MINISTÉRIO DA CIÊNCIA E TECNOLOGIA
INSTITUTO NACIONAL DE PESQUISAS ESPACIAIS

sid.inpe.br/mtc-m21b/2015/05.19.16.39-TDI

UMA PLATAFORMA DE BUSCA DE LAYOUTS INICIAIS PARA O PROBLEMA DE ALOCAÇÃO DE EQUIPAMENTOS EM SATÉLITES

Gustavo Furtado de Oliveira Alves

Dissertação de Mestrado do Curso de Pós-Graduação em Computação Aplicada, orientada pelos Drs. Sandra Aparecida Sandri, e José Carlos Becceneri, aprovada em 14 de maio de 2015.

URL do documento original:
<http://urlib.net/8JMKD3MGP3W34P/3JGM3K2>

INPE
São José dos Campos
2015

PUBLICADO POR:

Instituto Nacional de Pesquisas Espaciais - INPE
Gabinete do Diretor (GB)
Serviço de Informação e Documentação (SID)
Caixa Postal 515 - CEP 12.245-970
São José dos Campos - SP - Brasil
Tel.:(012) 3208-6923/6921
Fax: (012) 3208-6919
E-mail: pubtc@sid.inpe.br

MINISTÉRIO DA CIÊNCIA E TECNOLOGIA
INSTITUTO NACIONAL DE PESQUISAS ESPACIAIS

sid.inpe.br/mtc-m21b/2015/05.19.16.39-TDI

UMA PLATAFORMA DE BUSCA DE LAYOUTS INICIAIS PARA O PROBLEMA DE ALOCAÇÃO DE EQUIPAMENTOS EM SATÉLITES

Gustavo Furtado de Oliveira Alves

Dissertação de Mestrado do Curso de Pós-Graduação em Computação Aplicada, orientada pelos Drs. Sandra Aparecida Sandri, e José Carlos Becceneri, aprovada em 14 de maio de 2015.

URL do documento original:
<http://urlib.net/8JMKD3MGP3W34P/3JGM3K2>

INPE
São José dos Campos
2015

Dados Internacionais de Catalogação na Publicação (CIP)

Alves, Gustavo Furtado de Oliveira.

Al87p Uma plataforma de busca de layouts iniciais para o problema de alocação de equipamentos em satélites / Gustavo Furtado de Oliveira Alves. – São José dos Campos : INPE, 2015.

xxii + 81 p. ; (sid.inpe.br/mtc-m21b/2015/05.19.16.39-TDI)

Dissertação (Mestrado em Computação Aplicada) – Instituto Nacional de Pesquisas Espaciais, São José dos Campos, 2015.

Orientadores : Drs. Sandra Aparecida Sandri, e José Carlos Becceneri.

1. SMLDP. 2. MPCA. 3. HBAE. 4. SABH. 5. Satellite layout. I.Título.

CDU 004.8:629.784

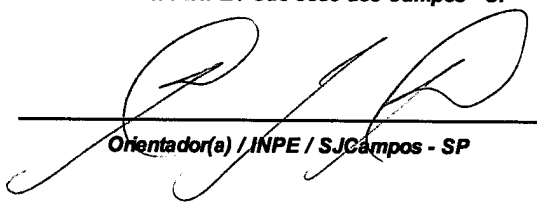
Aprovado (a) pela Banca Examinadora em cumprimento ao requisito exigido para obtenção do Título de ***Mestre*** em

Computação Aplicada

Dr. Luiz Antonio Nogueira Lorena

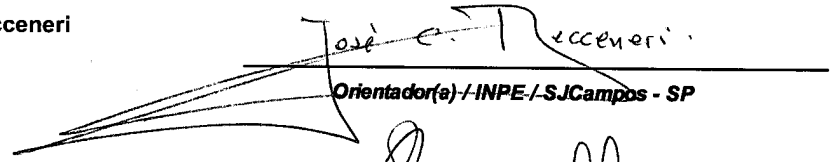
Presidente / INPE / São José dos Campos - SP

Dra. Sandra Aparecida Sandri



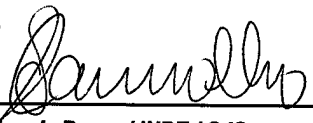
Orientador(a) / INPE / SJCampos - SP

Dr. José Carlos Becceneri



Orientador(a) / INPE / SJCampos - SP

Dr. Solon Venâncio de Carvalho

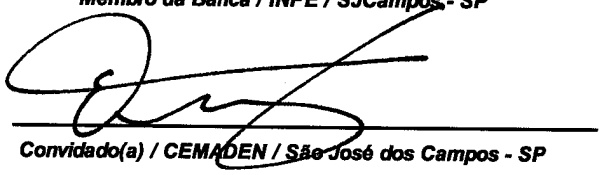


Membro da Banca / INPE / SJCampos - SP

Dr. Fabiano Luis de Sousa

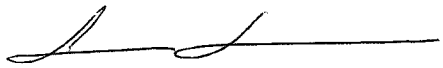
Membro da Banca / INPE / SJCampos - SP

Dr. Eduardo Fávero Pacheco da Luz



Convidado(a) / CEMADEN / São José dos Campos - SP

Dr. Luiz Leduíno de Salles Neto



Convidado(a) / UNIFESP / São José dos Campos - SP

Este trabalho foi aprovado por:

( ) maioria simples

(X) unanimidade

Título: ***" Uma Plataforma de Busca de layouts iniciais para o problema de alocação de equipamentos em Satélites."***

Aluno (a): ***Gustavo Furtado de Oliveira Alves***

São José dos Campos, 14 de Maio de 2015

"É melhor ser criticado pelos sábios do que ser elogiado pelos insensatos."

ECLESIASTES

A meus pais Vanda, Ernandes, irmão Philippe e a minha esposa Daiana.

AGRADECIMENTOS

Agradeço primeiramente a Deus pelo dom da vida e os demais dons a mim confiados.

Aos meus orientadores Sandra Sandri e José Carlos Becceneri, pela competência, dedicação, atenção, paciência, incentivo, ensinamentos e confiança. Posso, seguramente, afirmar que foram dois dos melhores professores que já tive. Com características complementares, me proporcionaram um amadurecimento acadêmico inimaginável.

A minha esposa, meus pais e irmão, pelo incentivo e apoio singulares em momentos importantíssimos para o sucesso deste trabalho.

Aos amigos que fiz nestes três anos de mestrado, pelo companheirismo, estudo e conhecimentos compartilhados.

Ao Instituto Nacional de Pesquisas Espaciais - INPE, pela oportunidade de estudo e utilização de suas instalações.

Aos professores que tanto me ensinaram nas aulas do curso.

Por fim, a todos aqueles que de alguma forma contribuíram para a conclusão deste trabalho.

RESUMO

Neste trabalho é proposta uma plataforma para criação de layouts iniciais para o problema de alocação de equipamentos em satélites artificiais, considerando o centro de massa e o momento de inércia do conjunto, além de restrições para o posicionamento de cada equipamento no satélite. A posição de cada equipamento, de massas diferentes, influencia diretamente a localização do centro de massa e o momento de inércia do satélite. Encontrar o melhor layout consiste em um problema de natureza "NP-Difícil". Ao desenvolver layouts de satélites também é importante considerar a dissipação térmica gerada pelos equipamentos e a conexão de cabos entre eles. As meta-heurísticas inspiradas na natureza *Ant Colony Optimization* e *Multi-Particle Collision Algorithm* foram implementadas para buscar soluções otimizadas e uma heurística chamada *Heurística de Balanceamento por Afinidade Espacial* foi desenvolvida para este problema, baseada na afinidade entre os equipamentos quando bons layouts são encontrados. O satélite brasileiro ITASAT foi utilizado como base para o desenvolvimento deste trabalho. A plataforma permite a introdução de conhecimento *a priori* no processo de exploração realizado pelos algoritmos de otimização, tais como fixar a posição de determinados objetos e restringir uma distância mínima e/ou máxima entre dois objetos.

Palavras-chave: SMLDP. ACO. MPCA. HBAE. SABH. Satellite Layout.

A PLATFORM FOR SEARCH INITIAL LAYOUTS FOR THE SATELLITE EQUIPMENTS ALLOCATION PROBLEM

ABSTRACT

In this work is proposed a platform for creating initial layout for the artificial satellite equipments allocation problem, considering the center of mass and the moment of inertia of the solution, as well as restrictions on the placement of each equipment. The positioning of equipments with different masses, directly influences the location of the center of mass and the moment of inertia of the satellite. Find the best layout consists of a problem of "NP-Hard" nature. At satellites layouts development is also important to consider the thermal dissipation generated by the equipment and the cable connection between them. The nature-inspired meta-heuristics *Ant Colony Optimization* and *Multi-Particle Collision Algorithm* were implemented to find optimized solutions and a new heuristic called *Spatial Affinity Balancing Heuristic* was developed for this problem, based on the affinity between the equipment when good layouts are found. The Brazilian satellite ITASAT was used as the basis for the development of this work. The platform allows the introduction of knowledge *a priori* in the exploration process performed by the optimization algorithms, such as setting the position of certain objects and add constraints for minimal and/or maximal distances between two objects.

Keywords: SMLDP. ACO. MPCA. HBAE. SABH. Satellite Layout.

LISTA DE FIGURAS

Pág.

LISTA DE TABELAS

Pág.

LISTA DE ABREVIATURAS E SIGLAS

INPE	–	Instituto Nacional de Pesquisas Espaciais
ACO	–	Ant Colony Optimization
MPCA	–	Multiple Particle Colition Algorithm
SMLDP	–	Satellite Module Layout Design Problem
OPT	–	Order-based Positioning Technique
C&P	–	Cutting and Packing
LOP	–	Layout Optimization Problems
HBAE	–	Heurística de Balanceamento por Afinidade Espacial
SABH	–	Spatial Affinity Balancing Heuristic
POL	–	Problema de Otimização de Layout

SUMÁRIO

Pág.

1 INTRODUÇÃO

Problemas onde um conjunto de itens devem ser alocados num recipiente qualquer, sem sobreposição, de acordo com um conjunto de objetivos e restrições, são conhecidos como Problemas de Otimização de Layouts (POL), um caso particular de Problemas de Corte e Empacotamento (C&P), e são geralmente NP-difíceis (DYCKHOFF, 1990). Posicionar equipamentos dentro de satélites artificiais é um tipo específico de POL conhecido como Problema de Projeto de Layout de Módulo de Satélites (SMLDP, do inglês *Satellite Module Layout Design Problem*) (WANG et al., 2009).

O SMLDP é um problema particularmente difícil pois, considerando tanto o momento do lançamento do satélite quanto sua operação durante a vida útil deste no espaço, tem complexidade derivada de um grande número de variáveis a serem otimizadas e de restrições de engenharia. Segundo Sousa et al. (2013b), na definição das arquiteturas estrutural, mecânica e elétrica de um satélite, o layout dos equipamentos nos painéis estruturais deve satisfazer os requisitos mecânicos e elétricos além das restrições envolvidas. Tais como, encontrar uma solução tal que produza um centro de massa do sistema o mais próximo possível de uma posição desejável, maximizar o momento de inércia de uma determinada direção, minimizar as interferências eletromagnéticas, etc.

Tudo deve ser planejado e projetado para funcionar em diversas situações críticas com grandes variações de radiação e temperatura. O projeto de layout de um satélite influencia diretamente o desempenho, a vida útil, o custo de montagem e a integração de todo o sistema. Por estas razões, encontrar a melhor disposição dos equipamentos em um satélite é fundamental para melhorar o desempenho global do mesmo durante sua missão espacial. Dentre os principais objetivos visados pelas equipes de engenharia ao desenvolver o layout de um satélite, otimizar a posição do centro de massa em relação a um ponto determinado pelos projetistas e maximizar o momento de inércia referente ao eixo principal de rotação são dois objetivos muito importantes para a estabilidade do satélite.

Como discutido por Cuco (2011), usualmente a arquitetura mecânica de um novo satélite é determinada ”manualmente” baseada na experiência da equipe de projetistas e em projetos implementados anteriormente para missões semelhantes. Há uma tendência de se utilizar sistemas automatizados para gerar um conjunto inicial de layouts otimizados, a partir dos quais os projetistas obtém um layout final ajustando as soluções iniciais (SOUSA et al., 2013b).

Técnicas de Inteligência Computacional têm sido muito utilizadas atualmente para encontrar soluções para problemas de otimização. Estes métodos utilizam estratégias de busca com balanço dinâmico entre exploração e diversificação ("*exploration*" e "*exploitation*" em inglês) para evitar o confinamento em mínimos ou máximos locais. Nas duas últimas décadas, muitos algoritmos de otimização foram desenvolvidos com inspiração em fenômenos naturais. Comportamentos de pássaros, vaga-lumes, formigas, colisão de partículas, evoluções genéticas, entre outros, tem sido observados por pesquisadores no intuito de utilizar a experiência de evolução da natureza para auxiliar na solução de problemas reais. Deste estudo tem surgido muitas heurísticas e meta-heurísticas que se mostraram eficientes para solucionar problemas de otimização (BECCENERI; SILVA NETO, 2009).

O Algoritmo de Otimização por Colônia de Formigas (ACO, do inglês "*Ant Colony Optimization*"), proposto por Dorigo (1992), é um desses algoritmos bio-inspirados que se baseia na forma como as formigas são capazes de encontrar um bom caminho, talvez o melhor, entre a colônia e a fonte de comida. Para isso, cada formiga, individualmente, trilha um caminho baseado na experiência das formigas que já fizeram o trajeto.

Em Xu et al. (2010) foi proposta a técnica de Posicionamento Baseado em Ordenação (OPT) em conjunto com o ACO para tratar o POL. Trata-se de um método construtivo que visa otimizar a ordem em que os objetos são alocados em uma solução candidata. Entretanto, conforme discutido em Alves et al. (2013a), diferentemente dos objetivos do problema apresentado por Xu et al. (2010), no SMLDP a ordem em que os objetos são alocados no container não é tão importante quanto a posição real do objeto na solução final. Pois, enquanto no primeiro busca-se otimizar o centro de massa e o raio resultante do envelope da solução, o que naturalmente requer que os objetos estejam mais próximos do centro, no problema tratado neste trabalho, objetos mais afastados do centro tendem a gerar soluções melhores devido a um dos objetivos da otimização em que busca-se maximizar o momento de inércia do satélite.

Outra meta-heurística inspirada na natureza, mais recente, que se baseia no comportamento de colisão nuclear de partículas num reator atômico, é o MPCA (*Multiple Particle Collision Algorithm*) proposto por Luz (2012) como uma variante populacional do PCA (*Particle Collision Algorithm*) apresentado por Sacco e Oliveira (2005). Diferentemente do ACO que é um algoritmo construtivo, o PCA e o MPCA partem de uma solução inicial e seguem alterando a solução a fim de melhorá-la.

De tempos em tempos uma nova solução "inicial" é criada de acordo com critérios probabilísticos que consideram a qualidade da solução. Uma característica interessante do PCA para o problema tratado neste trabalho, são as perturbações que o algoritmo aplica na solução, como se o layout sofresse uma chacoalhada com maior ou menor intensidade, dependendo do grau de perturbação decidido pelo algoritmo.

No SMLDP a região em que um equipamento é posicionado tem grande importância, principalmente quando se observa objetivos como interligação de equipamentos de um mesmo subsistema, interferências e dissipação térmica ou mesmo os principais objetivos a serem otimizados. Este trabalho também propõe uma heurística que, baseado na experiencia adquirida ao longo do processamento, considera o relacionamento entre os equipamentos ao selecionar uma boa região para alocá-los. A heurística nomeada "Heurística de Balanceamento por Afinidade Espacial" (HBAE) pode ser utilizada em conjunto com outros métodos populacionais.

Para facilitar a implementação de algoritmos de otimização para tratar o SMLDP e produzir layouts iniciais de satélites para as equipes de engenharia, foi desenvolvida neste trabalho, uma plataforma capaz de mostrar em tempo real os resultados obtidos pelos algoritmos. Utilizando recursos de Programação Orientada a Objetos, esta plataforma facilita a implementação de novos algoritmos de otimização.

Esta dissertação está dividida em 6 capítulos. O Capítulo 2 apresenta a motivação e a problemática do corrente trabalho. No Capítulo 3 são apresentados os métodos de otimização implementados e a heurística desenvolvida para o problema. No Capítulo 4 são apresentados os detalhes da implementação da plataforma de busca de soluções iniciais para o problema de layout de satélites. O Capítulo 5 apresenta os experimentos realizados e análise dos resultados obtidos. Finalmente, no Capítulo 6 são apresentadas as conclusões e as sugestões para trabalhos futuros de evolução da plataforma.

2 O PROBLEMA DE LAYOUT DE SATÉLITES

Projetos de desenvolvimento de satélite são desafiadores do início ao fim. Tudo deve ser planejado e projetado para funcionar em diversas situações críticas durante todo o ciclo de vida de uma missão. Entretanto na maior parte do ciclo de vida o satélite estará no espaço. Uma das principais características desse tipo de projeto concerne ao controle e acompanhamento da missão que deve ocorrer remotamente. Devido à grande dificuldade de reparação de equipamentos em órbita, muitas vezes impossível, os projetistas têm que prever situações de risco e manobras para toda a vida útil do satélite.

Durante a fase de concepção da arquitetura de satélites artificiais, um dos grandes desafios dos engenheiros é encontrar um layout ótimo para o posicionamento dos equipamentos no interior do satélite, visando otimizar um conjunto de objetivos. Estruturas de satélites desenvolvidas em projetos anteriormente bem sucedidos dão um bom ponto de partida para o desenvolvimento de layouts futuros (FORTESCUE et al., 2011). Segundo a metodologia apresentada por Cuco (2011), os principais fatores comumente usados para definir a disposição dos equipamentos no interior de um satélite são:

- obter a posição de centro de massa próximo dodo centro de gravidade definido pelos projetistas;
- atender requisitos de alinhamento de momentos de inércia;
- evitar a concentração equipamentos com alta dissipação térmica;
- agrupamento de equipamentos do mesmo subsistema;
- otimizar a disposição de cablagem e conectores;

Baseado nos fatore considerados na metodologia de Cuco (2011), pode-se definir alguns objetivos de otimização para direcionar a busca pelo layout ótimo do satélite, tais como:

- Minimizar a distância do centro de massa do satélite e um ponto previamente definido pela equipe de projetistas como requisito;
- Maximizar o momento de inércia do satélite em relação ao eixo principal (para satélites “spinados”);

- Minimizar interferências eletromagnéticas entre equipamentos eletrônicos;
- Maximizar a uniformidade de dissipação térmica dos equipamentos sobre o painel;
- Minimizar a distância entre equipamentos de um mesmo subsistema.

Além disso, o layout deve atender as seguinte restrições básicas:

- Todos os equipamentos devem estar contidos no interior do módulo;
- Não deve haver sobreposição entre equipamentos.

Problemas onde um conjunto de itens devem ser colocados num recipiente qualquer, sem sobreposição, de acordo com um conjunto de objetivos e restrições, são conhecidos como Problemas de Otimização de Layouts (*Layout Optimization Problems*, LOP), um caso particular de Problemas de Corte e Empacotamento (*Cutting and Packing*, C&P), e são geralmente NP-difíceis (DYCKHOFF, 1990).

O problema de posicionamento de equipamentos no interior do módulo de satélites artificiais que visa otimizar um conjunto de objetivos, enquanto satisfaz restrições espaciais e/ou de desempenho, é denominado Problema de Projeto de Layout de Módulo de Satélites ou SMLDP, sigla em inglês para "*Satellite Module Layout Design Problem*" (WANG et al., 2009), um tipo de LOP.

Em uma série de trabalhos desenvolvidos no INPE, ferramentas e metologias vem sendo apresentadas para auxiliar projetistas na definição de layouts de satélites. Em Lau et al. (2014) é apresentada uma ferramenta baseada em Excel, que utiliza a metologia de Cuco (2011). Esta ferramenta integra o Excel ao SolidWorks para calcular parâmetros de projeto e apresentar soluções candidatas graficamente. O conceito desta ferramenta foi apresentado na *22th International Congress of Mechanical Engineering (COBEM 2013)* Sousa et al. (2013b).

2.1 ITASAT

A plataforma de geração de layouts iniciais de satélites proposta neste trabalho teve inspiração em um satélite brasileiro de pequeno porte chamado ITASAT-1, por causa da simplicidade estrutural deste. Isso viabilizou o desenvolvimento desta primeira versão da plataforma como um trabalho de mestrado. O ITASAT encontra-se atualmente em desenvolvimento por um grupo liderado pelo Instituto Tecnológico de Aeronáutica (ITA).

Como é possível observar na Figura 2.1 e na Figura 2.2, a estrutura proposta para o ITASAT-1 consiste em uma prateleira única posicionada no centro do satélite, onde os equipamentos são alocados. Este projeto estrutural possibilita tratar o SMLDP do ITASAT-1 como um problema de duas dimensões. Ou seja, não é necessário validações de sobreposição na terceira dimensão, o que aumentaria consideravelmente a complexidade do projeto da primeira versão desta plataforma.

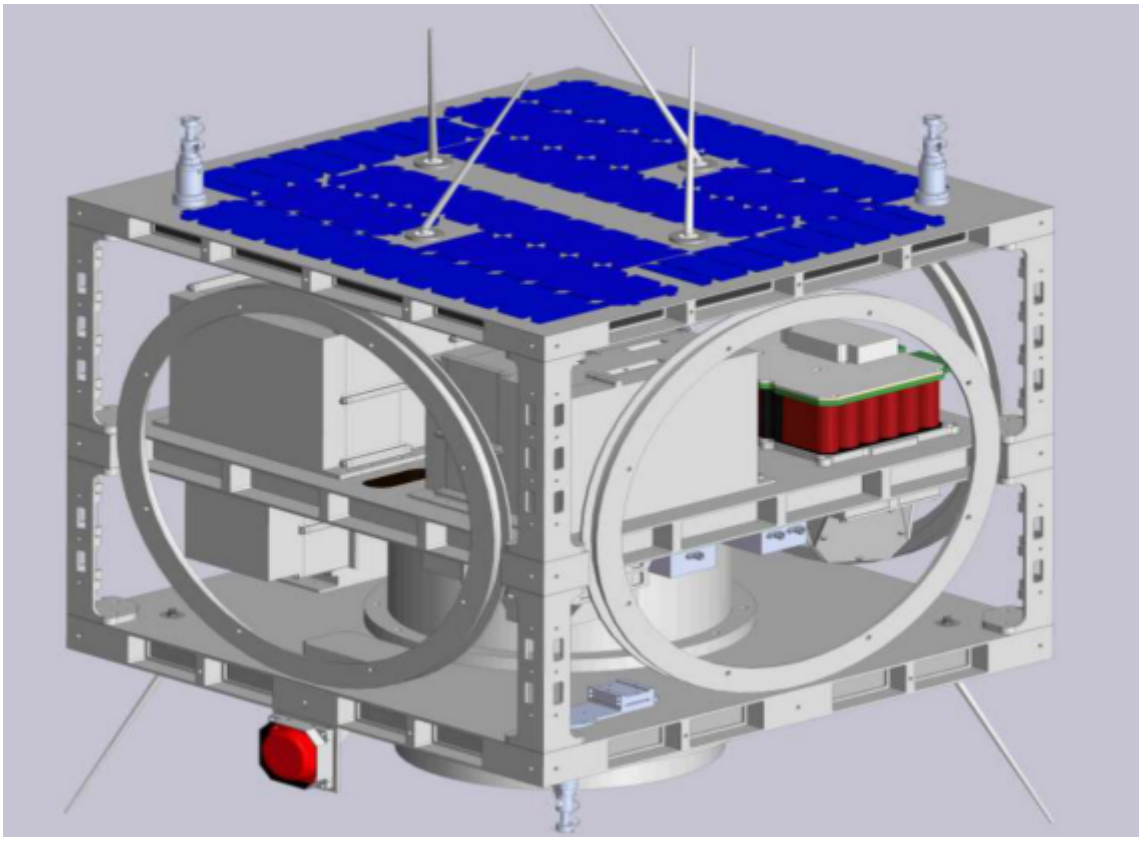


Figura 2.1 - Ilustração do projeto estrutural do ITASAT-1 com uma prateleira central de duas faces de alocação.

O projeto ITASAT-1 passou por readequações estratégicas na filosofia de desenvolvimento e na estrutura do equipamento, não previstas no início do desenvolvimento desta dissertação. Segundo Sato (2014) inicialmente, o ITASAT-1 foi pensado para ter de 80 a 100 kg. Na sua nova configuração, ele deverá ter menos de 10 kg.

O projeto estrutural inicialmente proposto e utilizado como estudo nesta dissertação a princípio não será mais utilizado na nova configuração do ITASAT-1, entretanto

pretende-se evoluir a plataforma desenvolvida durante este trabalho para tratar outras configurações estruturais de satélites, servido o corrente problema para viabilizar o desenvolvimento da primeira versão deste software.

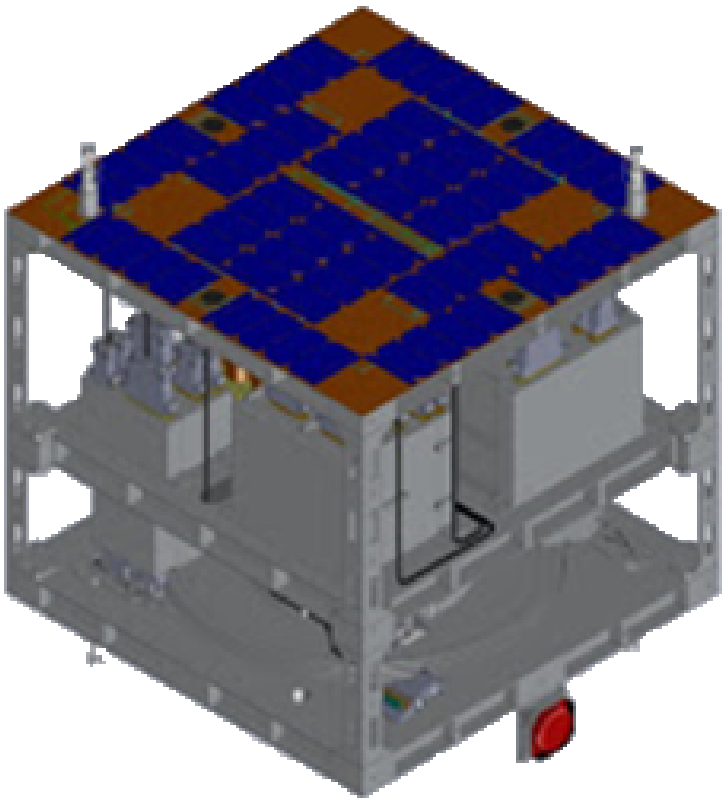


Figura 2.2 - Representação da estrutura proposta para o ITASAT-1

2.2 Objetivos de Otimização

Neste trabalho focamos em dois objetivos de projeto como principais para otimização. Ou seja, um problema bi-objetivo. A aproximação do centro de massa de um ponto pré-definido pela equipe de projetistas e a maximização do momento de inércia do conjunto em relação ao eixo principal de rotação do satélite. A seguir é explicado como calcular o centro de massa e o momento de inércia de um sistema.

2.2.1 Centro de Massa

Quando um corpo, ou sistema de corpos, se movimenta de forma linear, todas as suas partículas movem-se na mesma direção e com a mesma velocidade. Entretanto se o

sistema gira, cada partícula se move de forma diferente, com velocidades diferentes (CORRADI et al., 2010). A Figura 2.3 ilustra este comportamento em um sistema de partículas que se movimenta de forma linear (a) e o mesmo sistema em rotação (b).

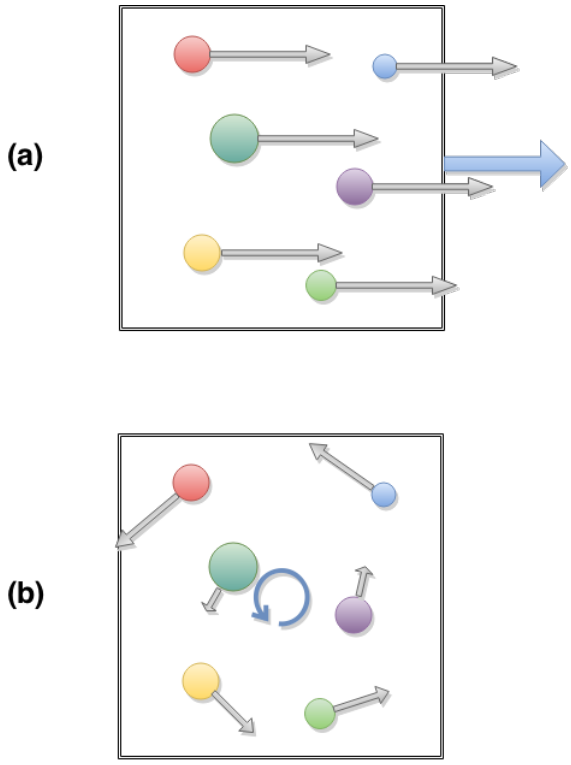


Figura 2.3 - Movimento das partículas de um sistema em (a) movimento linear e em (b) movimento rotacional

Mesmo quando um corpo gira, existe um ponto nesse corpo, chamado **centro de massa**, que se desloca da mesma maneira que se deslocaria uma única partícula, com a massa deste corpo e sujeita ao mesmo sistema de forças que ele, ainda que o sistema não seja um corpo rígido mas um conjunto de corpos (CORRADI et al., 2010). O centro de massa de um corpo ou de um sistema de corpos é o ponto que se move como se toda a massa estivesse concentrada nele e como se todas as forças externas fossem aplicadas neste ponto.

O cálculo para encontrar a posição do centro de massa de um sistema pode ser olhado como uma média ponderada da posição de cada partícula, onde o peso é a a fração da massa da partícula pela massa total do conjunto. Para um conjunto de duas partículas em uma única dimensão como ilustrado na Figura 2.4, o centro de

massa pode ser calculado conforme a Equação 2.1.

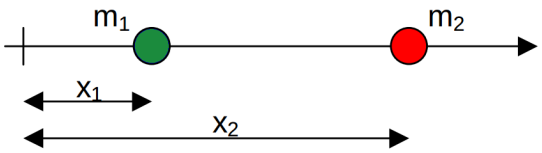


Figura 2.4 - Duas partículas em uma dimensão

$$x_{CM} = \left(\frac{m_1}{m_1 + m_2}\right) x_1 + \left(\frac{m_2}{m_1 + m_2}\right) x_2 \tag{2.1}$$

ou

$$x_{CM} = \frac{m_1 x_1 + m_2 x_2}{m_1 + m_2} \tag{2.2}$$

Para um conjunto de N partículas dispostas em uma linha reta, podemos estender a Equação 2.2:

$$x_{CM} = \frac{m_1 x_1 + m_2 x_2 + ... + m_N x_N}{m_1 + m_2 + ... + m_N} = \frac{\sum_{i=1}^{N} m_i x_i}{\sum_{i=1}^{N} m_i} \tag{2.3}$$

Para duas dimensões podemos considerar a Equação 2.4 por analogia à equação anterior. Sendo P_{CM} o ponto onde está localizado o centro de massa no plano.

$$P_{CM} = \left(\frac{\sum_{i=1}^{N} m_i x_i}{\sum_{i=1}^{N} m_i}, \frac{\sum_{i=1}^{N} m_i y_i}{\sum_{i=1}^{N} m_i}\right) \tag{2.4}$$

Considerando que seja o centro da superfície o ponto pré-definido para otimizar a distância do centro de massa. Podemos encontrar a distância do centro de massa e a origem através da Equação 2.5.

$$D_{CM-Origem} = \sqrt{\left(\frac{\sum_{i=1}^{N} m_i x_i}{\sum_{i=1}^{N} m_i}\right)^2 + \left(\frac{\sum_{i=1}^{N} m_i y_i}{\sum_{i=1}^{N} m_i}\right)^2} \tag{2.5}$$

2.2.2 Momento de inércia

A orientação de um satélite, isto é, a direção para a qual o satélite aponta é denominada de **atitude**. Compreender como se dá este movimento de atitude é importante para prever o comportamento do satélite. Uma das formas de manter o satélite numa dada orientação consiste em fazê-lo girar ao redor de um eixo. Assim como um pião permanece de pé quando posto para girar, um satélite tende a manter-se apontado para uma mesma direção quando estiver em rotação (CARRARA, 2012).

A inércia que um corpo apresenta para a rotação é chamada de **momento de inércia**. Ou seja, a inércia para adquirir uma aceleração relativa a um eixo rotacional (CORRADI et al., 2010). Para se obter uma estabilidade do satélite, o momento de inércia do conjunto deve ser maximizado com relação ao eixo principal de rotação.

Para um sistema de N partículas como o da Figura 2.3 (b), sendo m_i a massa de cada partícula e r_i a distâncias entre uma partícula e o eixo de rotação, o momento de inércia é dado por:

$$I = \sum_{i=1}^{N} m_i r_i^2 \tag{2.6}$$

Sua unidade é o produto da unidade de massa pelo quadrado da unidade de distância. No sistema SI, ela é $kg.m^2$. Nessa equação está representado o efeito da distribuição de massa do corpo através do produto da massa em um ponto pelo quadrado da distância deste ponto ao eixo. Em um sistema com vários corpos distribuídos, o momento de inércia é calculado considerando-se cada corpo como uma partícula, ou seja, as dimensões do objeto não são relevantes para a equação, apenas a posição de seu centro de massa no sistema. O momento de inércia de um corpo é um escalar.

2.2.3 Função Objetivo

Para realizar a otimização do layout dos equipamentos de um satélite, a maioria dos autores compõem os objetivos do problema em uma única função objetivo utilizando métodos de agregação(CUCO, 2011). Uma das formas mais simples de implementar este método é através da soma ponderada entre os objetivos, criando uma única

função-compromisso que agrega os objetivos através de uma combinação linear. Esta estratégia requer a definição prévia dos pesos de cada objetivo na otimização. A Equação 2.7 utiliza os fatores λ_1 e $\lambda_2 \in \mathbf{R}$ para ponderar o peso de cada objetivo na função final.

$$\max(f_{obj}) = \lambda_2 \max(I) - \lambda_1 \min(D_{CM-Origem}) \tag{2.7}$$

Sujeito a:

- λ_1 e $\lambda_2 \in \mathbf{R}$;
- I conforme Equação 2.6;
- $D_{CM-Origem}$ conforme Equação 2.5.

2.2.4 Restrições e Objetivos secundários

Para um layout ser considerado válido, este deve atender algumas restrições de projeto pré-definidas. Neste trabalho, é considerado que os equipamentos a serem alocados sobre o painel do satélite sejam retangulares.

A primeira verificação espacial para uma solução ser considerada válida, diz respeito a duas condições básicas para o posicionamento dos equipamentos no painel: É preciso garantir que os equipamentos alocados estejam inteiramente sobre a superfície do painel e também não pode haver sobreposição espacial entre os equipamentos.

A Figura 2.5 apresenta uma ilustração destas duas formas irregulares de alocação. O objeto 5 está sobrepondo um outro objeto da solução e o objeto 6 não está totalmente sobre a superfície do painel.

2.2.4.1 Alocação irregular de objetos

Como os equipamentos podem estar posicionados em qualquer orientação angular sobre a superfície do painel, para verificar se um equipamento O_i está inteiramente sobre a superfície S é necessário identificar a posição dos 4 vértices do equipamento. A Figura 2.6 apresenta um equipamento O_i, cujo centro geométrico está na posição $P(x_i, y_i)$, sobre a superfície S, com um ângulo θ_i em relação ao eixo x.

Pode-se calcular a posição de cada vértice v_{i1}, v_{i2}, v_{i3} e v_{i4} do equipamento geometricamente utilizando a posição do equipamento e o seu ângulo. A Figura 2.7 ilustra

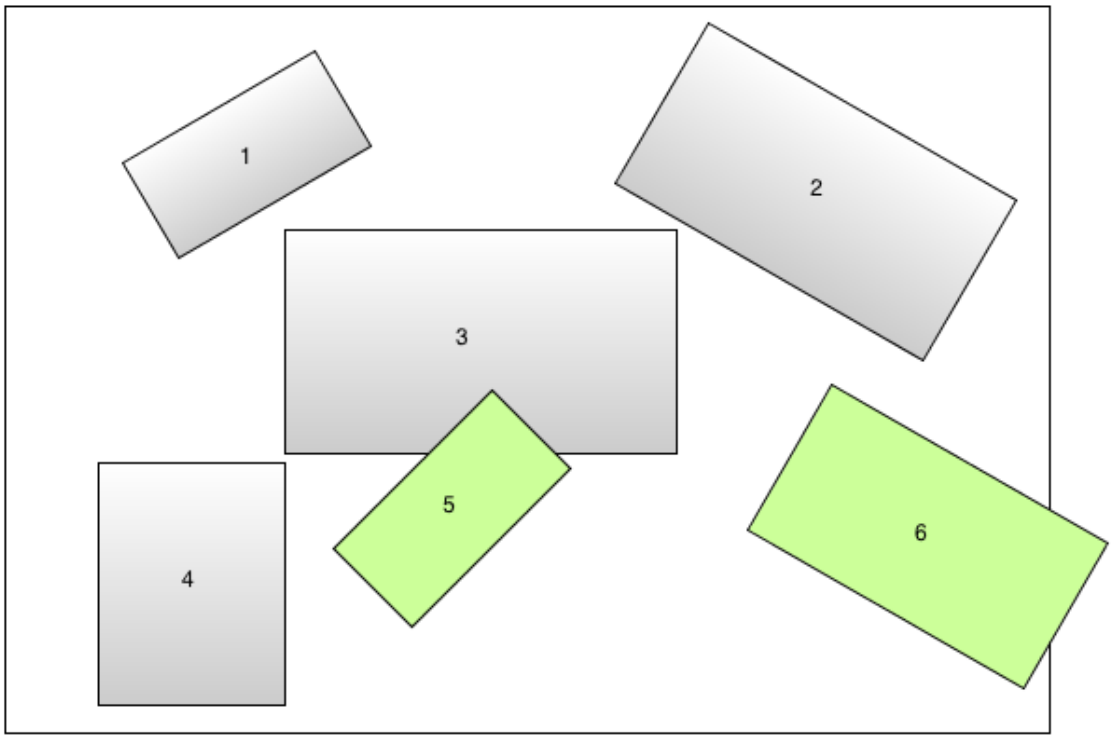


Figura 2.5 - Representação de um layout inválido.

a análise trigonométrica realizada para obter a abscissa do vértice v_{i1}.

Para encontrar o valor x_{i1} do ponto v_{i1} é necessário extrair $(b - a)$ de x_i. Utilizando o ângulo θ_i e trigonometria básica obtém-se o valor de a e b conforme as Equações 2.8 e 2.9

$$a = \sin(\theta_i)\frac{V_i}{2} \tag{2.8}$$

$$b = \cos(\theta_i)\frac{H_i}{2} \tag{2.9}$$

Substituindo a e b na equação para encontrar x_{i1}, tem-se:

$$x_{i1} = x_i - \cos(\theta_i)\frac{H_i}{2} + \sin(\theta_i)\frac{V_i}{2} \tag{2.10}$$

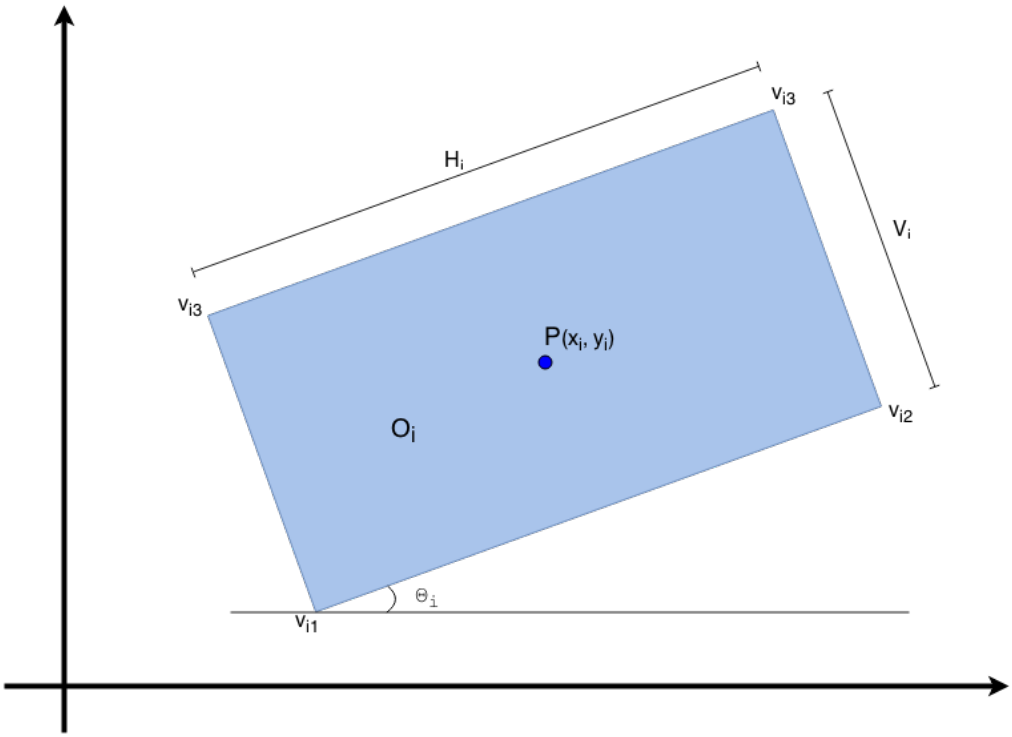


Figura 2.6 - Identificação dos vértices de um equipamento

O mesmo princípio é aplicado para encontrar a ordenada do vértice v_{i1}, Como ilustrado na Figura 2.8.

Para encontrar o valor y_{i1} do ponto v_{i1} é necessário extrair $(b + a)$ de y_i. Utilizando o ângulo θ_i e trigonometria básica obtém-se o valor de a e b conforme as Equações 2.11 e 2.12

$$a = \cos(\theta_i)\frac{V_i}{2} \tag{2.11}$$

$$b = \sin(\theta_i)\frac{H_i}{2} \tag{2.12}$$

Substituindo a e b na equação para encontrar y_{i1}, tem-se:

$$y_{i1} = y_i - \sin(\theta_i)\frac{H_i}{2} - \cos(\theta_i)\frac{V_i}{2} \tag{2.13}$$

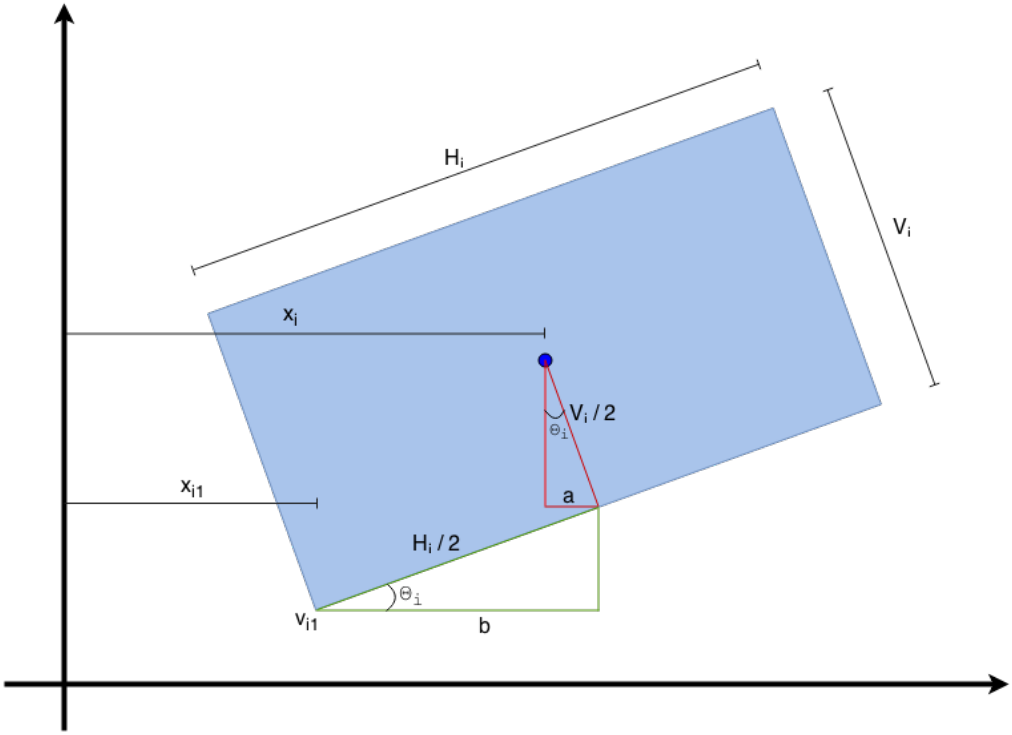


Figura 2.7 - Obtenção da abscissa do vértice v_{i1}

Unificando as Equações 2.10 e 2.13 em forma de matriz obtém-se a Equação 2.14.

$$\begin{bmatrix} x_{i1} \\ y_{i1} \end{bmatrix} = \begin{bmatrix} x_i \\ y_i \end{bmatrix} + \begin{bmatrix} \cos(\theta_i) & -\sin(\theta_i) \\ \sin(\theta_i) & \cos(\theta_i) \end{bmatrix} \begin{bmatrix} -\frac{H_i}{2} \\ -\frac{V_i}{2} \end{bmatrix} \tag{2.14}$$

A mesma análise geométrica pode ser feita para encontrar as coordenadas dos demais vértices do objeto. As Equações 2.15, 2.16 e 2.17 apresentam o resultado da aplicação deste princípio para encontrar os vértices v_{i2}, v_{i3} e v_{i4} do equipamento.

$$\begin{bmatrix} x_{i2} \\ y_{i2} \end{bmatrix} = \begin{bmatrix} x_i \\ y_i \end{bmatrix} + \begin{bmatrix} \cos(\theta_i) & -\sin(\theta_i) \\ \sin(\theta_i) & \cos(\theta_i) \end{bmatrix} \begin{bmatrix} \frac{H_i}{2} \\ -\frac{V_i}{2} \end{bmatrix} \tag{2.15}$$

$$\begin{bmatrix} x_{i3} \\ y_{i3} \end{bmatrix} = \begin{bmatrix} x_i \\ y_i \end{bmatrix} + \begin{bmatrix} \cos(\theta_i) & -\sin(\theta_i) \\ \sin(\theta_i) & \cos(\theta_i) \end{bmatrix} \begin{bmatrix} \frac{H_i}{2} \\ \frac{V_i}{2} \end{bmatrix} \tag{2.16}$$

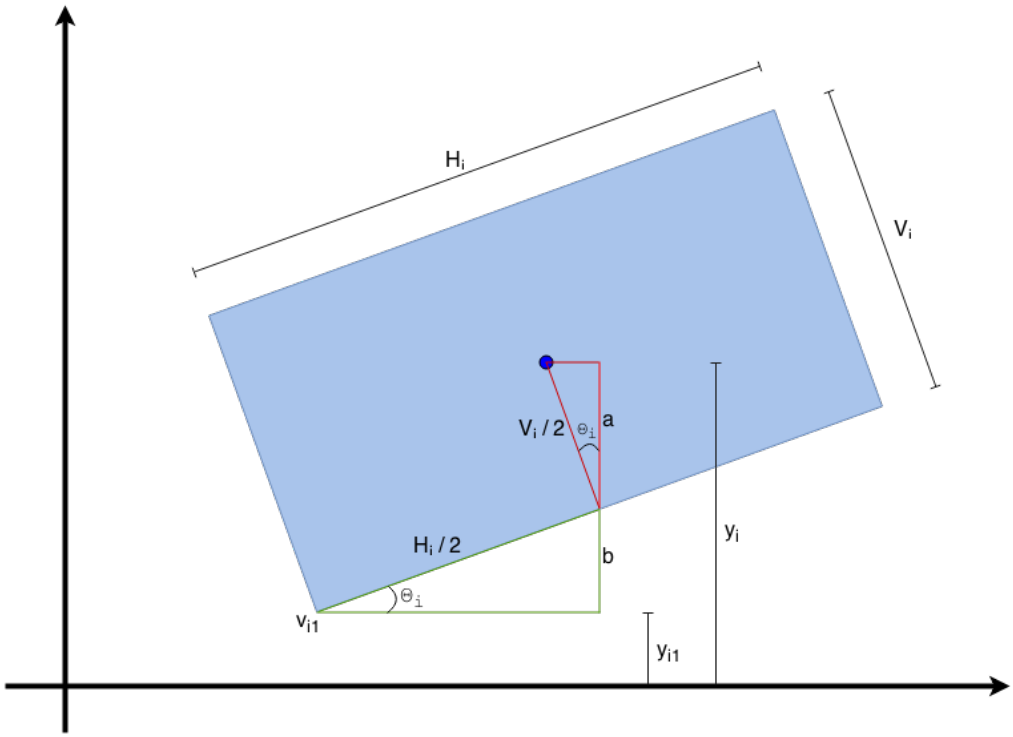


Figura 2.8 - Obtenção da ordenada do vértice v_{i1}

$$\begin{bmatrix} x_{i4} \\ y_{i4} \end{bmatrix} = \begin{bmatrix} x_i \\ y_i \end{bmatrix} + \begin{bmatrix} \cos(\theta_i) & -\sin(\theta_i) \\ \sin(\theta_i) & \cos(\theta_i) \end{bmatrix} \begin{bmatrix} -\frac{H_i}{2} \\ \frac{V_i}{2} \end{bmatrix} \tag{2.17}$$

2.2.4.2 Validação de alocação inteiramente no painel

Após identificar a posição dos vértices do objeto, é possível verificar se o objeto está inteiramente sobre a superfície de alocação. Este cálculo é feito em duas etapas, uma para o eixo horizontal e outra para o eixo vertical. Considerando o centro da Superfície S como sendo a origem do plano, basta calcular a distância euclidiana entre a abscissa do vértice mais extremo do objeto e o centro da superfície. E verificar se esta distância é menor que a metade do lado horizontal da superfície, como ilustrado na Figura 2.9.

A Equação 2.18 apresenta a verificação a ser feita para validar se o equipamento está inteiramente sobre a superfície da plataforma.

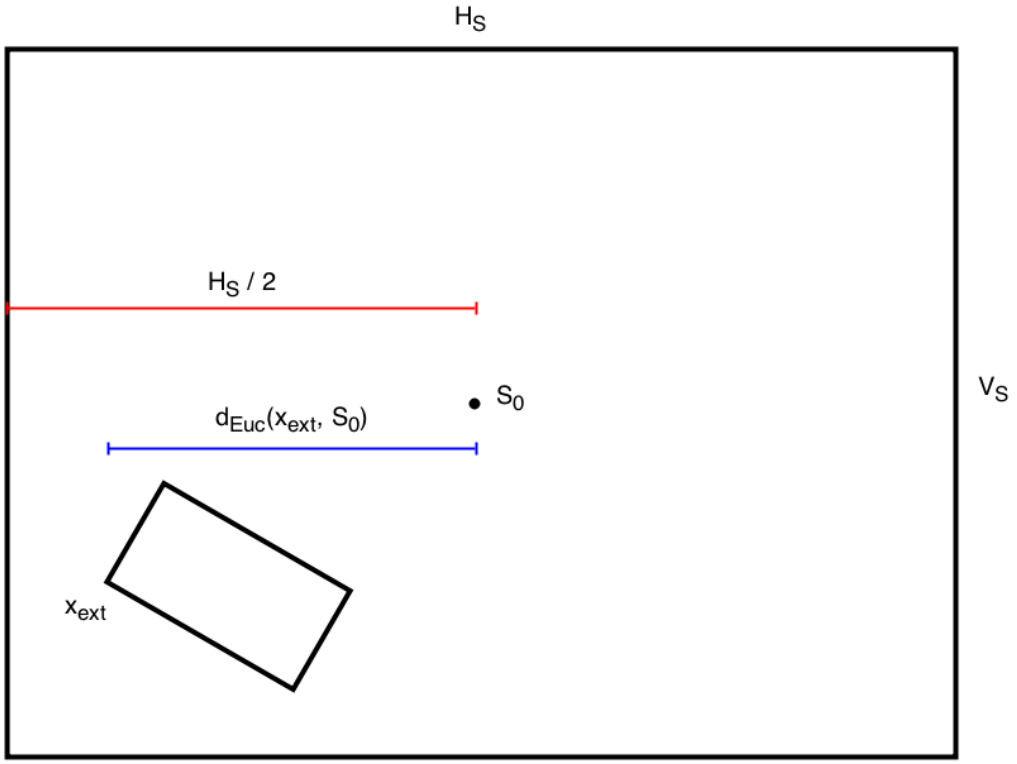


Figura 2.9 - Validação de posicionamento dentro da superfície

$$d_{Euc}(x_{ext}, S_0) < H_S/2 \tag{2.18}$$

Esta mesma verificação deve ser feita para o lado vertical.

2.2.4.3 Validação de sobreposição entre equipamentos

Para um layout ser considerado válido, além de verificar se todos os equipamentos estão inteiramente sobre a plataforma, também é necessário verificar se há sobreposição entre os equipamentos. Ou seja, dois objetos não podem ocupar o mesmo espaço.

Uma forma de fazer essa verificação entre dois objetos é tentar traçar uma linha entre eles, se for possível separá-los por uma linha significa que os mesmos não estão sobrepostos. Pode-se usar as próprias laterais dos objetos, encontrar a linha perpendicular a ela e projetar os vértices dos dois objetos sobre esta linha. Se as projeções dos vértices dos dois objetos forem separáveis significa que eles não estão

sobrepostos. A Figura 2.10 ilustra esta forma de verificar a sobreposição entre os objetos.

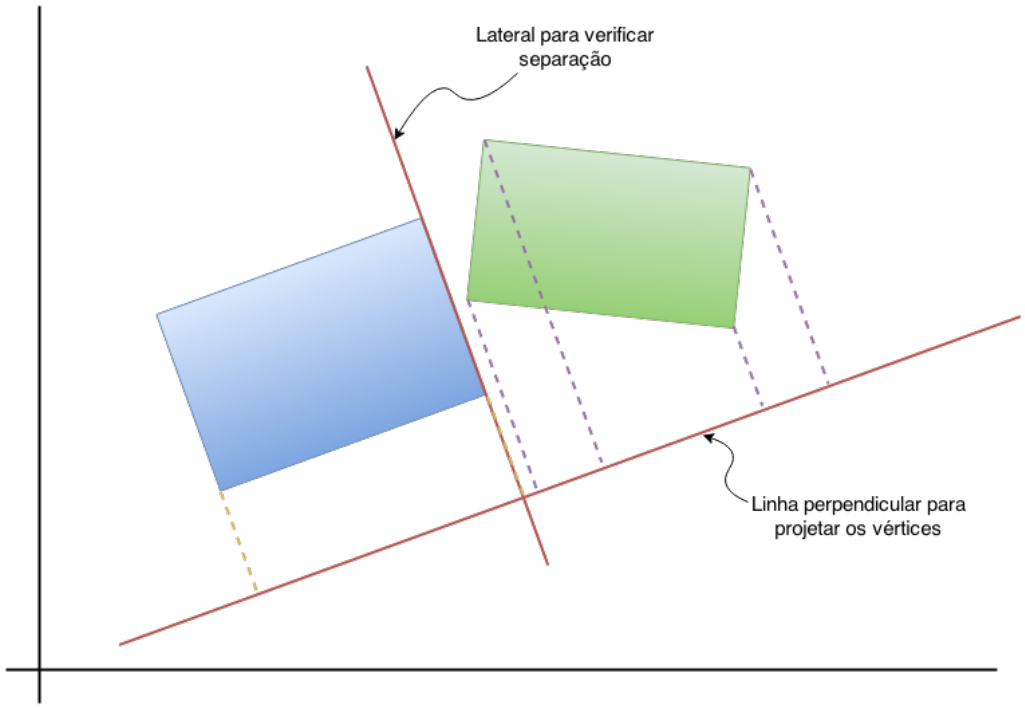


Figura 2.10 - Verificação de sobreposição entre objetos

Para criar uma linha perpendicular à linha lateral de um dos objetos, passando pelo ponto origem do plano, calcula-se o vetor normal aos dois vértices da lateral do objeto que está sendo verificada. O ponto normal de dois vértices $V_1(x_1, y_1)$ e $V_2(x_2, y_2)$ pode ser calculado conforme a Equação 2.19.

$$P_{Normal} = (y_2 - y_1, x_1 - x_2) \tag{2.19}$$

A Figura 2.11 ilustra a localização do vetor normal à linha formada por dois pontos determinados no plano cartesiano e a linha perpendicular que o a normal e a origem do plano formam em relação aos dois pontos que podem ser os vértices de um objeto.

Para fazer uma projeção dos vértices dos dois objetos (que estão sendo validados) na linha perpendicular formada pela origem e o ponto normal, basta somar a multiplicação de cada coordenada do ponto pela respectiva coordenada da normal, conforme

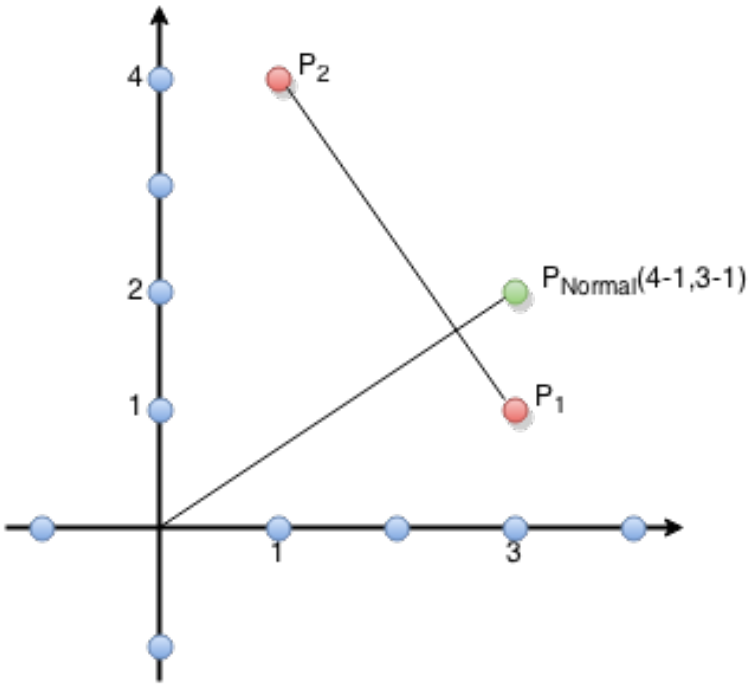


Figura 2.11 - Ilustração do ponto normal a dois pontos

a Equação 2.20.

$$Proj = x_k * x_{Normal} + y_k * y_{Normal} \tag{2.20}$$

Isso deve ser feito para cada lado dos dois objetos. Se em todas as verificações, as projeções dos vértices de cada objeto forem separáveis, os dois objetos não estão sobrepostos. A Figura 2.12 apresenta um método escrito na linguagem JAVA capaz de validar se dois polígonos a e b passados como parâmetro estão sobrepostos ou não usando a lógica apresentada.

2.2.4.4 Restrições extras e opinião do especialista

Muitas situações devem ser consideradas durante o desenvolvimento do layout de um satélite. Integração entre subsistemas, aquecimento e interferências gerados pela proximidade de equipamentos eletrônicos que compõem o módulo, equipamentos que já devem estar alocados de forma fixa em algum local, entre outros.

Para considerar estas situações, algumas restrições extras devem ser atendidas para

```java
private static boolean isPolygonsIntersecting(double[][] a, double[][] b) {
    Double minA, maxA, projected, minB, maxB;
    for (double[][] polygon : new double[][][] { a, b }) {
        for (int i1 = 0; i1 < polygon.length; i1++) {
            int i2 = (i1 + 1) % polygon.length;
            double[] p1 = polygon[i1];
            double[] p2 = polygon[i2];
            double[] normal = new double[] { p2[1] - p1[1], p1[0] - p2[0] };
            minA = maxA = null;
            for (int j = 0; j < a.length; j++) {
                projected = normal[0] * a[j][0] + normal[1] * a[j][1];
                if (minA == null || projected < minA) {
                    minA = projected;
                }
                if (maxA == null || projected > maxA) {
                    maxA = projected;
                }
            }
            minB = maxB = null;
            for (int j = 0; j < b.length; j++) {
                projected = normal[0] * b[j][0] + normal[1] * b[j][1];
                if (minB == null || projected < minB) {
                    minB = projected;
                }
                if (maxB == null || projected > maxB) {
                    maxB = projected;
                }
            }
            if (maxA < minB || maxB < minA) {
                return false;
            }
        }
    }
    return true;
}
```

Figura 2.12 - Validação de sobreposição entre polígonos escrito em JAVA

a solução ser considerada válida. Dois objetos que geram muito aquecimento a ser dissipado ou que interferem no funcionamento um do outro, por exemplo, devem ficar a uma distância mínima um do outro. Também alguns equipamentos de um mesmo subsistema devem ficar a uma distância máxima entre eles para facilitar o cabeamento. Pode ser necessário também, fixar a posição de alguns equipamentos que não devem ser movidos ao criar novas soluções.

A plataforma proposta neste trabalho, possibilita o acréscimo dessas informações *a priori* por um especialista. O caminho para encontrar o layout inicial ideal do satélite pode acontecer intercalando a execução dos algoritmos da plataforma e a

inserção desas informações especialistas.

3 OTIMIZAÇÃO

Este capítulo apresenta os conceitos-base sobre problemas de otimização, heurísticas e meta-heurísticas, descreve os algoritmos implementados neste trabalho e uma heurística desenvolvida como fruto deste trabalho.

3.1 Problemas de Otimização

Problemas de otimização consistem em achar a melhor combinação dentre um conjunto de variáveis para maximizar ou minimizar uma função, geralmente chamada de função objetivo ou função custo. Esses problemas podem ser divididos em três categorias: aqueles cujas variáveis assumem valores reais (ou contínuos), aqueles cujas variáveis assumem valores discretos (ou inteiros) e aqueles em que há variáveis inteiras e contínuas, classificados, respectivamente, como problemas de Otimização Contínua, Otimização Combinatória ou Discreta, e Otimização Mista (BECCENERI et al., 2008). O problema tratado neste trabalho é contínuo.

Matematicamente, otimizar um problema, ou conjunto de equações, significa encontrar um valor, ou conjunto de valores ótimos, que garantam um resultado mínimo ou máximo (LUZ, 2012). Uma função a ser minimizada ou maximizada pode ter vários ótimos locais e um único ótimo global conforme apresentado na Figura 3.1. Um ótimo local é o valor mínimo ou máximo de uma sub-região do espaço de interesse, enquanto o ótimo global é o valor mínimo ou máximo do espaço de busca total.

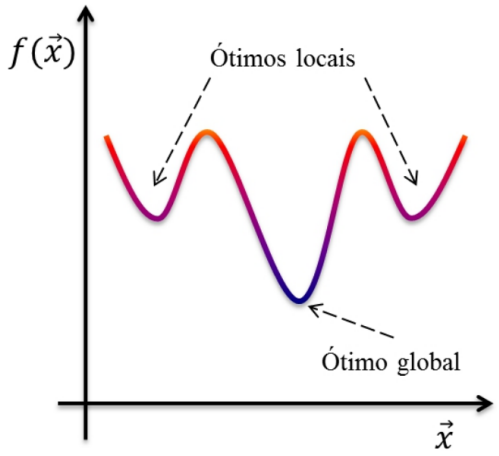


Figura 3.1 - Exemplo de ótimos locais e global para minimizar uma função

Fonte: Luz (2012).

A natureza da função influencia diretamente a escolha da estratégia de busca pelo ótimo global. Problemas de otimização contínua podem se tornar muito complexos como podemos observar nas funções de Rastriginm, Schwefel e Griewangk apresentada na Figura 3.2.

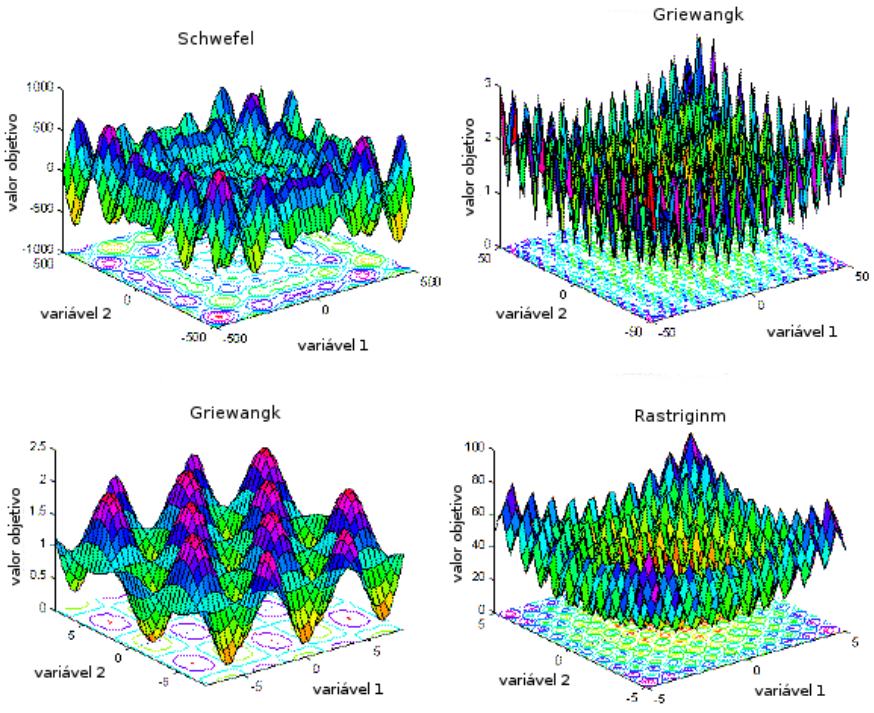


Figura 3.2 - Exemplo de funções de Rastriginm, Schwefel e Griewangk

Fonte: Pohlheim (2006).

3.2 Heurísticas e meta-heurísticas

O nome heurística é derivado da palavra grega *heuriskein*, que significa descobrir. Hoje, este termo é usado para descrever um método "que, baseado na experiência ou julgamento, parece conduzir a uma boa solução de um problema, mas que não garante produzir uma solução ótima". Este termo pode ser considerado como associado a um conhecimento circunstancial, não verificável matematicamente. (BECCENERI; SILVA NETO, 2009)

O prefixo "meta", em grego, significa "após", indicando um nível superior de desco-

berta. Uma meta-heurística é uma estratégia de busca, não específica para um determinado problema, que tenta explorar eficientemente o espaço das soluções viáveis deste problema. São algoritmos aproximados que incorporam mecanismos para evitar confinamento em mínimos ou máximos locais. (BECCENERI; SILVA NETO, 2009)

Uma boa definição para meta-heurísticas dada por Floudas e Pardalos (2001) diz que meta-heurística é um processo mestre iterativo que guia e modifica as operações de heurísticas subordinadas para eficientemente produzir soluções de alta qualidade. Ela pode manipular uma solução completa (ou incompleta) ou um conjunto de soluções a cada iteração.

As meta-heurísticas fazem um balanceamento dinâmico entre estratégias de buscas horizontais (largura) e verticais (profundidade), ou *exploração* e *intensificação*, do inglês *exploration* e *exploitation*. Um dos desafios na aplicação de meta-heurísticas é encontrar um equilíbrio ideal entre estes dois tipos de busca. Essa característica de balanceamento é importante para escapar de mínimos e máximos locais.

3.2.1 Classificação das Meta-heurísticas

Segundo Blum e Roli (2003), as meta-heurísticas podem ser classificadas de acordo com características básicas como:

- Inspirados na Natureza vs Não Inspirados na Natureza
- Baseado em população vs Busca de ponto único
- Função Objetivo Dinâmica ou Estática
- Estrutura de vizinhança única ou variável
- Métodos com ou sem uso de memória

Uma possível forma de classificar meta-heurísticas é baseando-se em sua origem, se inspiradas na natureza, como os *Algoritmos Genéticos* e os *Algoritmos das Formigas*, ou não-inspiradas na natureza, como as meta-heurísticas *Tabu Search* e *Iterated Local Search*. As meta-heurísticas implementadas neste trabalho são classificadas como inspiradas na natureza, sendo o ACO bio-inspirada.

Outra classificação muito usada refere-se ao comportamento de busca da meta-heurística. No método construtivo ou de trajetória, parte-se de um conjunto solução vazio e segue-se acrescentando elementos até obter uma solução viável para

o problema. Já os métodos populacionais partem de um conjunto de soluções iniciais (primeira geração) e seguem-se alterando os elementos deste conjunto a fim de encontrar soluções melhores. (LUZ, 2012)

3.3 Otimização por Colônia de Formigas - ACO

Cada vez mais os pesquisadores voltam um olhar para a natureza na tentativa de aprender como solucionar problemas reais. Muitos insetos, que individualmente têm comportamentos extremamente simples, são capazes de executar atividades extremamente complexas trabalhando em grupo. A capacidade de auto-organização em estruturas complexas chama a atenção de pesquisadores, pois não se identifica um coordenador para as atividades a serem executadas pelo grupo. A auto-organização resulta de comportamentos surpreendentemente simples executados por indivíduos possuindo apenas informação local (BECCENERI; SILVA NETO, 2009).

O método de Otimização por Colônia de Formigas (*Ant Colony Optimization*, ACO), proposto por Dorigo (1992), é um bom exemplo de algoritmo bio-inspirado. Originado da observação da capacidade das colônias de formigas em encontrar o menor caminho entre o ninho e fonte de comida, este algoritmo utiliza o método construtivo ou de trajetória para encontrar soluções otimizadas. Como é um tipo de inseto que vive em colônia, uma única formiga tem um comportamento muito simples, porém com a cooperação de todas as formigas, a colônia executa tarefas complexas.

Podemos descobrir como uma colônia de formigas pode encontrar o melhor caminho para uma fonte de comida observando cada formiga individualmente. Cada formiga da colônia, ao caminhar em busca de alimento, deposita em seu caminho uma substância chamada feromônio que exala um odor próprio. E ao decidir qual caminho percorrer a formiga leva em consideração o feromônio deixado pelas outras formigas que passaram por cada caminho anteriormente, como um caminho potencialmente bom. Desta forma, quanto mais formigas percorrem um caminho, mais atrativo ele se torna para as próximas formigas. Com o tempo o feromônio evapora, com isso um caminho muito percorrido no passado pode não ser mais tão atrativo no futuro, se um outro caminho (provavelmente mais curto) receber mais feromônio.

O que se nota na natureza é que, após um determinado tempo, a maioria das formigas decide trilhar um caminho mais curto (possivelmente o mais otimizado) entre o ninho e a fonte de alimento (BECCENERI; SILVA NETO, 2009). A Figura 3.3 mostra um exemplo de como as formigas encontram o melhor caminho até a fonte de comida quando um obstáculo é adicionado ao caminho. Com o passar do tempo, o caminho

mais curto tende a ser mais atrativos para todas as formigas. Embora, esporadicamente uma formiga pode escolher um caminho completamente novo, mesmo que não tenha nenhum feromônio.

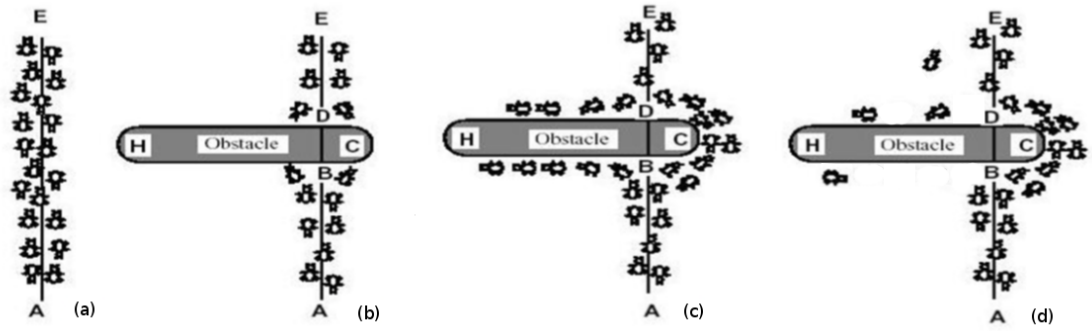


Figura 3.3 - Melhor caminho encontrado por um grupo de formigas entre o ninho (A) e a fonte de alimento (E) quando adicionado um obstáculo

Fonte: Souto et al. (2009).

Baseado neste comportamento das formigas na hora de decidir o caminho a trilhar, Dorigo (1992) fez uso de um mecanismo conhecido como reforço positivo para simular o feromônio depositado pelas formigas no caminho percorrido, também é implementado o reforço negativo para simular a evaporação do feromônio na natureza ao longo do tempo. Outro aspecto importante deste método é o comportamento cooperativo. A cada iteração do algoritmo, cria-se uma geração de formigas que trabalha simultaneamente criando diferentes soluções. As formigas que encontram as melhores soluções influenciam a exploração das futuras gerações de formigas através dos métodos utilizados para simular o depósito e evaporação do feromônio nos caminhos.

3.3.1 Descrição do Algoritmo

Existe hoje uma classe de algoritmos conhecida genericamente como "Algoritmo das Formigas", baseadas no primeiro algoritmo proposto por (DORIGO, 1992), com diferentes parâmetros e procedimentos. Há, por exemplo, várias estratégias para o depósito de feromônio virtual. Em geral, adota-se uma de duas possibilidades: ou todas as formigas depositam feromônios nos caminhos por elas percorridos, ou

apenas a melhor formiga de cada geração, algoritmos estes conhecidos como ACS (Ant Colony System) e ACO (Ant Colony Optimization), respectivamente.

A seguir, descrevemos o ACO (DORIGO et al., 1999), considerando o problema de percorrimento mínimo de um grafo direcionado acíclico. O problema consiste em encontrar o melhor caminho entre um nó inicial N_0 e um nó final N_f pré-estabelecidos, considerando que os arcos entre cada dois nós i e j estão associados a um custo $c(i, j)$. Cada formiga k estando no nó i calcula sua probabilidade de ir para o nó j através da Equação 3.1.

$$p_k(i,j) = \frac{\tau(i,j)^\alpha \cdot \eta(i,j)^\beta}{\sum_{l \in C^i} \tau(i,l)^\alpha \cdot \eta(i,l)^\beta} \tag{3.1}$$

Onde:

- $\tau(i, j)$ é a quantidade de *feromônio* entre os nós i e j;
- $\eta(i, j)$ é a informação heurística de atração entre os nós i e j. Este parâmetro é chamado de *desejabilidade* e no problema do percorrimento do grafo, é o inverso do custo associado ao arco entre os nós i e j;
- C^i é o conjunto de nós ainda não visitados pela formiga k;
- Os parâmetros α e β controlam o peso relativo entre feromônio e desejabilidade na decisão do caminho a ser seguido pela formiga.

Após o término de cada iteração, acontece a atualização do feromônio depositado nos arcos em todas as soluções. Baseado em uma regra pré-estabelecida, o feromônio de todos os caminhos evapora de acordo com uma constante de evaporação ρ, e as melhores soluções recebem um incremento de feromônio. A Equação 3.2 pode ser utilizada para esta atualização, mas outras estratégias para a atualização da quantidade de feromônio nos caminhos podem ser definidas. Uma discussão ampla sobre o assunto pode ser encontrada em Blum (2005).

$$\tau(i,j) = (1-\rho) \cdot \tau(i,j) + \sum_{k=1}^{m} \Delta\tau_k(i,j) \tag{3.2}$$

onde $\Delta\tau_k(i, j)$ é o incremento de *feromônio* se o arco (i, j) é percorrido pela formiga k.

No algoritmo ACO, utiliza-se também uma variável aleatória q, distribuída uniformemente entre $[0, 1]$, e um parâmetro de ajuste $q_0 \in [0, 1]$. Cada vez que uma formiga k está para escolher o próximo nó para visitar, um número q é sorteado entre 0 e 1. Se $q \leq q_0$, seleciona-se o arco que obteve o maior valor de probabilidade, segundo a Equação 3.1, privilegiando a memória armazenada na matriz de feromônio. Caso contrário, um arco é escolhido aleatoriamente, segundo a distribuição de probabilidades calculada na Equação 3.1. Obviamente, neste caso, o arco com maior probabilidade tem mais chance de ser escolhido que os demais. No algoritmo AS original, $q_0 = 0$ e todas as formigas depositam feromônio.

Os passos para a implementação do ACO para um problema de otimização são apresentados a seguir.

Passo 1: Inicialização do Sistema.

- Calcula-se a desejabilidade de cada nó.
- Define-se a quantidade de feromônio de cada arco com um valor τ_0 (por exemplo $1/N$).
- Distribui-se as formigas de modo aleatório entre os nós.

Passo 2: Para cada iteração (geração de formigas).

- Cada formiga percorre todos os nós do grafo, calculando, em cada nó, o próximo nó, conforme a Equação 3.1.

Passo 3: Verificar qual formiga obteve a melhor solução.

Passo 4: Atualizar a matriz de feromônio.

- Incrementa-se a quantidade de feromônio na melhor rota (depósito de feromônio);
- Decrementa-se a quantidade de feromônio em todas as rotas (evaporação).

3.3.2 Aplicação do ACO ao problema de alocação de equipamentos em satélite

Em Xu et al. (2010) é proposta a utilização do ACO em conjunto com a Técnica de Posicionamento Baseado em Ordenação (*Order-based Positioning Technique*, OPT).

A ideia principal do OPT é encontrar um layout mínimo a cada novo objeto acrescentado em uma solução parcial. Isso inclui aproximar o novo objeto dos demais já reunidos pela técnica, ou afastar o objeto, caso este tenha sido posicionado sobre o conjunto de objetos já alocados. Além disso, se o objeto for retangular, este é sempre rotacionado para um ângulo reto com a solução já encontrada, visando minimizar espaços vazios. A Figura 3.4, apresenta uma solução (a) gerada por um algoritmo de otimização que não utiliza a técnica OPT, e uma solução (b) onde a técnica OPT foi aplicada.

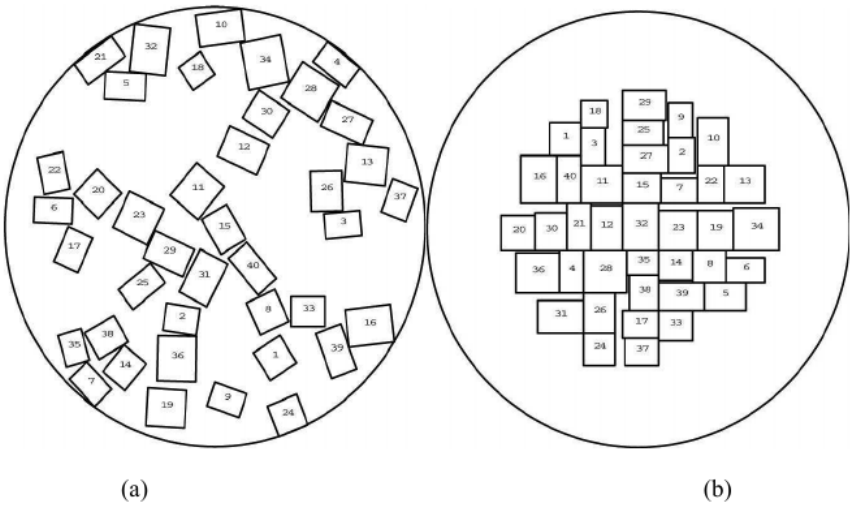


Figura 3.4 - Layouts gerados sem (a) e com (b) a utilização da técnica OPT. Objetivo: minimizar o centro de massa e o raio da solução. (XU et al., 2010)

O OPT apresentou prós e contras, mas gerou bons resultados para o problema discutido em Xu et al. (2010), cujos objetivos eram minimizar o envelope formado pelos objetos e minimizar a localização do centro de massa em relação ao centro do envelope. Também podemos observar que a heurística OPT pode ser considerada uma busca gulosa.

A metodologia para o uso do ACO no problema de alocação de equipamentos em satélites adotada neste trabalho é a mesma proposta em Xu et al. (2010), onde o ACO é aplicado na escolha da ordem em que os equipamentos são alocados. No problema de percorrimento de grafo, cada formiga deve encontrar uma sequência de nós para visitar. Aqui, no problema de alocação de equipamentos, a formiga deve encontrar uma boa ordem para alocar os equipamentos em uma solução candidata.

A implementação do ACO para o POL, denotado por ACO', é ilustrado na Figura 3.5.

```
Main_ACO'()
  Inicializa os parâmetros e a matriz de feromônio.
  Para n=0 até # de iterações
    Para k=0 até # de formigas
      Enquanto houver objetos a serem alocados pela formiga
        Determine o próximo objeto com probabilidade p_k(i,j)
        Aloque o objeto em uma posição aleatória na solução
      Fim-Enquanto
    Fim-Para
    Atualiza o feromônio na sequência de objetos
      usado pela melhor formiga
  Fim-Para
Fim
```

Figura 3.5 - Pseudo-código do ACO' (ACO para o POL)

3.4 Algoritmo de Colisão de Múltiplas Partículas - MPCA

O algoritmo de colisão de múltiplas partículas (*Multiple Particle Collision Algorithm, MPCA*) foi proposto por Luz (2012) como uma variante populacional do algoritmo de colisão de partículas (*Particle Collision Algorithm, PCA*) apresentado por Sacco e Oliveira (2005).

O PCA foi inspirado na física das reações de colisão de partículas em um reator nuclear, especialmente nos comportamentos de espalhamento (*scattering*) e absorção (*absorption*). A ideia principal do algoritmo é que, considerando por exemplo problemas de maximização, quando a partícula (solução candidata) encontra núcleos com valores altos para a função objetivo, esta é absorvida e a região próxima da partícula é explorada, por outro lado, quando a partícula encontra uma região de baixos valores para função objetivo, ela é espalhada para outra região do espaço de busca. Isso permite que o espaço de busca do problema seja amplamente explorado e as regiões mais promissoras recebam uma atenção maior do algoritmo.

Assim como a técnica de recozimento simulado (*Simulated Annealing - SA*), o PCA evolui com a exploração do espaço de busca partindo de uma única solução inicial, que sofre uma perturbação estocástica para gerar novas soluções. A qualidade das

novas soluções potenciais são avaliadas para determinar se serão absorvidas ou espalhadas. Caso a nova solução seja melhor que aquela antiga, a partícula é absorvida e ocorre uma exploração da vizinhança. Se a qualidade da nova solução for pior que a solução antiga, a partícula é espalhada para outra região do espaço de busca.

A grande vantagem do PCA está na quantidade de parâmetros requerida pelo algoritmo, enquanto o SA possui vários parâmetros altamente sensíveis, o PCA possui muito menos.

Basicamente, o MPCA introduz um loop de alto nível ao PCA para tratar cada partícula e um controle de processamento paralelo para manter a comunicação entre elas. (LUZ et al., 2008) As partículas do MPCA colaboram entre si compartilhando a solução de melhor valor para a função objetivo através da estratégia de *blackboard*. Um pseudo-código do PCA e do MPCA é apresentada na Figura 3.6, ilustrando como cada um trata a partícula.

As funções *Exploration()* e *Scattering()* utilizadas neste trabalho possuem uma diferença sutil para o caso da utilização conjunta do MPCA com a heurística discutida na próxima sessão. Aqui, de acordo com uma probabilidade, a função *Scattering* cria uma nova solução aleatória, enquanto ao utilizar a HBAE esta nova solução não é completamente aleatória. A Figura 3.7 apresenta o pseudo-código das funções *Exploration()* e *Scattering()* ilustrando a geração de uma solução aleatória do algoritmo original.

Durante a execução do PCA as soluções iniciais sofrem modificações estocásticas através das funções *Perturbation()* e *SmallPerturbation()* a fim de gerar novas soluções. Para aplicar o MPCA ao problema de otimização de layout, estas funções foram usadas para simular uma "chacoalhada" na solução, alterando a posição de cada equipamento em 5% e 1% respectivamente em relação ao tamanho da superfície de alocação.

O MPCA foi originalmente implementado usando a biblioteca MPI em um ambiente computacional de alto desempenho. Aqui, como a plataforma foi criada utilizando a linguagem de programação Java, utilizou-se Threads para gerenciar o processamento paralelo das partículas.

3.5 Heurística de Balanceamento

Uma forma de aplicar o ACO em problemas de otimização de layouts é otimizar a ordem em que os objetos são alocados. Em Xu et al. (2010) foi proposta a aplicação

```
Main_PCA()
  Gera uma solução inicial em Old_Config
  Best_Fitness = Fitness(Old_Config)
  Para n=0 até # de iterações
    Treat_particle()
  Fim-Para
Fim

Main_MPCA()
  Gera uma solução inicial em Old_Config
  Best_Fitness = Fitness(Old_Config)
  Atualiza Blackboard
  Para n=0 até # de iterações
    Para p=0 até # de partículas
      Treat_particle()
    Fim-para
    Atualiza Blackboard
  Fim-Para
Fim

Treat_particle()
  Perturbation()
  Se Fitness(New_Config) > Fitness(Old_Config) Então
    Se Fitness(New_Config) > BestFitness Então
      Best_Fitness = Fitness(New_Config)
    Fim-Se
    Old_Config = New_Config
    Exploration()
  Senão
    Scattering()
  Fim-Se
Retorno
```

Figura 3.6 - Pseudo-código do PCA e do MPCA

do ACO em conjunto com a técnica de posicionamento baseado em ordenação (OPT) para resolver o POL. Enquanto o ACO busca otimizar a ordem em que os objetos são alocados, o OPT visa construir um layout minimamente viável com cada objeto que é adicionado na solução em uma posição aleatória. Apesar de esta técnica produzir bons resultados para o POL, resultados obtidos em Alves et al. (2013a) mostraram que, para o SMLDP, a ordem em que os objetos são alocados tem menor importância na otimização dos objetivos que efetivamente influenciar a escolha da posição de cada

```
Exploration()
  Para n=0 até # de iterações
    Small_Perturbation()
    Se Fitness(New_Config) > Fitness(Old_Config) Então
      Se Fitness(New_Config) > Best_Fitness Então
        Best_Fitness = Fitness(New_Config)
      Fim-Se
      Old_Config = New_Config
    Fim-Se
  Fim-Para
Retorno

Scattering()
  Pscattering = 1 - Fitness(New_Config) / Best_Fitness
  Se Pscattering > Random(0,1) Então
    Old_Config = Get_New_Solution()
  Senão
    Exploration()
  Fim-Se
Retorno

Get_New_Solution()
Retorna Solução aleatória
```

Figura 3.7 - Pseudo-código das funções *Exploration()* e *Scattering()* do PCA/MPCA

equipamento na solução.

É apresentada aqui uma heurística para otimizar a alocação de um conjunto de objetos O em uma superfície S (prateleiras, painéis, paredes, plataformas, etc.). A ideia desta heurística, denominada *Heurística de Balanceamento por Afinidade Espacial* (HBAE), é influenciar o posicionamento de cada objeto ao ser adicionado na solução, através da afinidade entre este e os demais objetos já posicionados na superfície. Ao logo das iterações a afinidade entre os objetos é ajustada considerando a qualidade da solução para a função objetivo.

3.5.1 Heurística de Balanceamento por Afinidade Espacial - HBAE

A HBAE pode ser usada isoladamente para o SMLDP, ou em conjunto com outros algoritmos de otimização. Neste trabalho a heurística foi aplicada em sua forma isolada, e híbrida associada ao ACO e o MPCA. As duas formas são descritas a

seguir. Inicialmente é dada a notação básica e alguns parâmetros utilizados, então o processamento básico e a associação desta heurística ao ACO e ao MPCA.

3.5.1.1 Notação básica

- Seja S uma superfície retangular em $\mathbb{R}^2$.
- Seja $Q = \{Q_1, ..., Q_m\}$ um conjunto finito de regiões que particiona S, i.e. $S = \cup_i Q_i$ e $\forall i, j, i \neq j, Q_i \cap Q_j = \emptyset$
- Seja $q : S \rightarrow \{1, ..., m\}$ a função que mapeia cada região em S para sua respectiva região relativa em Q: $\forall s = (x, y) \in S, q(s) = j, se(x, y) \in Q_j$
- Seja $\Pi = \{\pi_1, ..., \pi_z\}$ o conjunto de termos que descrevem as posições relativas entre as regiões (i.e mesma região, região oposta, lateral)
- Seja $O = \{o_1, ..., o_n\}$ um conjunto de objetos
- Seja R uma matriz m x m que contém termos Π que descrevem a posição relativa de dois objetos O em regiões Q.
- Seja $O' \subseteq O$ denota o conjunto de objetos já alocados em S e $O'' \subseteq O$ o conjunto de objetos não alocados em S, em dado momento do processamento.
- Seja $G = (O, E), E = O^2$ um grafo completo unidirecional em que os nós são O. Cada arco $(o_i, o_j) \in E$ é anexa a um vetor v_{ij} de z posições, que representa a afinidade espacial entre os objetos o_i e o_j.
- Seja c_i um vetor de m posições, que representa a afinidade resultante entre o objeto o_i e cada região em Q.
- Seja $p : Q'' \rightarrow S$ uma função que identifica a posição de um objeto alocado.

3.5.1.2 Parametrização

Para a aplicação da heurística ao problema discutido neste trabalho, os seguintes parâmetros foram considerados para divisão de Q e definição das regiões relativas:

- $m = 4$, com $Q = \{Q_1, Q_2, Q_3, Q_4\}$ (vide Figura 3.8)
- $z = 3$, com $\Pi = \{$*mesmo lado, lado oposto, lateral*$\}$ (vide Figura 3.8)

- $\forall i, j \in \{1, ..., m\}, R(i, j) = R(j, i)$
- $\forall i \in \{1, ..., m\}, R(i, i) = \textit{mesmo lado}$, $R(i, i+2) = \textit{lado oposto}$ $R(i, i+1)$ ou $R(i, i+3) = \textit{lateral}$ (vide Figura 3.8)

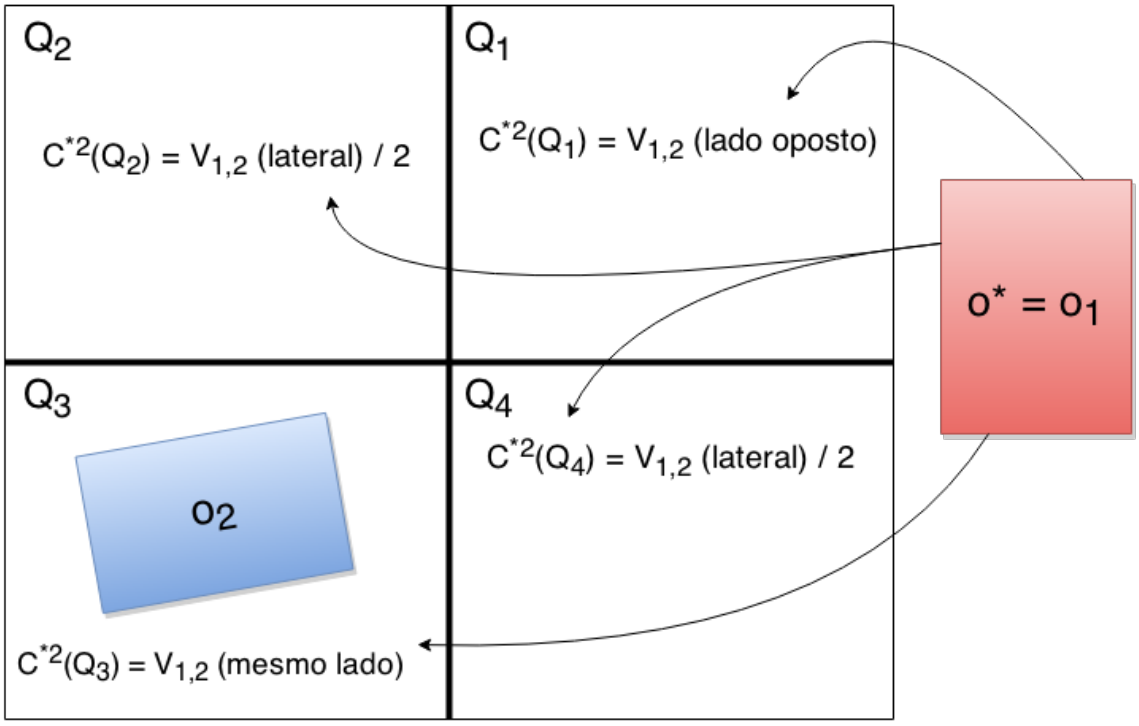


Figura 3.8 - Ilustração da alocação de um novo objeto em uma solução incompleta.

3.5.1.3 Processamento básico

- Associa-se um vetor v_{ij} com z posições a cada arco $(o_i, o_j) \in E$.
- Seja $o^* = o_i$ um objeto de O' a ser adicionado em S. Um vetor c^{*j} com m posições é criado para representar a afinidade entre o^* e cada região em Q.
- Seja $o_j \in O''$ um objeto alocado em uma região $Q^a \in Q$ em S, atualiza-se o vetor c^{*j} com as afinidades entre o^* e o_j (vetor v_{ij}), fazendo a correspondência entre posições relativas em Π com a atual região Q^a de O_j.
- Um vetor c^* é gerado a partir do conjunto de vetores c^{*j}, de acordo com a função f (Veja a seguir).

- Seja g a função usada para selecionar a região $Q^* \in Q$ onde o^* será alocado. A região Q^* é selecionada aleatoriamente de acordo com a probabilidade de cada região, considerando o vetor c^*. Quanto maior valor da região $Q_k \in Q$ em c^*, maior a probabilidade de Q_k se tornar Q^* selecionado.

- Finalmente, o objeto o^* é adicionado em S em uma posição aleatória da região Q^*. Caso após N tentativas de alocação o objeto não tenha sido inserido na região Q^* selecionada, gerando uma solução válida (conforme validações definidas no capítulo 2 desta dissertação), este objeto poderá então ser alocado em qualquer posição aleatória de S.

- Há duas versões para a função f (responsável por gerar o vetor c^*) que mantém as probabilidades de alocação de o^* em cada região em $Q_k \in Q$. Na versão I, um objeto $O^{ref} \in Q''$ é selecionado aleatoriamente e $c^* = c^{ref}$, considerando a função q. Na versão II, c^* corresponde ao somatório de todos os vetores c^{*j} relativos as relações entre o^* e todos os objetos $o_j \in O''$: $\forall Q_k \in Q, c^*(Q_k) = \sum_{o_j \in O''} c^{*j}(Q_k)$.

- Para criar um vetor c^{*j} de m posições (regiões em Q), a partir do vetor de afinidades v_{ij} com z posições (termos em Π), os termos de v_{ij} são relacionados à regiões Q através da matriz R. Por exemplo, suponhamos alocar o objeto $o^* = o_1$ em S, considerando que o objeto o_2 já esteja alocado na região Q_3 (vide Figura 3.8). Neste caso $o_2 \in O''$ e $q(p(o_2)) = Q_3$. Agora suponhamos que afinidade espacial entre o_1 e o_2 seja $v_{12}(\textit{mesmo lado}) = 10$, $v_{12}(\textit{lado oposto}) = 20$ e $v_{12}(\textit{lateral}) = 30$. Como $R(3,3) = \textit{mesmo lado}$ e $v_{12}(\textit{mesmo lado}) = 10$, temos que $c^{*2}(Q_3) = 10$. Do mesmo modo, obtemos $c^{*2}(Q_1) = 20$, pois $R(1,3) = \textit{lado oposto}$ (vide Figura 3.8). Para determinar $c^{*2}(Q_2)$ e $c^{*2}(Q_4)$ de forma justa, considerando que tanto Q_2 quanto Q_4 são vizinhos laterais de Q_3, neste trabalho foi tomada a decisão de dividir valores de $v_{ij}(\textit{lateral})$ $\forall o_j \in O''$ pela quantidade de regiões laterais de $q(p(o_j))$, pois sempre haverá duas regiões laterais em Q. Logo, usando esta regra, temos $c^{*2}(Q_2) = c^{*2}(Q_4) = v_{12}(\textit{lateral})/2 = 15$.

- Quando $Q' = \emptyset$, temos uma solução completa.

3.5.1.4 Processamento com múltiplos agentes

- No processamento básico, consideramos a criação de uma solução simples utilizando as afinidades espaciais entre os objetos, através dos vetores

v_{ij}. Entretanto, o objetivo desta heurística é aprender a criar os melhores layouts otimizando a afinidade espacial entre os objetos. Para isso, a heurística foi idealizada para ser utilizada com um conjunto de "agentes" originado de um algoritmo de otimização $\mathcal{A}$, criando soluções através do processamento básico descrito acima. Estes agentes dependem do algoritmo de otimização $\mathcal{A}$ utilizado. Podemos considerar, por exemplo, as formigas do ACO ou as partículas do MPCA como agentes HBAE. Quando o HBAE é utilizado sozinho, a quantidade de agentes deve ser definida.

- Inicialmente todas as afinidades são inicializadas com uma constante ϕ, i.e., $\forall o_i, o_j \in O$, $\forall \pi \in \Pi$, $v_{ij}(\pi) = \phi$.

- Os agentes compartilham as afinidades espaciais entre cada par de objetos. A cada iteração do algoritmo, todos os agentes geram uma solução baseado no processamento básico e, ao final da iteração, a solução que gerar melhor valor para a função objetivo é utilizada para ajustar as afinidades entre os equipamentos da seguinte maneira: Consideremos $sol^t = (p_1, ..., p_n)$ a melhor solução encontrada na t-ésima iteração de $\mathcal{A}$, onde p_i denota a posição do objeto o_i em S na solução sol^t. Ao final da t-ésima iteração, cada vetor v_{ij}, que mantém a afinidade espacial entre os objetos o_i e o_j, é atualizado conforme a expressão:

$$\forall \pi \in \Pi, v_{ij}^{t+1}(\pi) = \begin{cases} v_{ij}^t(\pi) \times (1+\gamma), & \text{se } \pi = R(q(p_i), q(p_j)) \\ v_{ij}^t(\pi), & \text{caso contrário} \end{cases}$$

  O parâmetro $\gamma \in [0, 1]$ (taxa de reforço/decaimento), fixado pelo usuário, é um balanceador entre velocidade e qualidade: valores baixos tendem a deixar o processo mais lento, enquanto valores altos podem resultar uma convergência prematura para ótimos locais.

3.5.1.5 HBAE puro *vs* hibridizações

- Quando usado *per se*, a cada iteração um conjunto de agentes simples cria soluções em paralelo, utilizando a heurística. A Figura 3.9 apresenta um pseudo-código do HBAE-puro.

- Quando usado com o ACO, as formigas são os agentes; ao final de cada iteração, além de atualiza a matriz de feromônios, a melhor formiga também atualiza a matriz de afinidades do HBAE. Após o ACO determinar o próximo objeto a ser alocado, o HBAE é usado para determinar a posição deste. A Figura 3.10 apresenta este algoritmo híbrido em pseudo-código.

- Quando usado com o MPCA, as partículas são os agentes; ao final de cada iteração, a matriz de afinidades do HBAE é atualizada pela melhor partícula. Sempre que uma partícula quer gerar uma nova solução (no início ou na função *Scattering*), a nova solução é gerada utilizando o método HBAE, como pode ser visto no pseudo-código da Figura 3.11.

```
HBAE_Puro()
  Inicializa a matriz de afinidades V com valores iguais para cada afinidade.
  Versão_HBAE = Configurar a versão da heurística (1 ou 2)
  Para n = 0 até # de iterações
    Para m = 0 até # de agentes
      Treat_agent()
    Fim-Para
  Fim-Para
Retorno

Treat_agent()
  Mistura a ordem dos objetos a serem alocados (vetor O'[m])
  Adiciona O'[m][0] em O'' (objetos alocados) em uma posição aleatória em S
  Para i = 1 até length(O') Faça
    C = Obtain_Probabilities(O'[i])
    O''[m][i] = Allocate(O'[i], C)
  Fim-Para
Retorno

Obtain_Probabilities(oi)
  Cria um vetor C de tamanho Q
  Inicializa o vetor C com 0
  Se Versão_HBAE = 1 Então
    j = um indice aleatório de O''[m]
    PointwiseSummation(oi, O''[m][j], C)
  Senão
    Para j = 0 Até length(O'') Faça
      PointwiseSummation(oi, O''[m][j], C)
    Fim-Para
  Fim-Se
Retorna C
```

Figura 3.9 - Pseudo-código do funcionamento do HBAE-puro

```
Main_ACO'_HBAE()
  Inicializa os parâmetros, a matriz de feromônios e
    a matriz de afinidades espaciais
  Para n = 0 até # de iterações Faça
    Para k = 0 até # de formigas Faça
      Enquanto houver objetos não alocados pela formiga k Faça
        Determinar o próximo objeto com probabilidade p_k(i, j)
        Alocar o objeto usando o método HBAE
      Fim-Enquanto
    Fim-Para
    Atualiza a matriz de feromônios na sequencia de objetos
      usado pela melhor formiga
    Atualiza a matriz de afinidades para o layout encontrado
      pela melhor formiga
  Fim-Para
Fim
```

Figura 3.10 - Pseudo-código do algoritmo híbrido ACO' e HBAE

3.6 Análise de complexidade da HBAE

A análise de complexidade de um algoritmo é muito importante para a avaliação de sua eficiência. Segundo Szwarcfiter (1989), estimar a complexidade de um novo algoritmo é um passo crucial para a validação e até mesmo valorização da técnica que está sendo apresentada.

Analisando o algoritmo do HBAE, considerando N o número de iterações, m o número de agentes (ou formigas, partículas, ...) e Ob[1] o número de objetos a serem alocados, podemos notar que o primeiro loop introduz N verificações para a função objetivo, o segundo loop m verificações e o terceiro loop Ob verificações. Logo, temos uma complexidade $O(N \times m \times Ob)$ para a versão I da heurística. A segunda versão introduz um novo loop de tamanho máximo Ob, tornando-se uma complexidade $O(N \times m \times Ob \times Ob)$. Importante observar também que não é comum a adoção de um número de agentes maior que o número de iterações, bem como um número de equipamentos maior que o número de iterações. A partir disso, podemos, portanto, considerar $O(N \times m \times Ob) < O(N^3)$ e $O(N \times m \times Ob \times Ob) < O(N^4)$ para o pior caso.

[1] A variável Ob foi adotada aqui como *objetos a serem alocados* para evitar conflito com a função de complexidade O

```
Main_MPCA_HBAE()
  Inicializa a matriz de afinidades espaciais
  Old_Config = Get_New_Solution()
  Best_Fitness = Fitness(Old_Config)
  Atualiza Blackboard
  Para n = 0 até # de iterações
    Para p = 0 até # de partículas
      Treat_particle()
    Fim-Para
    Atualiza Blackboard
    Atualiza matriz de afinidades para a melhor partícula
  Fim-Para
Fim

Get_New_Solution()
% Substitui Get_New_Solution do MPCA original
Retorna nova solução HBAE
```

Figura 3.11 - Pseudo-código do algoritmo híbrido MPCA e HBAE

4 PLATAFORMA DE BUSCA DE LAYOUTS INICIAIS

Para o INPE, um software como o desenvolvido neste trabalho pode ser muito útil por auxiliar no processo de desenvolvimento do layout de um satélite e ao mesmo tempo facilitar a implementação de novos algoritmos de otimização para testes e, quem sabe, gerar melhores resultados ante os métodos já implementados para este problema.

Este trabalho oferece uma plataforma que possibilita a aplicação de novos métodos de otimização a um problema prático da engenharia de construção de satélites, além de auxiliar projetistas já na fase conceitual de um projeto de satélite, com o fornecimento de um conjunto de layouts iniciais que previamente atendam requisitos cruciais do projeto. Ou seja, o desenvolvimento desta plataforma pode contribuir para encurtar o tempo de descoberta de novos métodos computacionais de otimização, e sua aplicação prática na produção de satélites.

Este capítulo apresenta detalhes usuais e técnicos da primeira versão desta *Plataforma de busca de layouts iniciais para o problema de alocação de equipamentos em satélites*.

4.1 Interfaces

Como apresentado do Capítulo 2, o desenvolvimento de layouts de satélites requer a otimização de variáveis de projeto fundamentais e o atendimento a requisitos essenciais para sua construção. Nesta seção veremos algumas características usuais do software supracitado, desenvolvido com tecnologia Java e JavaFX e projetado para simplicidade de utilização e facilidade de evolução. A Figura 4.1 apresenta a visão global do software.

A plataforma oferece recursos para configuração básica das dimensões e massa de cada equipamento e dimensões da superfície de alocação. Estas funcionalidades podem ser observadas na parte esquerda do software, destacado na Figura 4.1. Para cada objeto selecionado, a plataforma permite a configuração de certas restrições, tanto em relação a um outro objeto, quanto a qual face da superfície o objeto pode ser alocado. Por exemplo, como pode ser visto na Figura 4.1, o objeto 3 só pode ser alocado na face 1 e deve ser posicionado a uma distância mínima de 50 cm do objeto 1.

Além de determinar restrições de distâncias mínimas e máximas entre equipamentos e qual face da prateleira um equipamento deve ser alocado, o especialista pode

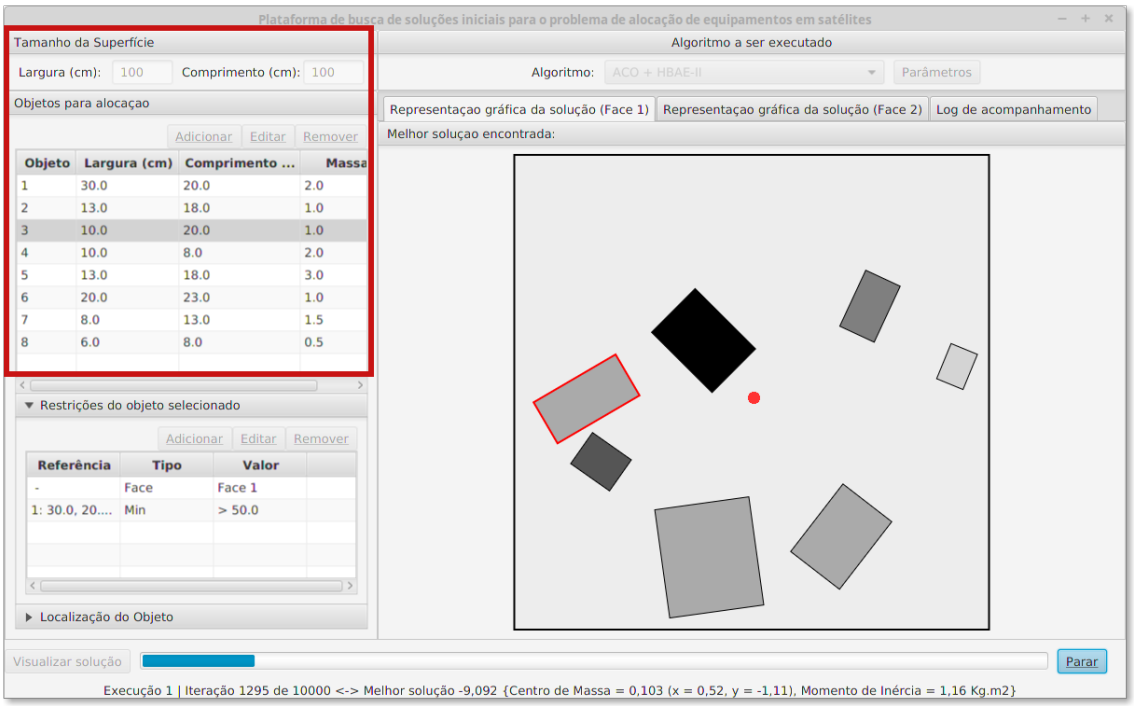


Figura 4.1 - Plataforma de busca de layouts iniciais para o problema de alocação de equipamentos em satélites - configuração básica

ainda especificar a posição e ângulo em que um determinado equipamento deve estar fixado, e executar o algoritmo de otimização para encontrar layouts com este(s) objeto(s) fixo(s), o que pode ser visto na Figura 4.2, através de componentes de interface que permitem marcar um objeto como fixo e movimentá-lo em posição e/ou ângulo. A plataforma ainda apresenta em tempo real as informações de centro de massa (ponto vermelho) e o momento de inercia (em relação ao eixo principal) da soluções atual.

Nas imagens geradas pela plataforma para representar o melhor layout, a cor dos objetos em escala de cinza é uma referência à massa de cada objeto. Os objetos mais escuros são mais pesados enquanto os mais claros são os mais leves. O ponto vermelho nas imagens representa a posição do centro de massa da solução.

Dependendo do algoritmo selecionado, a plataforma apresenta uma tela diferente para configuração dos parâmetros específicos deste algoritmo. Na Figura 4.3, é apresentada a tela que permite a configuração dos parâmetros do ACO. Nesta mesma interface é possível determinar quantas execuções do algoritmo se deseja executar e ao final a plataforma calcula o valor médio para a função-objetivo, o desvio padrão

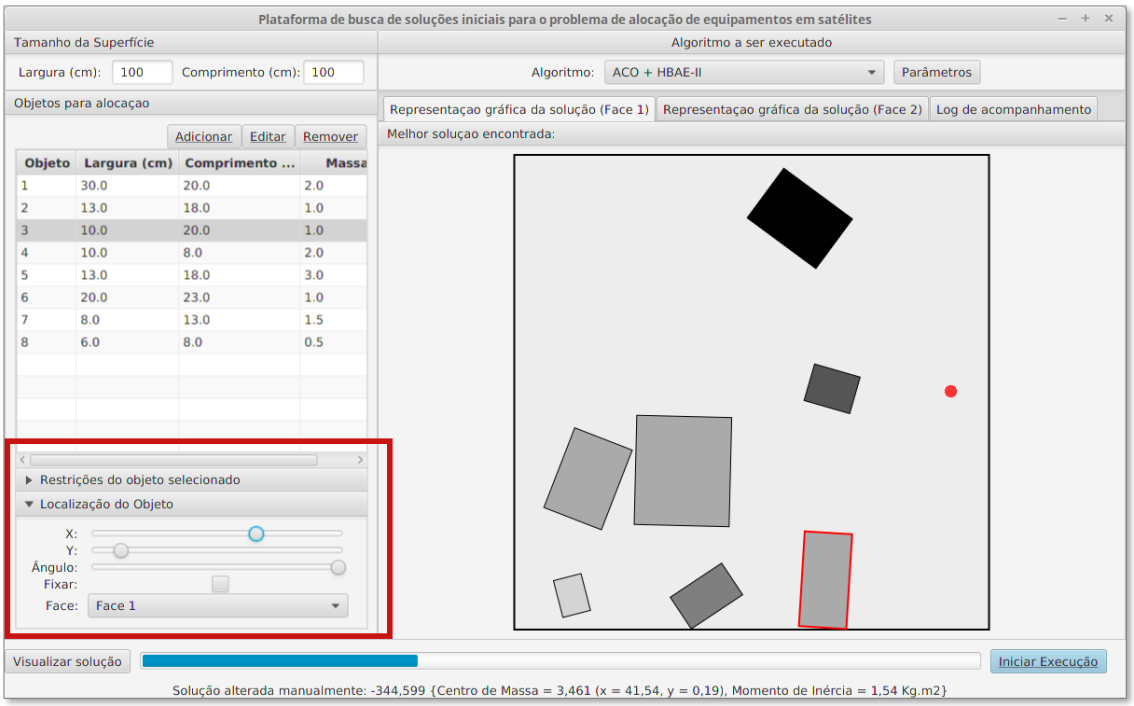


Figura 4.2 - Alteração e fixação de posição e ângulo de um equipamento

e o tempo médio. A semente da primeira execução também é definida pelo usuário/projetista (as demais sementes são determinadas aleatoriamente e salvas em log), bem como os pesos (λ_1 e λ_2) de cada objetivo e a pasta onde serão salvos os arquivos de log.

Após o algoritmo de otimização apresentar a melhor solução encontrada para o problema, o especialista pode realizar alterações na solução manualmente, alterando a posição, rotacionando os equipamentos ou mudando-os de face. A plataforma valida as modificações realizadas (incluindo as restrições de distância mínima, máxima, etc.) e apresenta informações de centro de massa e momento de inércia em tempo real. A Figura 4.4 apresenta um exemplo do resultado das modificações realizadas no objeto número 3 da solução.

É importante notar que apenas rotacionar um objeto não altera o centro de massa nem o momento de inércia da solução, pois foi considerado neste trabalho que o centro de massa de cada objeto encontra-se em seu próprio centro geométrico. Por isso, ao final da execução do algoritmo de otimização, pode-se alinhar os objetos em ângulos retos quando possível. A Figura 4.5 apresenta um layout gerado pela

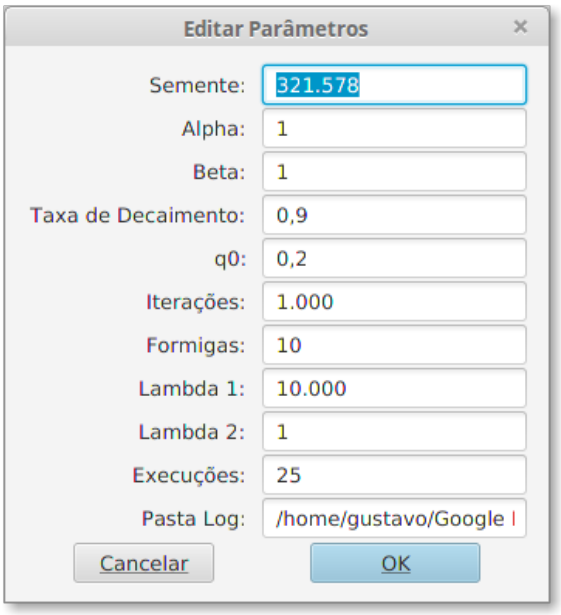


Figura 4.3 - Tela para configuração de parâmetros do algoritmo ACO

plataforma e o alinhamento reto de quase todos os equipamentos. Nota-se que não foi possível alinhar o ângulo de um dos equipamentos, pois geraria uma sobreposição de equipamentos.

Outra característica interessante da plataforma é a possibilidade de visualizar os equipamentos que estão alocados na outra face da prateleira. Note que na Figura 4.4 aparecem alguns objetos tracejados, isso indica que estes objetos estão na outra face da prateleira. Neste caso as restrições de sobreposição não são aplicáveis, por razões óbvias.

Todas as informações obtidas durante a execução do algoritmo selecionado são registradas em log, além de apresentadas em tempo de execução como observado na Figura 4.6. As soluções encontradas são registradas em um formato próprio com informações sobre a posição de cada equipamento, isso permite a reapresentação da solução em momento posterior na plataforma, caso desejado.

4.2 Arquitetura

Esta plataforma foi desenvolvida usando técnicas de programação orientada a objetos e desenvolvimento guiado por testes. Na estrutura básica da plataforma contém classes representando o container, os equipamentos, as restrições e as soluções. Estas

classes se relacionam entre si conforme diagrama da Figura 4.7.

Cada algoritmo pode precisar de suas próprias classes que se relacionam com as classes básicas da plataforma. Por exemplo o ACO que implementa um grafo onde cada aresta está ligada a um equipamento ou o MPCA que implementa uma classe Partícula (uma Thread, diga-se de passagem) que guarda as soluções que estão sendo trabalhadas. A Figura 4.8 apresenta estas classes e como elas se relacionam com as classes básicas da plataforma.

4.3 Validações e Testes Automatizados

Desenvolver software é uma atividade complexa, caracterizada por tarefas e requisitos que tendem a mudar constantemente (BOEHM; TURNER, 2003). Técnicas de desenvolvimento ágil de software nasceram para tratar esses problemas. Dentre estas técnicas destaca-se o TDD (*Test-Driven Development*).

O TDD oferece vários benefícios para o desenvolvimento de softwares. A prática ajuda o programador a evitar erros de regressão, em que a implementação de uma nova funcionalidade quebra uma outra funcionalidade já existente no sistema. Essa bateria de testes também provê segurança durante as constantes refatorações de código que são feitas durante o processo de desenvolvimento. (ANICHE, 2012)

O processo de validação e construção de layouts da plataforma, que é a base do software, foi desenvolvido com cobertura de testes de unidade, utilizando a técnica TDD. Isso dá segurança para refatoração a evolução do software em trabalhos futuros.

4.4 Algoritmos de Otimização

A implementação de algoritmos de otimização na plataforma para o problema discutido neste trabalho é um processo simples, uma vez que a toda a integração com a interface e as validações já estão implementadas em uma classe abstrata *OptimizationAlgorithm*, bastando estendê-la implementando o método abstrato *execute()* e informar à plataforma o progresso e as informações a serem registradas em log, a cada iteração e ao final da execução do algoritmo. Também é necessário criar a tela para edição dos parâmetros do algoritmo que está sendo implementado.

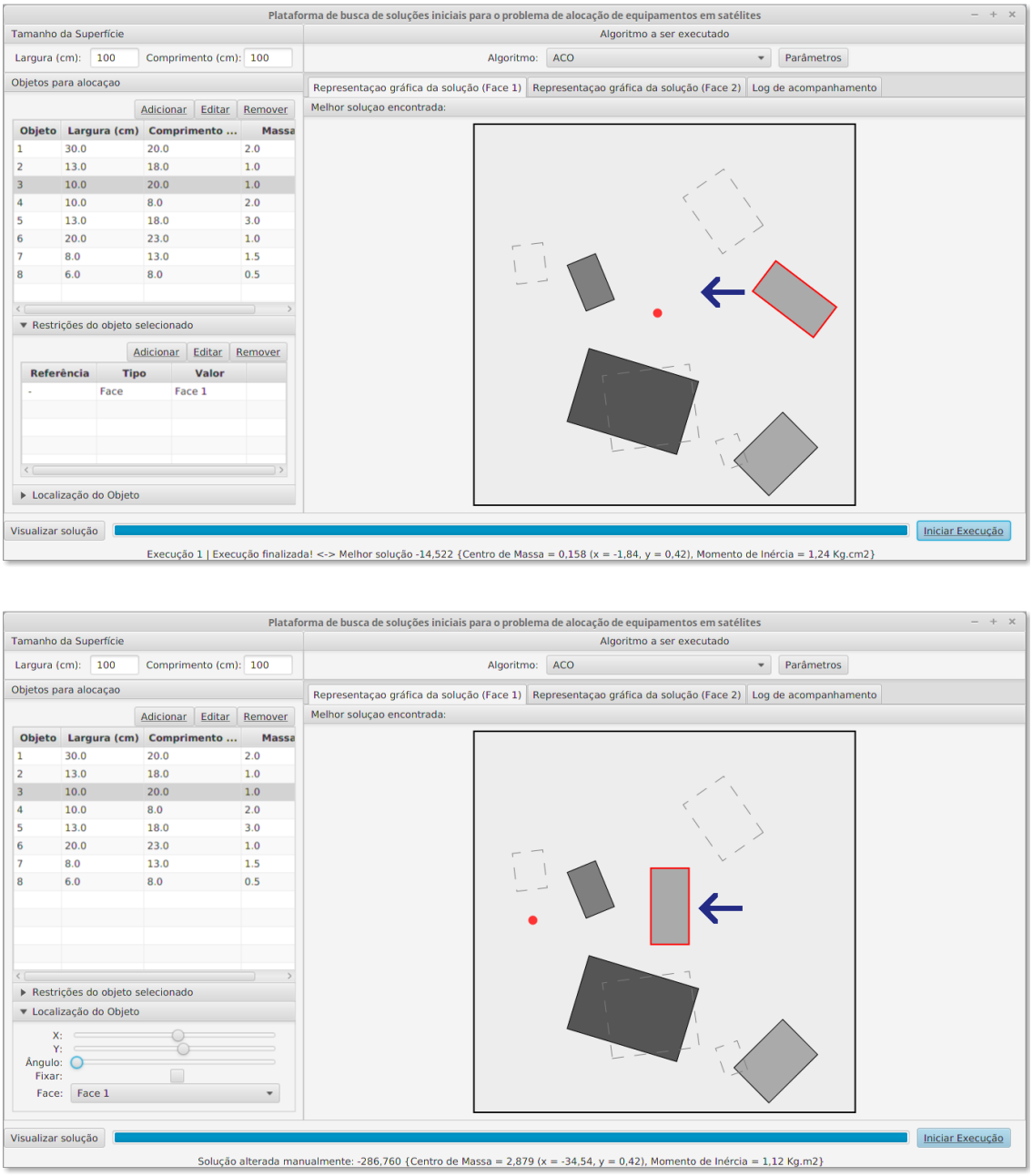


Figura 4.4 - Alteração de objetos na solução

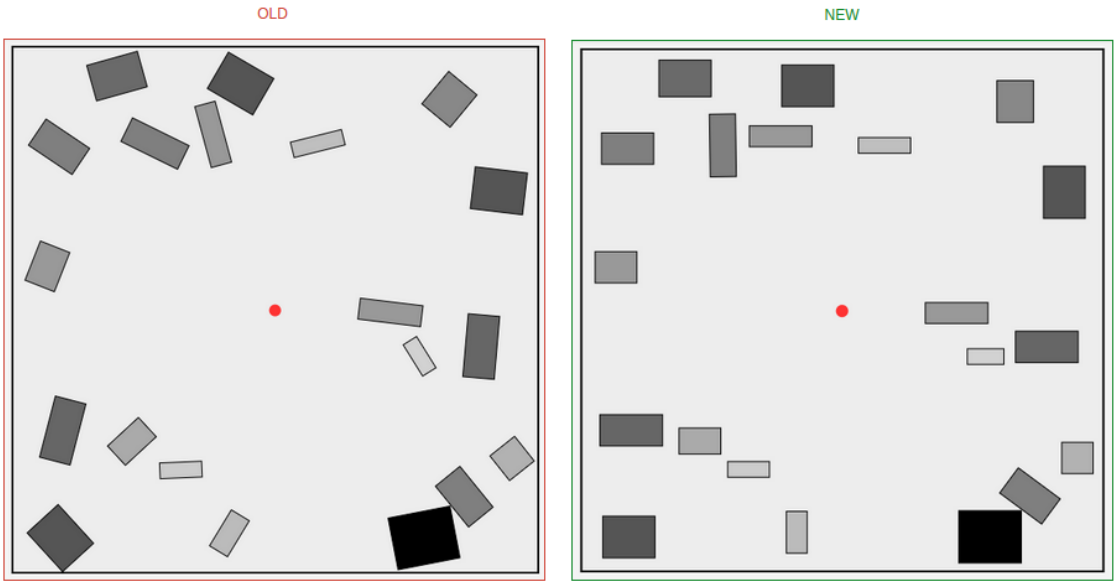


Figura 4.5 - Alinhamento de layout para ângulos retos.

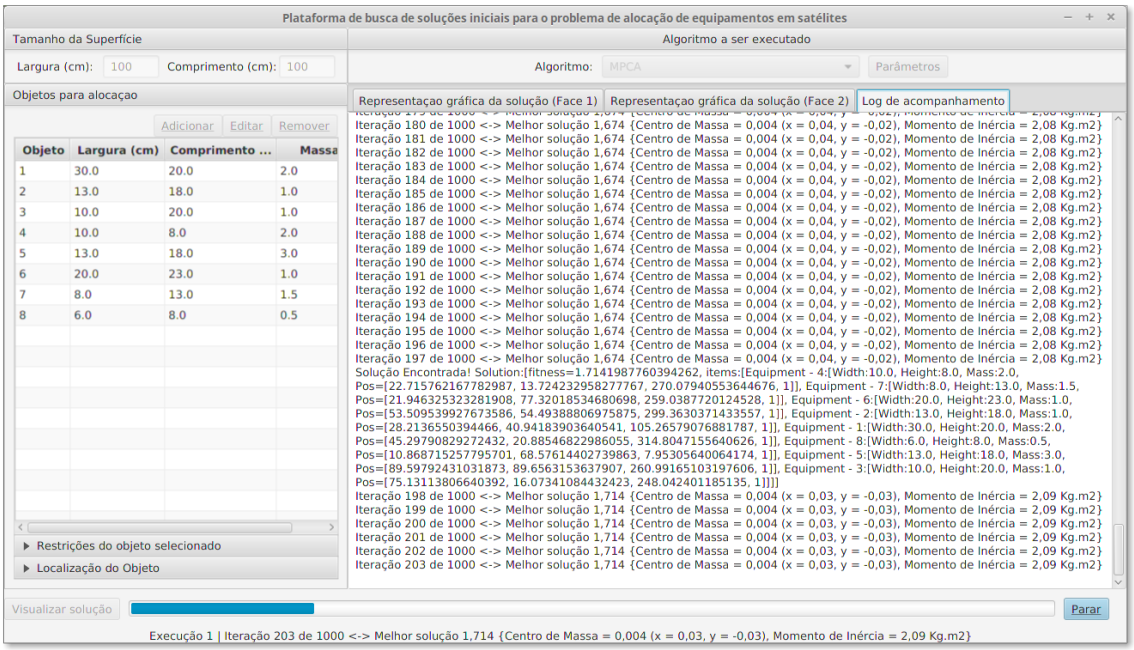


Figura 4.6 - Registro da execução de cada algoritmo

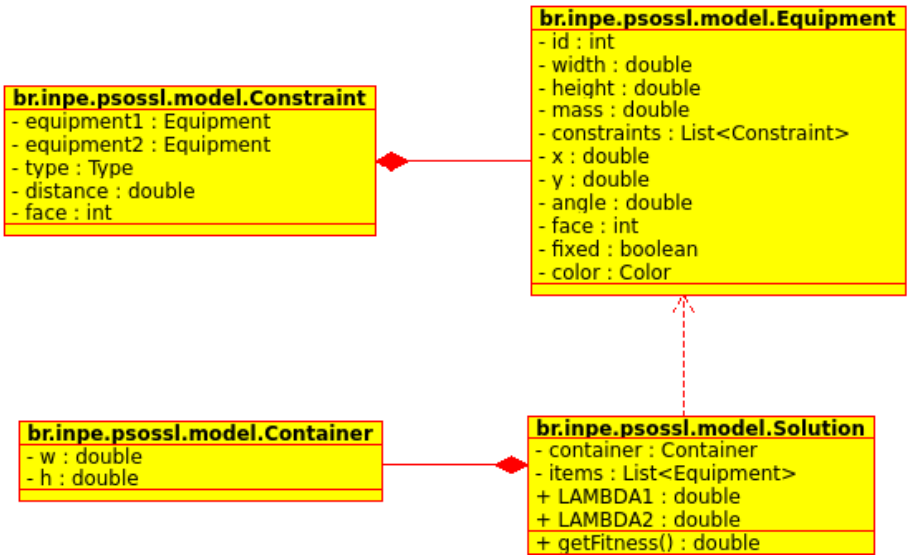


Figura 4.7 - Diagrama de classes da estrutura básica da plataforma

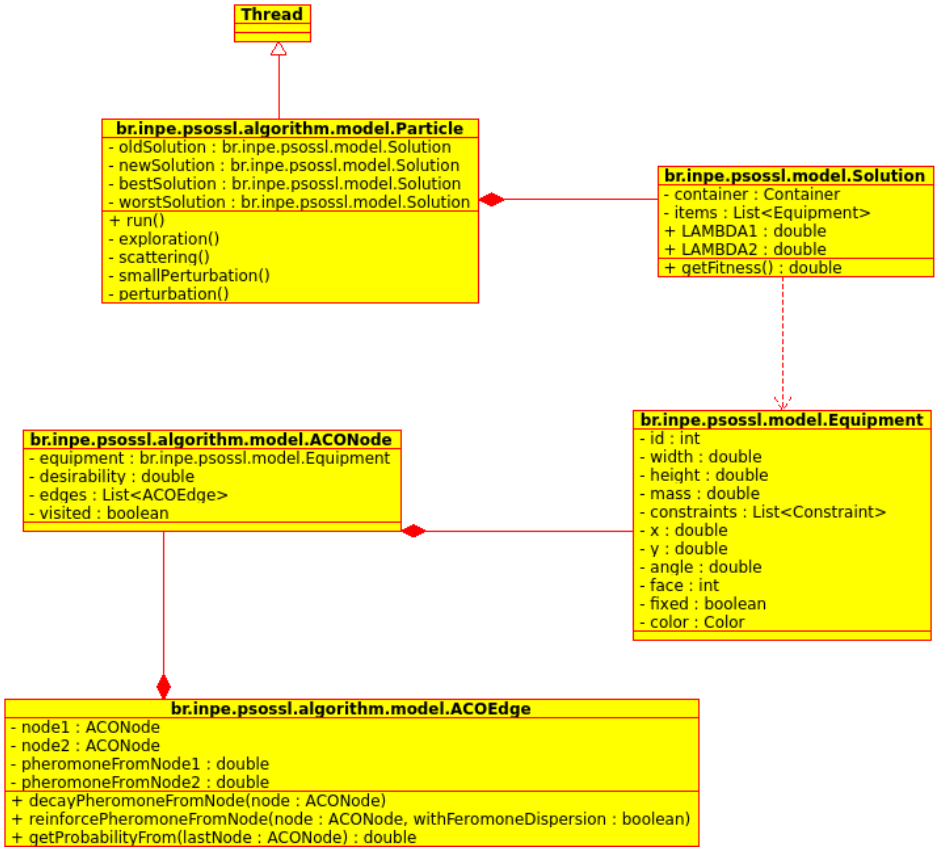


Figura 4.8 - Classes utilizadas pelos algoritmos de otimização

5 EXPERIMENTOS E RESULTADOS

Este capítulo apresenta os experimentos realizados, com o auxílio da plataforma, para diferentes casos, considerando-se o uso do ACO, do MPCA e do HBAE. As dimensões dos objetos foram determinadas hipoteticamente, baseado nos exemplos utilizados na literatura para este tipo de problema ((XU et al., 2007), (XU et al., 2010), (CUCO, 2011)). Sendo utilizada a função objetivo discutida na Seção 2.2.3.

Alguns parâmetros não tiveram alteração em todos os experimentos apresentados aqui. Os parâmetros λ_1 e λ_2 da função objetivo foram fixados em 10000 e 1 respectivamente, para ponderar os objetivos conforme Equação 5.1. A maioria dos parâmetros do ACO do HBAE e do MPCA não tiveram alteração e os valores utilizados são apresentados na Tabela 5.1.

$$\max(f_{obj}) = \lambda_2 \max(Mom.Inércia) - \lambda_1 \min(D_{CM-Origem}) \tag{5.1}$$

Para todos os experimentos foram executados 25 testes com cada algoritmo.

Tabela 5.1 - Parâmetros não alterados nos experimentos.

ACO					HBAE		MPCA
α	β	ρ	q_0	Formigas	Agentes	γ	Partículas
1	1	0,9	0,2	10	10	1%	10

Durante os testes também foi adotado um padrão para quantidade de iterações. No ACO, o número de iterações x número de formigas equivale à quantidade de soluções geradas pela execução do algoritmo. O mesmo vale para o HBAE. Entretanto, no MPCA não é possível saber a quantidade de soluções geradas em cada execução do algoritmo. Após alguns testes foi identificado que, no geral, poderia adotar para o MPCA, 1/10 de iterações do ACO/HBAE, para obter um tempo "parecido" entre eles.

Relembrando que nas imagens geradas pela plataforma para representar o melhor layout, a cor dos objetos em escala de cinza é uma referência à massa de cada objeto. Os objetos mais escuros são mais pesados enquanto os mais claros são os mais leves. O ponto vermelho nas imagens representa a posição do centro de massa da solução.

No geral, o tempo computacional para os experimentos de 1 a 5 com 1000 iterações

(100 para MPCA) variaram na casa de uma unidade de segundos e alguns algoritmos do experimento 5 (com mais equipamentos) atingiram mais de 10 segundos. Para os mesmos experimentos com 10000 iterações (1000 para o MPCA), o tempo computacional variou de 10 segundos a 1 minuto e meio. Já nos experimentos em que o tamanho da superfície foi mais proporcional ao tamanho dos equipamentos, o tempo computacional aumentou bastante, passando de 500 segundos em alguns casos. Isso se deve à dificuldade de conseguir construir uma solução válida e movimentar os equipamentos em um espaço menor.

O computador utilizado para os experimentos foi um Intel(R) Core(TM) i7 CPU 1,73 GHz 4-cores e 6 MB de memória cache, com memória de 6 GB. No sistema operacional Debian Linux.

5.1 Experimentos com objetos apenas em uma face e prateleira de tamanho fixo

Para os experimentos 1, 2, 4 e 5 foram adotadas as configurações de equipamentos (dimensões e massas) utilizadas por Xu et al. (2007). Para gerar soluções rapidamente, foi adotado uma prateleira de 100cm x 100cm. Também, foi adotada uma restrição padrão para todos os equipamentos, eles só podem ser alocados na face 1 da prateleira. As subseções a seguir apresentam as configurações (dimensões e massa) dos objetos utilizados em cada experimento, os resultados para 1000 e 10000 iterações e a visualização da melhor solução encontrada.

Os resultados apontaram o MPCA e o MCPA+HBAE muito superior ao ACO, HBAE-Puro E ACO+HBAE. Para alguns experimentos, os piores resultados do MPCA e do MCPA+HBAE foram melhores que a média dos outros. Nota-se também que quanto mais "preenchido" o espaço de alocação, melhor é a acurácia dos experimentos.

Nestes experimentos, como a superfície foi configurada com um tamanho relativamente grande, o tempo de execução dos algoritmos foram rápidos, pois há menos probabilidade de gerar soluções inválidas ao acrescentar um novo objeto.

5.1.1 Experimento 1: 5 objetos

Os objetos utilizados neste experimento são apresentados na Tabela 5.2, as Tabelas 5.3 e 5.4 para 1000 e 10000 iterações respectivamente, e a Figura 5.1 apresenta a melhor solução, encontrada pelo algoritmo MPCA com 1000 iterações.

Tabela 5.2 - Informações sobre os 5 equipamentos utilizados no experimento 1.

ID	Comprimento (cm)	Largura (cm)	Massa (Kg)
1	8	6	12
2	8	8	16
3	10	6	15
4	12	4	12
5	6	6	9

Tabela 5.3 - Resultados do experimento 1: 1000 iterações (100 iterações para MPCA)

	ACO	HBAE-I	HBAE-II	ACO+HBAE-I	ACO+HBAE-II	MPCA	MPCA+HBAE-I	MPCA+HBAE-II
Média	-4,958	-1,851	-5,533	-4,336	-6,113	8,094	**9,413**	9,251
Desvio padrão	6,573	6,993	7,446	8,460	6,054	2,534	3,385	1,624
Melhor solução	4,765	10,483	9,960	9,200	3,122	11,866	15,617	13,080
Melhor m. de inércia	11,036	13,111	11,331	12,085	6,002	12,298	15,899	13,331
Melhor c. de massa	0,063	0,026	0,014	0,029	0,029	0,004	0,003	0,003
Pior solução	-17,109	-16,784	-21,027	-20,151	-17,686	1,096	2,469	6,088
Pior m. de inércia	10,861	12,543	14,173	7,171	6,522	12,336	15,570	8,435
Pior c. de massa	0,280	0,293	0,352	0,273	0,242	0,112	0,131	0,023

Tabela 5.4 - Resultados do experimento 1: 10000 iterações (1000 iterações para MPCA)

	ACO	HBAE-I	HBAE-II	ACO+HBAE-I	ACO+HBAE-II	MPCA	MPCA+HBAE-I	MPCA+HBAE-II
Média	3,142	6,238	5,337	5,519	5,707	16,087	16,227	**16,400**
Desvio padrão	4,359	2,274	3,100	2,574	2,841	0,760	0,912	0,855
Melhor solução	10,003	11,900	10,127	10,136	10,238	18,391	18,388	18,207
Melhor m. de inércia	12,122	14,862	12,768	13,640	15,458	18,491	18,701	18,346
Melhor c. de massa	0,021	0,030	0,026	0,035	0,052	0,001	0,003	0,001
Pior solução	-8,349	2,287	-0,119	-0,397	-0,836	14,768	14,579	15,446
Pior m. de inércia	7,304	14,001	7,526	8,843	10,612	14,983	14,658	15,571
Pior c. de massa	0,157	0,117	0,076	0,092	0,114	0,002	0,001	0,001

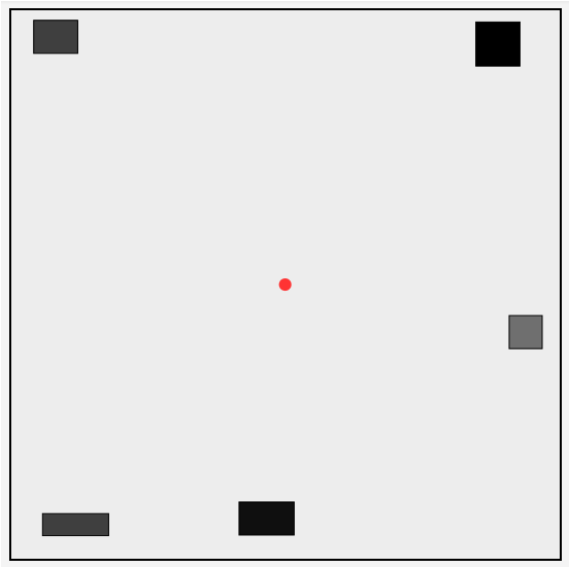


Figura 5.1 - Melhor solução encontrada no experimento 1

5.1.2 Experimento 2: 6 objetos

Os objetos utilizados neste experimento são apresentados na Tabela 5.5, as Tabelas 5.6 e 5.7 para 1000 e 10000 iterações respectivamente, e a Figura 5.2 apresenta a melhor solução, encontrada pelo algoritmo MPCA+HBAE-II com 1000 iterações.

Tabela 5.5 - Informações sobre os 6 equipamentos utilizados no experimento 2.

ID	Comprimento (cm)	Largura (cm)	Massa (Kg)
1	8	6	12
2	8	8	16
3	10	6	15
4	10	8	20
5	10	10	25
6	12	6	18

Tabela 5.6 - Resultados do experimento 2: 1000 iterações (100 iterações para MPCA)

	ACO	HBAE-I	HBAE-II	ACO+HBAE-I	ACO+HBAE-II	MPCA	MPCA+HBAE-I	MPCA+HBAE-II
Média	1,958	4,524	-0,352	3,205	2,035	14,081	**14,905**	14,105
Desvio padrão	6,815	5,805	7,095	6,925	8,512	3,887	3,123	3,250
Melhor solução	13,344	16,546	15,242	13,503	18,388	22,161	20,307	20,159
Melhor m. de inércia	18,612	18,911	23,206	18,281	20,675	22,564	20,435	20,283
Melhor c. de massa	0,053	0,024	0,080	0,048	0,023	0,004	0,001	0,001
Pior solução	-13,065	-7,842	-20,693	-9,576	-16,826	6,117	7,431	6,449
Pior m. de inércia	13,658	15,273	14,747	9,678	9,222	6,633	8,888	7,688
Pior c. de massa	0,267	0,231	0,354	0,193	0,260	0,005	0,015	0,012

Tabela 5.7 - Resultados do experimento 2: 10000 iterações (1000 iterações para MPCA)

	ACO	HBAE-I	HBAE-II	ACO+HBAE-I	ACO+HBAE-II	MPCA	MPCA+HBAE-I	MPCA+HBAE-II
Média	11,097	12,373	12,539	12,101	12,464	24,812	24,977	**25,396**
Desvio padrão	3,022	3,242	2,506	3,088	2,284	1,032	1,276	1,604
Melhor solução	15,954	21,903	18,051	18,481	17,487	26,914	27,581	29,529
Melhor m. de inércia	18,247	24,853	19,369	23,404	19,097	27,025	28,415	29,595
Melhor c. de massa	0,023	0,029	0,013	0,049	1,016	0,001	0,008	0,001
Pior solução	4,578	6,030	9,099	7,490	6,797	22,990	22,563	23,313
Pior m. de inércia	17,623	21,348	16,391	15,749	19,373	23,137	22,813	23,340
Pior c. de massa	0,0130	0,153	0,073	0,083	0,126	0,001	0,003	0,000

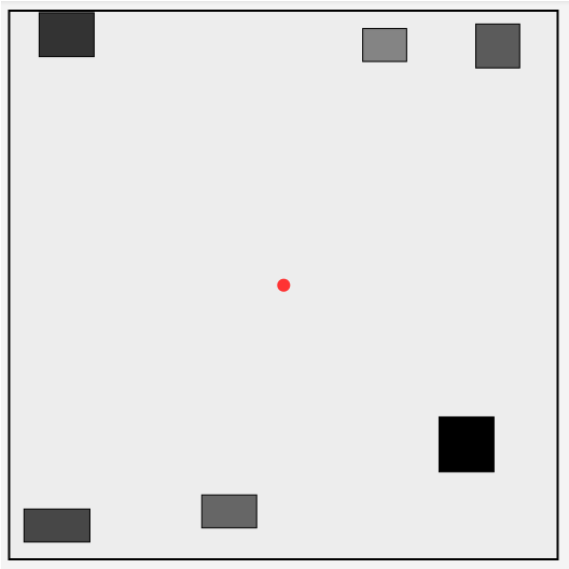


Figura 5.2 - Melhor solução encontrada no experimento 2

5.1.3 Experimento 3: 8 objetos

Os objetos utilizados neste experimento são apresentados na Tabela 5.8, as Tabelas 5.9 e 5.10 para 1000 e 10000 iterações respectivamente, e a Figura 5.3 apresenta a melhor solução, encontrada pelo algoritmo MPCA+HBAE-II com 1000 iterações.

Tabela 5.8 - Informações sobre os 8 equipamentos utilizados no experimento 3.

ID	Comprimento (cm)	Largura (cm)	Massa (Kg)
1	30	20	2
2	13	18	1
3	10	20	1
4	10	8	2
5	13	18	3
6	20	23	1
7	8	13	1,5
8	6	8	0,5

Tabela 5.9 - Resultados do experimento 3: 1000 iterações (100 iterações para MPCA)

	ACO	HBAE-I	HBAE-II	ACO+HBAE-I	ACO+HBAE-II	MPCA	MPCA+HBAE-I	MPCA+HBAE-II
Média	-7,599	-7,172	-7,915	-8,356	-5,625	1,088	1,187	**1,202**
Desvio padrão	4,899	4,992	6,289	5,212	2,890	0,269	0,346	0,457
Melhor solução	-1,651	0,617	0,170	-0,186	-1,177	1,588	1,830	1,946
Melhor m. de inércia	1,018	1,192	1,392	1,702	1,826	1,844	1,930	2,101
Melhor c. de massa	0,027	0,006	0,012	0,019	0,030	0,003	0,001	0,002
Pior solução	-21,714	-17,631	-27,907	-20,555	-12,273	0,538	0,448	-0,442
Pior m. de inércia	1,532	1,280	1,206	1,022	1,612	1,433	1,584	1,939
Pior c. de massa	0,232	0,189	0,291	0,216	0,139	0,009	0,011	0,024

Tabela 5.10 - Resultados do experimento 3: 10000 iterações (1000 iterações para MPCA)

	ACO	HBAE-I	HBAE-II	ACO+HBAE-I	ACO+HBAE-II	MPCA	MPCA+HBAE-I	MPCA+HBAE-II
Média	-1,543	-0,533	-0,977	-1,255	-1,669	2,199	2,178	**2,261**
Desvio padrão	1,244	1,449	1,468	1,303	1,469	0,085	0,104	0,101
Melhor solução	0,382	1,241	1,692	1,246	0,850	2,339	2,422	2,485
Melhor m. de inércia	1,093	1,877	1,976	1,957	1,071	2,471	2,510	2,578
Melhor c. de massa	0,007	0,006	0,003	0,007	0,002	0,001	0,001	0,001
Pior solução	-4,708	-4,583	-4,830	-3,817	-6,351	1,999	2,011	2,126
Pior m. de inércia	0,999	1,211	1,823	1,393	1,706	2,127	2,115	2,136
Pior c. de massa	0,057	0,058	0,067	0,052	0,081	0,001	0,001	0,000

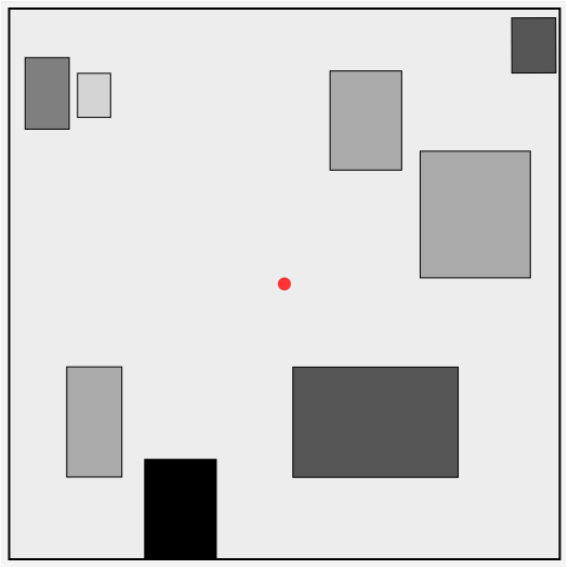


Figura 5.3 - Melhor solução encontrada no experimento 3

5.1.4 Experimento 4: 9 objetos

Os objetos utilizados neste experimento são apresentados na Tabela 5.11, as Tabelas 5.12 e 5.13 para 1000 e 10000 iterações respectivamente, e a Figura 5.4 apresenta a melhor solução, encontrada pelo algoritmo MPCA com 1000 iterações.

ID	Comprimento (cm)	Largura (cm)	Massa (Kg)
1	8	6	12
2	8	8	16
3	10	6	15
4	10	8	20
5	10	10	25
6	12	6	18
7	12	4	12
8	12	8	24
9	12	10	30

Tabela 5.11 - Informações sobre os 9 equipamentos utilizados no experimento 4.

Tabela 5.12 - Resultados do experimento 4: 1000 iterações (100 iterações para MPCA)

	ACO	HBAE-I	HBAE-II	ACO+HBAE-I	ACO+HBAE-II	MPCA	MPCA+HBAE-I	MPCA+HBAE-II
Média	14,611	16,090	13,919	12,793	14,213	25,257	24,334	**26,074**
Desvio padrão	4,137	5,250	5,031	5,353	7,329	5,174	2,967	5,778
Melhor solução	20,669	22,882	22,923	23,053	31,664	36,367	30,149	36,739
Melhor m. de inércia	22,333	33,878	25,547	27,229	34,404	36,500	30,503	37,317
Melhor c. de massa	0,017	0,110	0,026	0,042	0,027	0,001	0,004	0,006
Pior solução	4,424	4,623	3,901	2,029	-6,918	15,284	16,071	9,233
Pior m. de inércia	16,739	18,455	15,894	19,417	21,012	16,859	16,625	20,408
Pior c. de massa	0,123	0,138	0,120	0,174	0,279	0,016	0,006	0,112

Tabela 5.13 - Resultados do experimento 4: 10000 iterações (1000 iterações para MPCA)

	ACO	HBAE-I	HBAE-II	ACO+HBAE-I	ACO+HBAE-II	MPCA	MPCA+HBAE-I	MPCA+HBAE-II
Média	22,226	23,307	24,548	24,110	24,730	36,564	37,460	**37,518**
Desvio padrão	2,911	2,047	2,607	3,144	3,093	1,692	1,856	1,490
Melhor solução	26,671	26,506	28,922	32,891	32,165	42,365	41,044	40,631
Melhor m. de inércia	30,630	29,913	31,158	33,738	35,014	42,479	41,084	40,784
Melhor c. de massa	0,040	0,034	0,022	0,008	0,028	0,001	0,000	0,002
Pior solução	14,548	17,640	19,720	20,666	20,179	33,777	35,148	35,343
Pior m. de inércia	23,804	20,937	29,522	28,734	33,842	33,858	35,290	35,437
Pior c. de massa	0,093	0,033	0,098	0,081	0,137	0,001	0,001	0,001

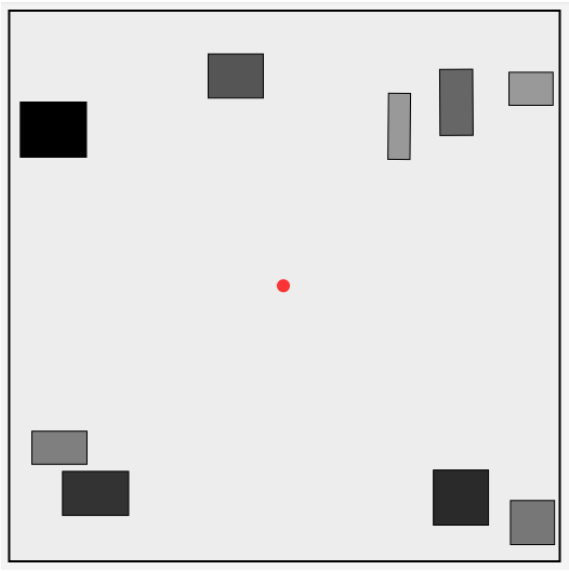


Figura 5.4 - Melhor solução encontrada no experimento 4

5.1.5 Experimento 5: 20 objetos

Os objetos utilizados neste experimento são apresentados na Tabela 5.14, as Tabelas 5.15 e 5.16 para 1000 e 10000 iterações respectivamente, e a Figura 5.5 apresenta a melhor solução, encontrada pelo algoritmo MPCA+HBAE-II com 1000 iterações.

Tabela 5.14 - Informações sobre os 20 equipamentos utilizados no experimento 5.

ID	Comprimento (cm)	Largura (cm)	Massa (Kg)
1	8	5	10
2	4	8	8
3	10	6	15
4	7	8	14
5	10	3	7,5
6	12	6	18
7	12	4	12
8	12	6	18
9	8	10	20
10	7	3	5,25
11	8	6	12
12	8	3	6
13	10	6	15
14	10	8	20
15	10	7	17,5
16	12	5	15
17	12	4	12
18	10	8	20
19	12	10	30
20	6	6	9

Tabela 5.15 - Resultados do experimento 5: 1000 iterações (100 iterações para MPCA)

	ACO	HBAE-I	HBAE-II	ACO+HBAE-I	ACO+HBAE-II	MPCA	MPCA+HBAE-I	MPCA+HBAE-II
Média	33,767	34,848	36,233	35,459	36,489	42,322	42,686	**42,797**
Desvio padrão	3,767	4,502	5,048	3,235	4,636	4,368	4,252	4,207
Melhor solução	40,678	42,207	46,300	41,471	47,987	50,939	53,222	52,812
Melhor m. de inércia	44,587	43,047	50,223	45,884	52,630	51,470	53,886	53,150
Melhor c. de massa	0,039	0,008	0,039	0,044	0,046	0,005	0,007	0,003
Pior solução	26,465	22,864	27,823	28,713	27,633	35,055	32,594	35,534
Pior m. de inércia	44,546	39,394	43,713	48,041	42,878	35,443	45,024	36,026
Pior c. de massa	0,181	0,165	0,159	0,193	0,152	0,004	0,124	0,005

Tabela 5.16 - Resultados do experimento 5: 10000 iterações (1000 iterações para MPCA)

	ACO	HBAE-I	HBAE-II	ACO+HBAE-I	ACO+HBAE-II	MPCA	MPCA+HBAE-I	MPCA+HBAE-II
Média	42,187	41,484	43,697	41,504	43,236	53,931	53,387	**55,927**
Desvio padrão	3,224	2,295	2,311	1,635	2,685	1,504	1,399	2,119
Melhor solução	48,663	46,577	49,368	44,696	49,859	57,575	56,951	61,001
Melhor m. inércia	52,923	48,759	50,706	48,558	51,221	57,747	57,025	61,050
Melhor c. massa	0,043	0,022	0,013	0,039	0,014	0,002	0,001	0,000
Pior solução	36,580	37,336	39,835	38,415	39,334	51,473	51,160	52,663
Pior m. inércia	40,553	41,384	49,935	43,492	47,357	51,534	51,185	52,689
Pior c. massa	0,040	0,040	0,101	0,051	0,080	0,001	0,000	0,000

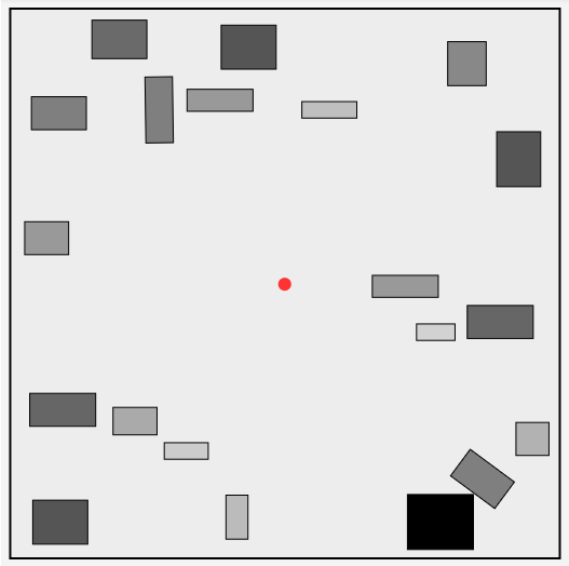


Figura 5.5 - Melhor solução encontrada no experimento 5

5.2 Experimentos com objetos em apenas uma face e prateleira de tamanho proporcional

Nestes experimentos, foram adotados os mesmos equipamentos dos experimentos 1, 2, 4 e 5 da seção anterior, entretanto o tamanho das superfícies foram configurados para serem proporcionais ao espaço ocupado pelos equipamentos. Isso aumenta o tempo de execução dos algoritmos, pois torna-se mais difícil resultar uma solução válida ao acrescentar novos objetos.

A qualidade dos experimentos melhoraram nestes resultados. Podemos notar também, que a diferença dos resultados encontrados pelo MPCA e MCPA+HBAE em

relação aos demais diminuiu. Ainda assim, MPCA e MPCA+HBAE mostraram-se bem mais eficiente. Novamente para alguns experimentos, os piores resultados dos algoritmos MPCA e MCPA+HBAE foram melhores que a média dos demais.

5.2.1 Experimento 6: 5 objetos

Os objetos usados neste experimento são os mesmos apresentados na Tabela 5.2 do experimento 1. A superfície utilizada foi configurada com 25cm x 25cm. Os resultados deste experimento são apresentados na Tabela 5.17 para 1000 iterações. Na Figura 5.6 é apresentada a melhor solução encontrada no experimento 6, pelo algoritmo MPCA+HBAE-I.

Tabela 5.17 - Resultados do experimento 6: 1000 iterações (100 iterações para MPCA)

	ACO	HBAE-I	HBAE-II	ACO+HBAE-I	ACO+HBAE-II	MPCA	MPCA+HBAE-I	MPCA+HBAE-II
Média	-0,492	-1,083	-0,808	-0,938	-1,033	0,386	**0,391**	0,376
Desvio padrão	0,640	0,877	0,573	0,700	0,633	0,036	0,044	0,045
Melhor solução	0,276	0,214	0,333	0,085	0,118	0,488	0,498	0,497
Melhor m. de inércia	0,361	0,502	0,464	0,445	0,464	0,519	0,560	0,547
Melhor c. de massa	0,001	0,003	0,001	0,004	0,003	0,000	0,001	0,001
Pior solução	-2,495	-4,025	-1,542	-2,931	-2,469	0,306	0,326	0,304
Pior m. de inércia	0,406	0,338	0,333	0,504	0,496	0,512	0,426	0,555
Pior c. de massa	0,029	0,044	0,019	0,034	0,030	0,002	0,001	0,003

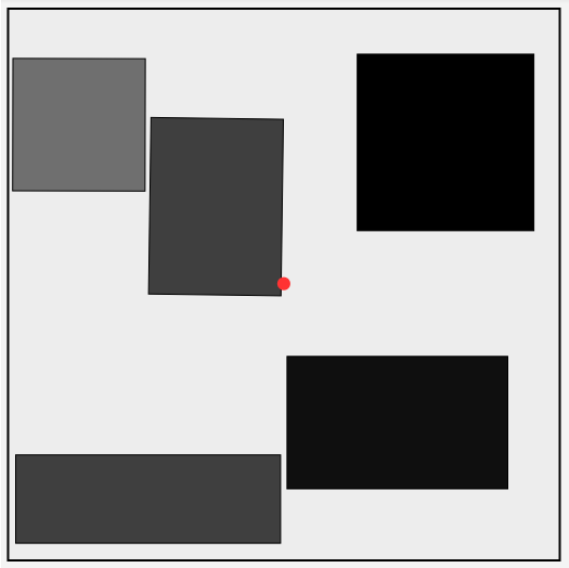


Figura 5.6 - Melhor solução encontrada no experimento 6

5.2.2 Experimento 7: 6 objetos

Neste experimento foram utilizados os mesmos 6 equipamentos apresentados na Tabela 5.5 do experimento 2, entretanto com uma plataforma menor, proporcional aos equipamentos, com 30cm x 30cm. Os resultados deste experimento são apresentados na Tabela 5.18 para 1000 iterações. Na Figura 5.7 é apresentada a melhor solução encontrada pelo algoritmo MPCA+HBAE-II.

Tabela 5.18 - Resultados do experimento 7: 1000 iterações (100 iterações para MPCA)

	ACO	HBAE-I	HBAE-II	ACO+HBAE-I	ACO+HBAE-II	MPCA	MPCA+HBAE-I	MPCA+HBAE-II
Média	-0,446	-0,415	-0,201	-0,012	-0,536	1,106	1,102	**1,128**
Desvio padrão	0,712	0,608	0,640	0,682	0,688	0,052	0,073	0,065
Melhor solução	0,655	0,586	1,015	0,858	0,616	1,226	1,274	1,327
Melhor m. de inércia	1,025	1,170	1,218	1,050	1,048	1,259	1,312	1,337
Melhor c. de massa	0,004	0,006	0,002	0,002	0,004	0,000	0,000	0,000
Pior solução	-2,187	-1,962	-1,566	-2,627	-2,083	0,982	0,990	1,005
Pior m. de inércia	1,029	1,128	1,040	1,086	1,109	1,116	1,149	1,356
Pior c. de massa	0,032	0,031	0,026	0,037	0,032	0,001	0,002	0,004

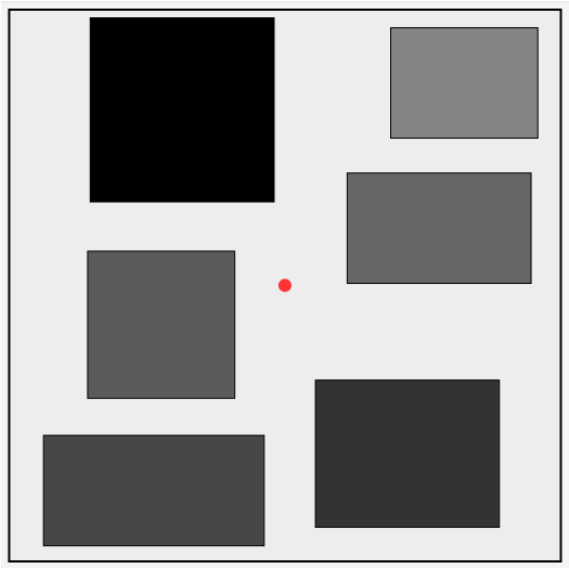


Figura 5.7 - Melhor solução encontrada no experimento 7

5.2.3 Experimento 8: 9 objetos

Neste experimento foram utilizados os mesmos 9 equipamentos apresentados na Tabela 5.11 do experimento 4, e uma prateleira com 40cm x 40cm. Os resultados deste experimento são apresentados na Tabela 5.19 para 1000 iterações. A melhor solução encontrada pelo MPCA é apresentada na Figura 5.8.

Tabela 5.19 - Resultados do experimento 8: 1000 iterações (100 iterações para MPCA)

	ACO	HBAE-I	HBAE-II	ACO+HBAE-I	ACO+HBAE-II	MPCA	MPCA+HBAE-I	MPCA+HBAE-II
Média	1,696	2,075	1,718	1,906	1,855	3,716	3,727	**3,733**
Desvio padrão	0,642	0,700	0,775	0,709	0,871	0,165	0,156	0,136
Melhor solução	2,886	3,120	3,531	3,052	3,710	4,249	4,035	3,974
Melhor m. de inércia	3,797	3,552	3,610	3,632	3,863	4,452	4,457	4,137
Melhor c. de massa	0,009	0,004	0,001	0,006	0,002	0,002	0,004	0,002
Pior solução	0,447	0,583	0,348	0,389	-0,026	3,494	3,455	3,457
Pior m. de inércia	3,166	3,177	3,542	3,233	3,115	3,547	3,663	3,791
Pior c. de massa	0,027	0,026	0,032	0,028	0,031	0,001	0,002	0,003

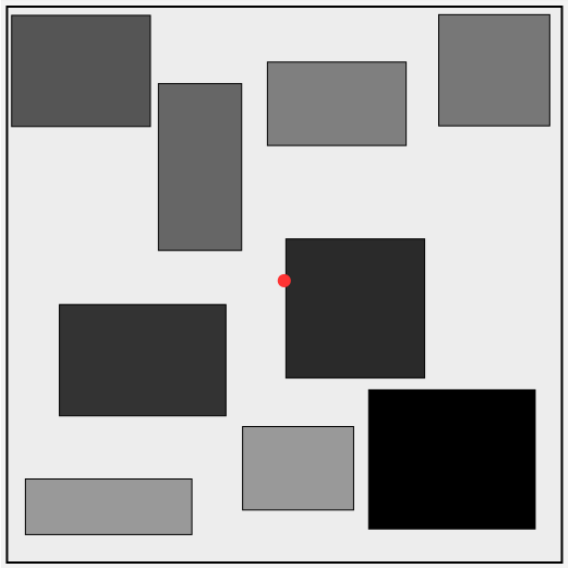


Figura 5.8 - Melhor solução encontrada no experimento 8

5.2.4 Experimento 9: 20 objetos

Neste experimento foram utilizados os mesmos 20 equipamentos apresentados na Tabela 5.14 do experimento 5, e uma plataforma de 60cm x 60cm. Os resultados deste experimento são apresentados na Tabela 5.20 para 1000 iterações. Na Figura 5.9 é apresentada a melhor solução encontrada no experimento 9, pelo algoritmo MPCA+HBAE-I.

Tabela 5.20 - Resultados do experimento 9: 1000 iterações (100 iterações para MPCA)

	ACO	HBAE-I	HBAE-II	ACO+HBAE-I	ACO+HBAE-II	MPCA	MPCA+HBAE-I	MPCA+HBAE-II
Média	11,920	11,827	11,973	11,819	12,118	15,280	**15,353**	15,331
Desvio padrão	1,225	1,247	1,019	1,343	1,347	0,343	0,434	0,384
Melhor solução	13,737	13,946	13,896	13,654	14,085	16,101	16,391	16,097
Melhor m. de inércia	14,172	14,381	15,300	14,174	16,444	17,263	16,831	16,597
Melhor c. de massa	0,004	0,004	0,014	0,005	0,024	0,012	0,004	0,005
Pior solução	9,234	8,543	9,257	8,223	8,253	14,681	14,715	14,539
Pior m. de inércia	14,243	12,458	13,002	12,786	13,653	14,730	15,800	14,923
Pior c. de massa	0,050	0,039	0,037	0,046	0,054	0,000	0,011	0,004

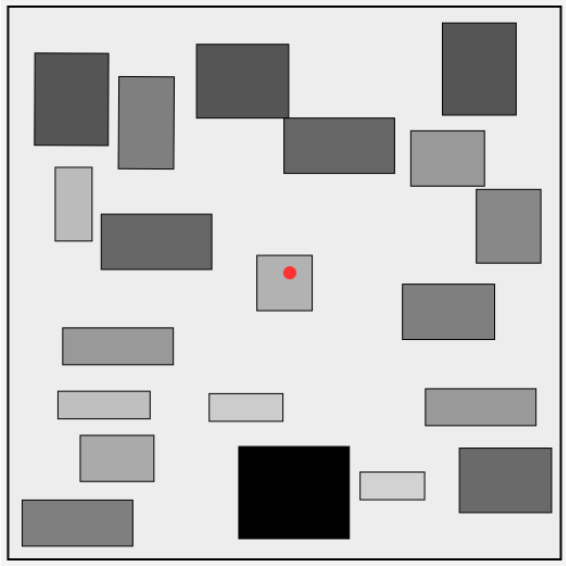


Figura 5.9 - Melhor solução encontrada no experimento 9

5.3 Experimentos com restrições entre objetos

Neste experimentos, foram adotados dimensões hipotéticas para os objetos. Utilizando uma das funcionalidades da plataforma, foram acrescentadas restrições de distância mínima e máxima Diferentemente dos experimentos realizados nas sessões anteriores, nestes foram utilizadas as duas faces da prateleira e também foram acrescentadas restrições sobre qual face da prateleira um objeto pode ser alocado. Lembrando que os equipamentos com borda tracejada estão alocados na outra face da prateleira. Foi considerado uma superfície com 100cm x 100cm.

Também com os experimentos envolvendo restrições e as duas faces da prateleira, o HBAE acrescentou uma pequena melhora na média dos resultados do MPCA. e os algoritmos que implementaram o MPCA obtiveram resultados superiores aos demais.

5.3.1 Experimento 10: 7 objetos

Foram utilizados 7 equipamentos para alocação nas duas faces da prateleira e com algumas restrições. As informações de cada equipamento deste experimento, bem como as restrições envolvidas, são apresentadas na Tabela 5.21. Os resultados deste experimento são apresentados nas Tabelas 5.22 e 5.23 para 1000 e 10000 iterações respectivamente. Na Figura 5.10 são apresentadas as duas faces da prateleira para a melhor solução, encontrada pelo algoritmo MPCA+HBAE-I com 1000 iterações.

Tabela 5.21 - Informações sobre os 7 equipamentos utilizados no experimento 10.

ID	Comprimento (cm)	Largura (cm)	Massa (Kg)	Restrições
1	40	35	8	Distância mínima com objeto 2: 50 cm
2	30	20	2	Distância mínima com objeto 1: 50 cm
3	50	30	15	Somente na face 2 da prateleira
4	10	15	0,6	Distância máxima com objeto 5: 40 cm
5	30	30	3	Distância máxima com objeto 4: 40 cm
6	20	25	2	
7	15	15	1	

Tabela 5.22 - Resultados do experimento 10: 1000 iterações (100 iterações para MPCA)

	ACO	HBAE-I	HBAE-II	ACO+HBAE-I	ACO+HBAE-II	MPCA	MPCA+HBAE-I	MPCA+HBAE-II
Média	-8,747	-8,909	-8,285	-8,397	-8,323	1,775	**1,846**	1,810
Desvio padrão	5,415	5,962	5,252	5,712	4,916	0,313	0,397	0,327
Melhor solução	-0,031	0,496	-1,311	-0,031	-1,640	2,374	2,816	2,361
Melhor m. de inércia	1,542	2,136	1,681	1,521	2,226	3,231	3,010	2,902
Melhor c. de massa	0,016	0,016	0,030	0,016	0,039	0,009	0,002	0,005
Pior solução	-18,272	-22,060	-25,948	-24,878	-19,964	1,126	1,068	1,237
Pior m. de inércia	1,930	2,058	2,526	1,931	2,329	2,831	2,004	1,735
Pior c. de massa	0,202	0,241	0,285	0,268	0,223	0,017	0,009	0,005

Tabela 5.23 - Resultados do experimento 10: 10000 iterações (1000 iterações para MPCA)

	ACO	HBAE-I	HBAE-II	ACO+HBAE-I	ACO+HBAE-II	MPCA	MPCA+HBAE-I	MPCA+HBAE-II
Média	-1,649	-1,964	-1,635	-1,256	-1,384	3,020	**3,126**	3,101
Desvio padrão	1,570	1,850	1,995	1,726	2,149	0,118	0,208	0,181
Melhor solução	0,981	1,008	2,102	2,144	1,880	3,234	3,525	3,410
Melhor m. de inércia	2,019	2,161	2,651	2,858	2,665	3,346	3,661	3,739
Melhor c. de massa	0,010	0,012	0,005	0,007	0,008	0,001	0,001	0,003
Pior solução	-5886	-6,359	-6,092	-5,940	-5,342	2,803	2,731	2,727
Pior m. de inércia	1,677	2,141	2,506	2,165	2,617	3,312	2,774	3,263
Pior c. de massa	0,076	0,085	0,086	0,081	0,080	0,005	0,000	0,005

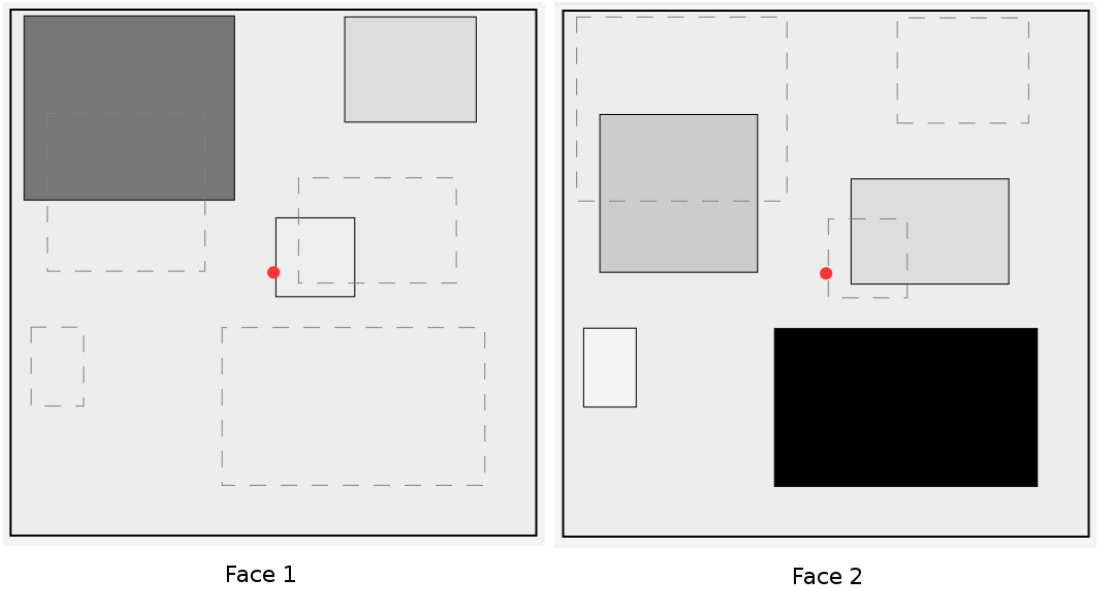


Figura 5.10 - Melhor solução encontrada no experimento 10: Face 1 e Face 2

5.3.2 Experimento 11: 20 objetos

Neste experimento foram utilizados 20 equipamentos para alocação nas duas faces da prateleira e com algumas restrições. As informações de cada equipamento deste experimento são apresentadas na Tabela 5.24, juntamente com as respectivas restrições. Foi considerado uma superfície com 100cm x 100cm. Os resultados deste experimento são apresentados nas Tabelas 5.25 e 5.26 para 1000 e 10000 iterações respectivamente. Na Figura 5.11 são apresentadas as duas faces da prateleira para a melhor solução, encontrada pelo algoritmo MPCA+HBAE-I com 1000 iterações.

Tabela 5.24 - Informações sobre os 20 equipamentos utilizados no experimento 11.

ID	Comprimento (cm)	Largura (cm)	Massa (Kg)	Restrições
1	8	5	10	Distância mínima com objeto 2: 50 cm
2	4	8	8	Distância mínima com objeto 1: 50 cm
3	10	6	15	Somente na face 2 da prateleira
4	7	8	14	Distância máxima com objeto 5: 40 cm
5	10	3	7,5	Distância máxima com objeto 4: 40 cm
6	12	6	18	
7	12	4	12	
8	12	6	18	
9	8	10	20	
10	7	3	5,25	Somente na face 1 da prateleira
11	8	6	12	Distância máxima com objeto 19: 30 cm
12	8	3	6	
13	10	6	15	
14	10	8	20	Distância mínima com objeto 15: 60 cm
15	10	7	17,5	Distância mínima com objeto 14: 60 cm
16	12	5	15	
17	12	4	12	
18	10	8	20	
19	12	10	30	Distância máxima com objeto 11: 30 cm
20	6	6	9	

Tabela 5.25 - Resultados do experimento 11: 1000 iterações (100 iterações para MPCA)

	ACO	HBAE-I	HBAE-II	ACO+HBAE-I	ACO+HBAE-II	MPCA	MPCA+HBAE-I	MPCA+HBAE-II
Média	30,999	35,435	33,192	36,619	32,613	50,173	**50,254**	49,828
Desvio padrão	5,318	4,561	4,091	4,155	6,742	1,872	2,381	2,687
Melhor solução	44,241	42,965	42,710	43,515	43,777	53,590	54,041	55,746
Melhor m. de inércia	49,013	50,958	46,002	47,538	46,793	54,933	54,731	57,736
Melhor c. de massa	0,048	0,080	0,033	0,040	0,030	0,013	0,007	0,020
Pior solução	23,341	26,354	26,974	27,910	19,090	46,199	46,267	45,446
Pior m. de inércia	40,091	40,650	47,738	39,185	45,656	50,287	50,505	47,773
Pior c. de massa	0,168	0,143	0,208	0,113	0,266	0,041	0,042	0,023

Tabela 5.26 - Resultados do experimento 11: 10000 iterações (1000 iterações para MPCA)

	ACO	HBAE-I	HBAE-II	ACO+HBAE-I	ACO+HBAE-II	MPCA	MPCA+HBAE-I	MPCA+HBAE-II
Média	42,070	41,927	34,356	41,545	33,313	55,901	**57,230**	55,591
Desvio padrão	3,495	2,259	4,049	2,532	5,742	1,758	1,836	1,405
Melhor solução	48,468	47,032	40,546	46,438	42,105	60,805	60,812	58,679
Melhor m. de inércia	51,276	48,457	44,555	47,581	47,697	61,546	61,565	59,404
Melhor c. de massa	0,028	0,014	24,064	0,011	0,056	0,007	0,008	0,007
Pior solução	33,336	37,367	24,064	37,106	21,834	53,255	54,227	52,852
Pior m. de inércia	40,880	44,583	43,622	44,134	48,553	53,344	55,149	53,982
Pior c. de massa	0,075	0,072	0,196	0,070	0,267	0,001	0,009	0,011

Figura 5.11 - Melhor solução encontrada no experimento 11: Faces 1 e Face 2

5.4 Conclusões dos Experimentos

Em todos os experimentos, o MPCA apresentou melhores resultados que o ACO. Isto se deve ao fato do MPCA executar movimentações na localização de cada objeto, através das funções *Perturbation* e *SmallPerturbation*. O ACO, aplicado à ordem em que os objetos são alocados, tem menor influência sobre a função objetivo utilizada neste trabalho.

Nota-se também que a heurística HBAE melhorou os resultados do MPCA em todos os experimentos, hora com a versão I, hora com a versão II da heurística. Estas

melhoras podem ser observadas na Tabela 5.27, que apresenta os resultados médios de todos os algoritmos para os experimentos realizados, destacando os melhores resultados.

Tabela 5.27 - Resultados médios gerais

	ACO	HBAE-I	HBAE-II	ACO+HBAE-I	ACO+HBAE-II	MPCA	MPCA+HBAE-I	MPCA+HBAE-II
Experimento 1	-4,958	-1,851	-5,533	-4,336	-6,113	8,094	**9,413**	9,251
Experimento 2	1,958	4,524	-0,352	3,205	2,035	14,081	**14,905**	14,105
Experimento 3	-7,599	-7,172	-7,915	-8,356	-5,625	1,088	1,187	**1,202**
Experimento 4	14,611	16,090	13,919	12,793	14,213	25,257	24,334	**26,074**
Experimento 5	33,767	34,848	36,233	35,459	36,489	42,322	42,686	**42,797**
Experimento 6	-0,492	-1,083	-0,808	-0,938	-1,033	0,386	**0,391**	0,376
Experimento 7	-0,446	-0,415	-0,201	-0,012	-0,536	1,106	1,102	**1,128**
Experimento 8	1,696	2,075	1,718	1,906	1,855	3,716	3,727	**3,733**
Experimento 9	11,920	11,827	11,973	11,819	12,118	15,280	**15,353**	15,331
Experimento 10	-8,747	-8,909	-8,285	-8,397	-8,323	1,775	**1,846**	1,810
Experimento 11	30,999	35,435	33,192	36,619	32,613	50,173	**50,254**	49,828

6 CONCLUSÕES E TRABALHOS FUTUROS

O objetivo deste trabalho foi desenvolver uma plataforma capaz de facilitar a aplicação de novas técnicas computacionais de otimização, a um problema real da engenharia de construção de satélites para projetos desenvolvidos no INPE. Uma heurística para o problema de alocação de equipamentos em satélites foi criada, gerando bons resultados. Também foi desenvolvida uma plataforma que pode ser de grande utilidade em projetos futuros. Com isso, o objetivo deste trabalho foi alcançado.

A arquitetura mecânica de um novo satélite, usualmente é determinada manualmente baseada na experiência da equipe de projetistas e em projetos anteriores para missões semelhantes. A ferramenta desenvolvida neste trabalho visa auxiliar este trabalho através da aplicação de técnicas de otimização desenvolvidas para este fim.

Nos experimentos, buscou-se utilizar dimensões e massas utilizados na literatura para problemas deste tipo. Também foram adotadas restrições para simular a necessidade de proximidade ou afastamento de equipamentos entre si, através de funcionalidades presentes na plataforma.

O Algoritmo da Formigas (ACO) e o de Colisão de Múltiplas Partículas (MPCA), duas meta-heurísticas inspiradas na natureza, foram implementados na plataforma, consolidando a facilidade de implementação de novos algoritmos de otimização, uma das principais características do software. Através disso foi possível concluir que, para este problema, o MPCA foi mais eficiente que o ACO por trabalhar diretamente na posição dos equipamentos na superfície.

Este trabalho gerou alguns resultados interessantes. A heurística aqui desenvolvida, apresentou resultados consistentes ao melhorar, na média, o desempenho do MPCA em todos os experimentos realizados. Isso mostra que a Heurística de Balanceamento por Afinidade Espacial é promissora na solução de problemas desta natureza. Foram submetidos dois artigos para congressos (Alves et al. (2013a), Alves et al. (2015)), descrevendo a heurística, sendo um deles internacional e ambos foram aceitos. Além disso, software criado pode ser de grande utilidade para projetos futuros do INPE como uma ferramenta de auxílio para o projetistas.

O desenvolvimento deste trabalho, pode ser de grande valia para o INPE, visto que o problema aqui tratado, quando bem resolvido, influencia diretamente os custos de produção, tempo de desenvolvimento, a performance e o tempo de vida do satélite. Além disso, como a computação está em constante evolução, trabalhos futuros a

partir deste visam evoluções tando para a plataforma, quanto para a heurística apresentada.

Para a plataforma, as evoluções sugeridas incluem: (i) integração com ferramentas CAD usada pelos projetistas, (ii) tratamento de objetos 3D, (iii) outros formatos de objetos, (iv) outros formatos de satélite (cúbico, painéis ligados em formato de pentágono, hexágono, etc...), (v) inclusão de regras fuzzy para as relações entre objetos, (vi) tratar o problema diretamente como multi-objetivo, apresentando gráficos com fronteira a fronteira de pareto. Visando garantir o suporte da plataforma para a maioria dos sistemas operacionais, esta foi desenvolvida em Java, e uma de suas últimas tecnologias de interface (JavaFX), entretanto, toda construção do código-fonte foi pensada em baixo acoplamento entre a interface e o *core* da plataforma. Isso facilitaria a (vii) extração de uma biblioteca para ser acoplada como complemento diretamente nas ferramentas utilizadas pelos projetistas.

Para a heurística, as evoluções propostas são: (i) paralelizar os agentes através de multi-processadores, (ii) aplicar a heurística para outros problemas do tipo Alocação de Objetos, incluindo o problema tratado em Xu et al. (2010), (iii) associá-la a outras heurísticas e meta-heurísticas, (iv) aplicá-la a problemas envolvendo vários painéis identificando outras regiões relativas para alocação.

REFERÊNCIAS BIBLIOGRÁFICAS

ALVES, G. F. d. O.; SANDRI, S.; BECCENERI, J. C. Algoritmo das formigas com heurística de balanceamento aplicado ao problema de alocação de equipamentos em satélites. In: ENCONTRO DE MODELAGEM COMPUTACIONAL. (EMC), 2013, Ilhéus-BA. **Anais...** Ilhéus: PPGMC/UESC, 2013. 2, 33, 71, 79

_____. Algoritmo de colônia de formigas aplicado ao problema de alocação de equipamentos em satélites. In: ENCONTRO NACIONAL DE INTELIGÊNCIA ARTIFICIAL E COMPUTACIONAL. (ENIAC), 2013, Fortaleza-CE. **Anais...** Fortaleza: BRACIS, 2013. 77

_____. A balancing heuristic for spacecraft equipment layout optimization. In: 2015 IEEE INTERNATIONAL CONFERENCE ON FUZZY SYSTEMS, Istanbul-Turkey. [S.l.], 2015. Submitted Paper. 71, 81

ANICHE, M. F. **Como a prática de TDD influencia o projeto de classes em sistemas orientados a objetos**. Dissertação (Mestrado em Ciência da Computação) — Universidade de São Paulo, 2012. 47

BECCENERI, J. C.; RAMOS, F. M.; VELHO, H. F. d. C.; SILVA, J. D. S. o. d.; LORENA, L. A. N.; VIJAYKUMAR, N. L.; SANTOS, R.; ROSA, R. R.; TRAVELHO, J. o. d. S. **Computação e matemática aplicada às ciências e tecnologias espaciais**. São José dos Campos, SP: INPE, 2008. 65-81 p. 23

BECCENERI, J. C.; SILVA NETO, J. A. **Técnicas de inteligência computacional inspiradas na natureza Aplicação em problemas inversos em transferência radiativa**. São Carlos, SP: SBMAC, 2009. 122 p. 2, 24, 25, 26

BLUM, C. Ant colony optimization: introduction and recente trends. **Physics of Life Review Letters**, p. 353–373, 2005. 28

BLUM, C.; ROLI, A. Metaheuristics in combinatorial optimization: overview and conceptual comparison. **ACM Computing Surveys**, v. 35, p. 268–308, 2003. 25

BOEHM, B.; TURNER, R. **Balancing agility and discipline:**: A guide for the perplexed. [S.l.]: Addison-Wesley Professional, 2003. 47

CARRARA, V. **Dinâmica de atitude de satélites artificiais**. 2012. Http://www2.dem.inpe.br/val/projetos/atdyn/index.html. [Acessado em 10/12/2014]. 11

CORRADI, W.; OLIVEIRA, W. S. de; NEMES, M. C.; TÁRSIA, R. D.; VIEIRA, S. L. A.; BALZUWEIT, K. **Fundamentos de física I**. [S.l.]: Editora URMG, 2010. 9, 11

CUCO, A. P. C. **Desenvolvimento de uma metodologia multiobjetivo para a otimização do layout de equipamentos em satélites artificiais**. 157 p. Dissertação (Mestrado) — Instituto Nacional de Pesquisas Espaciais, São José dos Campos, 2011-09-08 2011. Disponível em: <http://urlib.net/sid.inpe.br/mtc-m19/2011/08.09.01.46>. Acesso em: 02 jan. 2015. 1, 5, 6, 11, 51

DORIGO, M. **Ottimizzazione, Aprendimento Automático, ED Algoritmi Basati Si Metafora Naturale**. Tese (Doutorado) — Politécnico di Milano, Italy, 1992. 2, 26, 27

DORIGO, M.; CARO, G.; GAMBARDELLA, L. M. Ant algorithms for discrete optimization. **Massachusetts Institute of Technology**, 1999. 28

DYCKHOFF, H. A typology of cutting and packing problems. **European Journal of Operational Research**, v. 41, p. 145–159, 1990. 1, 6

FLOUDAS, C. A.; PARDALOS, P. M. **Encyclopedia of optimization**. [S.l.]: Kluwer Academic, 2001. 25

FORTESCUE, P.; SWINERD, G.; STARK, J. **Spacecraft Systems Engineering**. [S.l.]: Wiley, 2011. ISBN 9781119978367. 5

LAU, V.; SOUSA, F. L. d.; GALSKI, R. L.; ROCCO, E. M.; BECCENERI, J. C.; SANTOS, W. A.; SANDRI, S. A. A multidisciplinary design optimization tool for spacecraft equipment layout conception. **J.Aerosp. Technol. Manag.**, v. 6, n. 4, p. 431–446, 2014. 6

LUZ, E. F. P.; BECCENERI, J. C.; VELHO, H. F. C. A new multi-particle collision algorithm for optimization in high performance environment. **Journal of Computational Interdisciplinary Sciences**, 2008. 32

LUZ, E. F. P. d. **Meta-heurísticas paralelas na solução de problemas inversos**. 155 p. Tese (Doutorado) — Instituto Nacional de Pesquisas Espaciais, São José dos Campos, SP, 2012. 2, 23, 26, 31

POHLHEIM, H. **GEATbx: Genetic and Evolutionary Algorithm Toolbox for use with MATLAB Documentation**. 2006. Http://www.geatbx.com/docu/fcnindex-01.html. [Acessado em 13/11/2014]. 24

SACCO, W. F.; OLIVEIRA, C. R. E. A. A new stochastic optimization algorithm based on a particle collision metaheuristic. In: WORLD CONGRESS OF STRUCTURAL AND MULTIDISCIPLINARY OPTIMIZATION - WCSMO, 6., 2005, Rio de Janeiro-RJ. **Proceedings...** Rio de Janeiro: International Society for Structural and Multidisciplinary Optimization (ISSMO), 2005. v. 92, p. 657–659. 2, 31

SATO, L. H. S. **Notícia Satélite ITASAT-1**. 2014. http://www.ita.br/noticias/itasat1. [Acessado em 15/02/2015]. 7

SOUSA, F. L. d.; GALSKI, R. L.; ROCCO, E. M.; BECCENERI, J. C.; SANTOS, W. A. d.; SANDRI, S. A. A tool for multidisciplinary design conception of spacecraft equipment layout. In: INTERNATIONAL CONGRESS OF MECHANICAL ENGINEERING, 22. (COBEM), 3-7 Nov. 2013, Ribeirão Preto. **Proceedings...** [S.l.], 2013. 1, 6

SOUTO, R. P.; LUZ, E. F. P.; BECCENERI, J. C.; STEPHANY, S.; VELHO, H. F. C.; SANDRI, S. A.; SILVA NETO, A. J. A fuzzy ant colony optimization algorithm for the estimation of radiative properties in one-dimensional homogeneous participating media. In: INTERNATIONAL CONGRESS OF MECHANICAL ENGINEERING, 20. (COBEM)., 2009, Gramado-RS. **Proceedings...** Gramado: COBEM, 2009. 27

SZWARCFITER, J. L. **Estrutura de dados e seus algoritmos**. [S.l.]: UFMG, 1989. 40

WANG, Y.-S.; TENG, H.-F.; Y-J., S. Cooperative co-evolutionary scatter search for satellite module layout design. **International Journal for Computer - Aided Engineering and Software**, v. 26, n. 7, p. 761–785, 2009. 1, 6

XU, Y. C.; DONG, F. M.; LIU, Y.; XIAO, R. B.; AMOS, M. Ant colony algorithm for the weighted item layout optimization problem. **arXiv preprint arXiv:1001.4099**, 2010. xv, 2, 29, 30, 32, 51, 72

XU, Y.-C.; XIAO, R.-B.; AMOS, M. Particle swarm algorithm for weighted rectangle placement. In: INTERNATIONAL CONFERENCE ON NATURAL COMPUTATION (ICNC). **Proceedings...** [S.l.], 2007. v. 4, p. 728–732. 51, 52

ANEXO A - RESUMO DO ARTIGO PUBLICADO NO Encontro Nacional de Inteligência Artificial e Computacional (ENIAC 2013)

TÍTULO: ALGORITMO DE COLÔNIA DE FORMIGAS APLICADO AO PROBLEMA DE ALOCAÇÃO DE EQUIPAMENTOS EM SATÉLITES

Resumo: Descreve-se aqui um método de solução do problema de alocação de equipamentos em satélites utilizando o algoritmo de Otimização por Colônia de Formigas. Para cada equipamento considera-se a sua massa e sua geometria. O objetivo é posicionar os equipamentos de tal maneira que o centro de massa do satélite seja minimizado e seu momento de inércia seja maximizado. Dois estudos de caso são apresentados. (ALVES et al., 2013b)

ANEXO B - RESUMO DO ARTIGO PUBLICADO NO Encontro de Modelagem Computacional (EMC 2013)

TÍTULO: ALGORITMO DAS FORMIGAS COM HEURÍSTICA DE BALANCEAMENTO APLICADO AO PROBLEMA DE ALOCAÇÃO DE EQUIPAMENTOS EM SATÉLITES

Resumo: Este trabalho trata do problema de alocação de equipamentos em satélites, visando a minimização do centro de massa do satélite e a maximização do momento de inércia, utilizando-se o algoritmo de Otimização por Colônia de Formigas (ACO). Propõe-se aqui uma heurística de balanceamento de carga, acoplada ao ACO, que produz melhores resultados que a técnica de ordenação da colocação de objetos utilizada na literatura. Apresentamos também uma plataforma de busca de soluções otimizadas para o problema, correntemente em desenvolvimento. (ALVES et al., 2013a)

ANEXO C - RESUMO DO ARTIGO ACEITO PARA O IEEE International Conference on Fuzzy Systems (FUZZIEEE, 2015)

TÍTULO: A Balancing Heuristic for Spacecraft Equipment Layout Optimization

Abstract: We introduce a load balancing heuristic, called SABH, for allocating equipment in satellite modules. The goal is to position equipment such that the distance between the center of mass produced by the layout and the center of the module is minimized and its moment of inertia, relative to the module main axis, is maximized. A set of experiments compare the performance of the heuristic per se with its combined use with two nature-inspired optimization algorithms, Ant Colony Optimization (ACO) and Multi-Particle Collision Algorithm (M-PCA). (ALVES et al., 2015)