempenho até 7,35% inferior, para estes problemas. Merece destaque a solução obtida para o problema o7, igual à do melhor trabalho até o momento publicado. Entretanto, para o problema de 12 instâncias apresentou resultados insatisfatórios, enquanto para as maiores instâncias revelou-se inviável, pois não apresentou resultados factíveis, que não violassem as restrições de razão de aspecto.

O enxame de partículas de duplo estágio com matrizes de particionamento revelou-se consistente para solução de todas as instâncias estudadas. Foram geradas soluções factíveis para todos os problemas. A melhoria a partir do *layout* inicial revelou-se entre 10,01% e 61,87%. Em comparação com os *benchmarks*, apresentou resultados competitivos para problemas de até 14 instâncias, com diferença de desempenho entre 0,96% e 9,12%. Para os problemas de maiores instâncias o desempenho foi consideravelmente inferior, exceto para o problema de 62 instâncias, com resultado 13,3% maior.

Em relação ao processamento dos algoritmos, para o primeiro houve maior tempo dispensado para sua execução, no entanto os resultados foram comparáveis aos obtidos por outra pesquisa que se utiliza de inteligência de enxame, de autoria de Komarudin e Wong (2010).

Em síntese, o segundo algoritmo proposto apresentou uma *performance* mais razoável,

se consideradas todas as instâncias estudadas. Por outro lado, o primeiro algoritmo apresentou resultados 1,6% a 5,8% melhores que o segundo para os três menores problemas. Quanto ao tempo de processamento do algoritmo, foram apresentados resultados comparáveis com demais heurísticas baseadas em inteligência de enxame.

Como contribuição para a pesquisa acadêmica, apresentou-se uma modelagem do problema de *layout* de facilidades em duplo estágio, uma função para determinação do coeficiente inercial para o enxame de partículas e um método de agrupamento das facilidades através da medida ponderada da distância entre elas.

Para aprofundamento do método apresentado neste trabalho, sugere-se estudos envolvendo as diferentes formas de construção do *layout* com matriz de particionamento propostas por Kim e Kim (1998). Seria desejável investigar também a utilizações do coeficiente inercial variável, modelado com a função quadrática, em outras aplicações da literatura científica. Outra possibilidade de investigação envolve a utilização de problemas-teste para *layouts* sem restrição de dimensões de planta.

A criação e a discussão de novos métodos, presentes na pesquisa básica, são tão importantes e necessárias para o desenvolvimento da ciência quanto sua interpretação e adaptação a problemas reais, característicos da pesquisa aplicada. No entanto, a utilização prática na indústria dos métodos aqui apresentados tem de ser encarada com parcimônia. Embora o fator custo de transporte de material seja preponderante, na configuração do *layout* industrial há que se levar em conta o conhecimento do agente de negócio. Além de fatores subjetivos próprios do ambiente negocial que não foram abordados nesta pesquisa, outro fator não considerado neste trabalho é o custo de mobilização e desmobilização, que envolve as alterações promovidas no *layout* industrial ao longo do tempo.

Ainda assim, é perfeitamente viável a coleta de dados em um ambiente industrial cujo *layout* possa ser comparado aos paradigmas ora abordados. No entanto, como o escopo deste trabalho não envolveu a intervenção experimental em ambiente real, não é segura a recomendação de sua aplicação imediata. Com o intuito de aproximar a presente proposição de uma aplicação prática, seria possível a utilização conjunta destes métodos com a Simulação Computacional, em um futuro trabalho, o que favoreceria a observação e a atuação de um *stakeholder* durante o processo de concepção do *layout*.

REFERÊNCIAS

ANDERBERG, M.R. *Cluster Analysis for Applications*. New York, Academic Press, 1973.

ARENALES, Marcos; ARMENTANO, Vinícius Amaral; MORABITO, Reinaldo; YANASSE, Horácio Hideki. *Pesquisa operacional*. Rio de Janeiro: Campus/Elsevier, 2007.

ALVARENGA, A. G. de; NEGREIROS-GOMES, F. J.; MESTRIA, M. Metaheuristic methods for a class of the facility *layout* problem. *Journal of Intelligent Manufacturing*, Vitória, v. 11, 2000, p. 421-430.

ANJOS, Miguel F.; VANELLI, Anthony. An attractor-repeller approach to floorplanning. *Mathematical Methods of Operations Research*, v. 56, n. 1, ago. 2002, p. 3-27.

ARMOUR, G. C.; BUFFA, E. S. A heuristic algorithm and simulation approach to relative allocation of facilities. *Management Science*, v. 9, n. 2, 1963, p. 294–300.

ASL, A. D.; WONG, K. Y. Solving unequal-area static and dynamic facility layout problems using modified particle swarm optimization. Journal of Intelligent Manufacturing, *in press*, 2015, 1–20.

AZADEH, A; HAGHIGHI, S. Motevali; ASADZADEH, S. M. A novel algorithm for *layout* optimization of injection process with random demands and sequence dependent setup times. *Journal of Manufacturing Systems*, v. 33, fev. 2014, p. 287-302.

AZARBONYAD, Hosein; BABAZADEH, Reza. A Genetic Algorithm for solving Quadratic Assignment Problem(QAP). *Proceedings of 5th International Conference of Iranian Operations Research Society (ICIORS)*, Tabriz, Irã, 2012.

BANKS, Alec; VINCENT, Jonathan; ANYAKOHA, Chukwudi. A review of particle swarm optimization. Part I: background and development. *Natural Computing*, v. 6, p. 467-484.

BAYKASOGLU, A.; DERELI T.; SABUNCU, I. An ant colony algorithm for solving budget constrained and unconstrained dynamic facility *layout* problems. *Omega*, v. 34, n. 4, 2006, p. 385–396.

BAZARAA, M. S. Computerized layout design: A branch and bound aprouch. *AIIE Transactiosn*, v. 7, n. 4, 1975, p. 432-438.

CARVALHO, Danilo Ferreira; BASTOS-FILHO, Carmelo José Albanez. Clan particle swarm optimization. *International Journal of Intelligent Computing and Cybernetics*, v. 2, n. 2, 2009, p. 197-227.

CASTILLO, Ignacio; WESTERLUND, J.; EMET, S.; WESTERLUND, T. Optimization of block layout design problems with unequal areas: a comparison of MILP and MINLP optimization methods. *Computers and Chemical Engeneering,* v. 30, n. 1, 2005, p. 54-69.

CASTILLOS, Ignacio; WESTERLUND, T. An ε-accurate model for optimal unequal-area block layout design. *Computers & Operations Research,* v. 32, n. 3, 2005, p. 429–447.

CASTILLO, Ignacio; SIM, Thaddeus. A Spring-Embedding Approach for the Facility Layout Problem. *The Journal of the Operational Research Society,* v. 5, n. 1, 2004, p. 73-81.

CHEN, Yi-Kuang; LIN, Shih-Wei; CHOU, Shuo-Yan. An efficient two-staged approach for generating block layouts. *Computers and Operations Research,* v. 29, 2002, p. 489-504.

CHIANG, W. C.; KOUVELIS, P. An improved tabu search heuristic for solving facility *layout* design problems. *International Journal of Production Research,* v. 34, n. 9, 1996, p. 2565–2585.

CHWIF, L.; BARRETTO, M. R. Pereira; MOSCATO, L. A. A solution to the facility *layout* problem using simulated annealing. *Computers in Industry,* v. 36, n. 1–2, 1998, p. 125–132.

CLERC, M. The swarm and the queen: towards a deterministic and adaptive particle swarm optimization. *Proceedings of the 1999 Congress Evolutionary Computation*, IEEE Press, 1999, p. 1951–1957.

CLERC, M.; KENNEDY, James. The Particle Swarm-Explosion, Stability, and Convergence in a Multidimensional Somplex space. *IEEE Transactions on Evolutionary Computation,* v. 6, 2002, p. 58-73.

DORIGO, Marco. *Optimization, Learning and Natural Algorithms*. Ph.D. Thesis. Dipartimento di Elettronica, Politecnico di Milano, Milão, 1992.

DREZNER, Z. DISCON: A new method for the layout problem. *Operations Research,* v. 25, n. 6, 1980, p. 1375-1384.

DREZNER, Z. A heuristic procedure for the *layout* of a large number of facilities. *International Journal of Management Science*, v. 33, n. 7, 1987, p. 907–915.

DRIRA, Amine; PIERREVAL, Henri; HAJRI-GABOUJ, Sonia. Facility *layout* problems: a survey. *Annual Reviews in Control,* v. 31, nov. 2007, p. 255-267.

DUNKER, T.; RADONSB, G.; WESTKÄMPERA, E. Combining evolutionary computation and dynamic programming for solving a dynamic facility *layout* problem. *European Journal of Operational Research,* v. 165, n. 1, 2005, p. 55–69.

EBERHART, Russell C.; SHI, Yuhui. Comparing inertia weights and constriction factors in particle swarm optimization. *Proceedings of the 2000 Congress Evolutionary Computation*, IEEE Press, 2000, p. 84–88.

ENGELBRECHT, Andries P. *Fundamentals of Computational Swarm Intelligence.* West Sussex: John Willey & Sons, 2005. 599 p.

ESLAMI, Mahdiyeh. *A Survey of the State of the Art in Particle Swarm Optimization. Research Journal of Sciences, Engineering and Technology*, v. 4, n. 9, maio 2012, p. 1181-1197.

FARAHANI, Reza Zanjirani. STEADIESEIFI, Maryam. ASGARI, Nasrin Asgari. Multiple criteria facility location problems: A survey. *Applied Mathematical Modelling*, v. 34, 2010, p. 1689–1709.

FURTADO, João Carlos; LORENA, Luís Antonio Nogueira. Otimização de leiaute usando busca tabu. *Gestão & Produção*, v. 4, n. 1, abr. 1997b, p. 88-108.

_____. Otimização em Problemas de Leiaute. In: *XX Congresso Nacional de Matemática Aplicada e Computacional e 2ª Oficina Nacional de PCE*, 1997, Gramado, RS. Anais da 2ª Oficina Nacional de Problemas de Corte e Empacotamento, 1997a, p. 129-146.

GAREY, Michael. R.; JOHNSON, David .S. *Computers and intractability: a guide to the theory of Np-completeness*. New York: W.H. Freeman and Company, 1979.

GAU, K. Y.; MELLER, R. D. An interative facility layout algorithm. *International Journal of Production Research*, v. 37, n. 16, p. 3739- 3758.

GLOVER, Fred W. Tabu Search – part I. *ORSA Journal on Computing*, v. 1, 1989, p. 190-206.

_____. Tabu Search – part II. *ORSA Journal on Computing*, v. 2, 1989, p. 4-32.

HEPPNER, F.; GRENANDER, U. A stochastic nonlinear model for coordinated bird flocks. In: KRESNER, Saul. *The Ubiquity of Chaos.* New York: American Association for the Advancement of Science, 1990, p. 233-238.

HERAGU, Sunderesh S.; KUSIAK, Andrew. Efficient models for the facility layout problem. *European Journal of Operational Research*, v. 53, 1991, p. 1-13.

HERAGU, Sunderesh S. *Facilities design*. Boston: BWS, 1997.

HIGHAM, Desmond J.; HIGHAM, Nicholas J. *Matlab guide*. Philadelphia: SIAM, 2000. 283 p.

HOLLAND, John H. *Adaptation in natural and artificial systems*. University of Michigan Press: Ann Harbor, 1975, 183 p.

JANKOVITS, Ibolya; LUO, Chaomin; ANJOS, Miguel F.; VANELLI, Anthony. A convex optimisation framework for the unequal-areas facility layout problem. *European Journal of Operational Research*, v. 214, 2011, p. 199-215.

JIANG, S.; ONG, S. K.; NEE, Y. C. An AR-based hybrid approach for facility *layout* planning and evaluation for existing shop floors. *The International Journal of Advanced Manufacturing Technology*, v. 72, 2014, p. 457-473.

JIAO, B.; LIAN, Z.; GU, X. A dynamic inertia weight particle swarm optmization algorithm. *Chaos, Solitons & Fractals*, v. 37, 2008, p. 698-705.

KENNEDY, James; EBERHART, Russell C. Particle Swarm Optimization. In: *The 1995 IEEE International Conference on Neural Networks*, Perth,Australia, 1995, p. 1942–1948.

KENNEDY, James; EBERHART, Russell C.; Shi, Yuhui. *Swarm Intelligence*. San Francisco: Morgan Kaufmann/ Academic Press, 2001.

KIA, R.; KHAKSAR-HAGHANI, F.; JAVADIAN, N.; TAVAKKOLI-MOGHADDAM, R. Solving a multi-floor *layout* design model of a dynamic cellular manufacturing system by an efficient genetic algorithm. *Journal of Manufacturing Systems*, v. 33, jan. 2014, p. 218-232.

KIM, J. G.; KIM, Y. D. A space partitioning method for facility layout problems with shape constraints. *IIE Transactions*, v. 30, n. 10, 1998 p. 947–957.

KIM, J. G.; KIM, Y. D. A branch and bound algorithm for locating input and output points of departments on the block *layout. Journal of the operational research society*, v. 50, n. 5, 1999, p. 517–525.

KIRKPATRICK, S.; GELLAT, C. D.; VECHI, M. P. Optimization by Simulated Annealing. *Science, New Series*, v. 220, n. 4598, maio 1983, p. 671-680.

KOMARUDIN; WONG, Kuan Yew. Applying ant system for solving unequal área facility layout problems. *European Journal of Operational Research*, v. 202, n. 3, p. 730-746.

KOOPMANS, Tjalling C.; BECKMAN, Martin. Assignment Problems and the Location of Economic Activities. *Econometrica*, n. 25, 1957, p. 53-76.

KOTHARI, Ravi; GHOSH, Diptesh. A scatter search algorithm for the single row facility *layout* problem. *Journal of Heuristics*, v. 20, n. 2, 2014b, p. 125-142.

_____. An efficient genetic algorithm for single row facility *layout*. *Optimization Letters*,

v. 8, n. 2, fev. 2014a, p. 679-690.

KOUVELIS, Panagiotis; KIM, Michael W. Unidirectional loop network *layout* problem in automated manufacturing systems. *Operations Research,* n. 40, 1992, p. 533–550.

KULTUREL-KONAK, S.; KONAK. A. A new relaxed flexible bay structure representation and particle swarm optimization for the unequal area facility layout problem. *Engineering Optimization*, v. 43, n. 12, p. 1263–1287

LEE, R.; MOORE, J. M. CORELAP-computerized relationship *layout* planning. *The Journal of Industrial Engineering*, 1967, v. 18, p. 195–200.

LEE, Y. H.; LEE, M. H. A shape-based block *layout* approach to facility *layout* problems using hybrid genetic algorithm. *Computers & Industrial Engineering*, v. 42, 2002, p. 237–248.

LIEN, Li-Chuan; CHENG, Min-Yuan. Particle bee algorithm for tower crane *layout* with material quantify supply and demand optimization. *Automation in Construction,* v. 45, set. 2014, p. 25-32.

LIU, Q.; MELLER, R. D. A sequence-pair representation and mip-model-based heuristic for the facility layout problem with rectangular departments. *IIE Transactions,* v. 39, n. 4, 2007, p. 377-394.

MCKENDALL, A. R.; SHANG, J.; KUPPUSAMY, S. Simulated annealing heuristics for the dynamic facility *layout* problem. *Computers & Operations Research*, v. 33, n. 8, 2006, p. 2431–2444.

MELLER, Russell D.; NARAYANAN, Venkat;, VANCE, Pamela H. Optimal facility *layout* design. *Operations Research Letters*, v. 23, n. 3–5, out. 1998, p. 117–127.

MENDES, R.; KENNEY, James; NEVES, J. The fully informed particle swarm: simpler, maybe better. *IEEE Transactions on Evolutionary Computation,* v. 8, 2004, p. 204-210.

MORABITO, Reinaldo. Pesquisa Operacional. In: *Introdução à Engenharia de Produção.* Rio de Janeiro: Elsevier, 2008. p. 156-181.

MÜLLER, Viviane; FURTADO, João Carlos; NEIS, Jaime Furtado; CROSSETTI, Geraldo Lopes. Otimização do problema de *layout* de facilidades através da técnica enxame de partículas utilizando a modelagem attractor-repeller. *Anais do XXVI Encontro Nacional de Engenharia de Produção (ENEGEP)*, Fortaleza, 2006.

MÜLLER, Viviane. *Otimização de layouts industriais através do método enxame de partículas*. Dissertação (Programa de Pós-Graduação em Sistemas e Processos Industriais – Mestrado) – Universidade de Santa Cruz do Sul, Santa Cruz do Sul, 2007. 79 p.

NEGHABI, Hossein; ESHGHI, Kourosh; SALMANI, Mohammad Hassan. A new model for robust facility *layout* problem. *Information Sciences*, n. 278, abr. 2014, p. 498-509.

NEMATIAN, Javad. A robust single row facility *layout* problem with fuzzy random variables. *The International Journal of Advanced Manufacturing Technology*, v. 72, n. 1-4, fev. 2014, p. 255-267.

NORMAN, Bryan A.; SMITH, Alice E. Random Keys Genetic Algorithm with Adaptive Penalty Function for Optimization of Constrained Facility Layout Problems. *IEEE International Conference on Evolutionary Computation*, abr. 1997, p. 407-411.

ÖNUT, Semih; TUZKAYA, Umut R.; DOĞAC, Bilgehan. A particle swarm optimization algorithm for the multiple-level warehouse *layout* design problem. *Computers & Industrial Engineering*, v. 54, n. 4, maio 2008, p. 783-799.

POLI, R.; KENNEDY, J.; BLACKWELL, T. Particle Swarm Optimisation: an overview. *Swarm Intelligence Journal*, v. 1, n. 1, 2007, p. 33–57.

RAHBARI, Omid; VAFAEIPOUR, Majid; FAZELPOUR, Farivar; FEIDT, Michel; ROSEN, Marc A. Towards realistic designs of wind farm *layout*s: Application of a novel placement selector approach. *Energy Conversion and Management*, v. 81, mar. 2014, p. 242-254.

RODRIGUES, Eugénio; GASPAR, Adélio Rodrigues; GOMES, Álvaro. An approach to the multi-level space allocation problem in architecture using a hybrid evolutionary technique. *Automation in Construction*, v. 35, jul. 2013, p. 482-498.

ROSA, G. P.; CRACO, T.; REIS, Z. C.; NODARI, C. H. A reorganização do *layout* como estratégia de otimização da produção. GEPROS. *Gestão da Produção, Operações e Sistemas*, Bauru, ano 9, n. 2, jun. 2014, p. 139-154.

ROSENBLATT, M. J. The dynamics of plant layout. *Management Science*, v. 21, n.1, 1986, p. 76-86.

SAMARGHANDI, Hamed; TAABAYAN, Pouria. JAHANTIGH, Farzad Firouzi. A particle swarm optmization for the single row facility *layout* problem. *Computers & Industrial Engeneering*, v. 58, n. 4, maio 2010, p. 529-553.

SANTOS, Antonio Raimundo dos. *Metodologia científica: a construção do conhecimento.* Rio de Janeiro: DP&A, 2000. 3. ed. 139 p.

SARAVANAN, M.; KUMAR, S. Ganesh. Different approaches for the loop *layout* problems: a review. *The International Journal of Advanced Manufacturing Technology*, v. 69, n. 9-12, dez. 2013, p. 2513-2529.

SCHOLZ, D.; PETRICK, A.; DOMSCHKLE, W. A slicing tree and tabu search based heuristic for the unequal area facility layout problem. *European Journal of Operational Research*, v. 197, n. 1, 2009, p. 166-178.

SERAPIÃO, Adriane Beatriz de Souza. Fundamentos de otimização por inteligência de enxames: uma visão geral. *Controle e Automação*, v. 20, n. 3, set. 2009, p. 271-304.

SHAYAN, E.; CHITTILAPPILLY, A. Genetic algorithm for facilities layout problems based on slicing tree structure. *International Journal of Production Research*, v. 32, n. 3, 2004, p. 4055-4067.

SHERALI, H. D.; FRATICELLI, B. M.; MELLER, R. D. Enhanced Model formulations for optimal facility layout. *Operations Research*, v. 51, n. 4, p. 629-644.

SHI, Yuhui; EBERHART, Russell C. A modified particle swarm optimizer. *Proceedings of the IEEE International Conference on Evolutionary Computation. Piscataway*, IEEE Press, 1998, p. 69-73.

SOLIMANPUR, Maghsud; V, RATP.; SHANKAR, R. An ant algorithm for the single row *layout* problem in flexible manufacturing systems. *Computers & Operations Research*, v. 32, n. 3, 2005, p. 583–598.

SOLIMANPUR, Maghsud; JAFARI, Amir. Optimal solution for the two-dimensional facility *layout* problem using a branch-and-bound algorithm. *Computers and Industrial Engineering*, v. 55, 2008, p. 606-619.

SU, Yingsheng; FAN, Hua. An Improved Fuzzy Particle Swarm Optimization for Numerical Optimization. *Dynamics of Continuous, Discrete and Impulsive Systems, Series B: Applications & Algorithms*, v. 20, 2013, p. 173-188.

TAM, Kar Yan. Genetic Algorithms, function optmization and facility layout design. *European Journal of Operational Research*, v. 63, 1992, p. 322-346.

TATE, D. M.; SMITH, A. E. A genetic approach to the quadratic assignment problem. *Computers and Operations Research*, v. 32, 1995, p. 73-83.

TOMPKINS James A.; WHITE, John A.; BOZER, Yavuz A.; TANCHOCO, José M. *Facilities planning.* New York: Wiley, 2010.

TOMPKINS, James A.; REED Jr., Ruddell. An applied model for the facilities design problem. *International Journal of Production Research*, v. 14, n. 5, 1976, p. 583-595.

VAN CAMP, D. J.; CARTER, M. W.; VANELLI, A. A nonlinear optmization approach for solving facility layout problems. *European Journal of Operational Research*, v. 57, n. 2, 1992, p. 174-189.

VALDEZ, F.; MELIN, P.; CASTILLO, O. An improved evolutionary method with fuzzy logic for combinig Particle Swarm Optimization and Genetic Algorithms. *Aplied Soft Computing*, v. 11, 2010, p. 2625-2632.

WANG, Ming-Jaan; HU, Michael H,; KU, Meei-Yuh. A solution to the unequal area facilities layout problem by genetic algorithm. *Computers in Industry*, v. 56, 2005, p. 207-220.

WONG, Kuan Yew; KOMARUDIN. Solving facility layout problems using flexible baystructure representation and ant system algorithm. *Expert Systems with Applications*, v. 37, n. 7, p. 5523–5527.

XIE, Wei; SAHINIDIS, Nikolaos V. A branch-and-bound algorithm for the continuous facility *layout* problem. *Computers and Chemical Engineering*. v. 32, 2008, p. 1016-1028.

YANG. T; PETERS, B. A.; TU, M. *Layout* design for flexible manufacturing systems considering single-loop directional flow patterns. *European Journal of Operational Research*, v. 164, n. 2, 2005, p. 440–455.

ANEXO: Conjuntos de dados para teste

o7 – Custos de transporte de material entre as facilidades

	1	2	3	4	5	6	7
1	–	0	0	5	0	0	1
2	0	–	0	3	0	0	1
3	0	0	–	2	0	0	1
4	0	0	0	–	4	4	0
5	0	0	0	0	–	0	2
6	0	0	0	0	0	–	1
7	0	0	0	0	0	0	–

o7 – Área das facilidades

1	2	3	4	5	6	7
16	16	16	36	9	9	9

o8 – Custos de transporte de material entre as facilidades

	1	2	3	4	5	6	7	8
1	–	0	0	5	5	0	0	1
2	0	–	0	3	3	0	0	1
3	0	0	–	2	2	0	0	1
4	0	0	0	–	0	4	4	0
5	0	0	0	0	–	3	0	4
6	0	0	0	0	0	–	0	2
7	0	0	0	0	0	0	–	1
8	0	0	0	0	0	0	0	–

o8 – Área das facilidades

1	2	3	4	5	6	7	8
16	16	16	36	36	9	9	9

o9 – Custos de transporte de material entre as facilidades

	1	2	3	4	5	6	7	8	9
1	–	0	0	5	5	0	0	0	1
2	0	–	0	3	3	0	0	0	1
3	0	0	–	2	2	0	0	0	1
4	0	0	0	–	0	4	4	0	0
5	0	0	0	0	–	3	0	0	4
6	0	0	0	0	0	–	0	0	2
7	0	0	0	0	0	0	–	0	1
8	0	0	0	0	0	0	0	–	0
9	0	0	0	0	0	0	0	0	–

o9 – Área das facilidades

1	2	3	4	5	6	7	8	9
16	16	16	36	36	9	9	9	9

vC10 – Custos de transporte de material entre as facilidades

	1	2	3	4	5	6	7	8	9	10
1	–	0	0	0	0	218	0	0	0	0
2	0	–	0	0	0	148	0	0	296	0
3	0	0	–	28	70	0	0	0	0	0
4	0	0	0	–	0	28	70	140	0	0
5	0	0	0	0	–	0	0	210	0	0
6	0	0	0	0	0	–	0	0	0	0
7	0	0	0	0	0	0	–	0	0	28
8	0	0	0	0	0	0	0	–	0	888
9	0	0	0	0	0	0	0	0	–	59
10	0	0	0	0	0	0	0	0	0	–

vC10 – Área das facilidades

1	238
2	112
3	160
4	80
5	120
6	80
7	60
8	85
9	221
10	119

Ba12 – Custos de transporte de material entre as facilidades

	1	2	3	4	5	6	7	8	9	10	11	12
1	–	288	180	54	72	180	27	72	36	0	0	9
2	0	–	240	54	72	24	48	160	16	64	8	16
3	0	0	–	120	80	0	60	120	60	0	0	30
4	0	0	0	–	72	18	18	48	24	48	12	0
5	0	0	0	0	–	12	12	64	16	16	4	8
6	0	0	0	0	0	–	18	24	6	12	3	3
7	0	0	0	0	0	0	–	0	6	6	3	6
8	0	0	0	0	0	0	0	–	16	16	16	4
9	0	0	0	0	0	0	0	0	–	4	4	2
10	0	0	0	0	0	0	0	0	0	–	2	2
11	0	0	0	0	0	0	0	0	0	0	–	2
12	0	0	0	0	0	0	0	0	0	0	0	–

Ba12 – Área das facilidades

1	9
2	8
3	10
4	6
5	4
6	3
7	3
8	4
9	2
10	2
11	1
12	1

Ba14 – Custos de transporte de material entre as facilidades

	1	2	3	4	5	6	7	8	9	10	11	12	13	14
1	–	72	162	90	108	27	0	0	18	27	18	0	0	0
2	0	–	72	80	0	48	0	48	32	0	16	8	0	0
3	0	0	–	45	54	27	27	27	0	27	0	9	18	0
4	0	0	0	–	30	0	30	30	20	0	20	10	10	0
5	0	0	0	0	–	18	0	18	12	18	24	0	0	0
6	0	0	0	0	0	–	9	9	0	0	6	6	6	0
7	0	0	0	0	0	0	–	9	12	9	6	3	0	0
8	0	0	0	0	0	0	0	–	6	9	0	3	0	0
9	0	0	0	0	0	0	0	0	–	6	4	6	2	0
10	0	0	0	0	0	0	0	0	0	–	6	3	6	0
11	0	0	0	0	0	0	0	0	0	0	–	2	0	0
12	0	0	0	0	0	0	0	0	0	0	0	–	4	0
13	0	0	0	0	0	0	0	0	0	0	0	0	–	0
14	0	0	0	0	0	0	0	0	0	0	0	0	0	–

Ba14 – Área das facilidades

1	9
2	8
3	9
4	10
5	6
6	3
7	3
8	3
9	2
10	3
11	2
12	1
13	1
14	1

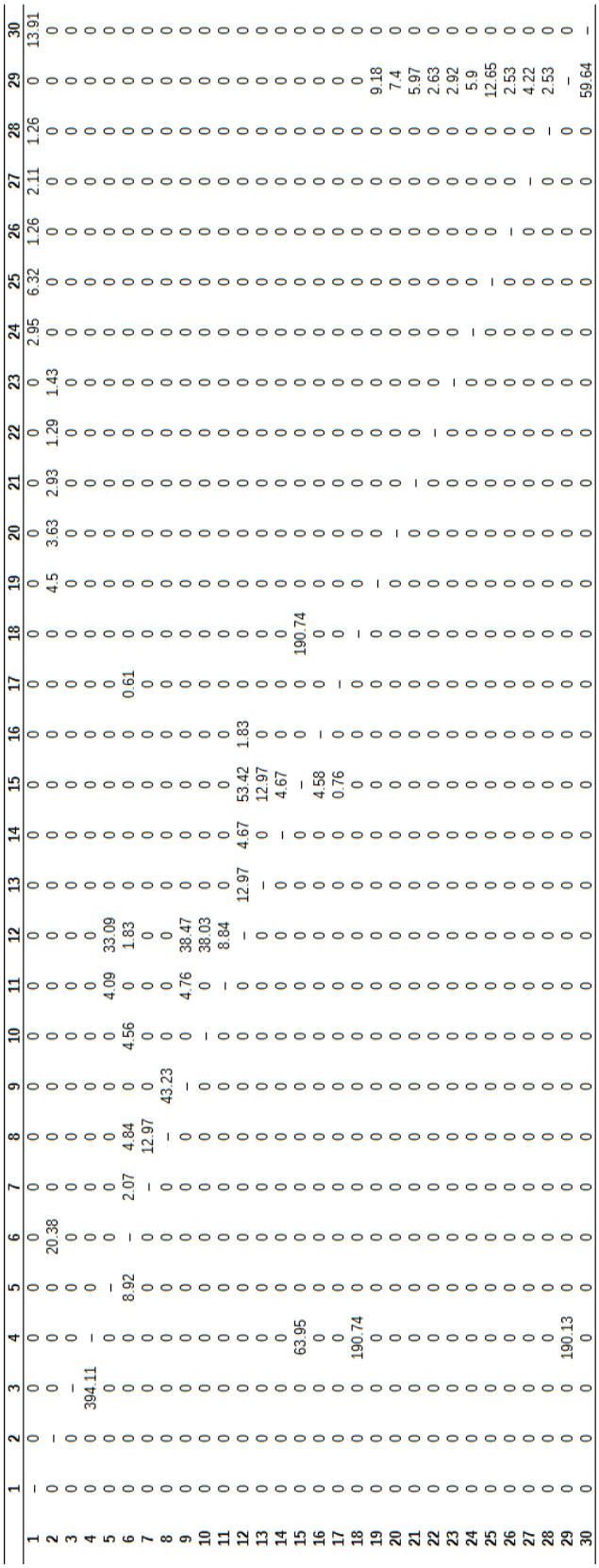
AB20 – Custos de transporte de material entre as facilidades

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	–	18	12	0	0	0	0	0	0	10.4	11.2	0	0	12	0	0	0	0	0	0
2	18	–	9.6	245	7.8	0	140	0	12	13.5	0	0	0	0	0	0	0	0	69	0
3	12	9.6	–	0	0	22.1	0	0	31.5	39	0	0	0	131	0	0	0	0	137	0
4	0	245	0	–	10.8	57	75	0	23.4	0	0	14	0	0	0	0	0	15	158	0
5	0	7.8	0	10.8	–	0	22.5	13.5	0	15.6	0	0	0	0	13.5	0	0	0	0	0
6	0	0	22.1	57	0	–	61.5	0	0	0	0	4.5	0	0	0	0	0	10.5	0	0
7	0	140	0	75	22.5	61.5	–	240	0	18.7	0	0	0	9.6	0	0	0	16.5	0	37.5
8	0	0	0	0	13.5	0	240	–	0	0	0	0	6	0	0	0	0	0	75	335
9	0	12	31.5	23.4	0	0	0	0	–	0	0	0	0	75	0	0	75	0	0	0
10	10.4	13.5	39	0	15.6	0	18.7	0	0	–	3.6	120	0	186	19.2	0	0	0	52.5	0
11	11.2	0	0	0	0	0	0	0	0	3.6	–	22.5	0	30	9.6	225	0	0	0	0
12	0	0	0	14	0	4.5	0	0	0	120	22.5	–	0	0	16.5	0	150	0	84	0
13	0	0	0	0	0	0	0	6	0	0	0	0	–	80	10.4	60	0	0	0	0
14	12	0	131	0	0	0	9.6	0	75	186	30	0	80	–	97.5	0	0	9	0	0
15	0	0	0	0	13.5	0	0	0	0	19.2	9.6	16.5	10.4	97.5	–	0	52.5	0	0	0
16	0	0	0	0	0	0	0	0	0	0	0	0	60	0	0	–	120	0	0	0
17	0	0	0	0	0	0	0	0	75	0	0	150	0	0	52.5	120	–	0	75	0
18	0	0	0	15	0	10.5	16.5	0	0	0	0	0	0	9	0	0	0	–	46.5	0
19	0	69	137	158	0	0	0	75	0	52.5	0	84	0	0	0	0	75	46.5	–	0
20	0	0	0	0	0	0	37.5	335	0	0	0	0	0	0	0	0	0	0	0	–

AB20 – Área das facilidades

1	2	3	4	5	6	7	8	9	10
0.27	0.18	0.27	0.18	0.18	0.18	0.09	0.09	0.09	0.24
11	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**
0.6	0.42	0.18	0.24	0.27	0.75	0.64	0.41	0.27	0.45

SC30 – Custos de transporte de material entre as facilidades



	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
1	–	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2.95	6.32	1.26	2.11	1.26	0	13.91
2	0	–	0	0	0	20.38	0	0	0	0	0	0	0	0	0	0	0	0	4.5	3.63	2.93	1.29	1.43	0	0	0	0	0	0	0
3	0	0	–	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0	0	394.11	–	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	0	0	0	0	–	0	0	0	0	0	4.09	33.09	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6	0	0	0	0	8.92	–	2.07	4.84	0	4.56	0	1.83	0	0	0	0	0.61	0	0	0	0	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	–	12.97	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	–	43.23	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	–	0	4.76	38.47	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0	0	–	0	38.03	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0	0	0	0	–	8.84	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
12	0	0	0	0	0	0	0	0	0	0	0	–	12.97	4.67	53.42	1.83	0	0	0	0	0	0	0	0	0	0	0	0	0	0
13	0	0	0	0	0	0	0	0	0	0	0	0	–	0	12.97	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
14	0	0	0	0	0	0	0	0	0	0	0	0	0	–	4.67	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
15	0	0	0	0	0	0	0	0	0	0	0	0	0	0	–	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4.58	–	0	0	0	0	0	0	0	0	0	0	0	0	0	0
17	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.76	0	–	0	0	0	0	0	0	0	0	0	0	0	0	0
18	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	–	0	0	0	0	0	0	0	0	0	0	0	0
19	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	–	0	0	0	0	0	0	0	0	0	0	0
20	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	–	0	0	0	0	0	0	0	0	0	0
21	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	–	0	0	0	0	0	0	0	0	0
22	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	–	0	0	0	0	0	0	0	0
23	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	–	0	0	0	0	0	0	0
24	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	–	0	0	0	0	0	0
25	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	–	0	0	0	0	0
26	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	–	0	0	0	0
27	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	–	0	0	0
28	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	–	0	0
29	0	0	0	190.13	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	–	0
30	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	–

SC30 – Área das facilidades

1	3
2	4
3	4
4	16
5	4
6	5
7	2
8	3
9	5
10	6
11	2
12	24
13	5
14	3
15	11
16	6
17	2
18	8
19	4
20	5
21	4
22	3
23	1
24	3
25	1
26	4
27	6
28	1
29	14
30	4

SC35 – Custos de transporte de material entre as facilidades

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35
1	–	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2.95	6.32	1.26	2.11	1.26	0	13.91	0	0	0	0	0
2	0	–	0	0	0	20.38	0	0	0	0	0	0	0	0	0	0	0	0	4.5	3.63	2.93	1.29	1.43	0	0	0	0	0	0	0	0	0	0	0	0
3	0	0	–	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0	0	225.65	–	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	0	0	0	0	–	0	0	0	0	0	4.09	33.09	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	72.89	0	0	0
6	0	0	0	0	8.92	–	2.07	4.84	0	4.56	0	1.83	0	0	0	0	0.61	0	0	0	0	0	0	0	0	0	0	0	0	0	3.93	0	0	0	0	0
7	0	0	0	0	0	0	–	12.97	43.23	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	–	–	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	–	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0	0	–	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0	0	0	0	–	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
12	0	0	0	0	0	0	0	0	0	0	0	–	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
13	0	0	0	0	0	0	0	0	0	0	0	0	–	0	12.97	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
14	0	0	0	0	0	0	0	0	0	0	0	0	0	–	4.67	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
15	0	0	0	0	0	0	0	0	0	0	0	0	0	0	–	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4.58	–	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
17	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.76	0	–	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
18	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	–	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
19	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	–	0	0	0	0	0	0	0	0	0	0	0	0	0	0	9.18	0
20	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	–	0	0	0	0	0	0	0	0	0	0	0	0	0	7.4	0
21	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	–	0	0	0	0	0	0	0	0	0	0	0	0	0	0
22	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	–	0	0	0	0	0	0	0	0	0	0	0	0	0
23	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	–	0	0	0	0	0	0	0	0	0	0	0	0
24	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	–	0	0	0	0	0	0	0	0	0	0	0
25	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	–	0	0	0	0	0	0	0	0	0	0
26	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	–	0	0	0	0	0	0	0	0	0
27	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	–	0	0	0	0	0	0	0	0
28	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	–	0	0	0	0	0	0	0
29	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	–	0	0	0	0	0	0
30	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	–	0	0	0	0	0
31	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	–	0	0	0	0
32	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	–	0	0	0
33	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	–	0	0
34	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	–	0
35	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	–

SC35 – Área das facilidades

1	3
2	4
3	4
4	16
5	4
6	5
7	2
8	3
9	5
10	6
11	2
12	24
13	5
14	3
15	11
16	6
17	2
18	8
19	4
20	5
21	4
22	3
23	1
24	3
25	1
26	4
27	6
28	1
29	14
30	4

Du62– Custos de transporte de material entre as facilidades

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
1	–	51	93	60	42	0	13	6	45	0	63	96	0	14	0	32	0	43	0	52	45	0	94	55	30	0	59	72	29	0	0
2	0	–	44	0	88	22	32	16	25	37	12	46	23	0	99	10	86	0	18	23	16	0	74	93	8	7	57	69	30	41	21
3	0	0	–	33	35	6	8	45	13	14	71	94	17	92	54	56	67	62	63	29	93	21	55	85	84	73	37	24	9	85	7
4	0	0	0	–	92	90	11	56	11	0	18	18	98	0	0	83	0	56	0	81	23	71	84	0	27	0	5	0	61	91	0
5	0	0	0	0	–	83	13	31	59	61	53	54	18	0	94	59	0	46	56	19	65	49	0	51	0	55	0	23	65	42	0
6	0	0	0	0	0	–	0	42	0	72	20	63	84	0	0	35	0	0	0	76	43	7	6	0	0	0	32	29	87	0	0
7	0	0	0	0	0	0	–	0	83	60	41	62	94	11	0	62	0	0	0	42	83	71	0	0	26	0	26	14	91	0	0
8	0	0	0	0	0	0	0	–	18	65	8	7	37	44	52	97	86	42	38	36	25	27	24	82	45	82	7	36	12	17	26
9	0	0	0	0	0	0	0	0	–	0	61	91	0	98	17	62	90	54	42	50	29	22	49	0	52	61	0	69	67	39	82
10	0	0	0	0	0	0	0	0	0	–	42	75	77	88	0	23	75	11	28	17	8	52	76	0	48	0	32	99	36	75	9
11	0	0	0	0	0	0	0	0	0	0	–	23	99	52	21	96	12	0	0	55	58	40	31	0	0	25	52	0	22	65	40
12	0	0	0	0	0	0	0	0	0	0	0	–	0	30	0	24	0	0	0	65	90	84	11	32	75	65	28	83	94	55	69
13	0	0	0	0	0	0	0	0	0	0	0	0	–	0	0	82	0	0	62	96	9	98	76	0	61	0	0	9	95	0	61
14	0	0	0	0	0	0	0	0	0	0	0	0	0	–	17	38	0	37	0	82	52	0	0	0	6	0	70	43	96	0	0
15	0	0	0	0	0	0	0	0	0	0	0	0	0	0	–	40	0	29	0	39	91	0	0	32	0	0	98	0	56	0	74
16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	–	87	81	17	56	92	17	72	42	82	87	53	72	87	36	83
17	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	–	0	0	26	46	0	0	0	0	0	44	0	27	0	0
18	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	–	90	97	70	59	90	98	0	10	0	0	52	95	0
19	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	–	27	13	0	0	0	0	0	16	0	29	78	0
20	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	–	91	71	14	36	95	96	51	97	32	14	12
21	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	–	98	76	98	25	15	41	18	11	60	60
22	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	–	66	86	0	77	37	19	14	0	0
23	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	–	79	49	31	27	79	72	69	59
24	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	–	77	0	67	82	46	0	0
25	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	–	53	35	0	61	0	0
26	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	–	17	0	20	0	0
27	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	–	35	11	15	30
28	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	–	37	0	0
29	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	–	0	29
30	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	–	74
31	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	–
32	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
33	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
34	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
35	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
36	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
37	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
38	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
39	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
40	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
41	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
42	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
43	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
44	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
45	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
46	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
47	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
48	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
49	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
50	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
51	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
52	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
53	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
54	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
55	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
56	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
57	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
58	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
59	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
60	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
61	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
62	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Du62– Custos de transporte de material entre as facilidades (continuação)

	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62
1	0	32	51	19	0	23	30	40	0	85	89	48	90	97	0	0	0	0	0	41	75	52	84	69	9	0	59	68	24	95	59
2	37	51	25	64	0	25	40	41	84	0	70	77	63	0	0	47	0	6	18	0	0	28	97	0	74	52	0	68	0	18	72
3	52	0	54	63	47	36	26	77	23	93	25	6	22	21	0	8	45	28	16	99	32	71	51	49	72	68	0	31	27	54	68
4	65	12	0	0	28	52	0	82	0	84	53	0	0	94	27	5	0	12	43	84	0	74	51	91	50	0	76	33	0	10	0
5	30	76	48	0	29	84	83	16	70	76	0	70	73	0	72	96	17	75	0	77	14	0	75	55	50	0	0	88	55	12	0
6	0	47	0	0	19	43	0	21	0	0	0	49	0	86	0	0	0	0	0	0	0	0	0	0	21	5	27	29	0	11	11
7	0	65	0	0	0	9	16	16	0	0	0	20	64	0	12	47	91	92	85	0	45	59	13	0	35	83	0	17	0	33	8
8	85	79	79	50	55	82	17	18	33	60	54	16	30	76	96	41	48	0	22	67	69	61	32	60	11	69	97	19	88	31	11
9	48	71	48	10	10	68	14	55	23	0	0	0	0	31	57	80	0	0	80	65	84	34	52	64	91	9	0	72	37	22	10
10	54	0	23	0	84	20	76	27	26	97	43	8	24	26	0	22	0	85	0	0	54	96	64	95	51	69	44	0	0	62	65
11	64	0	50	0	0	48	84	53	0	18	38	0	22	42	92	22	54	30	88	62	93	7	45	0	18	65	0	47	0	6	57
12	0	30	0	34	0	78	85	0	15	0	0	10	35	80	0	23	6	0	40	0	73	0	30	74	68	0	99	70	0	24	0
13	0	0	97	68	91	9	19	0	61	34	71	0	0	9	0	0	70	0	0	0	94	29	0	0	79	0	8	45	0	90	0
14	0	0	0	0	0	57	0	0	0	0	0	65	72	0	0	28	0	47	40	0	0	0	41	24	57	0	0	67	0	79	35
15	0	15	0	0	0	6	65	0	0	0	79	0	24	0	92	59	58	38	0	0	83	0	29	74	13	0	60	54	0	25	0
16	54	36	77	31	21	29	68	88	74	33	36	58	84	94	91	68	12	38	82	17	85	10	85	24	71	44	33	51	37	41	36
17	0	25	0	0	0	62	56	0	0	0	0	0	0	80	0	0	0	78	0	0	51	0	0	0	79	0	0	98	0	74	37
18	0	94	0	0	57	67	0	0	66	69	0	9	0	39	0	0	0	41	0	0	37	0	0	0	92	0	0	94	25	95	0
19	78	71	0	0	85	75	17	0	0	0	0	23	0	0	0	0	0	0	0	0	0	0	80	0	87	44	0	0	0	11	0
20	91	16	27	45	86	17	93	41	42	21	73	22	12	7	11	88	81	95	13	66	39	70	68	89	62	0	15	22	66	18	16
21	10	52	48	77	75	31	44	7	36	33	47	42	24	84	57	58	24	36	93	95	9	99	0	74	41	21	67	63	50	91	16
22	45	51	0	34	85	41	80	51	18	53	33	0	39	90	6	0	68	89	80	71	0	0	27	72	61	34	73	85	82	53	0
23	35	62	72	43	76	27	93	0	0	62	24	89	0	77	0	0	13	44	25	0	8	0	54	58	0	32	0	9	0	95	63
24	0	0	0	0	0	75	0	0	0	0	0	0	27	45	0	0	0	0	0	0	64	0	0	73	32	0	0	85	0	39	0
25	90	34	0	0	0	32	78	68	78	73	62	70	94	0	0	0	0	83	29	37	25	0	53	45	40	57	0	24	0	56	40
26	56	43	0	0	0	20	0	0	0	0	0	29	78	85	0	0	0	0	82	0	0	0	0	0	41	16	0	13	0	54	0
27	44	10	45	0	42	68	19	61	83	0	52	71	20	0	91	88	0	67	0	54	95	0	84	79	55	75	93	58	55	18	48
28	60	0	0	24	0	68	0	0	0	0	79	30	0	72	0	0	0	50	0	0	98	0	33	0	87	67	0	83	0	87	0
29	0	90	71	59	0	88	9	10	75	56	97	98	70	22	93	90	28	66	56	19	75	69	79	40	41	6	90	71	54	24	10
30	49	91	0	0	42	56	0	0	0	54	19	0	21	0	0	11	40	0	73	0	89	0	0	27	31	7	0	6	74	8	0
31	0	0	0	0	0	71	0	0	0	0	0	11	0	0	0	0	55	0	0	0	28	0	0	0	76	0	0	87	0	40	0
32	–	0	0	0	0	63	0	0	90	0	0	31	26	19	0	22	0	0	80	0	0	78	53	0	67	23	70	83	51	43	0
33	0	–	65	33	50	41	85	93	35	0	0	0	0	65	42	97	10	0	40	92	59	22	99	22	0	0	0	59	0	25	46
34	0	0	–	0	46	8	0	0	0	0	0	47	62	42	0	0	0	0	60	0	42	0	0	85	58	0	46	72	0	24	0
35	0	0	0	–	0	54	0	0	0	0	0	0	0	72	0	59	0	0	0	0	51	69	0	0	96	0	0	14	88	23	54
36	0	0	0	0	–	0	0	0	0	0	82	0	0	13	0	0	0	0	0	0	23	0	0	0	74	0	0	70	0	54	0
37	0	0	0	0	0	–	68	0	84	26	34	58	51	19	80	42	62	35	38	58	79	54	0	79	86	54	98	31	33	78	60
38	0	0	0	0	0	0	–	0	8	0	0	71	0	46	0	0	17	37	0	63	0	0	58	0	39	56	35	65	11	89	91
39	0	0	0	0	0	0	0	–	0	0	0	0	0	82	0	43	0	98	94	0	30	0	0	0	94	0	0	32	0	41	0
40	0	0	0	0	0	0	0	0	–	0	0	0	0	0	0	24	0	0	0	0	0	0	42	0	67	0	0	49	0	36	5
41	0	0	0	0	0	0	0	0	0	–	0	0	0	0	0	17	32	7	84	0	74	0	5	0	13	0	36	22	0	51	0
42	0	0	0	0	0	0	0	0	0	0	–	0	69	48	0	0	88	0	15	78	22	0	0	0	86	0	0	17	0	24	69
43	0	0	0	0	0	0	0	0	0	0	0	–	95	0	0	35	56	0	89	90	29	64	47	7	0	0	0	74	0	27	95
44	0	0	0	0	0	0	0	0	0	0	0	0	–	46	85	86	7	99	0	0	59	94	0	0	43	0	0	77	0	36	0
45	0	0	0	0	0	0	0	0	0	0	0	0	0	–	0	62	0	73	5	0	0	0	0	88	83	55	21	86	50	13	0
46	0	0	0	0	0	0	0	0	0	0	0	0	0	0	–	0	0	0	0	0	0	0	26	0	0	0	0	39	0	98	0
47	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	–	50	0	72	0	91	0	0	0	88	46	0	79	0	62	7
48	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	–	41	0	61	0	74	51	0	10	0	44	72	0	83	36
49	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	–	0	0	53	0	65	0	26	11	0	9	0	59	0
50	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	–	0	19	0	0	18	9	0	21	33	53	55	14
51	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	–	0	0	0	13	48	0	0	11	0	8	0
52	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	–	0	60	0	76	0	8	0	8	20	76
53	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	–	65	0	0	0	0	70	0	60	0
54	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	–	0	25	0	83	99	0	87	11
55	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	–	7	55	0	89	0	79	10
56	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	–	0	28	47	73	91	68
57	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	–	0	84	0	0	44
58	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	–	81	0	87	0
59	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	–	31	80	25
60	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	–	76	0
61	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	–	82
62	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	–

Du62 – Área das facilidades:

1	210
2	130
3	224
4	260
5	208
6	294
7	323
8	266
9	441

10	340
11	143
12	168
13	342
14	357
15	420
16	147
17	380
18	144

19	187
20	240
21	140
22	304
23	300
24	162
25	252
26	196
27	176

28	144
29	221
30	130
31	182
32	136
33	399
34	210
35	150
36	108

37	357
38	144
39	252
40	135
41	204
42	204
43	99
44	160
45	357

46	260
47	190
48	280
49	180
50	104
51	198
52	160
53	200
54	361

55	231
56	140
57	77
58	187
59	231
60	91
61	82
62	272